A Hybrid Method Based on Gravitational Search and Genetic Algorithms for Task Scheduling in Cloud Computing

Xiuyan ZHANG [1*]
Hengshui College of Vocational Technology, Hengshui, 053000, China

***Abstract*—Cloud computing has emerged as a novel technology that offers convenient and cost-effective access to a scalable pool of computing resources over the internet. Task scheduling plays a crucial role in optimizing the functionality of cloud services. However, inefficient scheduling practices can result in resource wastage or a decline in service quality due to under- or overloaded resources. To address this challenge, this research paper introduces a hybrid approach that combines gravitational search and genetic algorithms to tackle the task scheduling problem in cloud computing environments. The proposed method leverages the strengths of both gravitational search and genetic algorithms to achieve enhanced scheduling performance. By integrating the unique search capabilities of the gravitational search algorithm with the optimization and adaptation capabilities of the genetic algorithm, a more effective and efficient solution is achieved. The experimental results validate the superiority of the proposed method over existing approaches in terms of total cost optimization. The experimental evaluation demonstrates that the hybrid method outperforms previous scheduling methods in achieving optimal resource allocation and minimizing costs. The improved performance is attributed to the combined strengths of the gravitational search and genetic algorithms in effectively exploring and exploiting the solution space. These findings underscore the potential of the proposed hybrid method as a valuable tool for addressing the task scheduling problem in cloud computing, ultimately leading to improved resource utilization and enhanced service quality.**

Keywords—Cloud computing; task scheduling; genetic algorithm; gravitational search algorithm

I. INTRODUCTION

With the advent of cloud computing, most companies are moving their on-premises infrastructure to the cloud to efficiently provide network, storage, and computing services to their customers [1]. Different service models are available in this model, allowing users to access resources on demand [2]. The convergence of Internet of Things (IoT) [3, 4], artificial intelligence [5-7], differential equations [8], machine learning [9-12], smart grids [13], and Blockchain [14] has a profound impact on cloud computing, enabling enhanced connectivity, security, energy efficiency, data analytics, optimization, and intelligent decision-making capabilities, thereby revolutionizing the way resources are managed, services are delivered, and business operations are conducted in the digital era. The cloud paradigm requires a scheduling mechanism to provide on-demand access to cloud resources and provision of the resources needed by users [15]. Typically, in cloud environments, a scheduler is employed to identify suitable solutions for assigning constrained resources among incoming applications or tasks in order to achieve scheduling objectives, such as energy consumption, response time, and reliability [16]. As most scheduling problems are either NP-hard or NP-complete, implementing optimal or suboptimal solutions over a minimum timeframe requires a considerable amount of time [17]. Therefore, there are currently no polynomial time-scheduling algorithms available for optimizing the scheduling of restricted resources in the present computing environments. Taillard [18] provided a simple example of the dilemma we are faced with, indicating that approximately 0.02 percent of the candidate solutions take 1 to 1.01 times the total time required to obtain the most optimal solution. It is clear from this example that identifying the optimal solution to a complex problem is extremely difficult.

With an increase in the number of cloud clients, scheduling becomes quite challenging. The first scheduling algorithms were designed for grid computing, and due to their efficiency, most of them were customized for use in distributed computing [19]. Cloud computing offers users the opportunity to make use of numerous virtual resources, making it impossible to assign tasks manually to each user [20]. Commercialization and virtualization have led to cloud computing being able to handle task scheduling complexity at the virtual machine level [21]. Hence, cloud computing relies on scheduling to allocate resources efficiently and effectively to each task. Currently, a wide variety of scheduling mechanisms are available, including dynamic, static, workflow, and cloud service scheduling [22]. The cloud maintains both internal and external resource demands, with specifications for response time, resource costs, storage, and bandwidth varying by task [23].

Previous studies on cloud computing task scheduling have identified several significant gaps. Firstly, these studies often failed to adequately consider the unique characteristics of cloud computing, such as dynamic resource allocation and multi-tenancy, resulting in an inability to capture the inherent complexity and scalability requirements of cloud environments. Secondly, the lack of standardized benchmarks and evaluation metrics hindered the consistent comparison and assessment of different scheduling approaches, emphasizing the need for a standardized evaluation framework. Thirdly, the adaptability of scheduling algorithms to dynamic environments, including changing workloads and resource availability, was insufficiently addressed, leading to suboptimal resource utilization. Additionally, the diverse requirements of cloud tasks, such as varying resource demands and quality-of-service constraints, were often overlooked, resulting in inefficient allocation strategies and performance degradation. Lastly, the challenges associated with scaling scheduling algorithms to handle large-scale cloud environments were not adequately addressed, resulting in computational inefficiencies and scalability limitations. Addressing these gaps is crucial for developing effective and efficient task scheduling mechanisms in cloud computing.

While promising approaches have been developed to efficient cloud task scheduling, the problem remains NP-complete, implying its inherent complexity. This paper introduces a hybrid method that combines gravitational search and genetic algorithms to schedule multiple tasks within a cloud environment. By combining the strengths and capabilities of gravitational search and genetic algorithms, the paper seeks to overcome the limitations of previous approaches and provide a more effective solution. This hybridization allows for a more comprehensive exploration of the solution space and enhances the ability to find optimal task scheduling solutions. To assess the performance of the proposed algorithm, the paper conducts an experimental setup. The research provides a comparative analysis that showcases the advancements and improvements achieved by comparing the proposed algorithm with previous algorithms. Using several distributions to test the proposed algorithm adds value by providing insights into its performance trends and evaluating its effectiveness across different scenarios. The rest of the paper is structured as follows. Recent works on cloud task scheduling are reviewed in Section II. Section III describes the proposed algorithm. The simulation results are reported in Section IV. Section V concludes the paper.

II. Related Works

Lin, et al. [24] proposed an algorithm for scheduling divisible tasks in cloud computing environments that takes into account bandwidth constraints. A novel non-linear programming solution is presented to the multi-task scheduling problem. In this model, the solution yields an efficient allocation strategy that estimates the appropriate number of tasks to be assigned to each virtual resource node. An optimized allocation scheme is used to develop a heuristic algorithm for scheduling loads, known as the bandwidth-aware task scheduling algorithm (BATS). The proposed algorithm outperforms fair-based task scheduling, bandwidth-only task scheduling, and computation-only task scheduling algorithms. Chen and Guo [25] developed a real-time task scheduling approach based on the PSO algorithm. Optimization objectives encompass the imbalance degree, deadline rate, makespan, and cost. Using a utility function, tasks are assigned to machines with high performance in order to maximize the profit of cloud service providers.

Zhao, et al. [26] propose a method for scheduling tasks that considers energy and deadlines for data-intensive applications. As a first step, tasks are modeled as binary trees using a data correlation clustering approach. It takes into account the correlations generated from initial datasets as well as those generated from intermediate datasets. Thus, the global data transmission volume is substantially reduced, which contributes to a reduction in the rate of SLA violations. Second, using the determination of task requirement degree, a Tree-to-Tree task scheduling method is presented that enhances cloud system energy efficiency by minimizing the number of active machines, reducing data transmission time, and maximizing the utilization of its computing resources.

As a solution to cloud task scheduling problems, Li and Wang [27] proposed a multi-objective optimization algorithm based on the Analytic Network Process (ANP) framework. This algorithm was developed to overcome the weaknesses in mathematical analysis, limitations of optimization capabilities of conventional multi-objective optimization algorithms, and difficulty selecting Pareto optimal solutions in cloud task scheduling. First, they presented a theoretical analysis of cloud task scheduling based on matrix concepts. The improved NSGA-II multi-objective evolutionary algorithm has been applied to cloud task scheduling to search for the Pareto set among multi-objects by utilizing Gene Expression Programming (GEP). Lastly, the ANP model is coupled with the improved NSGA-II in order to address the problem of selecting Pareto solutions. The proposed algorithm can optimize multiple goals simultaneously and can effectively avoid additional iterations caused by changes in user preferences.

Using a bio-inspired intelligent model, Basu, et al. [28] discovered the optimal way to schedule IoT applications in cloud environments with heterogeneous multiprocessors. Evolutionary foraging traits and natural selection of genes have demonstrated that only the fittest species survive in nature. In this case, a fitness schedule is defined as one that complies with task order in a multiprocessor environment. Combining the genetic and ACO algorithms, only the most effective combinations of tasks are selected for each stage. The proposed scheduling algorithm is not preemptive and is based on the assumption that each task can be assigned to a single processor. It has been evaluated with different sizes of task graphs and various numbers of processors and has been demonstrated to be as effective as the traditional GA and ACO algorithms in a heterogeneous multiprocessor environment.

III. Proposed Method

Cloud computing is a computing pattern to meet the computing and storing needs of the final users. The cloud-based data centers need to improve their performance constantly because of increasing service requests. Task scheduling is an essential part of cloud computing for optimizing resource utilization, reducing energy consumption, minimizing response times, and maximizing energy efficiency. The users send their requests to a manager. The manager receives the requests and transmits them to all the VMs. Hence, this force is used as a tool for information exchange. Gravity law has a critical role in finding the most optimal path among the objects (VMs). We use GA's fitting function to select the best and most optimized VMs. Then, using GSA, a percentage of the population's chromosomes (some of VMs) are optimized and are known as the active children. In a repetition loop, some chromosomes are selected, and the GSA is called. We reduce the energy and cost for the active VMs.

A. Gravitational Search Algorithm

In Gravitational Search Algorithm (GSA), optimized finding using gravitational laws and moving in an artificial system is performed in discrete time. The system environment is the problem definition area. Hence, this power can be used as a tool for information exchange. The masses are determined using the objective function. The GSA is formed in two stages: a) proving a discrete-time artificial system in the problem environment, the objects' primary placement, providing the rules, and setting the parameters, b) time passing, objects' movement, and updating the parameters until the stop time [29].

The gravitational laws' role is critical to find the most efficient path among the objects. According to Newton's gravity laws, each object attracts other objects using gravitational power, and there is gravity between every two objects. The objects in the search area are defined by different features in the world, like accelerations (active or inactive), algebraic force, force vector, and distance between them. There are two laws that are regenerated in each object based on the power among the particles that perform based on acceleration.

- Newton's gravity law: each particle is attracted to another particle, and the gravitational interaction between two particles generates their mass and is proportional to the distance square between the particles inversely. The gravity force between two objects with masses of M_1 and M_2 and distance R is proportional to the multiplication of two objects' mass distance and inverse of the square of the distance between them. Newton obtained Eq. (1) for the gravitational force between two objects (F) by calculating the G constant, named the gravity constant [30].

$$F = G\frac{M_1M_2}{R^2} \quad (1)$$

- Newton's motion law: Force has a direct relationship to mass and internal acceleration. The next particle's velocity depends on the primary particle's velocity and the velocity change. A particle's acceleration (a) depends on the force and the mass of the particle [31].

The motion laws of Newton are the basic physics laws. Based on first Newton's law, each object maintains its stability or uniform movement on a direct line unless it is forced to change under one or some forces. Applying force to an object makes it accelerate based on its force and mass, according to the second Newton's law. More force leads to higher acceleration, and higher mass leads to less acceleration. Newton calculates the relation among acceleration, force, and mass using Eq. (2), where acceleration, force, and mass are presented by a, F, and M, respectively.

$$a = \frac{F}{M} \quad (2)$$

Acceleration is velocity changes in time unit, and velocity is passing a specified distance in a determined time duration. Therefore, particles that are heavier and closer to each other apply more gravitational force than lighter and farther particles. Another note is that in physics, there are three types of mass for an object. Active, passive, and inertia gravitational mass are equal for an object. Higher active gravitational mass for an object leads to higher gravitational force around it. The inactive gravitational mass shows the power of interaction in the gravitational field. Inertia mass is an object's resistance measure when changing its location and movement. Less Inertia, the mass of an object, makes more velocity changes. The amount of these three masses in physics is equal to each other.

B. Genetic Algorithm

Genetic Algorithm (GA) is a search technique in computer science to find the optimal solution. A solution for the considered problem is indicated using a list of parameters, namely chromosomes or genomes. The GAs uses the natural selection principle of Darwin to find the optimal formula for the patterns' prediction or comparison. In a nutshell, genetic analysis (GA) is a pattern-based programming approach that employs genetic evaluation to solve issues. The input is the issue, the solutions are programmed using the pattern, and the fitness function assesses each candidate solution that is essentially chosen randomly. The chromosomes are generally shown as a simple string of data, and other types of data structures also can be used. During each generation, each characteristic is evaluated, and the fitting value is measured using the fitting function. The stronger elements or the chromosomes with the fitting value near the optimal population have more chance to live during other periods and regeneration, and the weaker ones are destroyed. In other words, the algorithm saves the inputs near the optimal answer and ignores others. Then the algorithm enters a loop including four steps: selection, reproduction, mutation, and evaluation.

C. Hybrid Algorithm for Task Scheduling

A combination of GA and GSA is used in this section to solve the task scheduling problem in cloud computing. Both algorithms have advantages and disadvantages. For example, in GA, information about the selected person for hanging is destroyed, while GSA has a specific memory. In other words, in GA, the solutions are updated using the general operators, while GSA does not have such operators. In practical applications, the particles may lose variety during the algorithm execution because of parameter setting and executing the algorithm. Hence, they lose their ability to search in the search space. We conclude GSA works well in the primary stages of finding the solution. But it is trapped in the following stages. Thus, defining new operators for GSA is required to increase its efficiency in solving different non-linear criterion functions.

The new algorithm, GA-GSA, is proposed by combining the GA and GSA features. In this algorithm, GSA performance to find the optimal solution improves by adding genetic operators like selection, crossover, and mutation. First, the algorithm's solution is adjusted by the GSA. Then each solution is updated using genetic operators like selection, crossover, and mutation. The selection applies to find the best candidate based on the algorithm's fitting function. Then the solution is updated by applying the crossover and mutation. Elitism, a functional characteristic of GA, is a tool to select the best people for their children's reproduction and replacement. Exploration and exploitation abilities from the GSA algorithm improve by applying genetic operators. In this method, the advantages of GSA and GA by applying genetic operators to the GSA algorithm are embedded. First, the GSA algorithm is utilized to find the problem solution. Then the best solution is corrected during each period using the genetic operators for a balance between exploration and exploitation processes.

GSA and GA are two popular metaheuristic optimization techniques known for their successful application to a wide range of optimization problems. These algorithms are particularly suitable for addressing the cloud task scheduling problem due to the following reasons:

- Global optimization: Both GSA and GA are designed to search for global optima, enabling them to find optimal or near-optimal solutions for cloud task scheduling, which involves considering multiple tasks and resources.
- Population-based approach: GSA and GA work with a population of candidate solutions, allowing them to explore a diverse set of solutions simultaneously. This is beneficial for cloud task scheduling as it involves considering multiple scheduling possibilities and trade-offs.
- Non-deterministic search: GSA and GA employ a non-deterministic search strategy, making them flexible and suitable for the complex and dynamic nature of cloud task scheduling. They can handle uncertain arrival times and resource availability without relying on gradient information or assumptions about the problem's structure.
- Exploration and exploitation: GSA and GA strike a balance between exploration and exploitation, enabling them to explore different scheduling options while exploiting known good solutions. This balance is crucial for improving efficiency in cloud task scheduling.
- Scalability: GSA and GA can handle large-scale systems with numerous tasks and resources, which is essential for cloud task scheduling. They achieve scalability by parallelizing the evaluation and search processes, enabling efficient exploration and optimization in large-scale cloud environments.
- Adaptability: GSA and GA can be easily customized to incorporate problem-specific constraints and objectives. In cloud task scheduling, where various constraints such as task dependencies and resource availability exist, GSA and GA can adapt to handle these constraints and optimize specific objectives, making them flexible for different cloud environments.

1) Gravitational search algorithm: A particle force in the cloud is based on a gravitational constant, and G(t) is in a special time constant. G(t) specifies the particle potential and enhances the movement efficiency. The constant gravity help to exponentially increase the search space and efficiency improvement of the use of the resources. The gravitational constant G starts with a primary value and reduces over time.

2) Calculating GA fitness function: This method searches the problem space to find the best, not optimal answer. GA can be introduced as a general search method that imitates the natural biological evolution laws. This search usually uses to generate useful solutions for solving optimization problems. We use the GA concept to generate a difference among the Human Resources (HR) capabilities. HRs are important components of societies and organizations. Each organization's success depends on its HR. The organizations meet their goals using knowledge, experience, power, and human skills. Since HRs are distributed geographically, generating infrastructure is required to share knowledge, skills, and human experiences. In this method, each chromosome is considered a VM.

If S = {VM1, VM2, …, VMn} is the considered set of chromosomes, the chromosomes can be selected with the highest chance using the rank selection. In the rank-based selection plan, the chromosomes are saved in the population for the first time according to their fitting values based on Eq. (3), where pi is the probability of selecting the ith chromosome, n is the population size, R(n) is the best chromosome, and R(1) is the worst chromosome in the population.

$$pi = \frac{R(i)}{\sum_{i=1}^{n} R(i)} \qquad i = 1,2,\dots,n \tag{3}$$

3) Calculating resources costs: Assume Rj that is considered constant is the unit cost of the jth resource, and the max cost is the maximum cost of a user on the jth resource. Hence the cost of executing the ith task is estimated using Eq. (4). The max cost in this equation is the most expensive task. The total cost of a solution (chromosome) is calculated using Eq. (5), showing the cost of a chromosome in the population.

$$F_{cost}(I) = \frac{T(i,j) \times R_j}{Max\ cost^{1-a}} \tag{4}$$

$$F_{cost}(B) = \sum F_{cost}(j) \quad 1 \geq j \geq M \tag{5}$$

4) Calculating energy consumption: The energy model, including the system-level energy-saving techniques Dynamic Voltage Scale (DVS), acts based on a simple principle. It reduces the power supply voltage and the clock frequency of the CPU to reduce energy consumption. The energy model used in the Complementary Metal Oxide Semiconductor (CMOS) is used in this paper. The processor power is a dynamic estimate using Eq. 6, where A is the number of switches in each clock cycle, C_{ef} is the efficient load capacity, and V and f are the power supply voltage and the operational frequency, respectively. Based on the equation, the stored power supply voltage is a principal and important criterion. Hence, its reduction affects energy consumption.

$$P_{dynamic} = AC_{ef}V^2f \tag{6}$$

5) Updating the masses based on the evaluation function: The best position of the particles is returned to the tasks for scheduling. Then the tasks are assigned to the VMs based on their locations to execute on the data centers. Then, all the cloud GSA scheduling continues until all tasks on the VMs are executed to obtain the minimum time and cost of all the calculations. Best () saves the fitting value, which is the minimum value among the particles. Worst () saves the maximum fitting value among the particles in the search space. The new locations are considered the location of the new masses in the search space. The new masses' weights are normalized using the following equations. The best () and worst () values are calculated for the particles to minimize total cost and mass as follows.

$$m_i(t) = \frac{fit_j(t) - worst(t)}{best(t) - worst(t)} \tag{7}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{8}$$

In the above equations, fit_j shows the fitting of the i^{th} factor's mass at time t, worst () and best () show the amount of the merit of the worst and best factors of the population in time that is calculated using the following equations in the minimum finding problems.

$$best(t) = \frac{min}{j \in (1, \dots, N)} fit_j(t) \tag{9}$$

$$worst(t) = \frac{max}{j \in (1, \dots, N)} fit_j(t) \tag{10}$$

IV. Experiments' Results

CloudSim toolkit provides a convenient platform to model a virtualized cloud environment, which includes the components necessary to build virtual machines, brokers, hosts, and data centers. It is a flexible and adaptable instrument suitable for facilitating the exploration, simulation, and seamless modeling of emerging cloud computing infrastructures. Consequently, we decided to use it as our experiments' simulation toolbox. A comparison of the proposed scheduling algorithm with state-of-the-art algorithms, such as the hybrid PSO-ACO, ACO, and improved PSO algorithms, has been made to evaluate its performance.

Table I provides the experimental parameters for our experiments. To assess these scheduling techniques, we ran our trials with 20, 300, and 500 jobs over 20 virtual computer resources. In these experiments, each CPU can handle 500, 1000, and 1500 MIPS. The task length ranges between 400 and 600 MI. Fig. 1 to 3 illustrate the experimental results. By varying the number of iterations, the capabilities of the four algorithms were evaluated. A comparison is made between the costs associated with different data sizes.

TABLE I. Simulation Parameters

Parameter	Value
Number of tasks	100-500
Population size	100
Max iteration	200
Number of VMs	30
Memory size	512
MIPS	500-1500
Number of hosts	10

Fig. 1 presents a comparison of costs for four different algorithms with varying iterations. Among the metaheuristic algorithms, our algorithm demonstrates the lowest cost, followed by ACO, while IPSO yields the highest cost. Interestingly, the number of iterations does not significantly affect the performance of the PSO-based algorithm. In contrast, IPSO exhibits the highest costs, likely due to the presence of a jitter in the curve as the number of iterations increases. Conversely, both the ACO-PSO and our algorithm showcase smooth transitions between iterations, indicating their stability and efficiency.

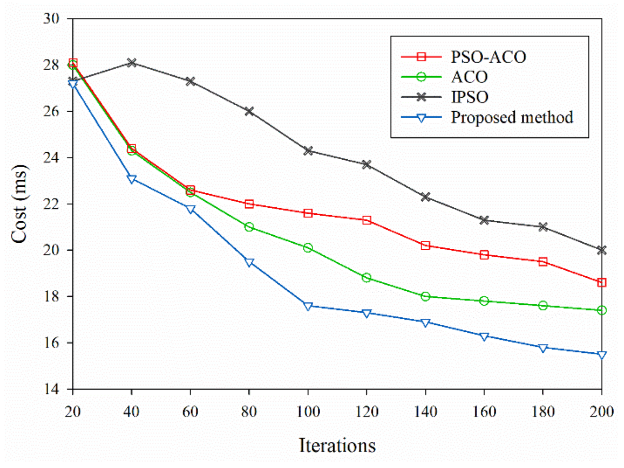

Fig. 1. Comparison for 100 tasks.

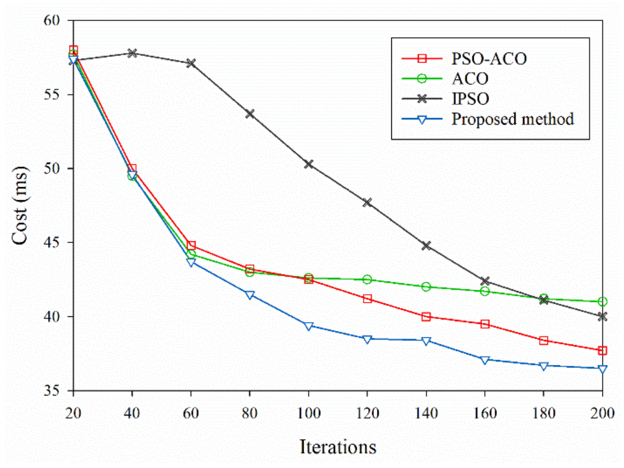

Fig. 2. Comparison for 300 tasks.

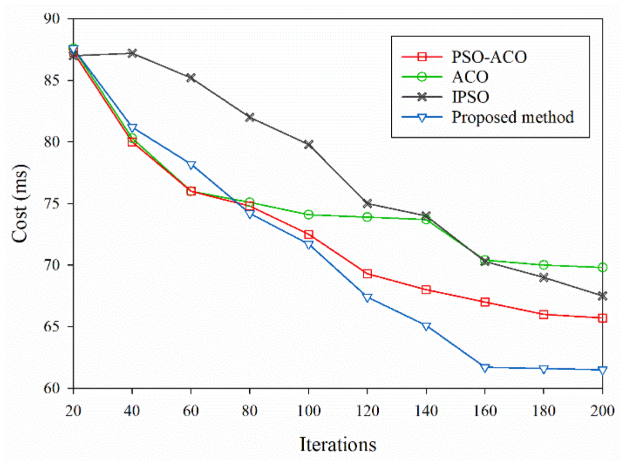

Fig. 3. Comparison for 500 tasks.

Moving on to Fig. 2, it illustrates the results obtained from different algorithms regarding total costs for a scenario involving 300 tasks. Our algorithm proves to be the most cost-effective, yielding the lowest amount of cost, while ACO represents the highest cost. It is worth mentioning that the PSO-ACO algorithm combines the local search and mutation procedures of both ACO and PSO simultaneously. In comparison to the other algorithms, our algorithm showcases a remarkable reduction in energy consumption. Specifically, it achieves a 6% reduction compared to IPSO, a 5% reduction compared to PSO-ACO, and an 8% reduction compared to ACO, all for a scenario involving 500 tasks and 200 iterations.

These findings demonstrate the superior energy efficiency of our algorithm. Furthermore, the observations from Fig. 3 reveal that as the task load increases, the optimization percentage of our algorithm improves significantly. This suggests that our algorithm adapts well to scenarios with higher task demands and exhibits a notable capability to optimize resource allocation efficiently. In summary, the comparisons and results presented highlight the strengths and advantages of our algorithm in terms of cost optimization, stability, energy consumption reduction, and adaptability to higher task loads.

V. Conclusion

This research paper has made several theoretical contributions to cloud computing scheduling. Firstly, it introduced a novel hybrid algorithm that combines genetic and gravitational search algorithms to address the task scheduling challenge in cloud environments. This hybrid approach leverages the strengths of both algorithms, providing a more efficient and effective scheduling mechanism. By integrating genetic operators and gravitational search principles, our method offers improved optimization capabilities and better adaptability to dynamic workload patterns and resource availability. The key results of this research demonstrate that our hybrid algorithm surpasses previous approaches in terms of energy consumption. By achieving better optimization and adaptability, the proposed algorithm provides an advanced solution for cloud task scheduling. These findings have practical implications for cloud service providers and users, enabling more efficient resource utilization, improved energy efficiency, and enhanced quality of service. However, it is important to acknowledge the limitations of this study. The experimental evaluation was conducted on a specific set of benchmarks and scenarios, which may not fully capture the diversity and complexity of real-world cloud environments. Therefore, further validation and testing on a broader range of datasets and workloads would be valuable for comprehensively assessing the algorithm's performance. There are several hints for future research in this area. Firstly, investigating the scalability of the proposed hybrid algorithm to handle large-scale cloud environments would be worthwhile. Exploring the algorithm's performance under different QoS constraints and diverse task profiles could also lead to further improvements. Furthermore, considering the impact of task dependencies and dynamic resource allocation on scheduling effectiveness would contribute to a more comprehensive understanding of cloud task scheduling.

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, 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.

[3] F. Kamalov, B. Pourghebleh, M. Gheisari, Y. Liu, and S. Moussa, "Internet of Medical Things Privacy and Security: Challenges, Solutions, and Future Trends from a New Perspective," Sustainability, vol. 15, no. 4, p. 3317, 2023.

[4] A. Peivandizadeh and B. Molavi, "Compatible authentication and key agreement protocol for low power and lossy network in IoT environment," Available at SSRN 4194715, 2022.

[5] H. Kosarirad, M. Ghasempour Nejati, A. Saffari, M. Khishe, and M. Mohammadi, "Feature Selection and Training Multilayer Perceptron Neural Networks Using Grasshopper Optimization Algorithm for Design Optimal Classifier of Big Data Sonar," Journal of Sensors, vol. 2022, 2022.

[6] 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.

[7] 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.

[8] P. Alipour, "The BEM and DRBEM schemes for the numerical solution of the two-dimensional time-fractional diffusion-wave equations," arXiv preprint arXiv:2305.12117, 2023.

[9] M. Sarbaz, M. Manthouri, and I. Zamani, "Rough neural network and adaptive feedback linearization control based on Lyapunov function," in 2021 7th International Conference on Control, Instrumentation and Automation (ICCIA), 2021: IEEE, pp. 1-5.

[10] R. N. Jacob, "Non-performing Asset Analysis Using Machine Learning," in ICT Systems and Sustainability: Proceedings of ICT4SD 2020, Volume 1, 2021: Springer, pp. 11-18.

[11] M. Sadi et al., "Special Session: On the Reliability of Conventional and Quantum Neural Network Hardware," in 2022 IEEE 40th VLSI Test Symposium (VTS), 2022: IEEE, pp. 1-12.

[12] W.-C. Yeh, Y.-P. Lin, Y.-C. Liang, C.-M. Lai, and C.-L. Huang, "Simplified swarm optimization for hyperparameters of convolutional neural networks," Computers & Industrial Engineering, vol. 177, p. 109076, 2023.

[13] S. Yumusak, S. Layazali, K. Oztoprak, and R. Hassanpour, "Low-diameter topic-based pub/sub overlay network construction with minimum–maximum node degree," PeerJ Computer Science, vol. 7, p. e538, 2021.

[14] S. Meisami, M. Beheshti-Atashgah, and M. R. Aref, "Using Blockchain to Achieve Decentralized Privacy In IoT Healthcare," arXiv preprint arXiv:2109.14812, 2021.

[15] 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.

[16] Y. Xu and K. Abnoosian, "A new metaheuristic-based method for solving the virtual machines migration problem in the green cloud computing," Concurrency and Computation: Practice and Experience, vol. 34, no. 3, p. e6579, 2022.

[17] I. Attiya, M. Abd Elaziz, L. Abualigah, T. N. Nguyen, and A. A. Abd El-Latif, "An improved hybrid swarm intelligence for scheduling iot application tasks in the cloud," IEEE Transactions on Industrial Informatics, 2022.

[18] E. Taillard, "Some efficient heuristic methods for the flow shop sequencing problem," European journal of Operational research, vol. 47, no. 1, pp. 65-74, 1990.

[19] A. Amini Motlagh, A. Movaghar, and A. M. Rahmani, "Task scheduling mechanisms in cloud computing: A systematic review," International Journal of Communication Systems, vol. 33, no. 6, p. e4302, 2020.

[20] J. Praveenchandar and A. Tamilarasi, "Dynamic resource allocation with optimized task scheduling and improved power management in cloud computing," Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 3, pp. 4147-4159, 2021.

[21] 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.

[22] G. Sreenivasulu and I. Paramasivam, "Hybrid optimization algorithm for task scheduling and virtual machine allocation in cloud computing," Evolutionary Intelligence, vol. 14, no. 2, pp. 1015-1022, 2021.

[23] F. Ramezani, M. Naderpour, J. Taheri, J. Romanous, and A. Y. Zomaya, "Task Scheduling in Cloud Environments: A Survey of Population-Based Evolutionary Algorithms," Evolutionary Computation in Scheduling, pp. 213-255, 2020.

[24] W. Lin, C. Liang, J. Z. Wang, and R. Buyya, "Bandwidth-aware divisible task scheduling for cloud computing," Software: Practice and Experience, vol. 44, no. 2, pp. 163-174, 2014.

[25] H. Chen and W. Guo, "Real-time task scheduling algorithm for cloud computing based on particle swarm optimization," in Second International Conference on Cloud Computing and Big Data in Asia, 2015: Springer, pp. 141-152.

[26] Q. Zhao, C. Xiong, C. Yu, C. Zhang, and X. Zhao, "A new energy-aware task scheduling method for data-intensive applications in the cloud," Journal of Network and Computer Applications, vol. 59, pp. 14-27, 2016.

[27] K. Li and J. Wang, "Multi-objective Optimization for Cloud Task Scheduling Based on the ANP Model," Chinese Journal of Electronics, vol. 26, no. 5, pp. 889-898, 2017.

[28] S. Basu et al., "An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment," Future Generation Computer Systems, vol. 88, pp. 254-261, 2018.

[29] Y. Wang, S. Gao, Y. Yu, Z. Cai, and Z. Wang, "A gravitational search algorithm with hierarchy and distributed framework," Knowledge-Based Systems, vol. 218, p. 106877, 2021.

[30] S. A. Rather and P. S. Bala, "A holistic review on gravitational search algorithm and its hybridization with other optimization algorithms," in 2019 IEEE International conference on electrical, computer and communication technologies (ICECCT), 2019: IEEE, pp. 1-6.

[31] A. Fathurohman, E. Susiloningsih, and A. Arianti, "Physics module based on STEM problem based learning on newton’s motion law material for senior high school," in Journal of Physics: Conference Series, 2021, vol. 1869, no. 1: IOP Publishing, p. 012155.