Contemporary Mathematics
http://ojs.wiserpub.com/index.php/CM/

Research Article

An Optimized Beluga Whale Approach for Migration to Reduce Power and Service Level Agreement in Real-Time System

Rukmini S[1*], Shridevi Soma[2]

[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
E-mail: rukminis@pdaengg.com

Received: 1 November 2023; **Revised:** 9 November 2023; **Accepted:** 13 November 2023

Abstract: Managing power consumption in cloud data centers has become a critical challenge. Live container migration is a technology supporting energy efficiency in this context. To address the hurdles of power consumption management, thereby mitigating carbon emissions and minimizing service level agreement (SLA) violations, we propose an approach utilizing a real-time server with the Beluga Whale Optimization Algorithm (BWOA) for container migration. The proposed approach aims to optimize energy consumption while ensuring SLA compliance. The BWOA, a machine learning-based method, is employed to predict the resource requirements of containers and migrate them to hosts with sufficient resources. We implemented the proposed approach in a real-time cloud server and compared its performance with other algorithms in terms of response time. The results demonstrate a remarkable 30% improvement in response time, leading to reduced SLA violations and optimized power consumption in containerized data centers.

***Keywords*:** virtual machine (VM), beluga whale optimization algorithm (BWOA), power optimization, service level agreement (SLA), container migration

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

Nomenclature

VM	Virtual Machine
BWOA	Beluga Whale Optimization Algorithm
SLA	Service Level Agreement
LL	Least Loaded
RR	Round Robin
RA	Random Allocation
VM	Virtual Machine

DOI: https://doi.org/10.37256/cm.6120253854

1. Introduction

Container migration involves the process of relocating a container from one host to another. This is a technique employed to optimize resource utilization in data centers. This migration can occur through two methods: cold migration, where the container is halted and then moved to another host, and live migration, which involves transferring the container while it is still operational [1]. The optimization of power in data centers is a crucial facet of energy conservation. Virtualization technology aids in achieving this by efficiently migrating virtual machines and containers among hosts, consequently decreasing the number of inactive hosts and overall energy consumption. Different live migration methods, including pre-copy, post-copy, and hybrid-copy, have been explored in reference [2]. As the challenge of enhancing power efficiency in data centers grows, minimizing SLA violations becomes another significant hurdle in cloud computing. The SLAs establish agreements between a cloud provider and their customers to ensure the uninterrupted operation of their businesses. To circumvent SLA violations, cloud services must maintain constant availability. One key aspect of Quality of Service (QoS) [3] is response time, directly proportional to SLA adherence. Prolonged response times lead to diminished cloud service availability, increased CPU utilization, and subsequently, heightened energy consumption in data centers. Existing research has predominantly focused on simulations or implemented Linux containers such as CRIU and Podman. Currently, limited research is conducted on real-time cloud servers due to the substantial costs associated with setting up a cloud environment.

This paper employs the Beluga Optimization Algorithm to implement container migration in cloud data centers, aiming to reduce both power consumption and Service Level Agreement violations in a real-time cloud server. The chosen cloud server for this study is the IBM Power 9, where virtual machines (VMs) are created, and a controller manages their activities. The migration process involves setting a threshold for the number of containers to be migrated, and these containers are then moved among various virtual machines. The analysis incorporates different migration strategies, including Least Loaded (LL), Round Robin (RR), Random Allocation (RA), and BWOA. The response time for each algorithm, considering various container quantities, serves as the basis for evaluating the performance of container migration across all implemented algorithms.

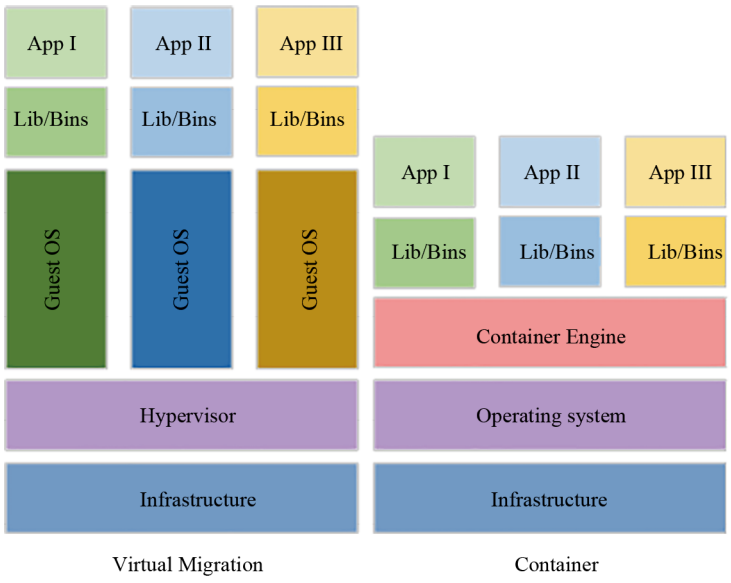

Figure 1. A pictorial representation of virtual migration and container

This manuscript is divided into the following sections: Section 2 covers related work of container migration, and the model of the system. Section 3 deals with the methodology of proposed BWOA. Sections 4 discusses the experimental

setup of the study and its parameters. Section 5 deals the results and discussion, where the performance of different algorithms under different parameters are demonstrated. The conclusion and future work are given in Section 6.

2. Related work

Power optimization and energy efficiency have become critical concerns in modern data centers as computational resources are in increasing demand and the rising costs of energy consumption. Containerization, a popular technology for resource management and isolation, offers benefits like improved scalability and flexibility. However, the dynamic nature of containerized environments poses challenges in effectively managing power consumption while meeting performance requirements. Recent research has focused on leveraging container migration techniques to optimize power consumption in data centers. Container migration involves moving containers between physical servers to consolidate workloads and improve resource utilization. Researchers aim to achieve power optimization without compromising performance or violating SLAs by dynamically reallocating containers based on workload patterns and power consumption profiles.

In a study by Wei Li et al. [4], a machine learning-based approach for container migration in power-constrained data centers was developed. Their algorithm predicts future workload patterns and proactively migrates containers to balance power consumption across servers, showing improved power efficiency and SLA compliance. The authors of [5] have introduced a hybrid approach combining workload prediction and container migration to optimize power consumption. The algorithm leverages workload forecasting models to predict future resource demands and strategically migrates containers to minimize power usage, demonstrating substantial energy savings and enhanced SLA adherence.

Further, researchers have been worked on container migration algorithms. The article [6] discusses different approaches for energy-optimized live migration using virtual machines and containers. The majority of the work has been carried out by researchers considering CPU utilization as the parameter to optimize energy consumption, while some works have used other parameters like memory, disk space, application execution time for active server along with CPU utilization. The survey also shows that most of the work has been implemented using CloudSim and there is more scope for developing solutions for optimal migration in virtual machines and containers. The container migration algorithm that utilizes machine learning techniques to optimize resource allocation in cloud data centers and the algorithm discussed in [7] predicts resource demands and migrates containers, accordingly, leading to improved resource utilization. In [8], the work presents a container migration algorithm based on reinforcement learning to achieve load balancing in edge computing environments. The algorithm learns optimal migration policies to distribute workload evenly across edge servers, enhancing performance and resource utilization. A multi-objective container migration algorithm for performance optimization in cloud data centers is discussed in [9]. This algorithm considers factors such as response time, resource utilization, and energy consumption to migrate containers and achieve a balance between performance and efficiency. The Authors [10] propose an adaptive container migration algorithm for QoS-aware resource management in cloud data centers and this dynamically migrates containers based on workload changes and QoS requirements to ensure efficient resource allocation and service quality. A power-aware container migration algorithm that considers both workload characteristics and power consumption profiles by intelligently migrating containers to minimize power usage while maintaining SLA requirements [11]. The experimental results demonstrated significant energy savings compared to traditional resource allocation methods. Another research article [12] discusses power optimization in virtualized data centers is “Optimization of power and migration cost in virtualized data centers using dynamic placement of virtual machines”. This work presents an optimization approach using dynamic placement of virtual machines in cloud computing approach. The literature in this context highlights the growing interest in power optimization through container migration in data centers. By dynamically reallocating containers based on workload patterns and power consumption profiles, researchers aim to achieve power efficiency while ensuring SLA compliance. However, further investigations need to be explored for practical implementation of these techniques in real-world data center environments.

3. Methodology

In this work, the system model consists of 5 VM's called Controller, Worker 1, Worker 2, Worker 3, Worker 4 among which one VM acts as controller which manages the activities of worker1. Worker1 is responsible to migrate the container to all the workers including worker1 based on the algorithm. The proposed system model is shown in Figure 2.

Containerized data centers have gained significant popularity in recent years because of their flexibility, scalability, and resource efficiency. However, the increasing demand for computational resources in data centers has led to a surge in power consumption, resulting in higher operational costs and environmental concerns. To address these challenges, researchers and industry practitioners have focused on developing efficient power optimization techniques and SLA management strategies. One promising approach in this domain is the utilization of the Beluga algorithm and container migration techniques.

The Beluga algorithm is an optimization algorithm [13] that aims to minimize power consumption in data centers by dynamically adjusting the allocation of computational resources. In addition, it considers factors such as workload characteristics, power consumption profiles, and SLA requirements to make intelligent resource allocation decisions [14]. Container migration, on the other hand, involves the movement of containers between physical servers to optimize resource utilization and reduce power consumption [15]. By migrating containers based on workload patterns and power consumption profiles, it is possible to consolidate workloads and dynamically allocate resources to meet SLA requirements while minimizing power usage. Several research studies have explored the integration of the Beluga algorithm and container migration techniques to achieve real-time power optimization and SLA management in containerized data centers. When applying the BWOA to container migration, the following procedure is needed to be implemented. The procedure starts with an initialization and ends with the output as elaborated the following steps (A)-(H). A pictorial representation of BWOA is shown in Figure 3.

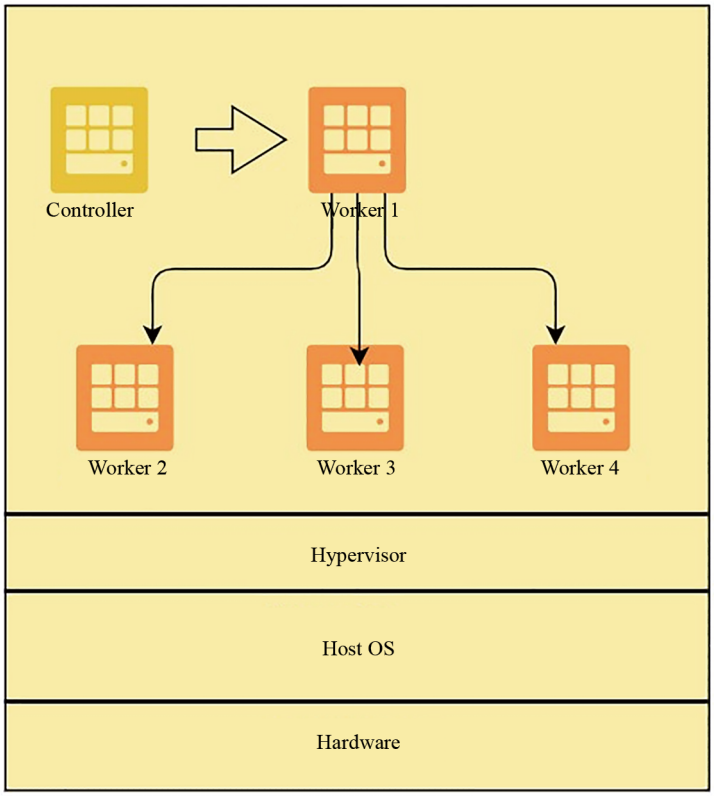

Figure 2. Proposed system model

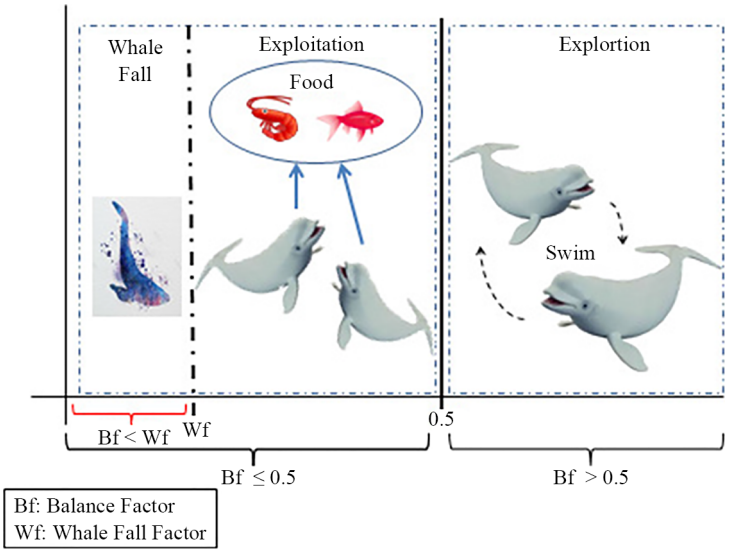

Figure 3. A pictorial representation of beluga whale optimization algorithm

A. Initialization: Initialize the population of the candidate solutions, where each solution represents a possible placement of containers on computing resources. This can be done randomly or using some heuristic initialization method.

$$X = \begin{bmatrix} x1,\ 1 & x1,\ 2 & \cdots & \cdots & x1,\ d \\ x2,\ 1 & x2,\ 2 & \cdots & \cdots & x2,\ d \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ xn,\ 1 & xn,\ 2 & \cdots & \cdots & xn,\ d \end{bmatrix} \tag{1}$$

B. Evaluation: Evaluate the fitness of each candidate solution. The fitness function should consider various factors relevant to container migration, such as resource utilization, network latency, load balancing, or any other objective you want to optimize.

$$R = \int_{n=0}^{n=N} r/n \tag{2}$$

In equation (2) the considered fitness function of this work is the response time, which is optimized, nth number of containers migrated assigned from 0 to N. N-24, and R is the response time optimized, and r is response time of i th iteration.

C. Social Interaction: A social interaction among beluga whales is simulated to enhance exploration and exploitation in BWOA. This can be achieved by allowing the candidate solutions to interact and exchange information. For container migration, this interaction can be implemented by allowing the candidate solutions to exchange information about their placements or migration strategies. The exploration and exploitation phase can be decided based on balance factor.

$$Bf = B0(1 - T/2Tmax) \tag{3}$$

In (3) T is nth current iteration, Tmax Iteration T is the current iteration, Tmax is the maximum iteration, and $B0$ is randomly chosen each iteration changes between (0, 1). During the exploration phase > 0.5 identifies the balance factor, while $Bf < 0.5$ signals the exploitation phase. Overloaded virtual machines are selected during exploration, and underloaded virtual machines are selected during exploitation.

D. Migration Operation: Apply migration operations to the candidate solutions based on the information exchanged during social interaction. This operation can involve moving containers from one VM to another, considering the constraints and objectives of the migration process. The migration process selects the VM for container to be migrated based on previous position and CPU utilization.

E. Update Population: Update the population by replacing some solutions with new ones generated through migration operations. This ensures diversity and allows for exploration of different migration strategies.

F. Termination Criteria: Determine the termination criteria for the algorithm. The termination criteria are 24 containers which is the max number of containers that can be migrated.

G. Repeat Steps 2-6: Repeat steps 2 to 6 until the termination criteria are met. The termination process is number of containers This iterative process allows the algorithm to search for better container placement solutions.

H. Output: Once the algorithm terminates, the best solution found during the optimization process represents the optimal placement of containers for migration.

Step 1: for each container $X_1, X_2, \ldots, X_k$ do

Step 2: for each VMs $CN_1, CN_2, \ldots, CN_v$ do

Step 3: If container X_j can be accommodated within XN_j, then

Step 4: Assign X_j to XN_j

Step 5: end if

Step 6: end for

Step 7: Configure the control parameters (problem dimension, maximum iterations, population size).

Step 8: Initiate the population with random values.

Step 9: Assess the fitness of the newly generated solutions.

Step 10: Identify the best solution obtained.

Step 11: While $T \leq T_{\max}$

Step 12: Retrieve the probability of a whale fall and the balance factor.

Step 13: For all beluga whale A_i

Step 14: If $Y_f(i) > 0.5\%$ $Y_f(i) > 0.5\%$ is exploration phase,

Step 15: Generate $e_j(j = 1, 2, \ldots, d))$ from the given dimension.

Step 16: Select a beluga whale randomly.

Step 17: Update the new position of the i^{th} beluga whale,

Step 18: If the $Y_f(i) \leq 0.5\%$ during the exploitation phase:

Step 19: Update the random jump strength C1 and calculate the levy flight function.

Step 20: Update the new position of the i^{th} beluga whale.

Step 21: end if

Step 22: ensure that the new positions remain within defined limits and evaluate their fitness values.

Step 23: End the loop for all beluga whales.

Step 24: For each beluga whale A_i during the whale fall phase,

Step 25: If $Y_f(i) \leq 0.5\, d_f$

Step 26: Update the step factor.

Step 27: Calculate the step size.

Step 28: Update the new position of the i^{th} beluga whale.

Step 29: ensure that the new positions are within specified limits and assess their fitness values.

Step 30: end if

Step 31: end the loop for all beluga whales.

Step 32: search for a new solution if it exhibits superior fitness.

Step 33: $f = f + 1$
Step 34: Continue the process until a specified condition is met.
Step 35: Assign VM to CM.

4. Experimental setup

This section demonstrates the experimental details of the virtual machine for the proposed study and other parameters used to do so. The workload/containers used for migration have a threshold value up to 24 as we have worker machines with each taking 280 MB and the rest 720 MB is for container execution. 2. The proposed beluga optimization algorithm is compared in terms of average response time with four algorithms like without migration. Round robin, least-loaded, random allocation. The setup is run for all the five algorithms with number of containers varying from 2 to 24 with an interval of 2. The size of each container of this experimental study is 30 MB. Specification of real-time server host is shown in Table 1 and Table 2 shows the virtual machine. As stated earlier, in this study, utilization of CPU, average response time and energy utilization are key factors to investigate and the same are expressed in (4), (5) and (6) respectively.

$$U = CPU\ max\ ram\ size/n\ (migrated\ container\ size) \tag{4}$$

In (4), the n is number of containers and the size of ram size and container is in MB.

$$R\ =\ It\ is\ time\ taken\ for\ N\ response/N \tag{5}$$

In (5), the N is number of workers.

$$E \alpha R \tag{6}$$

In (6), the R is response time, the taken to respond to any request and E is enegry consumed during processing. The energy consumption is direct proportional to response time as increase in response time indicates some part of energy is consumed by processing elements and majority is by CPU utilization [10].

Table 1. Specifications of host

Server	CPU	Core	Speed	Memory	OS	Hypervisor
POWER9 Processor	POWER9 Processor	20-core	2.7 GHz	32 GB RAM	Ubuntu 20.0	KVM

Table 2. Specifications of virtual machines

Virtual Machine (VM)	CPU	Core	Speed	Memory	OS	Tool
Controller VM	POWER9 Processor	2-core	2.7 GHz	2GB RAM	Centos 8	Ansible
Worker VM	POWER9 Processor	1-core	2.7 GHz	1GB RAM		

5. Results and discussions

This section deals with the real-time experimental results of the VM. As state earlier, the study focus on the improvement of response time by using the proposed BWOA. The effectiveness of the proposed study is verified by comparing with other techniques such as WM, RR, LL and RA. The Table 3 shows the response time for 24 containers of 2 to 24. From the Table 2, it is clear that the proposed algorithm performs better than the other methods. A diagrammatic representation of the Table 2 is shown in Figure 4.

The delay in response time can impact the power consumption of virtual machines in the cloud. When a VM is idle or not actively processing tasks, it can enter a low-power or sleep state to conserve energy. However, when a task is allocated to the VM and processing the workload is required, the VM's resources, such as CPU and memory, are utilized, resulting in increased power consumption. A delay in response time indicates that the VM is actively processing the workload for a longer duration. This prolonged activity leads to higher power consumption compared to a scenario where the response time is shorter.

Table 3. Response time of containers

Number of containers	Response time in seconds				
	WM	RR	L	RA	Proposed BWOA
2	0.08	0.08	0.07	0.09	0.08
4	0.14	0.15	0.08	0.08	0.07
6	0.3	0.18	0.09	0.12	0.08
8	0.49	0.16	0.12	0.13	0.09
10	0.74	0.21	0.14	0.21	0.1
12	0.85	0.26	0.27	0.12	0.11
14	1.08	0.25	0.22	0.15	0.13
16	1.24	0.25	0.24	0.18	0.16
18	1.39	0.31	0.33	0.24	0.18
20	1.58	0.28	0.41	0.25	0.22
22	1.74	0.28	0.55	0.44	0.27
24	1.93	0.29	0.53	0.43	0.27

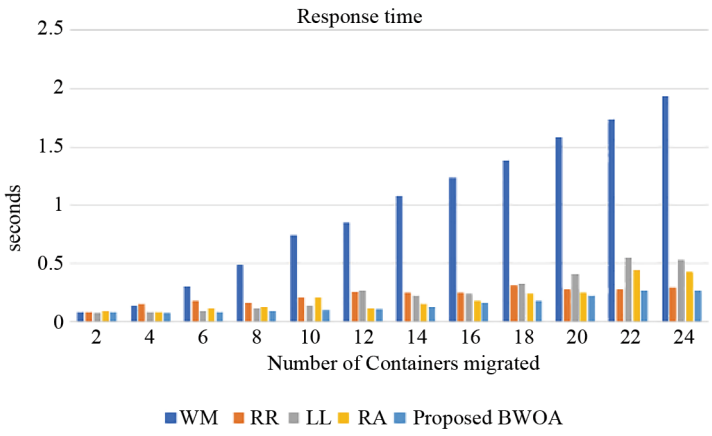

Figure 4. Comparison of response time of proposed BWOA with other works

Additionally, a delay in response time can also impact overall resource utilization in the cloud environment. If the workload is not efficiently managed and causes resource contention or inefficient resource allocation, it can further increase power consumption. In summary, a delay in response time can increase power consumption in the cloud as it prolongs the active processing of the VM, leading to higher resource utilization and potentially inefficient resource allocation. From Table 2 and Figure 3, the proposed BWOA has a very low response time. As shown in Figure 4, CPU utilization and response time increase with increasing CPU utilization, while the proposed BWOA has a shorter response time. According to Figure 5, CPU utilization increases as the number of containers increases. According to Figure 6, the Proposed BWOA has a reduced SLA due to a shorter response time. As shown in Figure 7, the average response time of all algorithms is observed, and it is noted that Proposed BWOA has a very low response time and smaller CPU utilization, hence, Proposed BWOA consumes less energy than others. The energy consumption of the proposed BWOA while comparing with the other approaches are shown in Figure 8, it is evident that the proposed BWOA performance in terms of less consumption is superior to the other approaches.

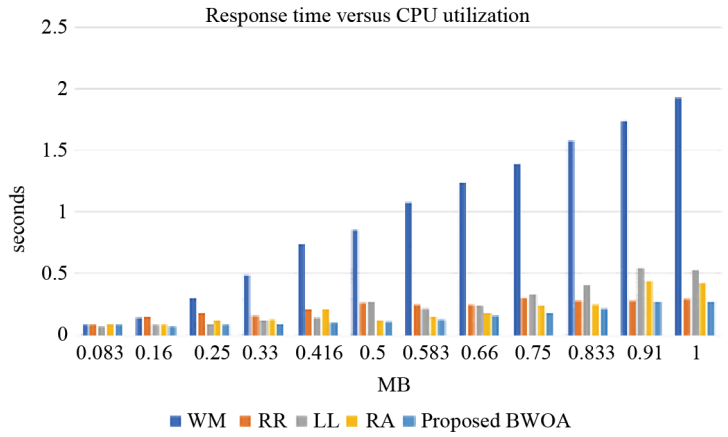

Figure 5. Comparison of response time with CPU utilization

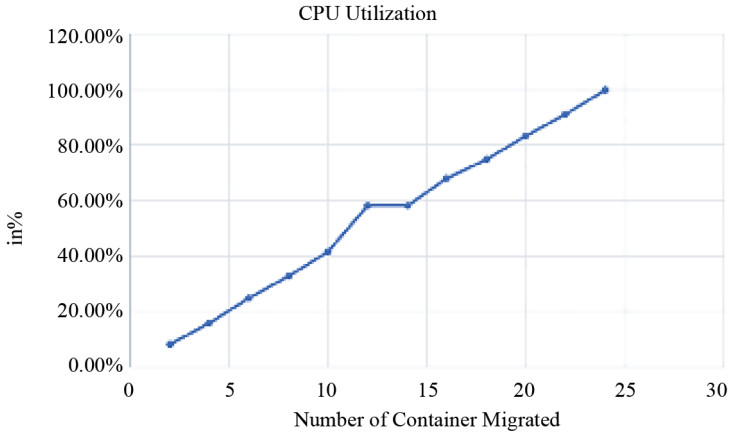

Figure 6. Calculations of CPU utilization

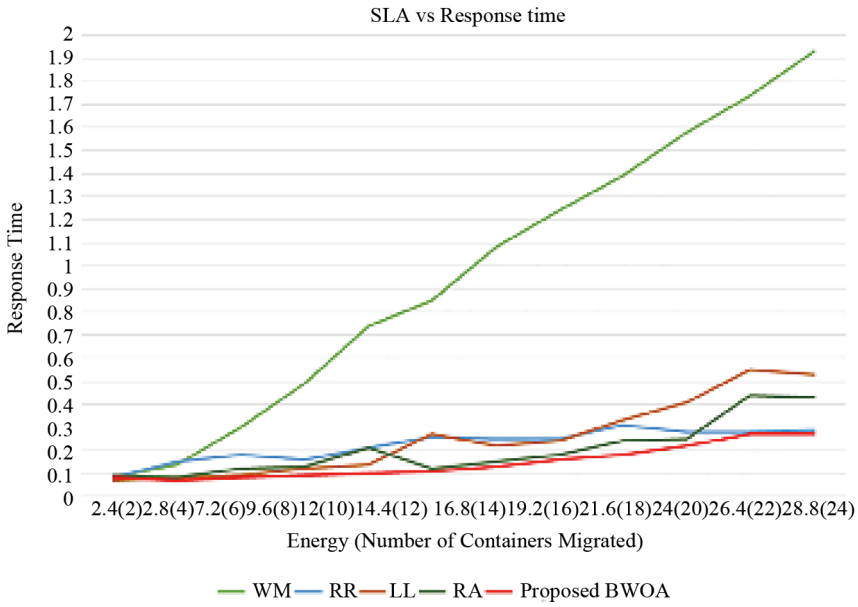

Figure 7. Comparison of response time with SLA

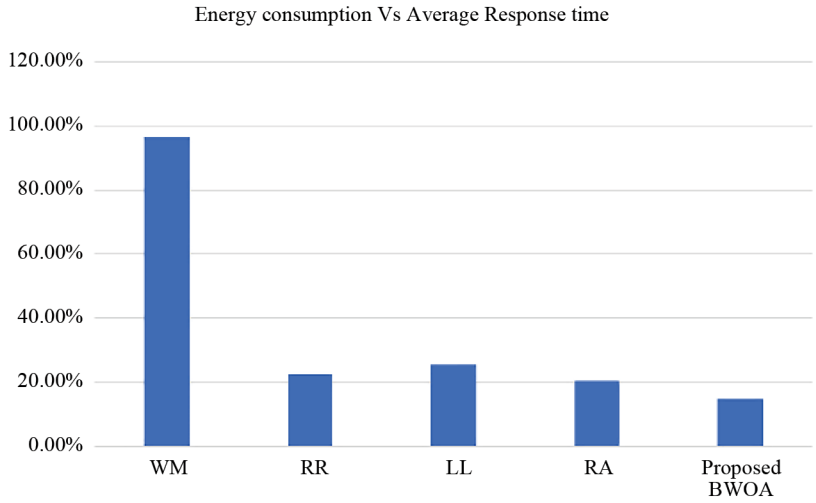

Figure 8. Energy consumption

Figures 9 to 12 represent CPU utilization. These figures show the average response of each algorithm's behaviour for different numbers of containers ranging from 02 to 24 (the threshold value). It is observed in the figures 9-12 that the proposed Beluga Optimization Algorithm outperforms all four algorithms. The Without Migration (WM) algorithm has a longer response time, as the containers are with worker 1 and have not been migrated to other workers.

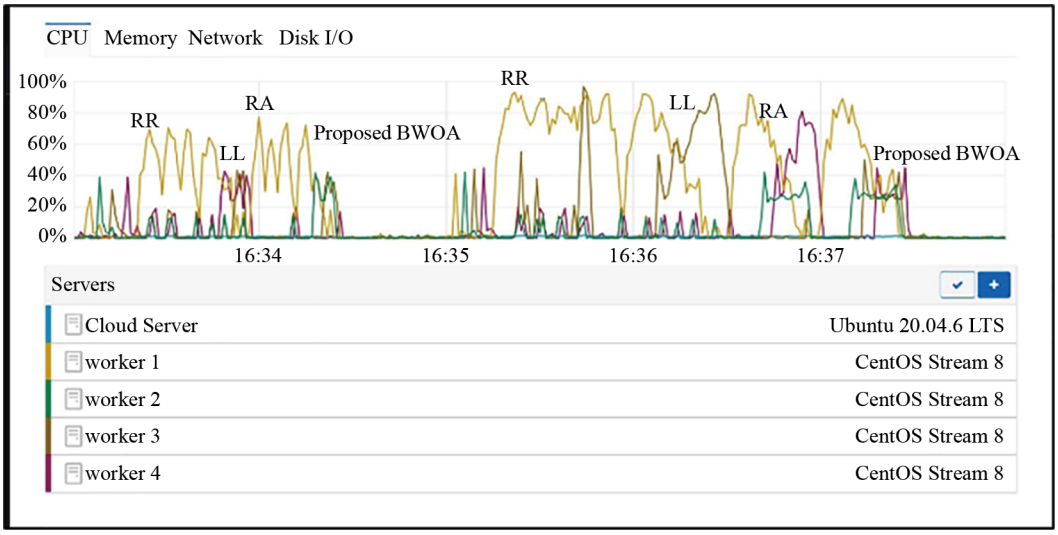

Figure 9. CPU utilization of 2 and 4-containers after migration of all the algorithms

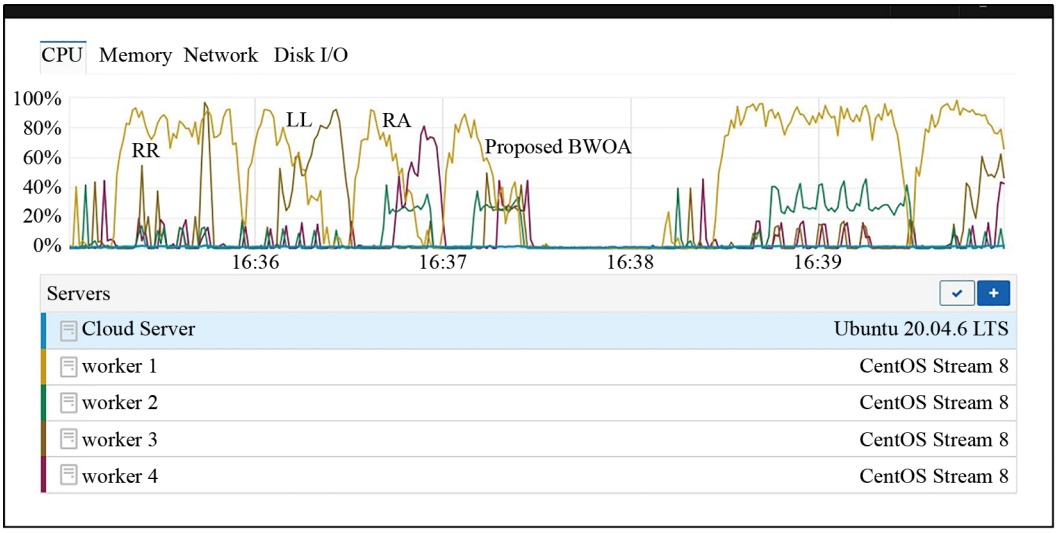

Figure 10. CPU utilization of 6 containers after migration of all the algorithms

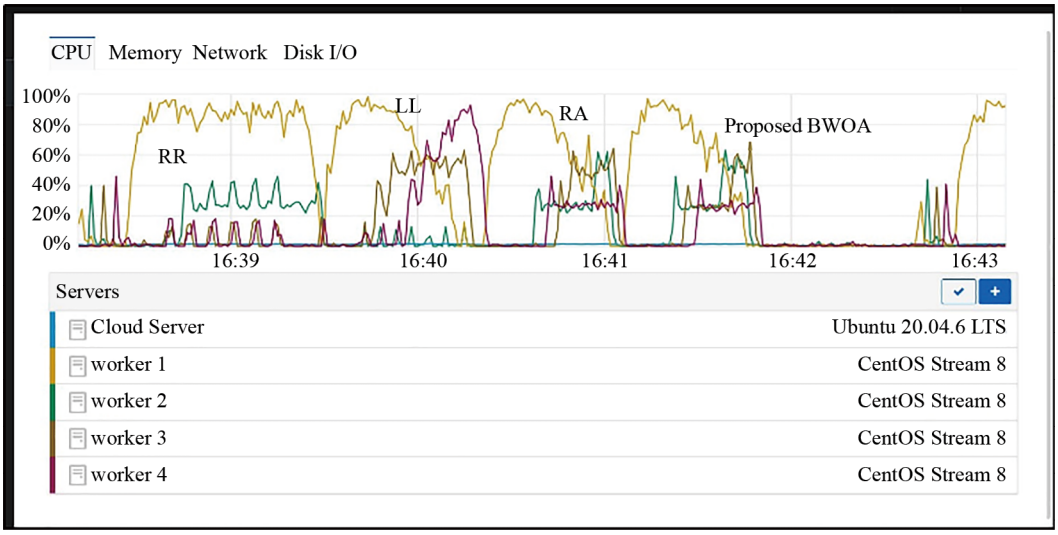

Figure 11. CPU utilization of 8 containers after migration of all the algorithms

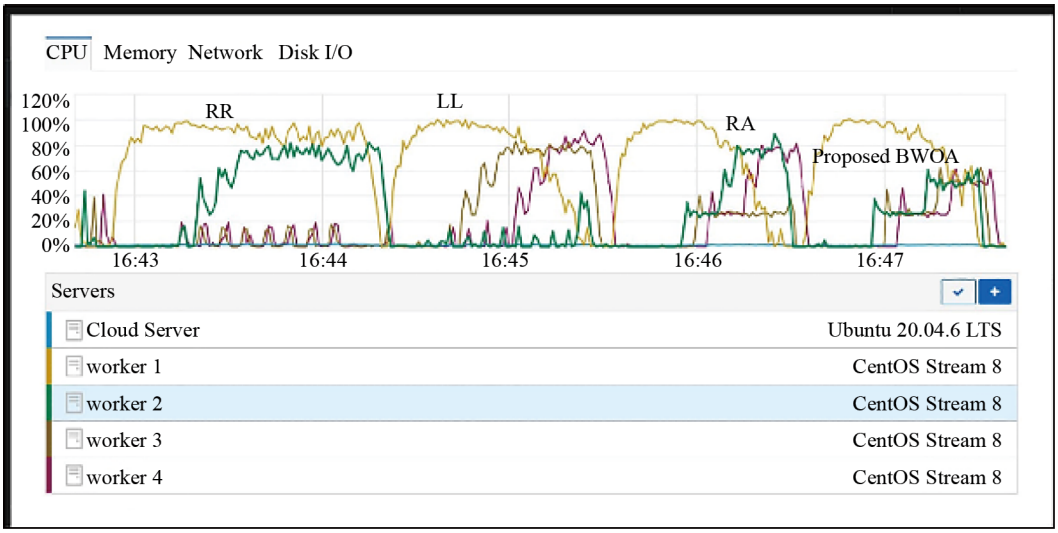

Figure 12. CPU utilization of 10- containers after migration of all the algorithms

6. Conclusions

In this work the Beluga Whale Optimization algorithm for container migration is proposed. From the experimental results it is shown that the proposed method reduces the response time by 30% as comparison to the response time of other algorithms as considered in this study. By doing so, the SLA will be reduced, and the power consumption of the virtual machine will be optimized during container migration. The algorithm is implemented in a real-time cloud server with the benefit of SLA and power optimization.

Future Scope:

- Can be considered for checking other data center data set like PlanetLab and Google data center.
- The threshold value used in this work is the number of containers against the memory size of the virtual machine; further work could be conducted based on the CPU utilization as the threshold value.

- Adaptive to the renewable energy applications by integrating the other optimization techniques [16–20].

Conflict of interest

The authors declare there is no conflict of interest at any point with reference to research findings.

References

[1] Clark C, Fraser K, Hand S, Hansen JG, Jul E, Limpach C, et al. A short course in soil and rock slope engineering. *NSDI'05: 2nd Symposium on Networked Systems Design & Implementation*. London: Thomas Telford Publishing; 2001. p.273-286.

[2] Xiong Y, Chen Y, Jiang K, Tang Y. An energy-aware task consolidation algorithm for cloud computing data centre. *International Journal of High-Performance Computing and Networking*. 2017; 10(4-5): 352-358.

[3] Maarouf A, Marzouk A, Haqiq A. A novel penalty model for managing and applying penalties in cloud computing. *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*. Marrakech, Morocco: IEEE; 2015. p.1-6.

[4] Li W, Fan Q, Cui W, Dang F, Zhang X, Dai C. Dynamic virtual machine consolidation algorithm based on balancing energy consumption and quality of service. *IEEE Access*. 10th ed. IEEE; 2022. p.80958-80975.

[5] Ahamed Z, Khemakhem M, Eassa F, Alsolami F, Basuhail A, Jambi K. Deep reinforcement learning for workload prediction in federated cloud environments. *Sensors*. 2023; 23(15): 6911.

[6] Rukmini S, Shridevi S. An optimal solution to reduce virtual machine migration SLA using host power. *Measurement: Sensors*. 2023; 25(12): 100628.

[7] Khan T, Tian W, Zhou G, Ilager S, Gong M, Buyya R. Machine learning (ML)-centric resource management in cloud computing: A review and future directions. *Journal of Network and Computer Applications*. 2022; 204: 103405.

[8] Hashemi SM, Sahafi A, Rahmani AM, Bohlouli M. Energy-aware resource management in fog computing for IoT applications: A review, taxonomy, and future directions. *Software: Practice and Experience*. 2023; 54(1): 1-40.

[9] ANSIBLE. Available from: https://www.ansible.com/ [Accessed 10th September 2023].

[10] Hazra A, Donta PK, Amgoth T, Dustdar S. Cooperative transmission scheduling and computation offloading with collaboration of fog and cloud for industrial IoT applications. *IEEE Internet of Things Journal*. 2022; 10(5): 3944-3953.

[11] 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. Available from: https://doi.org/doi:10.1016/j.simpat.2020.102127.

[12] Zhou Z, Abawajy J, Chowdhury M, Hu Z, Li K, Cheng H, et al. Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms. *Future Generation Computer Systems*. 2017; 86(6): 836-850.

[13] Usmani Z, Singh S. A survey of virtual machine placement techniques in a cloud data center. *Procedia Computer Science*. 2016; 1(78): 491-498.

[14] Zhong C, Li G, Meng Z. Beluga whale optimization: A novel nature-inspired metaheuristic algorithm. *Knowledge-Based Systems*. 2022; 251: 109215. Available from: https://doi.org/doi:10.1016/j.knosys.2022.109215.

[15] Zhang W, Zhang H, Chen H, Zhang Q, Cheng AM. Improving the QoS of web applications across multiple virtual machines in cloud computing environment. *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*. Shanghai, China: IEEE; 2012. p.2247-2253.

[16] Satyanarayan R. A novel approach for energy-efficient container migration by using gradient descent namib beetle optimization. *Contemporary Mathematics*. 2023; 5(1). Available from: https://doi.org/doi:10.37256/cm.5120243085.

[17] Krishna VBM, Vuddanti S. Identification of the best topology of delta configured three phase induction generator for distributed generation through experimental investigations. *International Journal of Emerging Electric Power Systems*. 2021; 23(3): 329-341.

[18] Krishna VBM, Sandeep V. Design and simulation of voltage sensor-based electronic load balance controller for SEIG based isolated load applications. *Journal of Advanced Research in Dynamical and Control Systems*. 2020; 12(3): 345-352.

[19] Krishna VM, Sandeep V, Murthy SS, Yadlapati K. Experimental investigation on performance comparison of self excited induction generator and permanent magnet synchronous generator for small scale renewable energy applications. *Renewable Energy*. 2022; 195(4): 431-441.

[20] Vatambeti R, Dhal P. Synergistic optimization of unit commitment using PSO and random search. *Contemporary Mathematics*. 2024; 5(1): 698-710.