Bulletin of Electrical Engineering and Informatics
Vol. 14, No. 3, June 2025, pp. 2066~2076
ISSN: 2302-9285, DOI: 10.11591/eei.v14i3.8098

Chaotic grey wolf optimization based framework for efficient task scheduling in cloud fog computing

Shreyas J.[1], Reena S. Kharat[2], Rajesh N. Phursule[3], Venkata Bhujanga Rao Madamanchi[4], Dhananjay S. Rakshe[5], Gaurav Gupta[6], Malik Jawarneh[7], Sammy F.[8], Abhishek Raghuvanshi[9]

[1]Department of Information Technology, Manipal Institute of Technology Bengaluru, Manipal Academy of Higher Education, Karnataka, India
[2]Department of Computer Engineering, Pimpri Chinchwad College of Engineering, Savitribai Phule Pune University, Pune, India
[3]Department of Information Technology, Pimpri Chinchwad College of Engineering, Savitribai Phule Pune University, Pune, India
[4]Department of Information Technology, RVR & JC College of Engineering, Chowdavaram, Andhra Pradesh, India
[5]Department of Computer Engineering, Pravara Rural Engineering College Loni, Savitribai Phule Pune University, Pune, India
[6]G L Bajaj Institute of Technology and Management Greater Noida, Uttar Pradesh, India
[7]College of Computer Science and Informatics, Amman Arab University, Amman, Jordan
[8]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India
[9]Department of Computer Engineering, Mahakal Institute of Technology, Ujjain, India

Article Info

Article history:

Received Dec 26, 2023
Revised Jun 1, 2024
Accepted Nov 22, 2024

Keywords:

Chaotic grey wolf optimization
Cloud computing
Makespan
Resource utilization
Task scheduling

ABSTRACT

Task scheduling is an essential component of any cloud computing architecture that seeks to cater to the requirements of its users in the most effective manner possible. It is essential in the process of assigning resources to new jobs while simultaneously optimising performance. Effective job scheduling is the only method by which it is possible to achieve the essential goals of any cloud computing architecture, including high performance, high profit, high utilisation, scalability, provision efficiency, and economy. This article gives a framework based on chaotic grey wolf optimization (CGWO) for efficiently scheduling tasks in cloud fog computing. Task scheduling is done with CGWO, ant colony optimization (ACO), and min-max algorithms. CloudSim is used to implement task scheduling algorithms. Makespan time required by CGWO algorithm for 500 tasks is 73.27 seconds. CGWO is taking minimum resources to accomplish the tasks in comparison to ACO and min-max methods. Response time of CGWO is also 3745.2 seconds. CGWO is performing better in terms of Makespan time, response time and resource utilization among the methods used in the experimental work.

Corresponding Author:

Abhishek Raghuvanshi
Department of Computer Engineering, Mahakal Institute of Technology
Ujjain, India
Email: abhishek14482@gmail.com

1. INTRODUCTION

Users can be located in any part of the world and still be able to make requests to execute applications thanks to cloud computing. The processing power of the computer is utilized by each of the programmes so that they can successfully complete one or more of their assigned duties. The problem of how to divide the available processing power in an equitable manner arises as the number of requests continues to grow. Because of the high volume of requests for resource allocation, the primary focus of the scheduler is on processing these requests and allocating the resources that are required to fulfill them [1].

The process of scheduling determines when, where, how many, and what kind of computing resources should be made available to a particular activity. Also included in the scheduling process is when the activity should take place. Computing resources, such as virtual machines (VMs), are typically deployed by providers onto nodes in their datacenters in accordance with customer preferences regarding the type and quantity of the resource. The constraints, such as processing capacity, projected completion time, and deadline, among others, are satisfied, the task is carried out without a hitch, and the requested type of computing resource is a good match for the available workload characteristics of resources [2]. When working in an elastic environment such as the cloud, where users may request or return resources on a more ad hoc basis, it is equally as important to evaluate whether or not such adjustments should be made. The task scheduler is the component that is accountable for setting priorities and distributing system resources among the tasks that are currently operating [3]. The execution of a task will start as soon as it is provided with the required amount of computational resources. Making a scheduling decision on the cloud for this reason is a more difficult challenge due to the increased complexity [4].

As can be seen from Figure 1, task scheduling is an essential component of any cloud computing architecture that seeks to cater to the requirements of its users in the most effective manner possible. It is essential in the process of assigning resources to new jobs while simultaneously optimising performance [5]. Effective job scheduling is the only method by which it is possible to achieve the essential goals of any cloud computing architecture, including high performance, high profit, high utilisation, scalability, provision efficiency, and economy. These goals are essential for any cloud computing architecture [6].

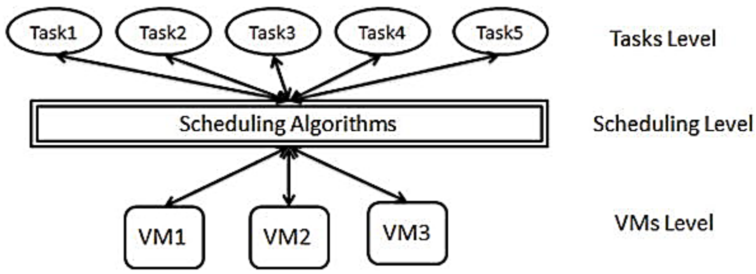

Figure 1. Task scheduling in cloud computing

Thetask scheduler module of the cloud framework starts looking for an appropriate VM to run the job on as soon as the user submits the job to the cloud framework. When scheduling, problems can develop because it can be tempting to place light work on a powerful VM or heavy work on a machine with limited resources. Both of these scenarios are problematic. This could potentially have a detrimental effect on the overall performance of the system as a result of the probable rise in makespan as well as the lengthening of waiting periods [7]. When seen from the perspective of the cloud provider, this results in a reduction in the amount of VM utilisation, earnings, and throughput. As a user of cloud services, this will have a negative influence on your experience because it will result in longer wait times, greater charges, and a failure to meet the quality of service (QoS) expectations you have set for yourself [8]. Therefore, in order to make everyone in the cloud happy, the algorithm that is used to schedule tasks needs to be improved. To reduce the amount of time needed to finish a work as well as the amount of money spent doing so is a primary objective of any good scheduling algorithm. This may be accomplished by allocating tasks to the VM resources that are used in the most effective manner [9].

Because the cloud is a dynamic environment that is always changing, its properties are also subject to change over time. When it comes to developing a system for scheduling work, there are a number of challenges that need to be surmounted. The administration of VMs in the cloud is handled by discrete administrative hosts, and the VMs themselves are hosted in separate physical data centres. At all times, compliance with the rules and regulations that have been established by each of the host organisations is required [10]. The scheduler needs to be able to accommodate these regional parameters because distinct cloud environments each have their own set of rules for the administration of resources and the provisioning of access [11].

There is a wide range of variety when it comes to computing hardware, data storage, memory, networking resources, and so on. When dealing with complex issues, you'll need a wide array of tools, many of which may be stored in several physical locations or be used on a variety of computer systems running different software distributions. The implementation of individual resource management systems results in the development of a variety of capabilities, some of which are significant. If a large application is segmented into a number of smaller jobs, then it will be able to operate many instances of a VM simultaneously. In an autonomous system, it is the role of the scheduler to keep track of everything, from the allocation of tasks and the commencement of their execution, all the way through failure recovery, computation management, and the monitoring of task progress [12].

This article provides chaotic grey wolf optimization (CGWO) based framework for efficient task scheduling in cloud fog computing. User's task or processes are input to task manager. Task manager maintains priorities of task and priorities of VMs on the basis of their response time and throughput. Initially all VMs are available. Task scheduler maintains task queue, the order in which processes will be executed. Task manager assigns VMs to task. VMs details are made available to task scheduler via resource manager. CGWO, ant colony optimization (ACO) and min-max algorithms are used for task scheduling. CloudSim is used for experimental work. MakeSpan, waiting time and resource utilization parameters are used to compare the performance of various task scheduling techniques used in the study. Performance of CGWO is better in terms of makespan, resource utilization and overall waiting time.

2. LITERATURE SURVEY

Ramezani *et al.* [13] presented a method for task scheduling known as task-based system load balancing using PSO (TBSLB-PSO) in order to divide workload among VMs. In the cloud, various tasks are carried out via VMs, also known as VMs. There is a possibility that certain VMs are overworked because more responsibilities have been delegated to them, while other VMs might just be lightly burdened or might possibly be doing nothing at all. It is necessary to divide the work over multiple VMs if you want to get the most out of your available resources, such as your VMs. It is occasionally necessary for VMs to move from their primary host to a secondary host when the primary host becomes overloaded. This is a time-consuming procedure. The utilisation of the TBSLBPSO algorithm, which transfers only the excessive workloads from the overworked VMs to the lightly loaded or idle VMs, results in a reduction in the amount of downtime experienced by overworked VMs. When optimising performance, bandwidth, SLA details, and VM properties are all taken into consideration alongside one another. When compared to other, more conventional ways, it reduces costs, saves time, and eliminates downtime.

Ebadifard and Babamir [14] have developed the PSO-based honeybee behavioral model (PSO-HBM) with the intention of ensuring that work is distributed in an equitable manner across VMs. Honeybees gather nectar and pollen from a diverse range of plants and flowers to use as nourishment. When the bee's current supply of food runs out, it is necessary for it to search for other supplies. Processing the task on a VM that is thus overloaded is comparable to a bee trying to gather food from a source that is completely devoid of it, given that the VM stands in for the food supply. PSO-HBM is the method that is utilised in order to reassign jobs from the overburdened VM to one of the accessible alternatives that has a lower level of workload. It achieves superior load balancing among VMs, quicker maketimes, and higher resource utilisation when contrasted with the PSO algorithm and the round-robin algorithm. It does not address concerns of inequality or discuss solutions to reduce expenses while organising responsibilities.

Panda and Pani [15] advise using a Hungarian method in order to plan paired jobs that are running on the cloud. Within the same cloud environment, the jobs are bundled together and then scheduled in whatever order that the user desires. The amount of time spent doing nothing between tasks is referred to as "layover time," and the overall amount of time spent doing nothing between tasks is equal to the sum of all layover durations for all pairs of tasks. The Hungarian algorithm does not concentrate on how long it takes to complete a task or how much it costs; rather, it considers when the task must begin and when it must stop before making any decisions. The assignment decision takes into account both the transfer time as well as the amount of time that is the shortest in between the lease term and the converse lease time. It is necessary to take into consideration both internal and external services in order to successfully adjust to the constantly shifting requirements of the cloud. When compared to the FCFS approach, the Hungarian algorithm was shown to have a much lower amount of layover time, according to the findings of an exhaustive study that examined the algorithm's performance on numerous datasets.

The FA has been put to use by SundarRajan *et al.* [16] in the process of planning workflow-based activities. Every firefly reveals its strategy, and the amount of time it takes to execute it is a component of the firefly's overall fitness score. Each time through, the VM with the quickest execution time is selected to obtain the tasks that have been queued up. After each cycle, the value of the firefly's fitness is determined, and the winner is chosen in accordance with that value. In addition to this, the optimal distance from the firefly is continuously checked and modified in order to account for any changes. The optimal solutions would be ranked, and the one that came out on top would be selected. The execution time and completion time of jobs planned using the firefly method are much reduced when compared to those of PSO and the cat swarm optimization technique. The question of cost has not been resolved as of yet.

To solve the issue of time-sensitive workflow scheduling, Liu *et al.* [17] propose a genetic algorithm (GA) with coevolution as a solution. These arguments are similar to the previous ones. When two or more populations are able to adapt to one another through time, this is an example of coevolution. It plays a role in the process of fine-tuning the crossover and mutation probabilities in order to speed up the convergence process and avoid premature optimization. The GA that incorporates coevolution emerges victorious when compared to both PSO and its own counterpart, GA. However, it does not take into consideration some details such as the cost and how data is transported.

In a manner analogous to this, Shen *et al.* [18] have adopted GA in order to enhance the performance of cloud computing and lower its carbon footprint. Cloud computing raises important questions about the energy efficiency of data centres. Turning off VMs that aren't being utilised is one approach to cut down on energy use, but this isn't always the most effective solution. Any adjustments that are made to save energy must not have a detrimental effect on the overall functionality of the system. In the GA-based method, the new solution is evaluated based on how well it performs two separate fitness functions. The ability to save energy is taken into consideration by the first fitness function, while performance is taken into consideration by the second. The evaluation shows that GA enhances the VM's energy efficiency as well as its overall performance balance.

In addition, Agarwal and Srivastava [19] scheduled jobs in the cloud with the CS algorithm. The most recent cuckoo egg is presumed to be the solution. It is generally accepted that the cuckoo lays only one egg at a time, and there are a set amount of nesting sites available. It seems likely that whichever host discovers the most efficient cure will be the one to pass it on to future generations of their progeny. In this manner, the CS technique develops the optimal schedule because it reduces the makespan of the schedule to its smallest possible value.

Using the CS method, Tavana *et al.* [20] have successfully completed the identical task with the cloud-based variant of the consolidation problem. In the cloud, consolidation can take place on three distinct levels: at the VM level, at the task level, and at the server level. The CS-based strategy performs better in terms of resolving the consolidation issue and minimising energy usage, task relocation, and penalty cost. This is in comparison to both the GA and the round-robin methods, which both perform poorly in this regard.

In the event that there is an uneven distribution of tasks across the VMs, the amount of time spent waiting for a particular task or delaying its completion is likely to increase. As the delay rises, it will affect the amount of time it takes to complete the task, as well as the amount of time it takes to complete the schedule as a whole. Gabi *et al.* [21] came up with the idea of using an algorithm called the orthogonal Taguchi based cat algorithm (OTCA) to schedule work in the cloud. When carrying out an OTCA, the Taguchi method, in conjunction with the cat's tracing mode, is utilised in order to ascertain which resource requires the shortest amount of time in order to complete the activity at hand. As a consequence of this, OTCA is in a position to address the issue of task imbalance by assigning the appropriate resources to the appropriate positions. OTCA shortens both the amount of time spent waiting and the amount of time spent manufacturing.

The lion optimization algorithm (LOA) is a population-based algorithm that achieves optimal results by taking its cues from the natural laws that govern the universe [22]. It distinguishes between two types of lions: those that remain in one place and those who travel around. The nomads are more likely to move in smaller groups or by themselves, in contrast to the permanent residents who like to gather. Lion prides frequently hunt in groups in order to improve the overall success rate of their assaults. The starting population is formed depending on the location of the lions, which is decided by a random number generator. This location is then used to produce the rest of the population. The migratory lions make up a small percentage of the overall population, while the rest are regarded to be residents. Because the lion is free to move about during each cycle, the ideal location for the lion is dynamically shifting all the time. The LOA shall continue to be carried out in its entirety up until the point where the termination conditions are satisfied.

On Almezeini and Hafez's [23] research, the LOA was utilised so that jobs may be scheduled in the cloud. The "fitness value" for any timetable may be found in the makespan, which is represented by a lion in this illustration. The most productive schedule of the lion has been recorded, which can also be thought of as the sites that the animal visits most frequently. Specifically, each pride would be responsible for doing the activities of hunting, wandering, mating, and defending its territory. It's probably safe to assume that every nomad goes through the same cycles of exploring new territory, finding a mate, ensuring their safety, moving around, and eventually settling down. At the end of each cycle, the lion that has been determined to be the most suitable answer is chosen. For the purpose of determining which strategy is most effective, we put LOA, PSO, and GA through their paces using three metrics: makespan, utilisation, and degree of imbalance. The LOA does not take into account cloud pricing models, despite the fact that it outperforms both the PSO and the GA.

3. METHOD

This section presents a methodology for efficient job scheduling in cloud fog computing via CGWO. The methodology is depicted in Figure 2. The task manager receives the user's tasks or processes as input. The task manager is responsible for managing the priorities of tasks and VMs, taking into consideration their reaction time and throughput. From the outset, all VMs are accessible. The task scheduler is responsible for managing the task queue, which determines the arrangement of processes to be executed. VMs are assigned to tasks by the task manager. The task scheduler can access the details of VMs using the resource management. The task scheduling methods employed are CGWO, ACO, and min-max. CloudSim is utilized for pragmatic research. In this study, the performance of different task scheduling strategies is compared using criteria such as MakeSpan, waiting time, and resource consumption.

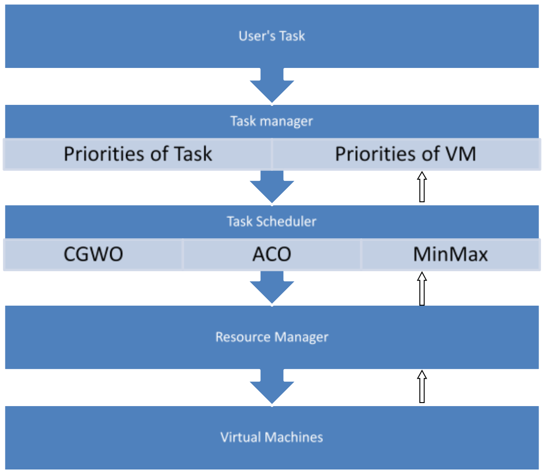

Figure 2. CGWO based framework for efficient task scheduling in cloud fog computing

The GWO has a commendable convergence rate, although it lacks proficiency in identifying the global optimum, resulting in a slower progression of the process. Kohli and Arora [24] devised the CGWO algorithm to mitigate this phenomenon and enhance the efficiency of the GWO algorithm. They achieved this by incorporating chaos theory into the GWO algorithm. This facilitated the attainment of both of these objectives. A state of chaos can be observed in a system that exhibits characteristics of being non-linear, dynamic, non-periodic, non-convergent, and bounded, as it demonstrates behavior that is simultaneously random and deterministic. The mathematical notion of chaos is employed to elucidate the stochastic nature of a dynamic system characterized by a limited number of variables that exhibit perfect independence from each other. Algorithmic optimization strategies incorporate randomness through the utilization of various chaotic maps, each derived from a unique set of mathematical equations. Chaotic maps have become increasingly prominent in the field of optimization in the past decade due to their dynamic character. The prevalence of chaotic maps can be attributed to their ability to facilitate optimization algorithms in conducting a comprehensive and dynamic investigation of the search space. In recent times, a diverse range of chaotic maps has been devised by scientists and mathematicians for the purpose of optimization. These maps have been specifically tailored to cater to distinct human domains. Most of the existing chaotic maps have been utilized in algorithms rather of being directly applied in real-world scenarios. The chaotic maps can be assigned an initial value that falls within the range of the maps, encompassing values ranging from 0 to 1, as well as any other number that falls inside that range. Nevertheless, it is imperative to consider that the beginning value might exert a substantial influence on the overall volatility pattern observed in the chaotic map. It is vital to bear this in mind consistently. A decision was made to employ a compilation of chaotic maps exhibiting diverse behaviors, with each map being assigned an initial value of 0.7. The incorporation of chaos into the feasible zone, which is initially predictable for a limited duration and becomes stochastic over extended epochs, enhances the convergence rate of the GWO approach. This objective is achieved by producing disorder inside the attainable area.

The most well-known example of a bio-inspired algorithm is referred to as an ACO in this context. This method, which is then applied to the scheduling of tasks in order to determine the most optimal timetable, was inspired by the behaviours of ants that seek out the shortest path, which served as the method's source of inspiration. The number of jobs that have to be planned in the cloud frequently exceeds the number of VMs that are available. Because of this, it is the responsibility of the scheduling algorithm to finish all of the tasks by utilising the VMs that are available [25].

The voyage of each individual ant starts with a unique VM and group of activities. A database would be used to record the tasks that were assigned as well as the VMs. When determining which group to select from next, the list is used as a reference point. The amount of free time made available by each VM is kept track of in a matrix with one dimension, which we will refer to as. The availability of the VM for the next job is represented in the matrix of resources that are available. Throughout the course of this inquiry, we will refer to the ith task as Ti, and the ithVM as VMj. We use the minimization function Fk to find the VM in ant k that is the best fit for the problem at hand. Following each iteration of the process, the element of the attainable matrix that is found to be the most significant is inserted into the minimization function Fk. We are able to calculate the problem-specific heuristic information, which is denoted by ij, by utilising the time constraints that the VM provides.

The ACO algorithm starts by processing the list of jobs that have not yet been scheduled and continues doing so until the list is empty. This procedure is repeated until the list is complete. During the initialization phase of the system, information about the available VMs and tasks is obtained. The second thing that it does is determine the ETC, which stands for "Estimated Time of Completion," for each job Ti that is contained in VMj. As a result, it employs a value for the initial pheromone deposit of 0.01, a value for the tuned parameter of 0.05, and a value for the initial pheromone evaporation of 1. This algorithm makes use of a total of four ants. Additionally, the proposed algorithm for work scheduling in the cloud makes the assumption that all of the VMs are available at the beginning of the process; consequently, the availability matrix is set to zero when the procedure begins.

4. IMPLEMENTATION AND RESULT ANALYSIS

Cloudsim is used to simulate experimental work. The ACO, min-max, and CGWO algorithms are run on a 2.4 GHz Intel Core i5 CPU with 8 GB of RAM under the Windows operating system. The results are compared based on characteristics such as makespan, VM utilization, degree of imbalance, and response time. Makespan refers to the overall time required to complete a collection of jobs in the queue. The term "VM utilization" relates to how busy a VM is at any particular time. It might range from 0 to 100%. The degree of imbalance measures the imbalance in load among the VMs. The cloud simulation environment starts with 16 VMs, and the number of jobs varies from 100 to 500. Initially, all VMs are exposed to the task scheduler.

The results are provided in Figures 3 to 5, and Tables 1 to 3. From Table 1 results, it is clear that the makespan time required by CGWO algorithm to collect all jobs in the queue is lesser that make span time time required by ACO and min-max algorithm. CGWO is taking minimum resources to accomplish the tasks in comparison to ACO and min-max methods. Response time of CGWO is also minimum among the methods used in the experimental work. The CGWO method requires a makespan time of 73.27 seconds for 500 tasks. The CGWO approach requires fewer resources to complete tasks compared to the ACO and min-max methods. The CGWO's response time is 3745.2 seconds. Among the approaches employed in the experimental study, CGWO demonstrates superior performance in terms of Makespan time, response time, and resource utilization.

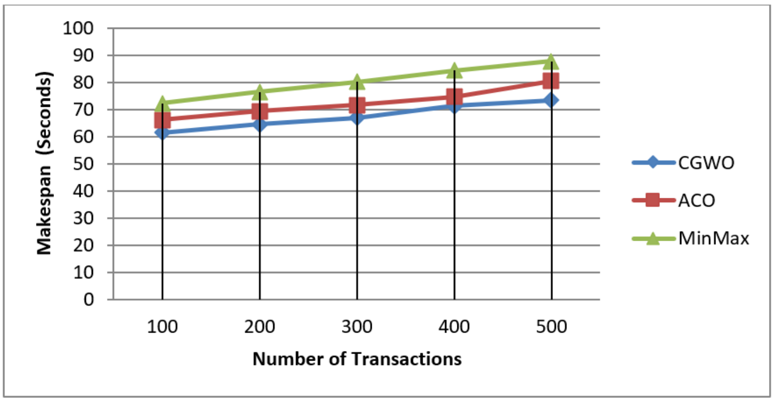

Figure 3. Makespan comparison of CGWO, ACO, and min-max algorithms for cloud task scheduling

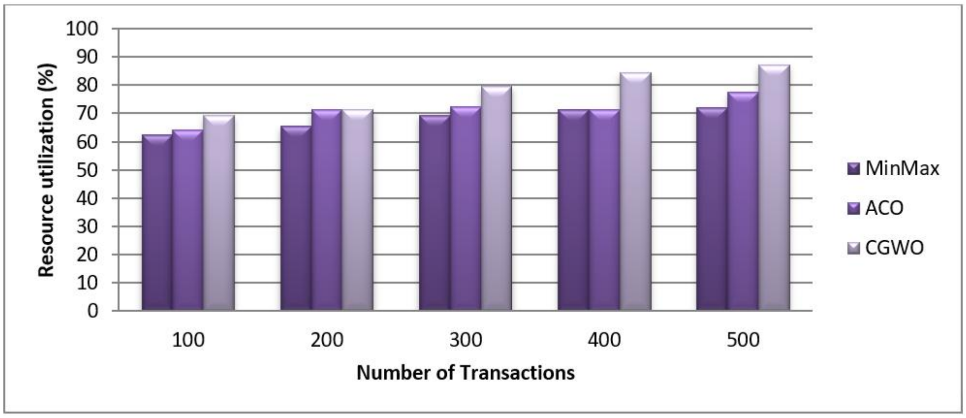

Figure 4. VM utilization comparison of CGWO, ACO, and min-max algorithms for cloud task scheduling

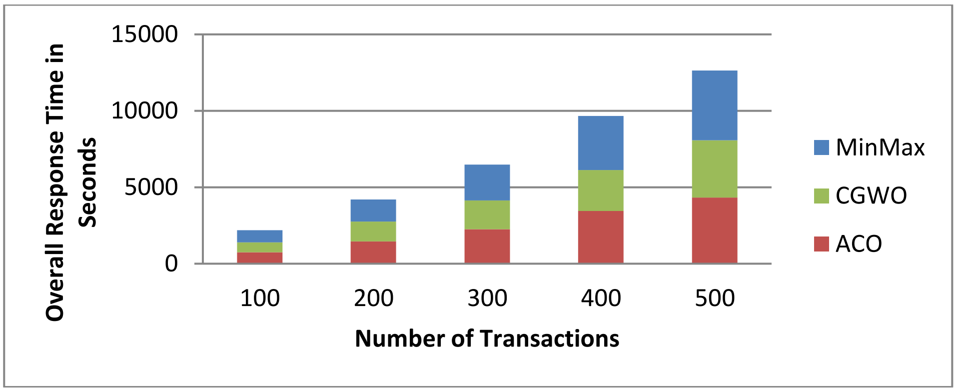

Figure 5. Response time comparison of CGWO, ACO, and min-max algorithms for cloud task scheduling

Table 1. Makespan (in seconds) comparison of CGWO, ACO, and min-max algorithms for cloud task scheduling

Number of tasks	CGWO	ACO	Min-max
100	61.3	66.1	72.27
200	64.5	69.5	76.4
300	66.8	71.6	80.2
400	71.3	74.5	84.3
500	73.27	80.3	87.8

Table 2. Resource utilization (in %) comparison of CGWO, ACO, and min-max algorithms for cloud task scheduling

Number of tasks	Min-max	ACO	CGWO
100	62.3	64.2	69.2
200	65.5	71.2	71.2
300	69.4	72.3	79.5
400	71.2	71.2	84.3
500	72.2	77.4	87.2

Table 3. Response time (in seconds) comparison of CGWO, ACO, and min-max algorithms for cloud task scheduling

Number of tasks	Min-max	ACO	CGWO
100	785.2	753.2	652.3
200	1455.2	1460.2	1289.4
300	2345.7	2247.1	1894.1
400	3554.2	3440.2	2678.4
500	4568.4	4325.1	3745.2

5. CONCLUSION

This article provides a methodology for successfully scheduling jobs in cloud fog computing that is based on CGWO. Users provide task manager with information regarding the processes and activities they need to complete. The order of tasks and VMs is maintained by the task manager, which ranks them according to how quickly they reply and how much they are capable of accomplishing. In the beginning, any and all virtual computers may be utilised. The task queue is the order in which tasks are scheduled to be executed, and the task scheduler keeps track of this order. The jobs are delegated to the VMs by the task manager. The task scheduler receives information regarding the specifics of the VMs from the resource management. The CGWO, ACO, and min-max algorithms are utilised in the process of task scheduling. Utilizing CloudSim allows for the testing of novel concepts. This study uses MakeSpan, waiting time, and resource utilisation as criteria to examine the effectiveness of various approaches for task scheduling. The performance of CGWO is superior in comparison to other variants in terms of MakeSpan, the amount of resources used, and the total amount of time spent waiting.

ACKNOWLEDGMENTS

We are deeply thankful to our institutions for providing the necessary resources and facilities that enabled the successful completion of this research.

FUNDING INFORMATION

No funding received for this research work

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Shreyas J.	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Reena S. Kharat		✓				✓		✓	✓	✓	✓	✓		
Rajesh N. Phursule	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Venkata Bhujanga Rao Madamanchi	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Dhananjay S. Rakshe		✓				✓		✓	✓	✓	✓	✓		
Gaurav Gupta	✓	✓	✓	✓		✓	✓			✓	✓		✓	✓
Malik Jawarneh	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Sammy F.	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Abhishek Raghuvanshi	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

We have obtained informed consent from all individuals included in this study.

ETHICAL APPROVAL

Ethical approval was not required for this study, as it did not involve human or animal subjects.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] M. Armbrust *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010, doi: 10.1145/1721654.1721672.

[2] M. Cusumano, "Cloud computing and SAAS as new computing platforms," *Communications of the ACM*, vol. 53, no. 4, pp. 27–29, Apr. 2010, doi:10.1145/1721654.1721667.

[3] G. B. H. Bindu, K. Ramani, and C. S. Bindu, "Energy aware multi objective genetic algorithm for task scheduling in cloud computing," *International Journal of Internet Protocol Technology*, vol. 11, no. 4, p. 242, 2018, doi:10.1504/ijipt.2018.095408.

[4] X. Chen and D. Long, "Task scheduling of cloud computing using integrated particle swarm algorithm and ant colony algorithm," *Cluster Computing*, vol. 22, suppl. 2, pp. 2761–2769, 2019, 10.1007/s10586-017-1479-y.

[5] W. Jing, C. Zhao, Q. Miao, H. Song, and G. Chen, "QoS-DPSO: QoS-aware task scheduling for cloud computing system," *Journal of Network and Systems Management*, vol. 29, no. 1, pp. 1–29, 2021, doi: 10.1007/s10922-020-09573-6.

[6] X. Chen *et al.*, "A WOA-based optimization approach for task scheduling in cloud computing systems," *IEEE Systems Journal*, vol. 14, no. 3, pp. 3117-3128, Sept. 2020, doi: 10.1109/JSYST.2019.2960088.

[7] P. Zhang and M. Zhou, "Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 772-783, April 2018, doi: 10.1109/TASE.2017.2693688.

[8] A. K. Shukla, A. Shukla, and R. Singh, "Neural networks based face recognition system for biometric security," *Indian Journal of Engineering*, vol. 20, no. 53, pp. 1–9, May 2023, doi: 10.54905/disssi/v20i53/e16ije1640.

[9] H. B. Alla, S. B. Alla, A. Touhafi, and A. Ezzati, "A novel task scheduling approach based on dynamic queues and hybrid meta-heuristic algorithms for cloud computing environment," *Cluster Computing*, vol. 21, no. 4, pp. 1797–1820, May 2018, doi: 10.1007/s10586-018-2811-x.

[10] H. Singh, S. Tyagi, P. Kumar, S. S. Gill, and R. Buyya, "Metaheuristics for scheduling of heterogeneous tasks in cloud computing environments: Analysis, Performance Evaluation, and Future Directions," *Simulation Modelling Practice and Theory*, vol. 111, p. 102353, Sep. 2021, doi: 10.1016/j.simpat.2021.102353.

[11] A. Kanade *et al.*, "Analysis of wireless network security in internet of things and its applications," *Indian Journal of Engineering*, vol. 21, no. 55, pp. 1–12, Apr. 2024, doi: 10.54905/disssi.v21i55.e1ije1675.

[12] T. Bezdan, M. Zivkovic, M. Antonijevic, T. Zivkovic, and N. Bacanin, "Enhanced flower pollination algorithm for task scheduling in cloud computing environment," *Machine Learning for Predictive Analysis Proceedings of ICTIS 2020*, Oct. 2020, pp. 163–171, doi: 10.1007/978-981-15-7106-0_16.

[13] F. Ramezani, J. Lu, and F. K. Hussain, "Task-based system load balancing in cloud computing using particle swarm optimization," *International Journal of Parallel Programming*, vol. 42, no. 5, pp. 739–754, Oct. 2013, doi:10.1007/s10766-013-0275-4.

[14] F. Ebadifard and S. M. Babamir, "A pso-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 12, Dec. 2017, doi:10.1002/cpe.4368.

[15] A. Panda and S. Pani, "A symbiotic organisms search algorithm with adaptive penalty function to solve multi-objective constrained optimization problems," *Applied Soft Computing*, vol. 46, pp. 344–360, Sep. 2016, doi: 10.1016/j.asoc.2016.04.030.

[16] R. SundarRajan, V. Vasudevan, and S. Mithya, "Workflow scheduling in cloud computing environment using Firefly algorithm," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, India, 2016, pp. 955-960, doi: 10.1109/ICEEOT.2016.7754828.

[17] L. Liu, M. Zhang, R. Buyya, and Q. Fan, "Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 5, Sep. 2016, doi: 10.1002/cpe.3942.

[18] Y. Shen, Z. Bao, X. Qin, and J. Shen, "Adaptive task scheduling strategy in Cloud: When energy consumption meets performance guarantee," *World Wide Web*, vol. 20, no. 2, pp. 155–173, Feb. 2016, doi: 10.1007/s11280-016-0382-4.

[19] M. Agarwal and G. M. Srivastava, "A cuckoo search algorithm-based task scheduling in cloud computing," in *Advances in Computer and Computational Sciences Proceedings of ICCCCS 2016*, Sep. 2017, pp. 293–299, doi: 10.1007/978-981-10-3773-3_29.

[20] M. Tavana, S. Shahdi-Pashaki, E. Teymourian, F. J. Santos-Arteaga, and M. Komaki, "A discrete cuckoo optimization algorithm for consolidation in cloud computing," *Computers & Industrial Engineering*, vol. 115, pp. 495–511, Jan. 2018, doi: 10.1016/j.cie.2017.12.001.

[21] D. Gabi, A. S. Ismail, A. Zainal, and Z. Zakaria, "Solving task scheduling problem in cloud computing environment using orthogonal taguchi-cat algorithm," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 3, pp. 1489-1497, Jun. 2017, doi: 10.11591/ijece.v7i3.pp1489-1497.

[22] M. Yazdani and F. Jolai, "Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm," *Journal of Computational Design and Engineering*, vol. 3, no. 1, pp. 24–36, Jun. 2015, doi: 10.1016/j.jcde.2015.06.003.

[23] N. Almezeini and A. Hafez, "Task scheduling in cloud computing using lion optimization algorithm," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 11, 2017, doi: 10.14569/ijacsa.2017.081110.

[24] M. Kohli and S. Arora, "Chaotic grey wolf optimization algorithm for constrained optimization problems," *Journal of Computational Design and Engineering*, vol. 5, no. 4, pp. 458–472, 2017, doi: 10.1016/j.jcde.2017.02.005.

[25] M. S. Khan and R. Santhosh, "Task scheduling in cloud computing using hybrid optimization algorithm," *Soft Computing*, vol. 26, no. 23, pp. 13069–13079, 2021, doi: 10.1007/s00500-021-06488-5.

BIOGRAPHIES OF AUTHORS

Dr. Shreyas J. received the B.E. degree in M.Tech. degree from Visvesvaraya Technological University. He has received full time Ph.D. degree from Bangalore University in 2021. He is currently working as Assistant Professor, Department of Information Technology, Manipal Institute of Technology Bengaluru, Manipal Academy of Higher Education, Manipal, India. His current research lies in the area of sensor networks, artificial intelligence of things, swarm intelligence, and machine learning. He can be contacted at email: shreyas.j@manipal.edu.

Dr. Reena S. Kharat is Associate Professor in Department of Computer Engineering in Pimpri Chinchwad College of Engineering, Savitribai Phule Pune University, Pune, India. She has 20 years teaching experience. Her research interests are machine learning, deep learning, and network security. She can be contacted at email: reenakharat@gmail.com.

Dr. Rajesh N. Phursule is Associate Professor in Department of Information Technology in Pimpri Chinchwad College of Engineering, Savitribai Phule Pune University, Pune, India. He is having 18 years of vast teaching experience. He has published many research articles in the area of machine learning, and cloud computing. He can be contacted at email: rphursule@gmail.com.

Venkata Bhujanga Rao Madamanchi is working as an Assistant Professor in Department of Information Technology in RVR & JC College of Engineering, India. He is having 17 years of teaching experience. He has published many research articles over cutting edge technologies like- cloud computing, internet of things in Scopus and web of science indexed journals. He can be contacted at email: bujji0618@gmail.com.

Dhananjay S. Rakshe is Assistant Professor in Department of Computer Engineering in, Pravara Rural Engineering College Loni, Savitribai Phule Pune University, Pune, India. He is having research interests in cloud computing, machine learning, and deep learning. He is having many publications in journals of repute. He can be contacted at email: jay.rakshe@gmail.com.

Gaurav Gupta is an Assistant Professor from 2013 in the Department of Computer Science and Engineering in G.L. Bajaj Institute of Technology & Management, Greater Noida. He has done his MCA from Ajay Kumar Garg Engineering College, Ghaziabad & M.Tech. from Monad University, Hapur. He has many journal and conference publications in machine learning, deep learning, and internet of things security. He can be contacted at email: gaurav.mintu@gmail.com.

Dr. Malik Jawarneh is an esteemed Associate Professor of Computer Science with extensive experience in academia and research, particularly in artificial intelligence and data science. He holds a Ph.D. from the National University of Malaysia (2016), a Master's in Information Technology from the Northern University of Malaysia (2008), and a Bachelor's in Computer Science from Al-Albayt University, Jordan (2006). He has a rich background in teaching, curriculum development, and research, with notable positions in Oman and Malaysia. He can be contacted at email: M.jawarneh@aau.edu.jo.

Dr. Sammy F. is Assistant Professor in Department of Computer Science and Engineering in Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India. She is astute in writing and publishing research articles in areas like- machine learning and network security. Her Ph.D. research is also in the area of network security with several Scopus and sci publications. She can be contacted at email: sammy@kluniversity.in.

Dr. Abhishek Raghuvanshi is working as Dean Academics and Professor in Department of Computer Science and Engineering in Mahakal Institute of Technology in Ujjain in India. He is having teaching experience of 18 years, research experience of 14 years and administrative experience of 9 years. His research areas include- machine learning, internet of things security, and health care analytics. He is having many publications in SCI and Scopus indexed journals. He has also worked on many governments of India funded research projects. He can be contacted at email: abhishek14482@gmail.com.