Improved Cat Swarm Optimization Algorithm for Load Balancing in the Cloud Computing Environment

Wang Dou*
Department of Engineering Management,
Sichuan College of Architectural Technology, Deyang 618000, Sichuan, China

***Abstract*—Recently, cloud computing has gained recognition as a powerful tool for providing clients with flexible platforms, software services, and cost-effective infrastructures. Cloud computing is a form of distributed computing that allows users to store and process data in a virtual environment instead of a physical server. This is beneficial because it allows businesses to quickly scale up or down their computing capacity, reducing the need to invest in expensive hardware. As cloud tasks continue to grow exponentially and the usage of cloud services increases, scheduling these tasks across diverse virtual machines poses a challenging NP-hard optimization problem with substantial requirements, including optimal resource utilization levels, a short execution time, and a reasonable implementation cost. The issue has consequently been addressed using a variety of meta-heuristic algorithms. In this paper, we propose a new load-balancing approach using the Cat Swarm Optimization (CSO) algorithm in order to distribute the load among the various servers within a data center. Statistical analyses indicate that our algorithm is superior to previous research with regard to energy consumption, makespan, and time required up to 30%, 35%, and 40%, respectively.**

***Keywords*—Cloud computing; resource utilization; load balancing; optimization**

I. INTRODUCTION

Cloud computing provides on-demand, convenient network access to a range of customizable computing resources, including services, applications, storage, and servers, instantly available by service providers without much effort on management's part [1]. A cloud computing model can be deployed in four ways: public, private, community, and hybrid, representing its underlying structure. As the name suggests, a public cloud is a cloud environment publicly accessible by many cloud customers without any restrictions placed upon them by the cloud service provider [2]. There are two scenarios in which a private cloud can be deployed. The cloud service provider may manage and maintain a private cloud exclusively within an organization. In the second scenario, the private cloud is deployed exclusively within a single organization and is managed and controlled by the organization. Regardless of the scenario, a private cloud is used privately by a single organization within a controlled environment. Community clouds are cloud environments that restrict access to organizations and cloud customers who share the same objectives [3]. Members of the community may jointly maintain and manage this type of cloud, or a separate cloud service provider may provide its services to accommodate this type of cloud. Hybrid clouds are cloud environments that combine multiple cloud deployment models. Public and private clouds may be combined in this manner. Organizations and customers often opt for this cloud environment due to security concerns, whereas less sensitive data may be stored in a public cloud environment [4].

Cloud computing is also comprised of three key services, Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). IaaS is concerned with providing hardware as a service. IaaS provides users with the ability to rent servers, storage, and networks, while PaaS provides developers with an on-demand platform to develop, test, and host applications. Lastly, SaaS provides users access to applications hosted on the cloud, usually through a web browser. SaaS is the most popular service, offering users the most convenience and scalability. It eliminates the need to manage hardware and software and allows users to access software on demand [5]. Cloud computing brings together several key attributes, including measured service, rapid elasticity, resource pooling, broad network access, and on-demand self-service. In general, cloud services can be quantified through various metrics, such as bandwidth, data, and time. The cost of cloud computing services is usually determined by the resources used, and the cost savings can be significant compared to traditional IT solutions [6]. The level of elasticity in cloud computing refers to the ability of the system to adapt to changes in workloads through automated provisioning and de-provisioning and the availability of resources based on that. In a resource pooling model, virtual and physical resources are pooled to serve multiple consumers through a multi-tenant model, with resources allocated and reallocated dynamically according to consumer demand. Broad network access involves locating resources on the network and accessing them via different computing platforms and devices, including tablets, smartphones, laptops, and various types of computers. The concept of on-demand self-service involves allowing users to access data and resources in the cloud whenever they need to without requiring human assistance [7].

In order to improve the performance of a cloud, the workload must be balanced among the servers. Due to heavy processing tasks or numerous processes, some servers may be under high utilization. At the same time, some other servers may be idle. This situation would result in a decrease in the efficiency of the network, while the idle servers would waste the network's energy. It is also necessary to wait for the busy server to complete its in-process tasks before sending requests to an overloaded server [8]. The reason for this delay is that a server can only handle a limited number of requests

simultaneously. This results in a decrease in service availability, which causes dissatisfaction. The availability of the service refers to the user's ability to access the desired service at any time. In this regard, load balancing can be used to optimize energy consumption and enhance performance. By transferring virtual machines from an overloaded host to an idle host without interruption, the workload is balanced between the servers, resulting in improved network efficiency [9].

In the domain of cloud load balancing, the significance of IoT, machine learning, artificial intelligence, meta-heuristic algorithms, association rule mining, deep learning, and feature selection lies in their collective potential to revolutionize resource allocation, performance optimization, and adaptability in cloud computing environments. The integration of IoT devices enables real-time data collection and monitoring of cloud resources, facilitating dynamic load distribution based on real-time demands. Machine learning and artificial intelligence techniques empower load balancing systems to autonomously learn from historical data and network patterns, enhancing their ability to predict resource demands and make informed load distribution decisions [10-12]. Meta-heuristic algorithms play a pivotal role in fine-tuning load balancing parameters and optimizing resource utilization for complex and dynamic cloud workloads [13]. Deep learning, with its capability to automatically extract intricate patterns from cloud data, enhances load balancing efficiency and enables the identification of performance bottlenecks and anomalies [14-16]. Association rule mining is of utmost significance as it allows for the identification of intricate relationships and dependencies among various cloud resources, enabling more efficient resource allocation and load distribution strategies to optimize the overall performance and utilization of the cloud infrastructure [17]. Feature selection techniques help identify the most relevant and discriminative cloud attributes, enabling streamlined load balancing processes and reduced computational overhead [18]. By harnessing the potential of these cutting-edge technologies, cloud load balancing achieves enhanced scalability, energy efficiency, and response times, ensuring optimal utilization of cloud resources, seamless user experiences, and the ability to adapt to changing cloud conditions effectively.

The proposed approach in this research aims to address the challenges of load balancing in cloud computing environments, which play a crucial role in optimizing resource utilization and enhancing overall system performance. Cloud computing offers a versatile and scalable platform, but efficient load distribution among servers is essential to avoid underutilization and overutilization issues. The motivations for this study stem from the need to overcome inefficiencies caused by imbalanced workloads, which can lead to reduced service availability and energy wastage. The potential benefits of the proposed approach are multi-fold. Firstly, by utilizing the Cat Swarm Optimization (CSO) algorithm, inspired by the foraging behavior of cats, the approach introduces a novel nature-inspired metaheuristic to the field of cloud load balancing. CSO's adaptive and efficient load distribution capabilities enable optimized resource allocation, resulting in reduced energy consumption, improved makespan, and enhanced task execution times. Secondly, by conducting a comprehensive comparative analysis with Q-Learning and MPSO, well-established load balancing methods, the research demonstrates the superiority of the proposed CSO-based approach, highlighting its potential to outperform existing techniques.

The rest of the paper is organized in the following manner. Section II reviews the related works in the field of cloud load balancing. Section III details the proposed method. Experimental results are reported in Section IV. Finally, Section V concludes the paper and suggests some hints for upcoming research.

II. Related Work

Muteeh, et al. [19] presented a multi-resource load-balancing mechanism using the Ant Colony Optimization (ACO) algorithm for cloud computing environments to ensure a well-load-balanced system and reduce cost and makespan time. Experimental results from benchmark workflows are used to validate the algorithm. The results indicate that the execution time and cost are reduced while the available resources are efficiently utilized by maintaining a balanced workload among them. Based on the mimicking behavior of flocks of birds, Mishra and Majhi [20] have developed a load-balancing method in which tasks are viewed as birds and VMs as food patches for the birds. The proposed approach was evaluated using a dataset (GoCJ) logged by Google in 2018 as part of cloudlet execution. This method is intended to improve the performance of the system by decreasing response time and maintaining overall balance. A comparison is made between the proposed technique and previously developed techniques. The proposed approach demonstrates an improvement in resource utilization and reduces the time required for tasks to be completed.

Mapetu, et al. [21] proposed a low-cost and low-complexity version of the PSO algorithm to schedule and balance cloud computing tasks. An objective function is defined that calculates the maximum difference in completion time between heterogeneous virtual machines. Constraints related to updating and optimization affect the objective function. Then, a particle position update strategy is devised with respect to load balancing. The experimental results indicate that the proposed algorithm performs better in task scheduling and load balancing than existing meta-heuristic and heuristic algorithms. Kruekaew and Kimpan [22] proposed an independent approach to cloud computing task scheduling that utilizes the Q-learning and Artificial Bee Colony (ABC) algorithms. ABC algorithm is improved through the use of reinforcement learning techniques. The proposed method aims to distribute workload among virtual machines proportional to resource utilization, cost, and makespan. The experimental results showed that the proposed algorithm was superior to the competitive algorithms with respect to throughput, imbalance degree, cost, and makespan. Sefati, et al. [23] employed Grey Wolf Optimization (GWO) algorithm to maintain proper load balancing based on resource reliability capability. The GWO algorithm attempts to identify unemployed or busy nodes first and, after discovering these nodes, determines the threshold and fitness function of each node. The experiments conducted in CloudSim have demonstrated that this approach has lower response times and

costs than other methods. The algorithm is capable of efficiently optimizing the load balancing process which requires less time and fewer resources.

Due to the complexity and multitude of criteria involved in dynamic task allocation to heterogeneous resources, finding an optimal solution for every scheduling problem in real-time can be extremely challenging. In such cases, researchers often turn to meta-heuristic techniques, which aim to find near-optimal solutions within a reasonable timeframe while exhibiting outstanding performance for the task at hand. In this context, Mirmohseni, et al. [24] have proposed a hybrid method called Fuzzy Particle Swarm Optimization Genetic Algorithm (FPSO-GA). Their approach combines two powerful optimization techniques, namely fuzzy particle swarm optimization and genetic algorithms. By integrating these methods, the researchers seek to leverage the unique strengths of each technique and achieve better performance in dynamic task allocation.

Haris and Zubair [25] have introduced a dynamic load balancing algorithm called Mantaray modified multi-objective Harris hawk optimization (MMHHO). This algorithm utilizes a hybrid optimization approach, combining the benefits of two optimization algorithms: Harris Hawk Optimization (HHO) and Manta Ray Forging Optimization (MRFO). The aim of MMHHO is to address dynamic load balancing in cloud computing environments. The hybridization process involves updating the search space of HHO using MRFO, taking into account factors such as cost, response time, and resource utilization. By considering multiple objectives, the proposed algorithm seeks to enhance the performance of the system in terms of Virtual Machines (VMs) throughput, load balancing among VMs, and maintaining a balance among task priorities by adjusting the waiting time of tasks. The simulation results demonstrated that the MMHHO load balancing scheme outperforms other algorithms, highlighting its efficiency and effectiveness in achieving dynamic load balancing in cloud computing environments.

The reviewed approaches have demonstrated improvements in resource utilization, reduced response times, and overall load balancing efficiency. However, despite their advancements, there are certain gaps in these previous studies that present opportunities for further research. One of the gaps is the need for a more comprehensive comparison and evaluation of these load balancing techniques in diverse and dynamic cloud environments. While some studies have compared their proposed approach with existing techniques, there is a lack of a unified benchmark dataset or standardized evaluation criteria to assess their performance consistently across different cloud scenarios. Additionally, there is limited exploration of hybrid approaches that combine the strengths of multiple algorithms to achieve optimal load balancing results. Furthermore, existing studies predominantly focus on performance metrics such as response time, makespan, and resource utilization, but there is less emphasis on energy efficiency and cost-effectiveness. Considering the growing importance of sustainability and cost optimization in cloud computing, future research should explore load balancing methods that prioritize energy-efficient resource allocation and cost reduction.

III. Proposed Method

This section explains the Cat Swarm Optimization (CSP) algorithm. Then, the proposed load-balancing algorithm is described in detail. Finally, a detailed insight into the proposed algorithm is provided.

A. Cat Swarm Optimization Algorithm

Observations of cat behavior in the real world have driven the design of a novel swarm-based optimization algorithm. CSO mimics many of the behavior patterns of cats, such as their ability to move quickly, their ability to focus on a single target, and their tendency to explore their environment. CSO has been found to be effective in solving complex optimization problems. The CSO explains cats' behavior in two ways: by seeking and tracing. A local search is performed by the former, and a global search is conducted by the latter. In the seeking mode, the individuals are perturbed multiple times so that each can approach the local optimum. In the tracing mode, each cat traces the target at a certain speed and updates its position to match better the swarm's optimal position [26, 27]. Generally, the CSO algorithm consists of the following steps:

- Create N cats with a specified number of dimensions.
- Assign random velocities to each cat.
- Randomly place cats in seeking and tracing modes.
- Compute the fitness of all cats and keep track of the non-dominated cats.
- For each cat, if it is in seeking mode, execute seeking mode actions, else execute tracing mode actions and relocate it to its new location.
- If the termination condition is not satisfied, move to step 3, else stop.

B. Problem Statement

Cloud computing offers cloud services to cloud users, in which tasks are performed within a cloud environment. A cloud typically consists of a large number of Data Centers (DCs) called Physical Machines (PMs), each of which is equipped with a specific computing resource to fulfil the consumer's task. Each cloud consumer includes a variety of tasks to execute on virtual machines. The load balancing process assigns diverse users' tasks to virtual machines and constantly checks loads on virtual machines in the cloud computing environment. According to the time taken to complete each task, the VM load fluctuates as each task takes a different amount of time. Load balancing allocates tasks from an overloaded virtual machine to an underloaded one. A load balancing system in a cloud environment is illustrated in Fig. 1. Cloud services receive a wide range of requests from users, which requires setting up a dynamic environment for executing tasks. Load balancing strategies are applied when the load balancer receives requests from users. The load balancer selects the necessary virtual machines in response to a request. Load balancers can be configured to distribute consumer requests among multiple virtual machines. The Data Center Controller (DCC) is responsible for task management. The load balancer assigns the task to the appropriate virtual machine based on a load-balancing algorithm.

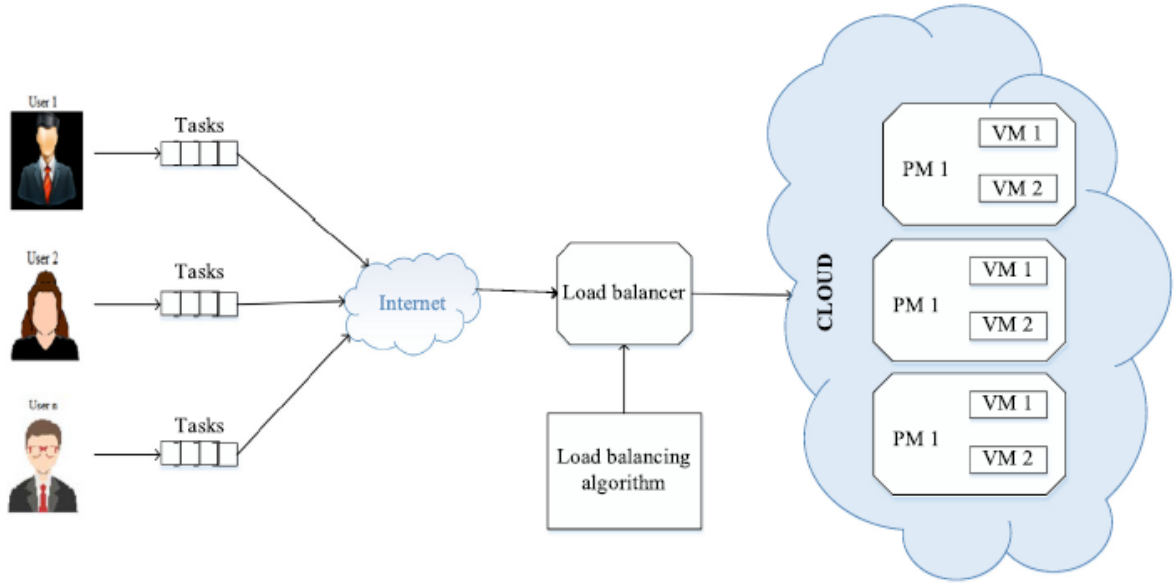

Fig. 1. An overview of cloud load balancing.

C. CSO-based Load Balancing Method

Load balancing is an excellent technique for reducing request waiting times and optimizing resource utilization. In this way, virtual machines can be prevented from becoming overloaded or underloaded, and load distribution can be uniformized among them. In the proposed method, cats represent solutions and attempt to achieve the target through iteration over different cats, striving to achieve the optimal solution. The load balancing problem is expressed according to the following terminology. The primary objective of the proposed method is to achieve optimal load balancing in the cloud. The considered cloud computing model comprises c number of PMs composed of m number of virtual machines. The suggested technique is to improve response time, resource utilization, and cost. Accordingly, Eq. 1 describes the objective function based on the above conditions, where w denotes the weight factor, and e represents the exponential function associated with each parameter. F1, F2, and F3 refer to objective functions for response time, resource utilization, and cost.

$$F = we^{(F1)} + we^{(F2)} + we^{(F3)} \quad (1)$$

1) Response time: It is defined as the time interval between a task arriving in the system and being completed. In terms of load balancing, response time refers to the time required to allocate VMs to lower load conditions in response to a user request. There is an inverse relationship between this factor and the efficiency of the system. The optimal response time is determined by the shortest makespan time as follows.

$$F1 = Fin_t - Arr_t + TDelay \quad (2)$$

In Eq. 2, Arr_t indicates a user demand's arrival time, Fin_t denotes a user demand's ending time, and TDelay signifies transmission delay.

2) Resource utilization: It refers to the utilization level of virtual machines. The goal is to achieve maximum resource utilization and minimize the makespan time. There is a reverse linear relationship between these two terms. Eq. 3 is used to calculate the average utilization of all virtual machines, where N indicates the total number of virtual machines and T_{VM_i} represents the time required to complete all tasks by a jth virtual machine.

$$F2 = \frac{\sum_{i=1}^{N}(T_{VM_i})}{MS \times N} \quad (3)$$

3) Cost: This indicator is calculated by Eq. 4. The cost varies in accordance with the tasks. The cost is determined by the complexity of the task and the resources needed. It is important to consider the cost when evaluating the effectiveness of a system. Optimizing the cost is essential for the success of any system.

$$F3 = \sum_{i=1}^{N}(VM_i^{time} \times VM_i^{cost}) \quad (4)$$

In Eq. 4, VM^{cost} denotes the cost involved in running a virtual machine to perform a task, VM^{time} denotes the duration of the execution of the task, and N represents the number of tasks involved in the workflow.

Suppose a solution comprises five PMs, each containing two VMs. Initially, incoming tasks are distributed randomly among virtual machines. All virtual machines should be balanced in this regard. Overloaded PMs should be migrated to underloaded ones. Fig. 2 illustrates the initial solution. The CSO algorithm is used to optimize the solutions. The CSO algorithm involves two primary steps, seeking and tracing.

PM1		PM2		PM3		PM4		PM5	
1	0	1	1	0	1	1	1	0	1

Fig. 2. Initial solution.

4) Seeking mode: Cats are placed in this mode following the fitness calculation. In this process, the cats look around and adjust their position. In this way, cats search for the best opportunity to capture prey. The CSO algorithm employs four variables: self-position consideration (SP), counts of different dimensions (CD), seeking collection of selected dimensions (SR), and seeking memory pool (MP). The steps for seeking mode are described below;

- Cats duplicate their position seeking Memory Pool (MP).
- Using Eq. 5, the number of dimensions changed (DC) can be calculated for these duplicated positions.

$$S = (1 \pm SR \times R) \times Sn \quad (5)$$

In Eq. 5, R is a random number ranging from 0 to 1, S represents the latest position, and Sn indicates the recent position.

- All cat positions are compared using fitness calculation values. All cats are set to 1 if they are identical. In any other case, we use the identifying probability Eq. 6, where FC_{min} refers to the minimum significance of fitness calculation, FC_{max} stands for the maximum assessment of fitness calculation, FC_i is the fitness calculation value of each cat, and P_v represents the probability of the latest cat.

$$P_v = \frac{|FC_i - FC_j|}{|FC_{max} - FC_{min}|} \quad 0 < i < j \quad (6)$$

5) Tracing mode: In tracing mode, cats are recreated based on their positions in order to update their velocity in dimension d. Eq. 7 described the updating of velocity in tracing mode, where Pn,d is the current location of the cat, Pbest,d is the cat's location with the best result, k is a random number between 0 and 1, and r represents a constant.

$$Vn,d = Vn,d + k \times r(Pbest,d - Pn,d) \quad (7)$$

IV. Experimental Results

This section analyzes the efficiency of the suggested method in light of the simulation results. In order to conduct the cloud computing experiment, the CloudSim simulator was installed on a system configured with an Intel Core i5 CPU, 4 GB RAM, and Windows 8 operating system. Performance has been evaluated in terms of makespan, energy consumption, idle time of tasks, response time of tasks, and number of tasks migrated. Table I presents the simulation environment of the experiments.

TABLE I. Simulation Parameters

Type	Count	Parameter	Value
VM	300	Bandwidth	1 GB/s
		Processor speed	9725MIPS
		Memory	0.5 GB
		VM monitor	Xen
Datacenter	1	OS	Linux
		VM monitor	Xen
		Cost Per Memory	0.05
		Arch	X86
		Cost	3.0
Host	4	Bandwidth	15 GB/s
		MIPS	117.720
		Storage	4 TB
		RAM	8 GB
		Cores	5
		VM monitor	Xen

Fig. 3 and 4 show the energy consumption during load balancing for three different algorithms: Q-Learning, MPSO, and CSO. According to these figures, CSO consumes the least amount of energy during load balancing in comparison to other algorithms. Makespan assesses the performance in another parameter. It measures users' response time for specific tasks in cloud computing, and the response time determines the quality of service. Consequently, the service provider is able to provide the client with a high-quality of service. Fig. 5 and Fig. 6 present a comparison of algorithms according to their makespan under various task and virtual machine configurations. According to these figures, CSO performed significantly better than both Q-Learning and MPSO. Therefore, CSO was able to achieve stability and load balance in a highly efficient manner.

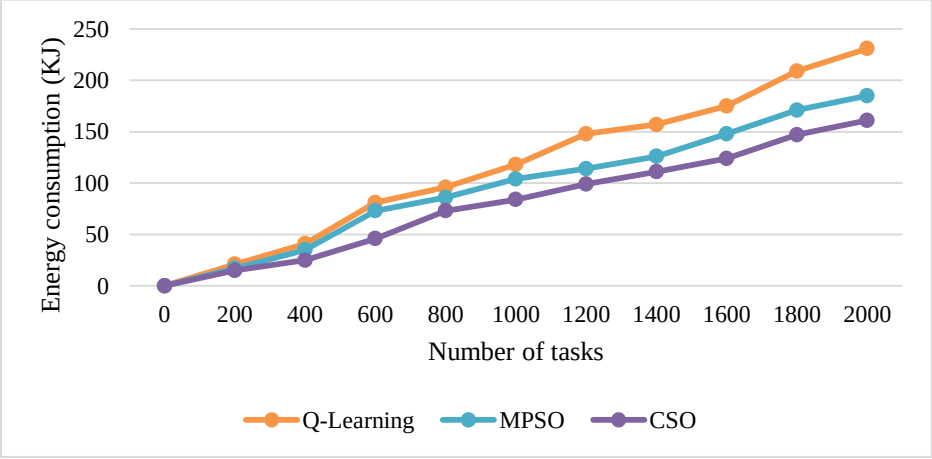

Fig. 3. Energy consumption comparison vs. number of tasks.

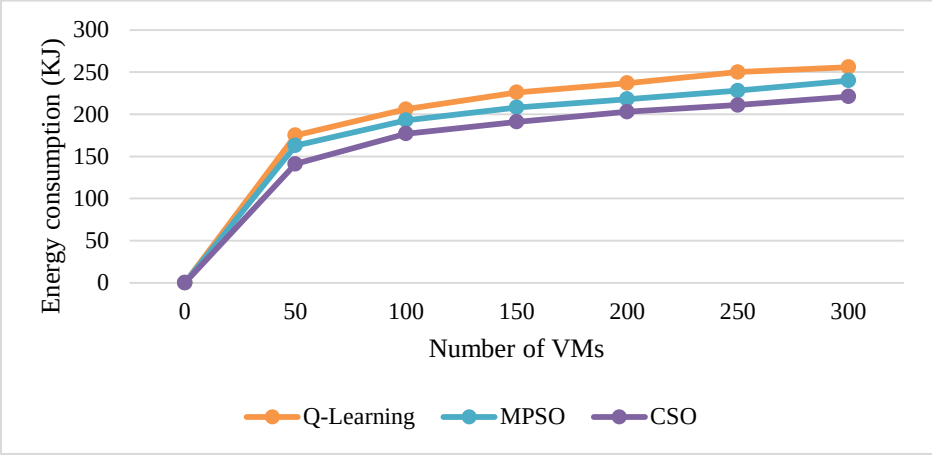

Fig. 4. Energy consumption comparison vs. number of VMs.

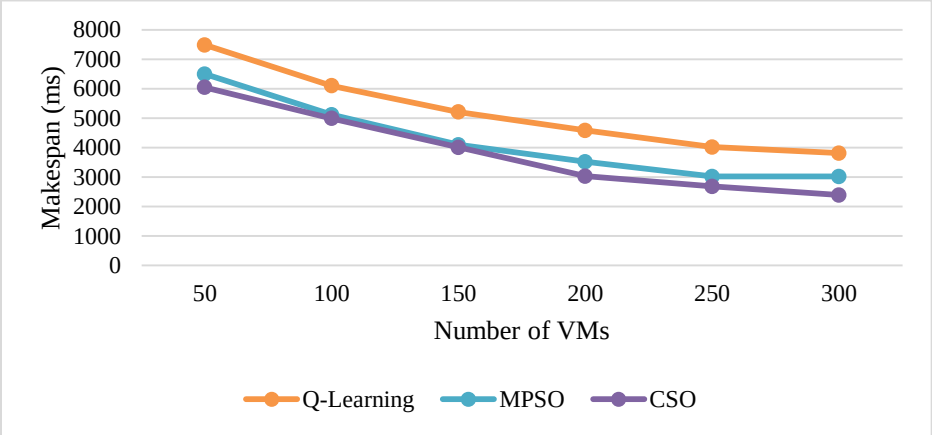

Fig. 5. Makespan comparison vs. number of VMs.

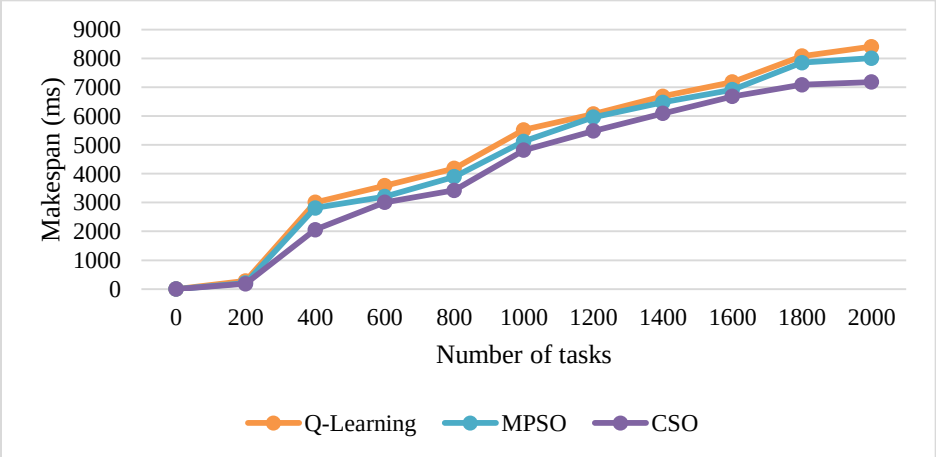

Fig. 6. Makespan comparison vs. number of tasks.

A longer waiting time can be expected if only one task can be executed simultaneously while other tasks are waiting in the ready queue. By increasing the processing power of VMs and allocating more VMs, the waiting time for all tasks will be reduced. Fig. 7 and 8 illustrate the idle time of tasks and the processing time of VMs, respectively. Based on Fig. 7, CSO exhibits a lower idle time than Q-Learning and MPSO for all tasks. Fig. 8 illustrates the processing power of VMs. The processing power has been increased from 200 MIPS to 2500 MIPS, resulting in a reduction in processing time from 9400 ms to 100 ms. Based on the results of the analysis, it can be concluded that CSO balances the load within the cloud network in an effective and efficient manner.

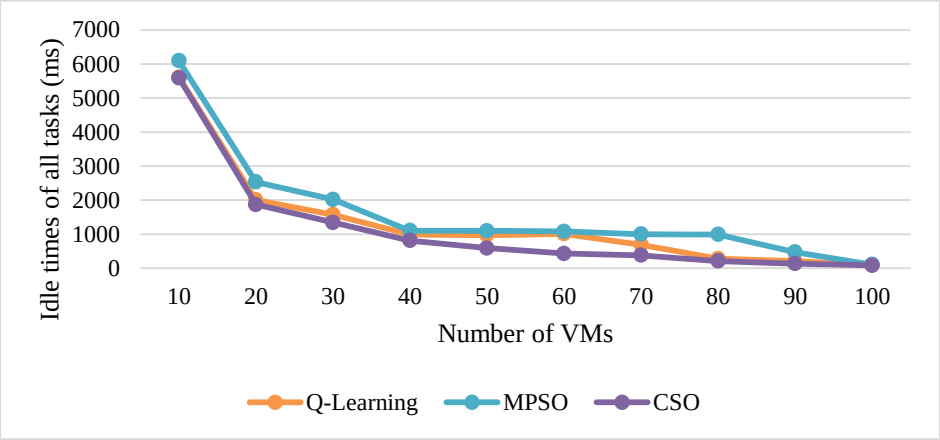

Fig. 7. Idle time of tasks vs. number of tasks.

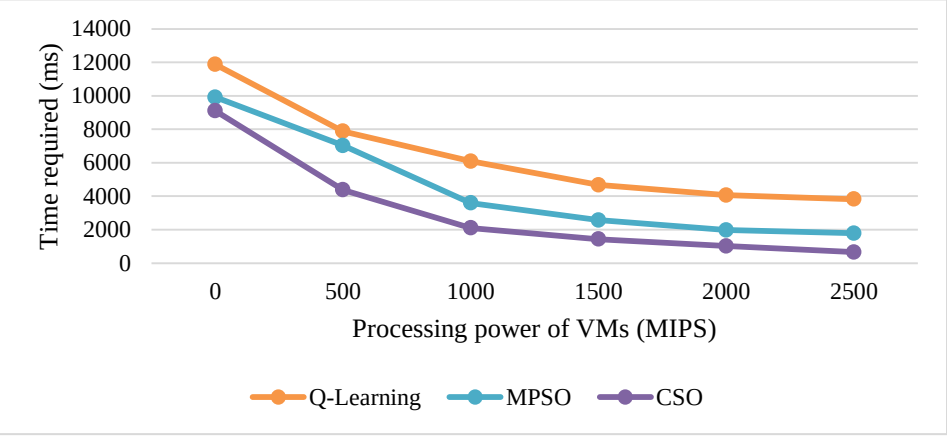

Fig. 8. Time required vs. processing power of VMs.

The presented experimental results demonstrate the superior performance of the proposed CSO-based approach in terms of energy consumption, makespan, and task idle time. CSO's ability to optimize load distribution and achieve efficient resource utilization makes it a promising solution for load balancing in cloud computing environments. The findings support the significance of the proposed method in enhancing the quality of service and overall cloud network performance. The comprehensive analysis of the simulation results reaffirms the potential benefits of adopting CSO for load balancing in cloud computing environments. However, further investigations and real-world validations may be necessary to ascertain its scalability and effectiveness in diverse and dynamic cloud infrastructures.

V. Conclusion

In recent years, cloud computing has experienced an upsurge of interest, primarily due to its usefulness and relevance to contemporary technological trends. Globally, it provides better computational services to its clients by being highly customizable. With the increase in demand for higher processing power, large amounts of data will put a considerable strain on the cloud computing environment. As a result, the need for an efficient scheduling algorithm for cloud computing tasks has become increasingly important. Cloud tasks continue to grow exponentially, and the number of cloud users rapidly increases, making scheduling and balancing these tasks among heterogeneous virtual machines a challenging NP-hard problem with significant constraints, including high resource utilization, quick scheduling, and low implementation cost. This paper proposed a novel load-balancing mechanism using the CSO algorithm, which distributes load among systems in a data center. As demonstrated by simulation results, our algorithm exhibits superior performance to previous research in terms of energy consumption, makespan, and time required by approximately 30%, 35%, and 40%, respectively. While the proposed CSO-based load balancing approach demonstrates promising results, it is essential to acknowledge its limitations. The scalability and adaptability of CSO in highly dynamic and large-scale cloud environments require further investigation. Additionally, the algorithm's performance might be sensitive to parameter settings, emphasizing the importance of proper tuning for optimal results.

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] V. Hayyolalam and A. A. P. Kazem, "A systematic literature review on QoS-aware service composition and selection in cloud environment,"

Journal of Network and Computer Applications, vol. 110, pp. 52-74, 2018.

[3] S. Bharany et al., "Energy efficient fault tolerance techniques in green cloud computing: A systematic survey and taxonomy," Sustainable Energy Technologies and Assessments, vol. 53, p. 102613, 2022.

[4] 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.

[5] J. Dizdarević, F. Carpio, A. Jukan, and X. Masip-Bruin, "A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration," ACM Computing Surveys (CSUR), vol. 51, no. 6, pp. 1-29, 2019.

[6] E. J. Ghomi, A. M. Rahmani, and N. N. Qader, "Load-balancing algorithms in cloud computing: A survey," Journal of Network and Computer Applications, vol. 88, pp. 50-71, 2017.

[7] 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.

[8] F. Ebadifard and S. M. Babamir, "Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment," Cluster Computing, vol. 24, no. 2, pp. 1075-1101, 2021.

[9] G. Saravanan, S. Neelakandan, P. Ezhumalai, and S. Maurya, "Improved wild horse optimization with levy flight algorithm for effective task scheduling in cloud computing," Journal of Cloud Computing, vol. 12, no. 1, p. 24, 2023.

[10] 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.

[11] M. Bagheri et al., "Data conditioning and forecasting methodology using machine learning on production data for a well pad," in Offshore Technology Conference, 2020: OTC, p. D031S037R002.

[12] S. R. Abdul Samad et al., "Analysis of the Performance Impact of Fine-Tuned Machine Learning Model for Phishing URL Detection," Electronics, vol. 12, no. 7, p. 1642, 2023.

[13] S. Aghakhani, A. Larijani, F. Sadeghi, D. Martín, and A. A. Shahrakht, "A Novel Hybrid Artificial Bee Colony-Based Deep Convolutional Neural Network to Improve the Detection Performance of Backscatter Communication Systems," Electronics, vol. 12, no. 10, p. 2263, 2023.

[14] B. M. Jafari, X. Luo, and A. Jafari, "Unsupervised Keyword Extraction for Hashtag Recommendation in Social Media," in The International FLAIRS Conference Proceedings, 2023, vol. 36.

[15] 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.

[16] W. Anupong et al., "Deep learning algorithms were used to generate photovoltaic renewable energy in saline water analysis via an oxidation process," Water Reuse, vol. 13, no. 1, pp. 68-81, 2023.

[17] 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.

[18] M. Javidan, M. Yazdchi, Z. Baharlouei, and A. Mahnam, "Feature and channel selection for designing a regression-based continuous-variable emotion recognition system with two EEG channels," Biomedical Signal Processing and Control, vol. 70, p. 102979, 2021.

[19] A. Muteeh, M. Sardaraz, and M. Tahir, "MrLBA: multi-resource load balancing algorithm for cloud computing using ant colony optimization," Cluster Computing, vol. 24, no. 4, pp. 3135-3145, 2021.

[20] K. Mishra and S. K. Majhi, "A binary Bird Swarm Optimization based load balancing algorithm for cloud computing environment," Open Computer Science, vol. 11, no. 1, pp. 146-160, 2021.

[21] J. P. B. Mapetu, Z. Chen, and L. Kong, "Low-time complexity and low-cost binary particle swarm optimization algorithm for task scheduling and load balancing in cloud computing," Applied Intelligence, vol. 49, pp. 3308-3330, 2019.

[22] B. Kruekaew and W. Kimpan, "Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning," IEEE Access, vol. 10, pp. 17803-17818, 2022.

[23] S. Sefati, M. Mousavinasab, and R. Zareh Farkhady, "Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: performance evaluation," The Journal of Supercomputing, vol. 78, no. 1, pp. 18-42, 2022.

[24] S. M. Mirmohseni, C. Tang, and A. Javadpour, "FPSO-GA: a fuzzy metaheuristic load balancing algorithm to reduce energy consumption in cloud networks," Wireless Personal Communications, vol. 127, no. 4, pp. 2799-2821, 2022.

[25] M. Haris and S. Zubair, "Mantaray modified multi-objective Harris hawk optimization algorithm expedites optimal load balancing in cloud computing," Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 10, pp. 9696-9709, 2022.

[26] A. M. Ahmed, T. A. Rashid, and S. A. M. Saeed, "Cat swarm optimization algorithm: a survey and performance evaluation," Computational intelligence and neuroscience, vol. 2020, 2020.

[27] S.-C. Chu, P.-W. Tsai, and J.-S. Pan, "Cat swarm optimization," in PRICAI 2006: Trends in Artificial Intelligence: 9th Pacific Rim International Conference on Artificial Intelligence Guilin, China, August 7-11, 2006 Proceedings 9, 2006: Springer, pp. 854-858.