ISSN 2047-3338

A Comparative Study of GA and PSO Algorithm in Cloud Computing and IoT Technology

Sharjeel Tariq[1], Muhammad Shahzad Ashraf Rana[2], M. Junaid Arshad[3]
[1,2,3]Department of Computer Science, University of Engineering and Technology, Lahore
2022mscs225@student.uet.edu.pk, 2021mscs582@student.uet.edu.pk, mjunaiduet@gmail.com

***Abstract*– Web administration organizations are excellent at organizing creative applications for various Internet-based business arrangements. This paper uses two metaheuristic calculations, specifically Genetic Algorithm (GA) and Particle Swarm Optimization Algorithm (PSO), to handle QoS-based assistance organization issues. Quality of service has transformed into a fundamental issue in the administration of web administrations, given the significant number of administrations that outfit comparative usefulness yet with different qualities. This paper compares two popular optimization algorithms: Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). To assess the performance of the two algorithms, they are tested on two well-known benchmark functions and two engineering optimization problems, namely, the welding sequence problem and the design optimization of a truss structure. The results indicate that both algorithms can find the optimal global solutions for the benchmark functions, but PSO shows a better accuracy and faster convergence speed. However, when applied to engineering optimization problems, the superiority of PSO is not as evident. GA demonstrates better performance in finding the optimal solution with fewer evaluations. The authors conclude that selecting the appropriate algorithm for solving an optimization problem depends on the specific characteristics of the problem, such as the number of variables, the complexity of the objective function, and the constraints.**

***Index Terms*– Genetic Algorithm, Particle Swarm Optimization Algorithm, Workflow, Saas, Iaas, Paas, Quality of Service, IoT, Service Composition and Workflow Applications**

I. INTRODUCTION

THE need for processing and large storage capacity is rapidly increasing. As a result of high computing services and facilities given to customers (SaaS), (IaaS), and (PaaS), cloud computing attracts attention (PaaS). Multiple applications may be described as workflow applications and a bunch of tasks having multiple dependencies among them, in this way that dependent tasks must finish their execution first before one task can run [1]. Workflow applications are employed in various fields, including astronomy, bioinformatics, and catastrophe modeling and prediction. Furthermore, complex challenges, such as sophisticated scientific applications, have lately emerged due to combining various methodologies and techniques into a single solution. Fortunately, with the rise of cloud computing, similar workflow applications may now be run remotely [3].

Depending on some technical rules, from one job to the next, to attain a general purpose, the activities in the workflow application are interdependent, with one task's output becoming the input of another. As a result, when allocating tasks to Virtual Machine processors, particularly in a multiprocessor environment, the order in which they are executed must be considered [6]. The workflow application scheduling processes are a multi-objective optimization challenge. Consumers may seek to reduce the throughout cost of the application—the time it takes to execute it by utilizing effective load balancing among Virtual Machines in a cloud environment. The trade-off between the three objectives is the best decision for multi-objective workflow optimization. As a result, consumers must value the objectives to identify the best Pareto solutions. The total cost may lead to maximum execution time and load on a single virtual machine. [7, 8]. The problem of a workflow scheduling is a classic challenge, especially from heterogeneous computing settings, for which several research initiatives to address the scheduling problem have been performed [9]–[11]. However, computing environments that are heterogeneous are challenging when configuring, and their potential to deliver more consistent performance with fewer failures is restricted compared to cloud computing environments [12], [13]. Furthermore, the primary goal is to get the algorithm with the highest efficiency and less completion time.

This paper aims to introduce a comparative analysis between GA and PSO Algorithm, resulting in the best solution overall. Both algorithms should run a benchmark test to calculate the execution time and load balancing over the VMs with a minimum overall cost.

II. RELATED WORK

The IoT is coursed, flexible, significantly novel, conflicting w ith new things, and individual things going this way and that from the IoT. Organizational synthesis is one of the best significant issues in IoT. Li et al. [21] have kept an eye on the assisting course of action with giving using Multicriteria considering the quality of service limits. This computation contemplates a piece of the quality of service limits to find the ideal response for assistance with organizing issues.

The challenge we face while planning the energy-productive calculation is dealing Cloud-based applications enable the coordination of diverse devices with differing functionalities, ensuring seamless interoperability among them. Fortino et al. [25] proposed a specialist-based undeniable level plan for IoT-based frameworks. They stressed using mists to add high computational capacity to IoT frameworks. It also supported decentralized and dynamic systems. Chen et al. [26] made sense of a trust-the-board convention that upholds versatility and flexibility. It supports IoT frameworks based on service-oriented architecture (SOA) and has implemented a filtering approach to monitor responses from IoT hub devices that share common interests. The social bond is structured according to social similarity, with companionship, social contact, and shared community interests in order of priority. Additionally, a mobile channel is utilized to showcase entertainment options, and the flexible IoT trust is demonstrated to be superior to both EigenTrust and PeerTrust

Multiple suppliers' administrations on the cloud can be consolidated according to the requirements of the client bringing about SC issues. Kurdi et al. [22] have resolved the problem of managing and developing applications in multi-cloud environments. He has proposed the Multi Cuckoo calculation utilizing an exploratory system. Comparing Particle Swarm Optimization and Genetic Algorithm, PSO has performed well in several cases and is efficient and less time-consuming. The utilization of the cloud is to offer support to the client independent of its geological area.

III. ALGORITHMS & THEIR WORKING PRINCIPLES

Genetic Algorithm

As the era of cloud computing emerges, the optimization issue is rising. Optimization can only be improved by using the assets according to the need and in the best efficient method. This refers to maximizing or limiting qualities based on a function that, in a particular situation, is also referred to as an objective function from the range of available attributes. This refers to maximizing or limiting rates based on a process that, in a particular situation, is also referred to as an objective function from the range of available qualities. There are precisely two types of traits: positive and negative. Positive should be emphasized, while negative should be kept to a minimum. The Genetic Algorithm is one of the genetics-based artificial intelligence algorithms. The survival of the fittest is the fundamental tenet of this algorithm. Use it if the search area is significant because it is desirable. Over traditional optimization techniques, a genetic algorithm offers several advantages [23]. The fundamental idea of GA is based on genetics and revolves around the chromosomes and genetic architecture found in various cultures. GA encourages the survival of the fittest individuals in succeeding generations to tackle the issue. A collection of strings that resemble chromosomes can be found at every age.

Every iteration represents a prospective solution that can be implemented based on the value found in the search space. Subsequent to that, the individuals in the population undergo evolutionary processes. The genes that exhibit superiority over the previous generations' mean are utilized to generate solutions for the next generation. The fitness values are evaluated to identify the optimal solution when the population converges to a state where no further offspring are produced beyond what the previous generation had. In the Genetic Algorithm (GA), every member of the population represents a feasible response to the problem being addressed in the solution. Each individual is distinct, described as a finite length variable within a selected range in a binary format. In the context of the genetic metaphor, these individuals are viewed as chromosomes, and the variables correspond to genes. Thus, a solution, or chromosome, in the GA consists of multiple genes. The fitness values are determined using the objective's exploitation and allocated to each response, representing a unique solution [34]. After that, the ideal fitness value is found; GA aims to utilize the specific "propagation" of arrangements with the goal that the upcoming children are higher than the parent.

Steps for Applying Genetic Algorithm

Step 1: Randomly initialize the population and name it “P”

Step 2: Calculate the fitness score for P by utilizing an appropriate fitness function..

Step 3: Persist until achieving the best possible fitness value.

Step 4: Choose two parents and name them “P1” and “P2”..

Step 5: By conducting parental crossover, a new population (P + 1) is generated.

Step 6: Population (P + 1) is processed by the mutation operator.

Step 7: Evaluate the fitness of the new population (P + 1), compare it with the current best fitness value, and if the obtained fitness value is superior, update the optimal solution accordingly.

Particle Swarm Optimization Algorithm

Kennedy and Eberhart created the particle swarm optimization **(**PSO**)** algorithm in 1995, A cognitive bionic system inspired by the study of bird predation behavior. Finding the best answer through group collaboration and knowledge exchange is the fundamental tenet of PSO. Specifically, the position and speed of each bird are considered as independent variables, while the value of the objective function is determined by the food density at the corresponding location where the bird arrives. Depending on the discrepancy between the best site for its previous search and the population history search, each search will modify the direction and pace of its inquiryUltimately, the complete swarm of birds can gather around the optimal position in the search space, resulting in either the identification of the ideal solution or the convergence of the problem.

Some Advantages of the PSO Algorithm are listed down below:

1. PSO has a tremendous global search capability and a very rapid computing speed compared to the conventional technique [24].
2. The population size does not impact training speed

because PSO is not sensitive.

3. There are no constraints on the continuity, convexity, connectivity, or derivability of viable regions for the objective function, nor is it necessary to determine the function's gradient information when optimizing it.

IV. LITERATURE REVIEW

The Genetic Algorithm (GA) is a nature-inspired search and optimization technique widely applied in various fields, including cloud computing. Within the context of cloud computing, Genetic Algorithms (GAs) have been employed to enhance resource allocation, scheduling, load balancing, and energy consumption. This is done to improve the efficiency of resource utilization and enable smooth execution of applications. An example of GA's application in cloud computing is virtual machine placement, where the allocation of virtual machines to physical servers is optimized based on resource usage, workload, and energy consumption. This optimization can help reduce costs and improve overall infrastructure performance. GA can also be used for load balancing to ensure that the load on different servers is balanced, enhancing cloud applications' response time and reliability.

In this context, the GA algorithm's performance is primarily influenced by the solution encoding method, the fitness function employed, and the size of the population corresponding in number of iterations. Alajmi and Wright have shown that these parameter values significantly affect the algorithm's efficacy. In order to reduce complexity, the suggested approach employs the GA algorithm for the first half of the specified number of iterations. The optimal performance of the GA-PSO algorithm is attained when the iterations are equally divided between the GA and PSO algorithms.

At each iteration, the GA operators, namely selection, crossover, and mutation, progressively enhance the chromosomes. These chromosomes, referred to as particles, are then fed into the PSO algorithm for the second half of the specified iterations and gradually refined in each iteration. The particle with the minimum fitness value represents the optimal solution for the workflow task problem. By running a few test trials, the algorithm's performance can be fine-tuned by adjusting the parameter values. Leveraging the GA-PSO algorithm in cloud computing has the potential to optimize resource utilization, reduce costs, and enhance the performance and dependability of cloud-based applications.

A comparative study of multiple algorithms from multiple authors has been made that defines the algorithm's objective, advantages, and limitations. The table I draws fit results:

Table 1: Shows research on PSO and GA Algorithm

Authors	Algorithms	Objective	Advantages	Limitations
Braun et al. [15]	Min-Min Algorithm	Time	12%-14% more efficient than GA	Slow in large scale tasks
Kumar and Verma [19]	The fusion of the min-min and max-min approaches within the context of the Genetic Algorithm	Time	speedy as compared to GA	Time-consuming
Guo et al. [20]	Particle Swarm Optimization (PSO) algorithm	Transfer time and execution	More efficient than the M-PSO and L-PSO algorithms when dealing with large-scale problems	Stuck in Local Optimal Solution
Gen M. [23]	Heuristic Algorithm based on particle swarm optimization	Time and cost	It offers a cost reduction that is three times better than that of BRS and achieves a balanced distribution of the workload across available resources	Stuck in the optimal local solution
R.Xu [24]	Heterogeneous Budget-Constrained Scheduling (HBCS) algorithm	Execution time and cost	Achieving a 30% decrease in execution time without exceeding the allocated budget	Not considering the load on resources
A.Verma [19]	Bicriteria Priority-Based Particle Swarm Optimization Algorithm	Execution time and cost	Reducing the execution expenses in contrast to BHEFT and PSO.	Not considering the load on resources
R.Xu. [24]	Heuristic Algorithm based on the min-min Algorithm	The fault recovery, the time, and the cost	The recovery from faults can greatly affect both performance criteria	Better choice only when both cost and makespan are considered

S.Chitra [25]	The PSO algorithm	Calculate Load balance	Better than PSO and GA	Time-consuming
Y.Mao [18]	The Genetic Algorithm	Calculate Load balance	Superior to FIFO	Time-Consuming to reach the optimal solution
Fard et al. [28]	The heuristic algorithm	Calculate Energy consumption and reliability	More efficient	The algorithm's efficiency may not be optimal when dealing with a small number of tasks and processes.
Ge and Wei. [26]	The algorithm known as Revised Discrete Particle Swarm Optimization (RDPSO).	Determine the costs of communication and computation.	The algorithm performs better than the PSO and BRS (Best Resource Selection) Algorithm.	Inefficient in dealing with a large search space.

The presentation of these algorithms relies upon wellness values. The environment in which multiple systems are attached plays a significant role in looking at the exhibition of the two algorithms.

Particle swarm optimization (PSO) is a population-based optimization technique applied in various fields, including the Internet of Things (IoT). In IoT, PSO can be used to optimize resource allocation, energy consumption, and network performance. These tasks are essential in IoT, as they help ensure that devices efficiently utilize resources, conserve energy, and communicate effectively. One example of using PSO in IoT is sensor placement optimization. By using PSO, the placement of sensors in an IoT network can be optimized to achieve better coverage and reduce redundancy. This optimization can help conserve energy and improve the accuracy of data collection. PSO can also be used in routing optimization in IoT networks. Using PSO, the optimal route for data transmission can be determined based on factors such as energy consumption, signal strength, and congestion. This optimization can improve network performance and reduce energy consumption. Using PSO in IoT can help optimize resource utilization, conserve energy, and improve network performance.

When GA creates poor solutions, the PSO Algorithm stores the best and worst solutions in memory, which might help with quick solution convergence. The GA-based scheduling algorithm represents the problem's scheduling solution through multiple chromosomes, each of which has a length equal to the total number of workflow jobs. The chromosomes consist of various genes that represent the virtual machines of the hosts. In each iteration, the GA applies three operators, namely selection, crossover, and mutation, to the chromosomes

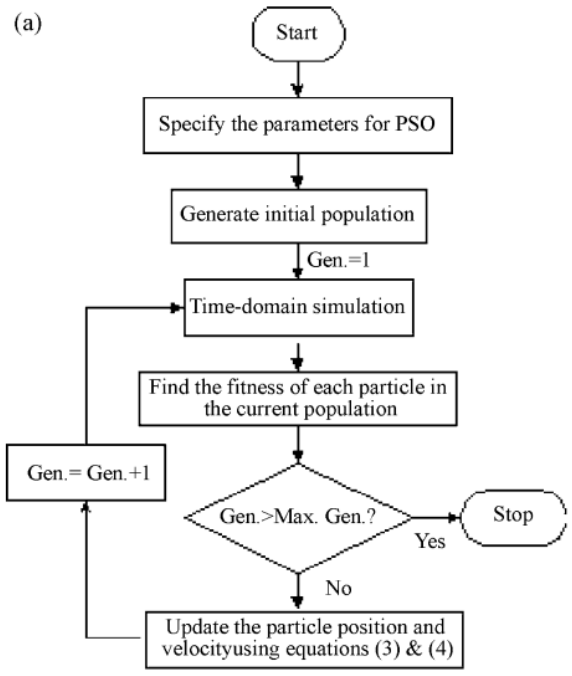
In this context, the Mutation Operator plays a crucial role. Its main objective is to introduce unpredictable changes to the new chromosomes generated by the previous crossover operator, resulting in chromosomes with a higher fitness value than the current ones. The Mutation Operator acts on the chromosome selected by the selection method, and its occurrence is determined by the mutation rate variable. At the beginning of the mutation process, a random number is generated, and if it is less than or equal to the mutation rate, the mutation operator is applied to the chromosome. Its ultimate goal is to create novel solutions that may improve the overall fitness of the population.

In-Depth Comparison of GA & PSO Algorithm

GA and PSO are population-based optimization algorithms that solve complex problems. Here is a comparison table of the GA and PSO algorithms (Table II):

Table II: Shows multiple parameters of the PSO and GA Algorithm

Parameter	GA	PSO
Population size	Large	Small to medium
Encoding scheme	Binary or real-valued	Real-valued
Selection process	Fitness-proportionate	Best individuals
Mutation operator	Probability-based	N/A
Crossover operator	Binary or real-valued	Real-valued
Particle updating	N/A	Based on personal and global best
Search intensity	Low to High	Medium to high
Convergence speed	Slow	Fast
Exploration capability	Low	High
Complexity	High	Low to medium


(a)
Start
Specify the parameters for PSO
Generate initial population
Gen.=1
Time-domain simulation
Find the fitness of each particle in the current population
Gen.= Gen.+1
Gen.>Max. Gen.?
Yes
Stop
No
Update the particle position and velocityusing equations (3) & (4)

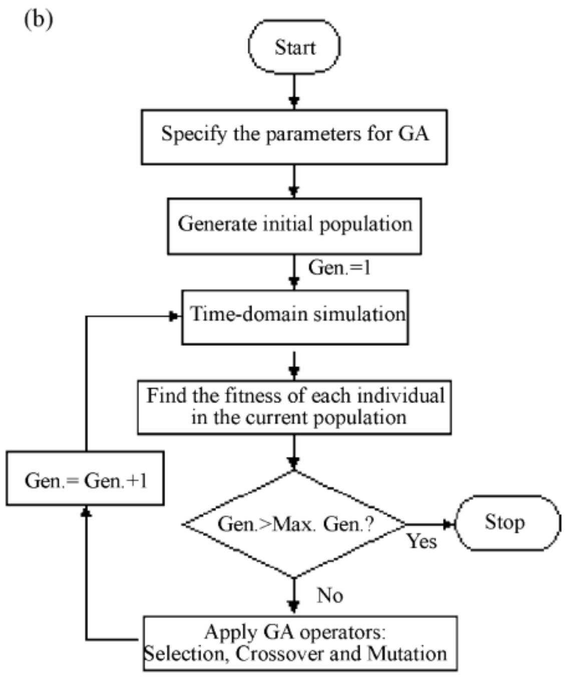

(b)
Start
Specify the parameters for GA
Generate initial population
Gen.=1
Time-domain simulation
Find the fitness of each individual in the current population
Gen.= Gen.+1
Gen.>Max. Gen.?
Yes
Stop
No
Apply GA operators: Selection, Crossover and Mutation

Fig. 1: Shows a flow chart diagram of the PSO and GA Algorithm [27]

Performance Evaluation of Genetic Algorithm

Genetic Algorithms (GAs) are popular optimization techniques that imitate the mechanism of natural selection to discover the best solution to a problem.. Performance evaluation of GA involves assessing its effectiveness in finding a near-optimal solution to a problem and its efficiency in terms of time and resources required.

Here are some commonly used methods for the performance evaluation of GA:

1. *Fitness Function Analysis:* The fitness function evaluates the degree to which a solution meets the requirements of the problem. Analyzing the fitness function helps in understanding the quality of solutions found by the GA
2. *Convergence Analysis:* Analyzing the convergence of a GA involves evaluating how close the population is to reaching the optimal solution as the algorithm progresses. This analysis helps determine the number of iterations required to achieve a solution that satisfies the problem requirements.
3. *Parameter Tuning:* The performance of GA is affected by the selection of various parameters, such as population size, mutation rate, and crossover rate. Tuning these parameters involves experimenting with different combinations to identify the most effective ones.
4. *Comparison with other algorithms:* Assessing the performance of the GA can involve comparing its results with those of other optimization algorithms that solve the same problem. This allows for a fair comparison and helps to identify the strengths and weaknesses of GA.
5. *Scalability Analysis:* GA performance can be evaluated by analyzing how it performs as the problem size increases. This helps to determine whether GA is scalable to more significant problems and whether it can maintain a reasonable level of performance as the problem becomes more complex.

Overall, the performance evaluation of GA involves analyzing various aspects of the algorithm, including its ability to find reasonable solutions, convergence speed, parameter sensitivity, scalability, and comparison with other algorithms.

Performance Evaluation of Particle Swarm Optimization (PSO) Algorithm

The performance efficiency of the Particle Swarm Optimization (PSO) Algorithm can be evaluated based on the following factors:

1. *Convergence speed:* The PSO algorithm's convergence speed is generally faster than other optimization algorithms. The algorithm quickly converges to an optimal solution using the swarm intelligence technique.
2. The PSO algorithm's ability to obtain high-quality solutions relies on the fitness function used to assess those solutions. The PSO algorithm can provide high-quality solutions if the fitness function is well-defined and appropriate.
3. *Robustness:* PSO algorithm is robust and can easily handle noisy and non-linear optimization problems. This makes it suitable for real-world applications with complex and dynamic optimization problems.

4. *Scalability:* The PSO algorithm can handle large and complex optimization problems by scaling up its swarm size and the number of iterations, thereby increasing its computational complexity..
5. *Ease of implementation:* PSO algorithm is relatively easy to implement and does not require a lot of computational resources. This makes it efficient for solving optimization problems in various domains.

Fitness Tests between GA and PSO Algorithm

PSO Parameters	Genetic Parameters
Swarm size: 30	Population size: 30
Maximum number of generations: 200	Maximum number of generations: 200
D1, D2=3.0, 3.0	Type of selection: normal geometric [0.09] Type of crossover: arithmetic [2]
Wstart, Wend=0.9, 0.4	Type of mutation: non-uniform [2, 100, 3]

Values	PSO	GA
Best	500.7721	496.321
Average	512.1185	507.112
Worst	503.12	515.71

Table III: [27] Shows fitness Tests of the PSO and GA Algorithm

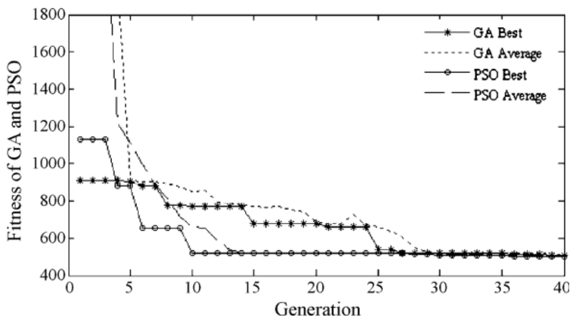


Fig. 2: [27] Shows a flow chart diagram of the GA and PSO Algorithm

Table IV: [27] Shows Workflows, Average Results, and Executed Experiments of the PSO and GA Algorithm

The traits that define the workflows in Cloud Computing			
Scenes	Tasks	Edges	Average Data Size (MB)
Scene One	26	96	3.44
Scene Two	51	207	3.37
Scene Three	101	434	3.25
Scene Four	1001	4486	3.23
The algorithms were compared based on their average results in makes pan, execution cost, and load balancing			
Methods	Avg Makes	Avg Execution Time	Avg Load Balance
GA	799.131	451.331	74.591
PSO	578.0671	353.741	42.8821
Results of Executed Experiments			
Algorithm	Make span	Execution Cost ($)	Load Balance (rate)
Scenario One			
GA	197.651	52.682	52.584
PSO	101-211	18.163	21.334
Scenario Two			
GA	250.89	86.34	61.93
PSO	155.31	62.86	18.23

Tabular Evaluation of Genetic Algorithm (GA) & Particle Swarm Optimization (PSO) Algorithm

Table V: Shows a Tabular Evaluation of the PSO and GA Algorithm

Evaluation Measures	Genetic Algorithm	Particle Swarm Optimization
Representation	Binary string	Vector
Search Strategy	Evolutionary	Swarm-based
Convergence	High probability of global optimum but slower	Faster but may get stuck in local optima
Complexity	Computationally expensive due to fitness evaluation	Relatively simple and computationally efficient
Parameter Tuning	Many parameters, such as population size, mutation rate, and crossover rate	Fewer parameters, such as swarm size and inertia weight
Convergence speed	Slow but a high probability of finding global optima	Fast but may get stuck in local optima
Exploration ability	Strong due to diversity maintenance and mutation operation	Moderate due to the stochastic search
Exploitation ability	Strong due to the selection and crossover operations	Moderate due to relying on local and global best positions
Scalability	High due to population-based approach	Limited due to swarm size limitation
Complexity	High due to the need for fitness function evaluation	Low due to the simple update rule
Parameter tuning	The problem is complex due to the multitude of parameters involved, such as population size, crossover rate, and mutation rate.	Simple with only a few parameters, such as swarm size and inertia weight
Robustness	Strong due to the diversity maintenance	Moderate due to the stochastic search
Applicability	Wide range of optimization problems	Suitable for continuous optimization problems
Performance	Comparable to other optimization algorithms	Competitive with other optimization algorithms
Convergence speed	Slow but a high probability of finding global optima	Fast but may get stuck in local optima
Exploration ability	Strong due to diversity maintenance and mutation operation	Moderate due to the stochastic search
Exploitation ability	Strong due to the selection and crossover operations	Moderate due to relying on local and global best positions
Scalability	High due to population-based approach	Limited due to swarm size limitation

V. CONCLUSION

GA and PSO are powerful optimization algorithms using different techniques to find the ideal scenario. GA uses the principles of natural selection and genetics to evolve a population of candidate solutions, while PSO uses a swarm of particles to search the solution space. In general, GA is more suitable for problems that involve discrete variables and combinatorial optimization, while PSO is more suitable for problems that involve continuous variables and multi-modal optimization.

The results suggest that the PSO and GA algorithms are suitable for optimizing parameters. However, from an evolutionary standpoint, PSO outperforms GA in terms of performance. PSO exhibits faster convergence, requiring fewer generations to reach the final parameter values than GA. Furthermore, while PSO and GA show linear increases As the number of generations increases, the GA has a lower cumulative computational time than PSO. This is because of the communication that occurs between particles in PSO after each generation. It is worth noting that the success of these optimization techniques heavily depends on the selection of control parameters and objective functions, and choosing them appropriately is crucial.

The hybridization of GA and PSO algorithms has resulted in a robust optimization technique that incorporates the strengths of both algorithms. GA utilizes natural selection and genetic operators to explore the solution space, while PSO imitates the behavior of bird flocks or fish schools to locate the global optimum. By combining these two algorithms, a hybrid algorithm is created that is more effective in handling complex optimization problems. In this approach, GA performs a population-based search, and PSO carries out a local search.

This hybridization strategy increases the algorithm's exploration and exploitation capabilities by utilizing GA's global and PSO's local search ability. Furthermore, the hybrid algorithm can handle continuous and discrete optimization problems and is insensitive to the initial population, making it applicable to various real-world optimization problems. The PSO and hybrid GA algorithm is a widely accepted optimization technique that comprehensively solves complex optimization problems.

REFERENCES

[1] A. H. Aljammal, A. M. Manasrah, A. E. Abdallah, and N. M. Tahat, "A new architecture of cloud computing to enhance the load balancingg," *International Journal of Business Information Systems*, vol. 25, no. 3, pp. 393–405, 2007.

[2] J. Li, Z. Liu, X. Chen, F. Xhafa, X. Tan, and D. S. Wong, "L-EncDB: A lightweight framework for privacy-preserving data queries in cloud computing," *Knowledge-Based Systems*, vol. 79, pp. 18–26, 2015.

[3] A. M. Manasrah, T. Smadi, and A. ALmomani, "A Variable Service Broker Routing Policy for data center selection in cloud analyst," *Journal of King Saud University - Computer and Information Sciences*, vol. 29, no. 3, pp. 365–377, 2017.

[4] B. B. Gupta and T. Akhtar, "A survey on smart power grid: frameworks, tools, security issues, and solutions," *Annales des Te´le´communications*, vol. 72, no. 9-10, pp. 517–549, 2017.

[5] J. Yu, R. Buyya, and K. Ramamohanarao, "Workflow scheduling algorithms for grid computing," in *Metaheuristics for scheduling in distributed computing environments*, pp. 173–214, Springer, 2008.

[6] A. Verma and S. Kaushal, "Cost-Time Efficient Scheduling Plan for Executing Workflows in the Cloud," *Journal of Grid Computing*, vol. 13, no. 4, pp. 495–506, 2015.

[7] H. Ji, W. Bao, and X. Zhu, "Adaptive workflow scheduling for diverse objectives in cloud environments," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 2, Article ID e2941, 2017.

[8] A. M. Manasrah, "Dynamic weighted VM load balancing for cloud-analyst," *International Journal of Information and Computer Security*, vol. 9, no. 1-2, pp. 5–19, 2017.

[9] W.-N. Chen and J. Zhang, "An ant colony optimization approach to a grid workflow scheduling problem with various QoS requirements," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 39, no. 1, pp.29–43,2009

[10] A. K. M. K. A. Talukder, M. Kirley, and R. Buyya, "Multiobjective differential evolution for scheduling workflow applications on global Grids," *Concurrency and Computation: Practice and Experience*, vol. 21, no. 13, pp. 1742–1756, 2009.

[11] "Privacy-preserving outsourced classification in cloud comput- ing," *Cluster Computing*, pp. 1–10, 2017.

[12] C. Stergiou, K. E. Psannis, B.-G. Kim, and B. Gupta, "Secure integration of IoT and Cloud Computing," *Future Generation Computer Systems*, vol. 78, pp. 964–975, 2018.

[13] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, "A genetic algorithm (GA) based load balancing strategy for cloud computing," *Procedia Technology*, vol. 10, pp. 340–347, 2013.

[14] Z. Zhang and X. Zhang, "A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation," in *Proceedings of the 2nd International Conference on Industrial Mechatronics and Automation (ICIMA '10)*, vol. 2, pp. 240–243, May 2010.

[15] T. D. Braun, H. J. Siegel, N. Beck et al., "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," *Journal of Parallel and Distributed Computing*, vol. 61, no. 6, pp. 810–837, 2001.

[16] M. Rana, S. Bilgaiyan, and U. Kar, "A study on load balancing in cloud computing environment using evolutionary and swarm based algorithms," in *Proceedings of the 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT 2014*, pp. 245–250, India, July 2014.

[17] Z. Zhu, G. Zhang, M. Li, and X. Liu, "Evolutionary multi-objective workflow scheduling in cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1344–1357, 2016.

[18] Y. Mao, X. Chen, and X. Li, "Max–Min task scheduling algorithm for load balance in cloud computing," in *Proceedings of International Conference on Computer Science and Information Technology*, S. Patnaik and X. Li, Eds., vol. 225, pp. 457–465, Springer, New Delhi, India, 2014.

[19] P. Kumar and A. Verma, "Scheduling using improved genetic algorithm in cloud computing for independent tasks," in *Proceedings of the 2012 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2012*, pp. 137–142, India, August 2012.

[20] L. Guo, S. Zhao, S. Shen, and C. Jiang, "Task scheduling optimization in cloud computing based on heuristic algorithm," *Journal of Networks*, vol. 7, no. 3, pp. 547–553, 2012.

[21] Liu H., Zhong F., Ouyang B., Wu J., An approach for QoS-aware web service composition based on improved genetic Algorithm, International Conference on Web Information Systems and Mining, 2010 Oct 23, 1, 123-128

[22] Chen R., Guo J., Bao F., Trust management for SOA-based IoT and its application to service composition, IEEE Transactions on Services Computing, 2016 May 1, 9(3), 482-95.

[23] Gen M., Cheng R., Genetic Algorithms & Engineering Design, John Wiley& Sons, Inc., New York, 1997

[24] R. Xu, R. An and X. Geng, "Research intrusion detection based PSO-RBF classifier", *Proc. IEEE 2nd Int. Conf. Softw. Eng. Service Sci.*, pp. 104-107, Jul. 2011.

[25] S. Chitra, B. Madhusudhanan, G. R. Sakthidharan, and P. Saravanan, "Local minima jump PSO for workflow scheduling in cloud computing environments," *Lecture Notes in Electrical Engineering*, vol. 279, pp. 1225–1234, 2014.

[26] Y. Ge and G. Wei, "GA-based task scheduler for the cloud computing systems," in *Proceedings of the International Conference on Web Information Systems and Mining (WISM '10)*, vol. 2, pp. 181–186, IEEE, October 2010.

[27] . Panda, S., & Padhy, N. P. (2008). Comparison of particle swarm optimization and genetic algorithm for FACTS-based controller design. Applied soft computing, 8(4), pp. 1418.