*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

# Capuchin Search Particle Swarm Optimization (CS-PSO) based Optimized Approach to Improve the QoS Provisioning in Cloud Computing Environment

**Manila Gupta[1], Bhumika Gupta[2], Dr. Devendra Singh[3]**
[1]Dept. Of Computer Science
lIFTM UNIVERSITY
Lodhipur,Moradabad
manilagupta24@gmail.com
[2]Associate Professor, Computer Science & Engineering Department
G.B.P.I.E.T.
Pauri-Garhwal
mail2bhumikagupta@gmail.com
[3]Associate Professor,Dept. Of Computer Science
lIFTM UNIVERSITY
Lodhipur,Moradabad
dev625@gmail.com

**Abstract**—This review introduces the methods for further enhancing resource assignment in distributed computing situations taking into account QoS restrictions. While resource distribution typically affects the quality of service (QoS) of cloud organizations, QoS constraints such as response time, throughput, hold-up time, and makespan are key factors to take into account. The approach makes use of a methodology from the Capuchin Search Particle Large Number Improvement (CS-PSO) apparatus to smooth out resource designation while taking QoS constraints into account. Throughput, reaction time, makespan, holding time, and resource use are just a few of the objectives the approach works on. The method divides the resources in an optimum way using the K-medoids batching scheme. During batching, projects are divided into two-pack assembles, and the resource segment method is enhanced to obtain the optimal configuration. The exploratory association makes use of the JAVA device and the GWA-T-12 Bitbrains dataset for replication. The outrageous worth advancement problem of the multivariable capacity is addressed using the superior calculation. The simulation findings demonstrate that the core (Cloud Molecule Multitude Improvement, CPSO) computation during 500 ages has not reached assembly repeatedly, repeatedly, repeatedly, and repeatedly, respectively.The connection analysis reveals that the developed model outperforms the state-of-the-art approaches. Generally speaking, this approach provides significant areas of strength for a successful procedure for improving resource designation in distributed processing conditions and can be applied to address a variety of resource segment challenges, such as virtual machine setup, work arranging, and resource allocation. Because of this, the capuchin search molecule enhancement algorithm (CSPSO) ensures the success of the improvement measures, such as minimal streamlined polynomial math, rapid consolidation speed, high productivity, and a wide variety of people.

**Keywords**-Cloud computing, Resource allocation, Throughput, Response time,PSO.

## I. INTRODUCTION

Finding a number of boundary values that satisfy a certain percentage of improvement is a problem of improvement, independent of whether some exhibition files of the framework arrive at the base or most extreme.The broad range of application domains demonstrates the versatility and effectiveness of bionic calculation in solving diverse problems.The implementation of bionic calculation has resulted in significant financial and societal benefits. One of the notable advantages of bionic calculation is its ability to address complex problems that traditional improvement algorithms struggle to simplify. As the size and complexity of the application objects increase, bionic calculation becomes particularly valuable in providing viable solutions. It offers a unique and valuable resource for tackling problems that were previously challenging to solve using conventional methods. Overall, bionic calculation has emerged as a powerful tool for innovation and problem-solving across various industries. By emulating nature's designs and processes, it enables more efficient and effective solutions, leading to tangible benefits for both financial and societal aspects.

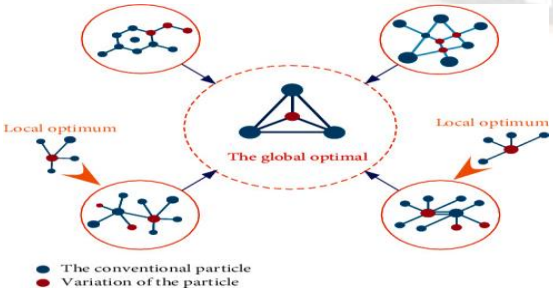
The Bionic calculation is a calculation model that considers human and natural behavior as well as the structure of material growth [21].Currently, bionic calculations include

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

local area calculations, molecular group advancement calculations, underground insect state enhancement calculations, counterfeit insusceptibility calculations, and more [22, 23].

Similar to the hereditary calculation and a laborious improvement calculation, particle swarm enhancement (PSO) is a type of bionic streamlining computation (see Figure 1). It begins with a number of strange configurations and then repeatedly searches for the best one. However, this is not quite the same as the inherited maxim that "the best makes due, the best gets by" in terms of developmental thinking. Molecule group calculations are easier to understand, have fewer limits that may be altered, and are simpler to do than other bionic calculations, such as hereditary calculations. PSO calculations are based on an integrated analysis of molecular group improvement calculations. At the moment, union models and intermingling research are required for the majority of advanced molecular group improvement estimates. Second, it is not difficult to enter the local enhancement problem into the molecular improvement calculation; that is, there is a problem with insufficient incorporation [24]. While tackling the problem of enhancing high-layered or super high-layered complex capabilities, molecule streamlining frequently runs into the problem of improper combination; in other words, when the population has not found the ideal arrangement, the particles have accumulated to a certain point and deteriorated.

The statements you provided touch upon two important aspects in the context of computational approaches: Escaping local minima: Local minima are points in a problem space where an algorithm gets stuck due to a suboptimal solution. It is indeed crucial to find mechanisms that enable computation to escape local minima. This is particularly important in optimization problems where the goal is to find the global optimum rather than settling for a suboptimal solution. Various techniques have been developed to address this issue, such as simulated annealing, genetic algorithms, and particle swarm optimization, which introduce randomness or exploration-exploitation trade-offs to help algorithms overcome local minima and search for better solutions.




The key field in the IT industry these days is distributed computing, if not the main area. Along with informational gathering servers and calculation servers, limit servers are one of the resources available in distributed processing [1]. These services are provided to customers through the cloud with a pay-more-only as expenses increase billing structure. Distributed computing, in general, refers to a PC platform that provides on-demand network access to a larger pool of framework organisation, taking care of, and restrict resources outside the web [2].This type of registration addresses cost reduction because the equipment isn't required for the specific needs of the customer [3]. Due to the unique concept of the assets offered by cloud specialist organisations, such as on-request advantages and broad organisation access, clients have difficulty selecting the appropriate assets [6].When customers need cloud administrations, they may purchase calculating resources from a cloud specialised company [4].Reputable companies like Google and Microsoft use the cloud's administrations as a wise alternative. Long execution times and rising costs are only two of the concerns raised by the growth of online apps and cloud organisations for cloud specialised co-ops [5]. One of the most urgent issues with the cloud is allocating resources according to the requirements of a client's application. The best solution to the asset designation problem is then provided using a variety of computations. An Amazon web administration offers a variety of services, each with a different pricing structure, such as information base, systems administration, processing, and capacity. As a result, the customer finds it difficult to select the asset for their own financial plan while also taking QoS into consideration [7].

In distributed computing, the following asset allocation processes are frequently used:

1. Static Portion: With this technique, resources are distributed to users or programmes in accordance with an appropriate designation strategy. When the obligation is consistent and predictable, this tactic performs exceptionally well.
2. Dynamic Allotment: Using this system, resources are allocated based on continuous interest and responsibility. In situations where the obligation is quirky and frequently changes, this strategy works exceptionally well.
3. Load adjusting: This process involves distributing the workload equitably over a number of servers to prevent any server from getting overloaded. Resource allocation on the servers can be improved by using computations for load change.
4. Virtualization: Virtualization enables several users or programmes to share a single real asset, such as a server or storage device. By creating virtual computers that may be deployed as assets depending on the scenario, virtualization enables flexible and effective asset categorization.
5. Hybrid Methodologies: To improve asset designation in light of responsibility characteristics and client requirements, several cloud suppliers use static and dynamic distribution methods.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

## II. LITERATURE SURVEY

The creators have provided a game-speculative framework for fair resource allocation in distributed economic organizations in [8]. Through calculation offloading administrations, handling assets anticipated by cars might be obtained, with the main focus of the earlier study being distributed computing or portable edge processing. The engineers have put forth a useful strategy [9] for automobile associations that employ both flexible edge handling and dispersed figuring. It is currently possible to offload acceptable estimates for NP-hard and non-bent issues. The results demonstrate how a practical strategy may be used to the suggested structure's presentation, but its primary drawback is its lengthy transmission time.

The authors of [10] presented a need-based, one-of-a-kind asset allocation strategy for distributed computing. The concept is meant to provide resources to different customers according to their needs rather than being completely constrained by their QoS requirements and administration-level agreements (SLAs). The designers suggest a need-based lining model and a novel asset identification computation that take the shifting accountability and asset accessibility in the cloud environment into account. They evaluate the suggested strategy through reenactment experiments and show how, in comparison to traditional techniques, it may provide superior QoS and asset usage. The study argues that the need-based strategy to distributing unique assets may improve the efficiency and presentation of distributed computing administrations while satisfying the various QoS requirements of various clients.

The authors in [11] have developed a sophisticated resource component model to fulfil client resource requirements while enhancing the plausibility of circulating processing and reducing transmission latency. The designers used the scattering multi-objective bug lion estimation (S-MOAL), a multi-objective request computation, to reduce costs and create a time of virtual machines. The energy consumption and response to internal dissatisfaction were also examined using the S-MOAL approach. The developers found that the S-MOAL computation was not particularly attractive for virtual machine tweaking or performing assurance.

In order to choose the optimal resource, a survey improvement approach in [12] suggested combining the fake bumble bee region (ABC) model with the replicated treating (SA) methodology. In response to growing need for dynamic asset distribution in virtual machines, the designers have developed a robust computation to overcome impedance difficulties in evaluating the display of asset part methodologies [13].In [14], the authors promoted a different levelled multi-specialist streamlining (HMAO) method to handle dispersed computing assets, with an eye on maximising asset use and minimising transmission capacity costs. This method combines multi-specialist streamlining with hereditary computations (GA) to identify administrative hubs with the best resource utilisation for job delivery. The HMAO method reduces the cost of data transmission by using decentralized-based MAO. This model's viability is compared to conventional tactics.

The authors of [15] have presented a project titled energy-compelling resource portion problem in cloud settings that aims to reduce energy consumption while still meeting the Idea of Organisation (QoS) requirements of cloud users. The designers suggest a novel method in light of the Bug Monkey Improvement (SMO) computation, which is motivated by the behaviour of eight-legged creature monkeys seeking food. The suggested approach calls for converting virtual hosts (VMs) in the cloud to real hosts.

The authors of [16] have promoted RAFL, a crossover metaheurisctic-based asset designation framework for load adjustment in distributed computing environments. The architecture incorporates three free metaheurisctic strategies—specifically, Firefly Calculation (FA), Dark Wolf Streamlining agent (GWO), and Molecule Multitude Advancement (PSO)—to accomplish effective load adjustment and asset allocation. There are two stages to the RAFL. At the primary level, the system uses a modified version of the K-Means grouping technique to organise virtual machines (VMs) according to their utilisation of computer CPU and memory.In the second step, cross-breed metaheuristic computations are used to evenly and effectively distribute the virtual machines (VMs) among the physical machines (PMs). To run tests to evaluate RAFL's presentation, the developers used CloudSim, a cloud simulation tool.

The developers released a brand-new load adjustment technique for distributed computing in [17], which was motivated by krill crowd behaviour. The suggested method uses a multi-objective improvement strategy to distribute tasks among cloud-based virtual machines while using the least amount of energy and available resources. The computation inspired by krill crowd behaviour functions by replicating krill's gathering behaviour in the pursuit space. In order to provide fresh replies and modify the hunt area, the computation makes use of a variety of strategies, including hybrid, change, and choice. Three measurements—makespan, energy usage, and burden balance—are used to evaluate the display of the suggested computation.

The makers in [18] advised the Cross breed Molecule Multitude Advancement (HPSO)- Numerous Hereditary Calculation (MGA) technique as a better method for identifying dynamic assets in cloud frameworks. The suggested calculation aims to reduce the energy consumption of the cloud environment by guaranteeing that the Nature of

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

Administration (QoS) needs of the cloud customers are met. The HPSO-MGA calculation is a half breed improvement technique that combines the benefits of PSO and GA. GA is used to improve the arrangement's accuracy and increase its combination rate, while PSO is used to swiftly study the search area and identify the optimal arrangement. The discoveries show that regardless of getting equivalent QoS fulfillment rates, the HPSO-MGA calculation beats different procedures concerning energy utilization.

The authors of [19] present a remarkable method for allocating resources in a distributed computing environment that combines a progressive asset portion plan with a grouping model, a multi-objective cross-breed Capuchin Search computation, and a hereditary calculation (GA). The main goal of the suggested technique is to improve the configuration of virtual machines (VMs) on cloud hubs while reducing energy consumption and speeding up response time. The optimal location for VMs is determined using the suggested method, taking energy consumption and response time into account.Every hub upgrades its asset allocation method using the GA.

Since many researchers are focused on studying streamlining problems, the number of improvement hypotheses and calculations is rapidly increasing. Currently, common streamlining methods include Newton's strategy, the simplex method, the form angle method, the trust locale method, the design search method, the Rosenbrock method, and the Powell method. These techniques are "powerless" in spite of these problems because they must traverse the entire search space and produce a combined search blast, which means that their computation speed, combination, and starting responsiveness fall far short of the requirements. As a result, accurate progress calculation has been one of the study interests of scientists [26]. The Sun et al. model has been used in several disciplines, including cunning administration and fluffy evaluation, to address the gap between the meaning of subjective information, the notion of worth, and its mathematical depiction. The key principles of species progress in nature are mirrored by the cloud model because it possesses the characteristics of fragility, vulnerability, solidity, and change in the outflow of information. As a result, the discipline of transformational processing has also begun to focus on cloud architecture [27].

This has the potential to significantly increase the calculation's capability for global inquiries [28]. In order to replace the traditional hybrid and change administrators in hereditary calculation, Zeng et al. presented a cloud hereditary calculation, which has achieved remarkable results in capability enhancement [29]. The issue of hereditary calculations is successfully addressed, and Kumari et al. combining hereditary calculations with cloud models delivers a cloud-based developmental calculation that successfully works with nearby enhancement and early combination [30]. Using molecule wellness and various dormancy weight development approaches, Omidinasab and Goodarzimehr offered a flexible cloud molecule swarm advancement calculation that effectiveladdressed the challenges of neighbourhood enhancement and an excessively rapid rate of calculation intermingling [31].

In order for the real molecule to be dynamic locally and globally searched, Zhu et al. advise that the momentum condition and space of a molecule in a population should be examined, appraised by the wellness worth of the molecule, and its speed altered by the wellness esteem [32].

This research suggests investigating molecular crowd computations and their applications in light of the review. A basic cloud administrator is used to concentrate on the development of particles and to make transformations to speed up the incorporation speed of the calculation. Through arrangement space modification, local enhancement and global streamlining are combined. It is usually clear from the reproduction results that the improvement measures focus on the accuracy of the population variety, search capabilities, and calculating reliability.

## III. PPROBLEM FORMULATION

This section lays out the suggested asset identification plans and provides comprehensive explanations of the general terminology and concepts of asset planning challenges. The key challenges of asset allocation in a cloud environment are how to efficiently organise, distribute, and assign a few tasks to several VMs with short execution times. The QoS requirement of each client is upgraded organically by this presentation enhancement. The cloud foundation includes actual machines (npm) and virtual machines (nvm), which are communicated as

Cloud system = [ PM1, PM2, … ,PMl, … , PMnpm ] , (1)

The notation PMl represents the physical machines processed in the cloud, where l is a variable that ranges from 1 to PMnpm. Let's break down the notation further:

PMl: This represents a specific physical machine in the cloud. The subscript "l" denotes the machine number or identifier, indicating that there are multiple physical machines in the system.

l = 1, 2, … ,PMnpm: This notation specifies the range of values that the variable "l" can take. In this case, "l" can assume values starting from 1 and incrementing by 1 until it reaches PMnpm. The notation PMnpm denotes the maximum number of physical machines in the cloud.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

To provide a concrete example, let's say PMnpm is equal to 5, meaning there are five physical machines in the cloud. In this case, the notation PMl would represent the individual machines, such as PM1, PM2, PM3, PM4, and PM5. The variable "l" would take values from 1 to 5, representing each of these physical machines.

PM = [ VM1, VM2, … ,VMm, … , VMnvm ] , (2)

Where VMm, m stands for the mth virtual machine, nvm stands for the number of virtual machines, and VMm is used to refer to the mth virtual machine that is present in the cloud. The qualities of VMm are assessed as

VMm = [ SidVm, Mipsm ] . (3)

the processing acceleration reports submitted by VMs when processing millions of instructions per second are denoted by the letters id and

Mipsm. T = [ Ts1, Ts2, … ,Tsl, … , Tsnts] . (4)

The user specifies the quantity of task l as nts, and the task sequence's lth task is represented as Tsl, which is stated as

Tsl = [ SidTl, Ts − lengthl, Ectl, Tspl ] . (5)

The character number for the lth task is denoted as SidTl, the length of the given undertaking is denoted as Ts - length l, the execution period of the task planning is denoted as Ectl, and the assignment's inclination in relation to the total number of assignmentnts is denoted as Tspl. The execution time predicted to complete an operation on a virtual machine is addressed by the anticipated execution time (Ect) examination of size ntsnvm. The standard asset part plans provide information on the QoS limits, but they are unable to take into account the nature of the undertaking and responsibility. The additional allocation of resources does not ensure efficacy because of the lack of knowledge regarding responsibility type. Because of the absence of assets in the cloud, asset provisioning is a dangerous errand in distributed computing.

The challenges of increasing asset utilization and reducing reaction time are dealt with in the proposed work using the CAPUCHIN SEARCH and Molecule SWAN advancement techniques. The CS-PSO approach that is being presented is designed to reduce the makespan by focusing on throughput and execution time. In this way, our suggested model's multi-complaint capacity limits organization costs and response times while improving execution times, asset utilisation, and throughput.

## IV. PROPOSED METHODOLOGY

We are concentrating on asset assignment and associated perspectives in order to improve network administrations and hence QoS in distributed computing. Throughput, makespan time, reaction time, time utilisation, usage % for distinct positions, and holding time are measurements that would affect asset assignment since issue-tolerant frameworks are linked to dependable frameworks. The adaptation to non-critical failure framework includes a number of key conditions that are included in reliability. In this section, we numerically justify asset classification in our framework model. Despite a few practical constraints, our aim is to improve asset utilisation and lower transfer speed costs in distributed computing. To ensure that cloud providers may effectively and effectively assign resources to suit the varying demands of their customers' applications, it is essential to address the fundamental problem of asset portion in distributed computing.

The problem arises from how limited cloud resources are allocated, and it needs to be done in a way that increases efficacy, lowers costs, and also ensures that exhibition is not compromised. This calls for the advancement of current asset distribution calculations and procedures that can gradually distribute resources to meet changing needs while also accurately representing the complex interactions between various asset types, such as computer chip, memory, stockpiles, and organisational data transfer facility. Additionally, when dispensing assets, cloud suppliers must take into account factors like information security, protection, and administrative consistency, further complicating the situation. This makes the problem of asset identification in distributed computing a complex and challenging one that calls for careful consideration and original solutions to ensure that distributed computing continues to be useful to organisations and associations.

### 4.A K-medoids algorithm

**The K-medoids method works in the following way as a clustering algorithm:**

Random Initialization: Choose K initial medoids at random from the dataset. The centroids or representatives of their respective clusters will be these medoids. The ideal number of clusters is K.

Calculate the distance between each data point in the dataset and each of the medoids as part of your assignment. Based on a distance metric, such as the Euclidean distance, assign the data point to the cluster that is represented by the closest medoid.

Step for an update: Calculate the total separation (or dissimilarity) between each cluster's data points and the medoid. The data point with the smallest total distance should be chosen as the replacement medoid for that cluster if replacing the existing medoid with any other data point in the cluster decreases the total distance.

Iteration: Repetition of stages 2 and 3 up until the fulfilment of one of the halting conditions. When the medoids stop

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

changing or when the allotted number of iterations has been achieved might serve as the terminating condition.

The K-medoids technique divides the data into clusters, with each data point belonging to the cluster represented by the nearest medoid, by repeatedly updating the medoids depending on the distances between data points and medoids.

4.B Capuchin-Search Particle Swarm Optimization (CS-PSO)

A hybrid optimization technique called CS-PSO, often referred to as Capuchin Search Molecule Swarm Optimization, combines the Capuchin Search algorithm with Molecule Swarm Optimisation (PSO) to ensure effective resource allocation in cloud computing settings. Resource allocation is an essential operation in cloud computing to guarantee resource efficiency and offer Quality of Service (QoS) to cloud users. Resource allocation aims to distribute resources to activities and services in a way that maximises system performance overall while maintaining effective resource utilisation. By improving the distribution of resources to activities and services, CS-PSO may be used for resource allocation in cloud computing. The algorithm can improve the allocation of CPU, memory, and storage resources to cloud services, maximizing overall system performance while meeting the QoS requirements of cloud users. The CS-PSO algorithm initiates by dividing the cloud resources into several clusters based on their characteristics such as processing power, memory capacity, and network bandwidth. It then assigns a group of particles to each cluster. These particles perform local searches and random explorations to find the best solution within their assigned cluster. The particles also communicate with each other to share information about their best solutions. The Capuchin Search update mechanism enables particles to coordinate their search efforts and prevent premature convergence. The PSO update mechanism is used to update the speed and position of the particles. The CS-PSO algorithm incorporates a fitness function that evaluates the quality of the resource allocation solution. The fitness function considers the QoS requirements of cloud users and the utilization of cloud resources.The Capuchin Search Molecule Swarm Optimisation (CS-PSO) algorithm's overall process for allocating resources in cloud computing may be summed up as follows:

- Initialization: Set the settings for the Capuchin Search and PSO updating methods as well as the algorithm's parameters for the number of particles, the number of clusters, the maximum number of iterations, and other factors.
- Cluster Formation: Based on factors like processor speed, memory size, and network bandwidth, group the cloud resources into several clusters. Give each cluster a set of particles.
- Particle Initialization: Each particle's location and speed inside its designated cluster are initialised at random.
- Particle Evaluation: Assess each particle's fitness using a fitness function that takes into account the QoS needs of cloud users and how they use cloud resources.
- Local Search: To obtain the best answer, each particle does a local search inside the cluster to which it has been allocated. Each particle's location is updated using the Capuchin Search update method.
- Global Communication: To convey information about its ideal solution, each particle interacts with other particles. The particles' search attempts are coordinated by the Capuchin Search update mechanism.
- Velocity and Position Update: Based on each particle's best solution and the best solutions of its neighbours, the PSO update mechanism is utilised to update each particle's velocity and position.

Termination: Check if the maximum number of iterations is reached or if the termination criteria are met. If not, go back to stage 4.

Output: Return the best solution found by the algorithm as the optimized resource allocation solution.

This iterative process continues until the termination conditions are satisfied, and the algorithm outputs the best resource allocation solution obtained.

Output: The algorithm returns the best arrangement found as the optimized resource allocation solution.

The CS-PSO technique enhances resource allocation in cloud computing by combining the advantages of Capuchin Search and Particle Swarm Optimisation (PSO). The PSO component

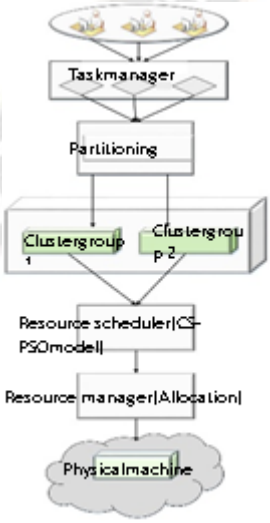



FIGURE1 Overall architecture of proposed methodology

optimizes the speed and position of the particles to converge towards the desired solution, while the Capuchin Search component aids particles in the algorithm in organizing their

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

search efforts and preventing premature convergence.The Partitioning Around Medoids (PAM) clustering algorithm, on the other hand, is a clustering technique that resembles K-means. The medoid, which is the most central point in a cluster, is used in place of computing the mean of the points inside each cluster. The data point with the lowest average dissimilarity to every other point in its cluster is represented by the medoid.

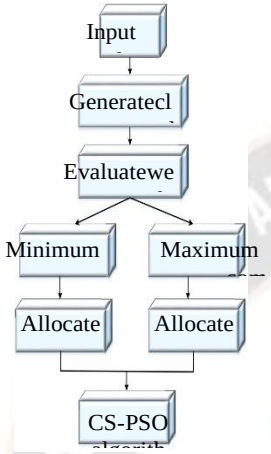



FIGURE2 -Process of PAKM clustering approach"

## V. RESULTS AND DISCUSSIONS

Five different virtual machine types (VMs) were used in the experiment, and they were divided into low- and high-performance groups according to their MIPS value. While VMs 4 and 5 were rated as high performers, VMs 1, 2, and 3 were rated as low performers. The GWA-T-12 Bitbrains dataset, which includes performance statistics for 1750 virtual machines in a Bitbrains data centre, was used in the simulation. Bitbrains is a company that offers services for managing computer operations for businesses. Each virtual machine's performance statistics files are included in the dataset and are arranged according to Rnd and fast storage cues. While 500 VMs were linked to either Network Attached Storage (NAS) or a fast SAN in the secondscenario, 1,250 VMs were attached to a fast Storage Area Network (SAN) device in the first storage systems. The performance metrics files provide data on resource utilisation and VM performance that may be used to assess how well resource allocation and scheduling algorithms work in the cloud. For academics and practitioners to create and test novel algorithms for resource allocation and scheduling in cloud computing environments, the dataset is a useful resource.

5.1 Make span

The amount of time needed to finish all jobs for all resources is measured by the make span. It is determined as the difference between a task's beginning and finish locations. A smaller make span value denotes resource allocation scheduling methods that are more effective.Figure 1 provides an illustration of the evaluation of make span within the framework of the suggested model. The suggested model's make span is contrasted with those of already-in-use methods including FCFS, PSO, KPSHOW, and KMPS. Four alternative task lengths—50, 100, 150, and 200—are taken into account in the evaluation. By cutting the make span at each level of the issue, the suggested model performs better than the other solutions, according to the comparison.

This suggests that the suggested model offers more efficient scheduling techniques forresource allocation, leading to faster total job completion times and more effective resource management. By reaching a smaller makespan value than the other tested approaches, the suggested model proves its capacity to optimise resource allocation and improve scheduling.

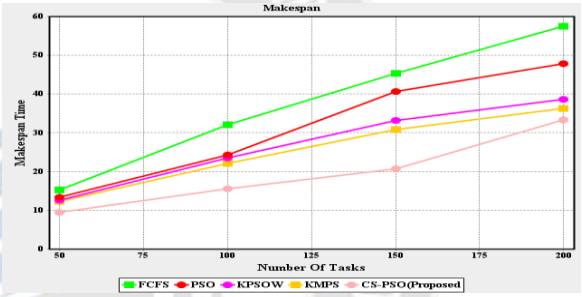



Figure 1 Makespan vs Number of tasks

Based on the provided information, the make span values of the proposed model and other existing methodologies can be observed in the above picture. The make span values for the proposed model are as follows:

10.45 seconds for 50 tasks
17.6 seconds for 100 tasks
25.67 seconds for 150 tasks
31.34 seconds for 200 tasks

Comparing these values to the make span values of the previous methodologies, it is evident that the proposed model achieves the smallest make span value at each task level. This indicates that the proposed model provides more efficient scheduling strategies for resource allocation and task completion, resulting in shorter overall completion times for the given number of tasks.The smaller make span values obtained by the proposed model demonstrate its effectiveness in optimizing resource allocation and scheduling, outperforming the previous methodologies in terms of efficiency and performance.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

5.2 Utilization of virtual machines

The assessment of virtual machine utilization was conducted for different numbers of tasks, ranging from 50 to 200, as depicted in Figure 2-4. The utilization of virtual machines in the proposed model was compared to that of existing techniques, including FCFS, PSO, KPSHOW, and KMPS. According to the assumption, virtual machines with lower performance are expected to handle less complex tasks, while those with higher performance should handle more complex tasks. As a result, the usage rate of VM1 should be lower than that of VM2 and VM3 for virtual machines with lower performance, while the usage rate of VM4 should be lower than VM5 for virtual machines with higher performance. Additionally, the overall make span time should be reduced.The proposed CS-PSO approach outperformed the existing methods by effectively managing virtual machine assignments and achieving a shorter make span. This indicates that the proposed approach successfully controlled the allocation of virtual machine resources, resulting in improved utilization rates and a more efficient scheduling of tasks. By accurately managing virtual machine jobs and achieving a shorter make span, the proposed CS-PSO approach demonstrates its superiority over the existing methods in terms of resource utilization and scheduling efficiency.

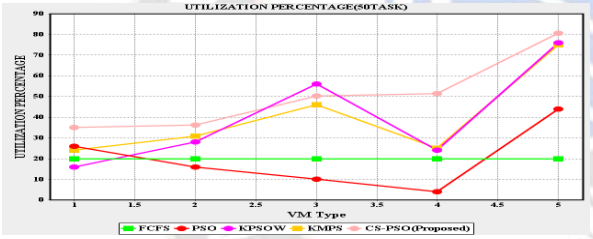



Figure 2Utilization vs VM types at 50 tasks

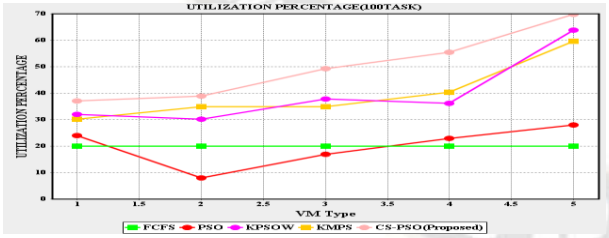



Figure 3Utilization vs VM types at 100 tasks

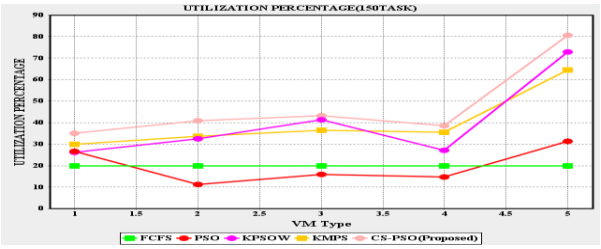



Figure 4Utilization vs VM types at 150 tasks

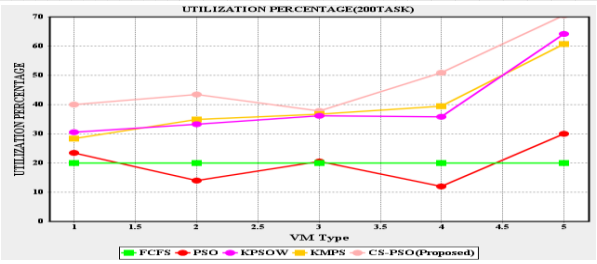



Figure 5Utilization vs VM types at 200 tasks

5.3 Waiting time

The amount of time that tasks spend in the queue for each designated virtual machine is referred to as the average waiting time, also known as the average queueing time. It reflects the typical amount of time a job must wait before the virtual machine can process it. The mean waiting time for all jobs in the queue may be used to numerically represent the average waiting time. Assume that there are 'n' tasks and that 't1, t2, t3,..., tn' represents the waiting periods for each job. The average waiting time, T_avg, may therefore be determined as follows:

T_avg = (t1 + t2 + t3 + ... + tn) / n

The average waiting time is calculated by adding together all of the waiting times and dividing the total by the total number of jobs. It is important to keep in mind that the precise waiting time values will depend on the individual simulation or experimental setting, and the average waiting time gives an indication of how long activities typically take to complete when they are in queue.

as, $$Aw_t = \frac{\sum Tw_t}{m}$$

The typical waiting time, denoted as 'T_avg', represents the average time that tasks spend waiting to be executed. It is calculated by dividing the total waiting time by the total number of tasks.In the given context, there are a total of 100 tasks, with 500 being the final test. The proposed approach achieves a lower waiting time compared to the mentioned existing methods for each task length. This indicates that the proposed model results in reduced waiting time for task allocation in the cloud server.

Figure 6 compares the typical waiting time of the proposed model to the currently used methods. The analysis of the experiments demonstrates that the recommended model significantly reduces the waiting time for task allocation compared to the existing strategies. The specific values for the waiting times and the comparison depicted in Figure 6 will depend on the experimental setup and the performance of the algorithms used in the evaluation. The lower waiting time achieved by the proposed model highlights its effectiveness in optimizing task allocation and reducing the waiting period in the cloud server environment.

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

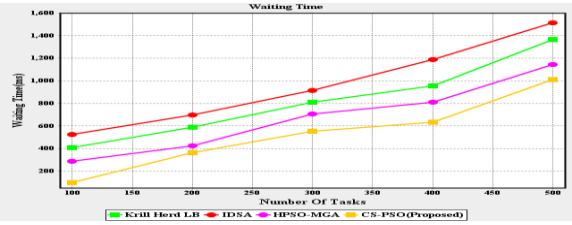


Figure 6 Waiting time vs Number of tasks

5.4 Response time

The waiting time and service time are added to determine the response time, which is the amount of time needed to reply to a service request. The proposed Capuchin Search Particle Swarm Optimisation (CS-PSO) method is compared to existing approaches like Krill Crowd Load Balancing (LB), Hybrid Particle Swarm Optimisation with Multiple Genetic Algorithms (HPSO-MGA), and Improved Differential Search Algorithm (IDSA) in Figure 7 of the paper. The comparison's findings show that the suggested methodology works better than the current approaches, resulting in a quicker response time for projects of various durations. The simulation analysis shows that, in terms of response time, the suggested strategy outperforms the present methods.

These results support the hypothesis that the suggested strategy improves cloud computing resource allocation and scheduling efficiency. The suggested method improves the general efficiency and efficacy of service delivery in the cloud computing environment by obtaining quicker reaction times.

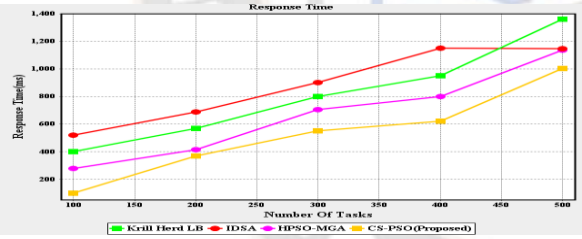


Figure 7 Response time vs Number of tasks

The evaluation of reaction time was conducted for a range of task lengths, specifically from 100 to 500. According to the provided information, the proposed model yielded the following response times:

105ms for a task length of 100
376ms for a task length of 200
555ms for a task length of 300
624ms for a task length of 400
1014ms for a task length of 500

These results indicate that the proposed model achieves faster response times compared to the current methodologies. The decreasing trend in response times as task length increases suggests that the proposed model effectively handles tasks of varying lengths, optimizing resource allocation and scheduling to minimize response times. By achieving faster response times for a range of task lengths, the proposed model demonstrates its superiority over the existing methodologies in terms of performance and efficiency. These results provide evidence of the improved effectiveness of the proposed model in resource allocation and scheduling for cloud computing environments.

5.5 Throughput

The total number of jobs finished within a certain amount of time is referred to as throughput. Figure 8 shows a throughput comparison of the suggested method's performance with those of known approaches like Improved Differential Search Algorithm (IDSA), Hybrid Particle Swarm Optimisation with Multiple Genetic Algorithms (HPSO-MGA), and Krill Crowd B.

Throughput's primary goal is to reduce the amount of time needed to finish a task. The quantity of requests or tasks being handled affects throughput performance. Figure 8 compares the suggested approach's throughput performance to the aforementioned current approaches based on the information supplied. It compares how many jobs each strategy accomplished in a specific amount of time. By achieving higher throughput, the proposed approach demonstrates its ability to efficiently process a larger number of tasks within the same time period compared to the existing methodologies. This indicates improved performance and efficiency in task processing and resource allocation in the context of distributed computing environments.

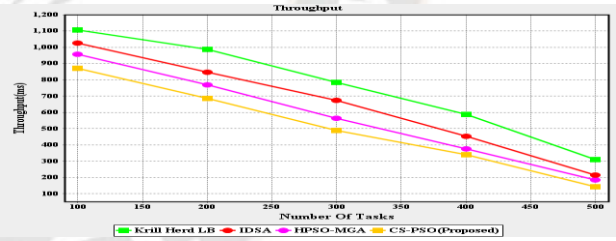


Figure 8 Throughput vs Number of tasks

The quantity of tasks being completed affects throughput performance, which seeks to reduce the amount of time needed to finish a process. The jobs in the context are between 100 and 500, and the throughput time is expressed in milliseconds. According to the facts given, the suggested model should take less time to perform the tasks than the approaches now in use. This shows that the suggested model performs better in terms of throughput than the current approaches. The suggested model proves its superior performance in resource allocation and schedule optimisation by attaining a greater throughput and completing the jobs more effectively. It successfully reduces the amount of time needed

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

to accomplish the activities and boosts the system's overall effectiveness.

5.6 Time utilization

The proposed Capuchin Search Particle Swarm Optimisation (CS-PSO) method is compared to other methods that are already in use, such as Krill Crowd Load Balancing (LB), Improved Differential Search Algorithm (IDSA), and Hybrid Particle Swarm Optimisation with Multiple Genetic Algorithms (HPSO-MGA), in Figure 9 of the paper. The simulation's findings show that the recommended technique produces a lower rate of time utilisation than the other approaches. This shows that the suggested method is more effective and takes less time to schedule and allocate resources in the setting of distributed computing. Lower time utilisation implies that the suggested model optimises resource allocation and use, enabling more effective task processing inside the system. The suggested method boosts the overall effectiveness and performance of resource allocation and scheduling in distributed computing systems by efficiently using resources and minimising time overhead. These results demonstrate how the suggested CS-PSO strategy is superior and more successful at delivering effective and timely resource allocation and planning.

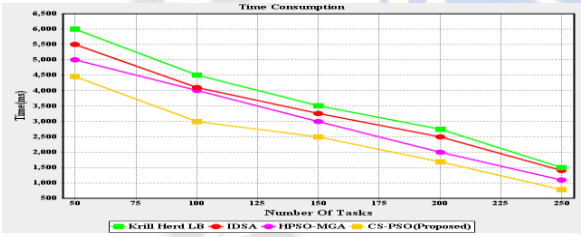



Figure 9 Time consumption vs Number of tasks

## VI. CONCLUSION

In conclusion, the suggested strategy based on Capuchin Search Particle Swarm Optimisation (CS-PSO) provides a useful and efficient way to improve Quality of Service (QoS) provisioning in distributed computing settings. The strategy makes the most of resource allocation and improves QoS provisioning by combining the advantages of the Capuchin Search algorithm with Particle Swarm Optimisation. With bandwidth utilisation rates of 0.75%, 0.86%, and 0.835 for 4, 6, and 16 jobs, respectively, the suggested model's attained results show its effectiveness. The suggested method also performs better in terms of time use and makespan than other current methods. The suggested paradigm improves a number of performance parameters, including response time, availability, and throughput, according to simulation findings. The suggested solution outperforms alternative approaches in terms of performance, improving resource allocation to satisfy various QoS criteria while preserving resource efficiency. These results demonstrate how the suggested strategy is more effective than current techniques in raising QoS performance metrics. The suggested method may be used by cloud service providers to improve resource allocation and QoS provisioning, which would boost customer satisfaction and overall performance in distributed computing systems.

## References

[1] Rashid, Aaqib, and Amit Chaturvedi. "Cloud computing characteristics and services: a brief review." International Journal of Computer Sciences and Engineering 7, no. 2 (2019): 421-426.

[2] Kollolu, Roopha. "Infrastructural Constraints of Cloud Computing." International Journal of Management, Technology and Engineering 10 (2020): 255-260.

[3] Alam, Tanweer. "Cloud Computing and its role in the Information Technology." IAIC Transactions on Sustainable Digital Innovation (ITSDI) 1 (2021): 108-115.

[4] Akintoye, Samson Busuyi, and Antoine Bagula. "Improving quality-of-service in cloud/fog computing through efficient resource allocation." Sensors 19, no. 6 (2019): 1267.

[5] Kumar, Mohit, SubhashChander Sharma, AnubhavGoel, and Santar Pal Singh. "A comprehensive survey for scheduling techniques in cloud computing." Journal of Network and Computer Applications 143 (2019): 1-33.

[6] Devarasetty, Prasad, and Satyananda Reddy. "Genetic algorithm for quality of service based resource allocation in cloud computing." Evolutionary Intelligence 14, no. 2 (2021): 381-387.

[7] Shrimali, B., & Patel, H. (2020). Multi-objective optimization oriented policy for performance and energy efficient resource allocation in Cloud environment. Journal of King Saud University-Computer and Information Sciences, 32(7), 860-869.

[8] Wei, G., Vasilakos, A.V., Zheng, Y. et al. A game-theoretic method of fair resource allocation for cloud computing services. J Supercomput **54**, 252–269 (2010). https://doi.org/10.1007/s11227-009-0318-1

[9] Zhao, Junhui, Qiuping Li, Yi Gong, and Ke Zhang. "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks." IEEE Transactions on Vehicular Technology 68, no. 8 (2019): 7944-7956.

[10] C. S. Pawar and R. B. Wagh, "Priority Based Dynamic Resource Allocation in Cloud Computing," 2012 International Symposium on Cloud and Services Computing, Mangalore, India, 2012, pp. 1-6, doi: 10.1109/ISCOS.2012.14.

[11] Belgacem, Ali, KaddaBeghdad-Bey, HassinaNacer, and Sofiane Bouznad. "Efficient dynamic resource allocation method for cloud computing environment." Cluster Computing 23, no. 4 (2020): 2871-2889.

[12] Muthulakshmi, B., and Krishnan Somasundaram. "A hybrid ABC-SA based optimized scheduling and resource allocation for cloud environment." Cluster Computing 22, no. 5 (2019): 10769-10777.

[13] Ramasamy, Vadivel, and SudalaiMuthuThalavai Pillai. "An effective HPSO-MGA optimization algorithm for dynamic

*International Journal on Recent and Innovation Trends in Computing and Communication*
*ISSN: 2321-8169 Volume: 11 Issue: 6s*
*DOI: https://doi.org/10.17762/ijritcc.v11i6s.6937*
***Article Received**: 22 March 2023 **Revised**: 12 May 2023 **Accepted**: 25 May 2023*

resource allocation in cloud environment." Cluster Computing 23, no. 3 (2020): 1711-1724.

[14] Gao, Xiangqiang, Rongke Liu, and Aryan Kaushik. "Hierarchical multi-agent optimization for resource allocation in cloud computing." IEEE Transactions on Parallel and Distributed Systems 32, no. 3 (2020): 692-707.

[15] Samriya, J. K. ., & Kumar, N. (2022). Spider Monkey Optimization based Energy-Efficient Resource Allocation in Cloud Environment. Trends in Sciences, 19(1), 1710. https://doi.org/10.48048/tis.2022.1710

[16] A. Thakur and M. S. Goraya, "RAFL: A hybrid metaheuristic based resource allocation framework for load balancing in cloud computing environment," Simulation Modelling Practice and Theory, vol. 116, p. 102485, 2022.

[17] Raed Abdulkareem HASAN*, Muamer N. MOHAMMED, A Krill Herd Behaviour Inspired Load Balancing of Tasks in Cloud Computing, Studies in Informatics and Control, ISSN 1220-1766, vol. 26(4), pp. 413-424, 2017.

[18] Ramasamy, V., Thalavai Pillai, S. An effective HPSO-MGA optimization algorithm for dynamic resource allocation in cloud environment. Cluster Comput **23**, 1711–1724 (2020). https://doi.org/10.1007/s10586-020-03118-x.

[19] K. K. Gola, B. M. Singh, B. Gupta, N. Chaurasia, and S. Arya, "multi-objective hybrid capuchin search with genetic algorithm based hierarchical resource allocation scheme with Clustering Model in cloud computing environment," Concurrency and Computation: Practice and Experience, vol. 35, no. 7, 2023.

[20] Doumari S. A., Givi H., Dehghani M., Montazeri Z., Leiva V. A new two-stage algorithm for solving optimization problems. Entropy . 2021;**23**(4):491–496. doi: 10.3390/e23040491. [PMC free article] [PubMed] [CrossRef] [Google Scholar]

[21] Hai T., Chen W. C., Wang X. N., Chen L. Multi-objective reservoir operation using particle swarm optimization with adaptive random inertia weights. Water Science and Engineering . 2020;**13**(2):136–144. doi: 10.1016/j.wse.2020.06.005. [CrossRef] [Google Scholar]

[22] Wang X. L., Xie W. X., Li L. Q. Interacting t-s fuzzy particle filter algorithm for transfer probability matrix of adaptive online estimation model. Digital Signal Processing . 2021;**110**(5):102944–102949. doi: 10.1016/j.dsp.2020.102944. [CrossRef] [Google Scholar]

[23] Gao X., Xie W., Chen S., Yang J., Chen B. The prediction of human abdominal adiposity based on the combination of a particle swarm algorithm and support vector machine. International Journal of Environmental Research and Public Health . 2020;**17**(3):1117–1123. doi: 10.3390/ijerph17031117. [PMC free article] [PubMed] [CrossRef] [Google Scholar]

[24] Lenin K. Tailored particle swarm optimization algorithm for solving optimal reactive power problem. International Journal of Regulation and Governance . 2020;**5**(12):246–255. doi: 10.29121/granthaalayah.v5.i12.2017.500. [CrossRef] [Google Scholar]

[25] Han L., Tang L., Tang Y. Sports image detection based on particle swarm optimization algorithm. Microprocessors and Microsystems . 2020;**80**(2):103345–103348. [Google Scholar]

[26] Du S., Deng Q. Unscented particle filter algorithm based on divide-and-conquer sampling for target tracking. Sensors . 2021;**21**(6):2236–2240. doi: 10.3390/s21062236. [PMC free article] [PubMed] [CrossRef] [Google Scholar]

[27] Sun S., Zhang H., Dong L., Fang X., Khan M. S. Multibyte electromagnetic analysis based on particle swarm optimization algorithm. Applied Sciences . 2021;**11**(2):839–843. doi: 10.3390/app11020839. [CrossRef] [Google Scholar]

[28] Song Y. A fractional pid controller based on particle swarm optimization algorithm. Journal of Autonomous Intelligence . 2020;**3**(1):1–7. doi: 10.32629/jai.v3i1.94. [CrossRef] [Google Scholar]

[29] Zeng W., Zhu W., Hui T., Chen L., Xie J. An imc-pid controller with particle swarm optimization algorithm for msbr core power control. Nuclear Engineering and Design . 2020;**360**(2):110513–110517. doi: 10.1016/j.nucengdes.2020.110513. [CrossRef] [Google Scholar]

[30] Kumari R., Gupta N., Kumar N. Cumulative histogram based dynamic particle swarm optimization algorithm for image segmentation. Indian Journal of Computer Science and Engineering . 2020;**11**(5):557–567. doi: 10.21817/indjcse/2020/v11i5/201105183. [CrossRef] [Google Scholar]

[31] Dimf, G. P. ., Kumar , P. ., & Manju, V. N. . (2023). An Efficient Power Theft Detection Using Modified Deep Artificial Neural Network (MDANN). International Journal of Intelligent Systems and Applications in Engineering, 11(1), 01–11. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2437.

[32] Omidinasab F., Goodarzimehr V. A hybrid particle swarm optimization and genetic algorithm for truss structures with discrete variables. Journal of Applied and Computational Mechanics . 2020;**6**(3):593–604. [Google Scholar]

[33] Zhu M., Chu S. C., Yang Q., Li W., Pan J. S. Compact sine cosine algorithm with multigroup and multistrategy for dispatching system of public transit vehicles. Journal of Advanced Transportation . 2021;**2021**(2):16. doi: 10.1155/2021/5526127.5526127 [CrossRef] [Google Scholar]