I. J. Computer Network and Information Security, **2025, 1, 68-81**
Published Online on February 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2025.01.06

Efficient Resource Allocation to Enhance the Quality of Service in Cloud Computing

Shubhangi Pandurang Tidake*
PG Department of Computer Science, Sant Gadge Baba Amravati University, Amravati, Tapovan Campus Amravati-444602, Maharashtra, India
E-mail: shubha.tidke@gmail.com
ORCID iD: https://orcid.org/0000-0002-4302-5115

Pramod N. Mulkalwar
PG Department of Computer Science, Sant Gadge Baba Amravati University, Amravati, Tapovan Campus Amravati-444602, Maharashtra, India
E-mail: pramodmulkalwar@gmail.com
ORCID iD: https://orcid.org/0009-0008-0886-2285

Received: 21 August 2023; Revised: 27 October 2023; Accepted: 28 June 2024; Published: 08 February 2025

Abstract: Pay-as-you-go models are used to grant users access to cloud services. While using the cloud, an imbalance workload on data centre resources degrades quality of service metrics like makespan, storage, high failure rate, and energy consumption. Hence proposed the heuristic based hybrid GA to enhance the QoS with resource allocation in cloud computing. The population is first initialized using the Binary encoding sorts the tasks according to priority. After that, the Best Fit algorithm compares the Best Fit with iterations of each fitness value depending on the computation time to shorten the make span. Heuristic crossover approach and mutation are then used to update the probability of the existing population with the new population lowers the failure rate by using the fitness value. Therefore, the proposed heuristic-based hybrid GA technique balanced the load and allocate the resources effectively to improve QoS performances. The outcome reveals that the proposed method of QoS performances attained less makespan, energy consumption, failure rate and execution time with effectively allocated the resources of 1% to 39% when compared to the previous methods in cloud computing.

Index Terms: Cloud Computing, Resource Allocation, Genetic Algorithm(GA), Binary Encoding, Virtual Machine(VM), Quality of Service(QoS).

1. Introduction

Cloud computing is a widely used computer technique that provides on-demand services, scalability, flexibility, and usability. Users pay for the services it consumes, and it allows them to access resources such as memory, platforms, apps, and architecture on demand across a network. [1]. The supply of computational resource services across a network (internet) is the basic tenet of cloud computing. Clients can use cloud computing services based on their needs rather than being required to have a big, complex computer system in order to perform their business [2]. A major problem with energy usage is the growing number of computers in cloud data centres. In these centres, load balancing, scalability, and performance all depend on task scheduling. It seeks to reduce process running time and maximize resource utilisation by matching the best resources to workloads [3]. Despite its promise, the cloud paradigm frequently suffers from delayed access since software, infrastructure, and data processing are all located remotely. This leads to lengthier transmission times and higher energy consumption, which makes data retrieval challenging [4].

Virtualization is regarded as a crucial management technology that makes it possible to allocate and in cloud computing, provide resources as needed [5]. With the use of virtual resources, cloud computing services provide an increasing number of scheduling possibilities as they grow in size. Building information-storing, energy-efficient data centres is essential to reducing global warming. There should be intelligent techniques for assessing and controlling cloud computing resources [6]. The bulk of pertinent resource allocation technology has concentrated on raising resource utilization rate and reducing power use through approaches including dynamic resource allocation, process performance analysis, and time series data forecasting of distributed host load [7]. Clients pay service providers for consumption, but the rise of data centres owing to worldwide computing demand has resulted in energy use difficulties, which have an

influence on the environment and costs. [8].

Cloud service providers can allocate the resources linked to service nodes that are accessible to the tasks that have been requested based on supply and demand. Each task requires physical resources like a CPU, RAM, or graphics processing unit (GPU). Additionally, data transit over several service nodes requires bandwidth [9]. Servicing providers' goals is impacted by the difficulty of managing virtual machines (VMs) in cloud data centres. These difficulties are associated with NP-hard optimization problems, and multi-objective optimization is a widely used method for resolving VMS and VMA problems [10]. Virtual machine (VM) configuration errors occur when VMs are configured repeatedly in the cloud, increasing setup time and resulting in overhead and SLA breaches. In order to prevent under- or overcrowded physical hosts in data centres, service providers prioritize deploying virtual machines (VMs) on the optimal host to satisfy end-user demands. [11]. Increased load levels on certain virtual machines have a detrimental effect on system performance, demanding resource division, restricting cloud resources, and slowing down operations [12].

Ineffective VM placement lowers the fitness function calculation's total complexity and uses more energy. A viable strategy for logically organizing VMs in PMs to speed up fitness function calculation while using less time and energy is still needed for faster GA in cloud data centers [13]. One strategy to balance the burden is to move virtual machines (VMs) from one overloaded virtual machine health centre (VMH) to other VMHs. This ensures that only VMs that require relocation are stopped and resumed during migration, which might lead to an increase in demand for the target VMHs [14]. Computational resources including CPU power, memory capacity, and I/O bandwidth are the main emphasis of virtual machines (VMs). Nevertheless, VM overloading might result in a scarcity of real machine resources in some scenarios, such as traffic jams or weather shifts. [15]. In the cloud environment, it is difficult to foresee how many requests will be sent in a second, therefore load balancing has become a significant problem. The unpredictable nature of the cloud was caused by its constantly shifting behavior [16]. However, load balancing in virtual machine is critical. Thus, the Balance approach for resource allocation must be used in cloud computing in order to balance the load and enhance QoS.

These are the ways in which the article contributed.

- In cloud computing the imbalance of workload in the data centre resources deteriorates the QoS performance such as makespan, storage, high failure rate and energy consumption. Hence, it is necessitating to introduce the heuristic based hybrid Genetic Algorithm (GA) technique for resource allocation to enhance QoS in cloud computing.
- Initially the Binary encoding is utilized to initialize the population which sort out the jobs based on priority and on the virtual resources, the chromosome is positioned in the top row of genes (tasks). Then Best fit algorithm takes place at every time for all genes, which compares the best fit with iteration of each fitness values based on the computation time to reduce the makespan.
- After that heuristic crossover and mutation takes place for update the probability of current population with new population using the fitness value which reduced the failure rate.
- Thus the proposed heuristic based hybrid GA technique balanced the load as well effectively allocate the resources which enhanced the QoS performances.

The remainder of the article is divided into five sections: Section 1, which emphasizes the introduction, Section 2, which shows the relevant works for the current methodology, Section 3, which outlines the suggested methodology, Section 4, which highlights the results of the proposed methodology, and Section 5, which wraps up the discussion.

2. Related Works

Narendrababu et.al [17] discussed a task scheduler should modify its scheduling approach to account for fluctuating tasks and a changing environment. This study modifies the modified ant colony optimization (MACO) algorithm-based cloud work scheduling policy. The shortening of the make span and the execution of the multi-objective task scheduling (MOTS) technique are the key advantages of the suggested approach. Compared to the conventional ACO algorithm, the MACO method will reduce makespan and degree of imbalance, improving task scheduling performance. However, this can be particularly problematic for task scheduling problems, where the optimal solution may be difficult to identify.

Pirozmand et.al [18] Genetic Algorithm and Energy-Conscious Scheduling Heuristic based on the Genetic Algorithm were developed as a two-step hybrid technique for scheduling jobs conscious of energy and time. Tasks are prioritized in the first stage, and tasks are assigned to the processor in the second. We created main chromosomes and prioritized tasks before allocating them to the processor using the energy-conscious Energy Conscious Scheduling Heuristic model. The objective of the aforementioned problem's solution is to offer an efficient and ideal method of work scheduling with the intention of decreasing energy and time usage. Moreover, in some cases, maximizing energy efficiency can conflict with other objectives such as meeting task deadlines or minimizing resource utilization. ECS-based approaches may not always find the optimal balance between these competing objectives such as minimizing makespan.

Shi et.al [19] suggested a list scheduling-based task duplication and insertion method that integrates dynamic finish time prediction, job replication, and task insertion. To expedite task execution, the system replicates work over many nodes, reduces transmission time, and inserts tasks into free time slots. It also predicts the completion time of activities based on the scheduling of previously scheduled jobs. However, the increased communication overhead may result in

longer task completion times and reduced efficiency also duplication of tasks can lead to an increase in resource usage, which can result in higher energy consumption, reduced efficiency, and increased operational costs.

Izadkhah et.al [20] addressed static processor scheduling in homogenous computing systems, the learner genetic algorithm (LAGA) was proposed. Two learning criteria were designed using terminology like Steepest Ascent Learning Criterion and Next Ascent Learning Criterion in mind, keeping in mind the notions of punishment and reward for learning. As a result, develop a search strategy that is effective for addressing scheduling problems, ensuring that scheduling research progresses rationally while avoiding becoming trapped in local optimal. In order to shorten the makespan, the reuse idle time criteria are also taken into account during scheduling. However, parameter adjustment is still a significant flaw in the genetic algorithm.

Akintoye et.al [21] presented the issues with task distribution and virtual machine placement in a single cloud/fog computing environment and suggested algorithmic task distribution and genetic algorithm-based virtual machine placement (GABVMP) as fixes for the problem models. The Hungarian Algorithm Based Binding Policy (HABBP) load balancing approach for connecting cloudlets to virtual machines was initially proposed. This problem was first represented as a linear programming problem. After discussing the virtual machine placement challenge, a GABVMP was suggested as a viable resolution for the model's optimization. However, significant impact on the QoS offered by the cloud and the availability of cloud nodes.

Rajagopalan et.al [22] a hybrid Firefly-Genetic meta-heuristic method was suggested. The recommended method produces a potent metaheuristic search algorithm by fusing the benefits of an evolutionary algorithm, such as the Genetic algorithm, with a mathematical optimization technique, such as Firefly. The proposed hybrid Firefly-Genetic algorithm was effective at task scheduling with the objective of all tasks being completed in the lowest possible time and fast converging to the nearly optimal solution. However, the hybrid procedure can be computationally expensive, especially for large-scale problems. The increased computational cost can result in longer run times and higher resource utilization.

Pang et.al [23] offered an estimation of distribution algorithm (EDA) and GA-based hybrid scheduling method. (genetic algorithm). To begin with, a predetermined range of workable options is generated using the probability model and sampling technique of EDA. Second, the search space of solutions is expanded using the crossover and mutation GA procedures. Finally, the ideal work scheduling approach for virtual machines is accomplished. This algorithm offers the benefits of quick convergence and powerful search capabilities. The volatility and unpredictability of the cloud computing environment, however, are not taken into account in this work also the three elements of load balancing, cost, and job completion time are not balanced.

Aziza et.al [24] discussed cloud computing's scheduling of scientific workflows into consideration. The model's primary function is to lower the computational cost while meeting deadlines and budgets by optimizing the amount of time required to complete a group of interrelated jobs in the cloud. In order to simulate and enhance a workflow scheduling issue in cloud computing, a hybrid solution based on genetic algorithms was presented. The heterogeneous earliest finish time (HEFT) algorithmic model interferes with the process of creating the initial population. While designing procedures in cloud systems, there was the problem of data centers' consumption of power.

As a result, the utilization of all of the VMs by allocating tasks to each one in accordance with their computing capacities, which plays a critical role in a load-balancing of all the VMs. Load deviation is the objective function that must be minimized in the optimization issue of load balance. Hence a Heuristic based Algorithm is necessary to overcome the drawbacks in cloud computing.

3. Enhance the Load Balancing in Cloud Computing Using Heuristic Based Hybrid Genetic Algorithm

Cloud computing is popular for scalable computation and big data processing to get around the limitations of specific computers of networked intelligent devices, and it is increasingly convincing for the issue of an imbalanced workload of data centers at any given time caused by the unpredictability of the location and quantity of access devices. However, still the load balancing in virtual machines which is insufficient to satisfy an appropriate balance among customer request and cloud provider supply that leads to deteriorates the performance metrics such as make span, execution time, processors, memory, storage and energy consumption. Hence proposed a Heuristic based Hybrid Genetic Algorithm establish an appropriate schedule for the workflow using a genetic algorithm, which also improves the response time and load balancing. Here two evaluation functions are used to create the solutions; one function determines the importance of each task in the workflow based on how it affects the others, and the other function assesses the worth of the solutions that are generated. It focuses on reducing application completion times, failure rates, and increasing load balance topics. Each outcome is saved as a chromosome, and each chromosome is made up of N genes, according to its length. Each schedule in workflow scheduling takes the shape of a chromosome. The tasks and the relevant resources are listed on each table. According to data center information, the Binary Encoding Technique is used to allocate each job to a specific resource from a virtual list of available resources in order to generate a chromosome. To build the first population, the occupations will be arranged according to priority and put in the first row of genes on the chromosome. From a list of available virtual resources, the one with the quickest running time will be selected for each job. All genes go through the Best-Fit algorithm every time. The priority of tasks could be determined using an ordering mechanism that bases it on task influence. The Heuristic Crossover Technique uses the fitness values of the two parent chromosomes to decide the direction of the search. Then the mutation takes place to mutate each location with probability and it replaces the current population with a new

population of chromosome which balances the load as well as enhances the storage, memory and energy consumption such as make-span with Efficient Resource Allocation in cloud computing. The suggested heuristic-based hybrid Genetic Algorithm's work flow is shown in Fig.1 below. The description and formulation for task scheduling is described in the below section.

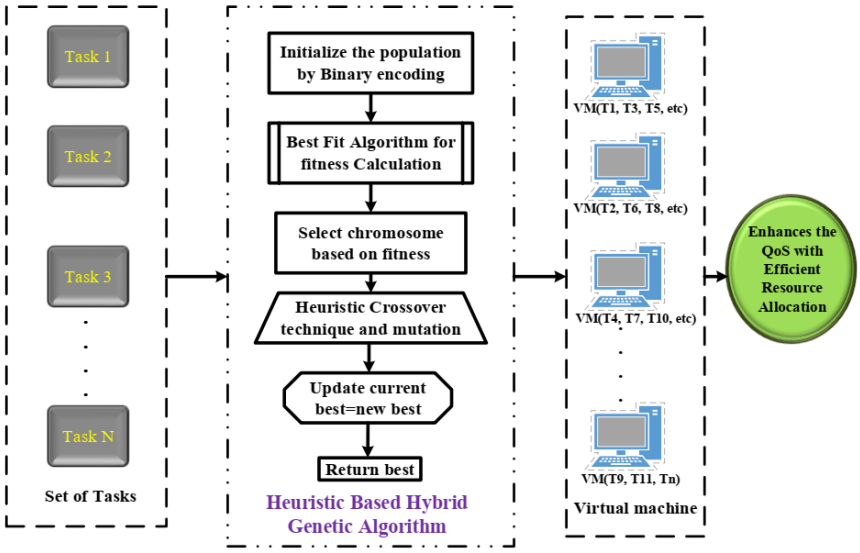

Fig.1. Work flow for the proposed heuristic based hybrid genetic algorithm

3.1. Description and Formulation for Task Scheduling

In this section, Under the IaaS concept, cloud resources are divided into several groups inside cloud data centers using a pool of virtualized homogenous and/or heterogeneous resources. A shared-time or shared-space VM scheduling strategy may be used to schedule one or more virtual machines (VMs) to operate in a pseudo-parallel fashion on every server in the cloud data centre. This virtualization layer provides resources with extremely high availability and scalability for cloud users. The main element of the cloud scheduling process, a data center broker, distributes user workloads to the data center's available resources. Submitted tasks are organized by the task management component, which also updates the person who submitted each work on its status. This happens when users send their cloudlets to the cloud provider. After receiving task requests from the task manager, the task scheduler uses the scheduling algorithm to allocate incoming jobs to suitable and available VMs. Using task and virtual machine (VM) information from the cloud information system (CIS), the task scheduler makes scheduling decisions. If VMs are not reachable for processing in this case, tasks will be transferred to the task queue. When users' cloudlets are being distributed across the available VMs in the cloud environment, the scheduling issue is raised. Such work is referred to be an NP-complete issue due to the variety of job needs and the dynamic nature of the heterogeneous resource availability in the cloud. With the cloud environment's increasing complexity, the scheduling problem gets increasingly complicated. In general, it is challenging to establish algorithms that generate the most effective task scheduling possibilities for cloud computing. The use of meta-heuristic algorithms to solve task scheduling has lately gained greater interest due to its capacity to deliver nearly optimum solutions in an acceptable period of time. The workloads are represented by jobs J, where each job $j_i \epsilon J$ is comprised of many tasks. Assume that there are m separate tasks $\{t_1, t_2, t_3, \ldots t_m\}$ with specific lengths of $l(t_i)$ in million instructions(MI) that have been submitted by cloud users and need to be processed. These tasks are distributed by the scheduler (broker) across a group of n cloud resources, designated as $VM_1, VM_2, VM_3, \ldots, VM_n$. The primary memory, storage, MIPS (million instructions per second) processing power, VM pricing, and number of cores are all different for each VM of these resources. The scheduling algorithm determines the duration of each task's execution using these configurations. Hence, the value of the task's expected execution time(EET) on a virtual machine is given as equation (1)

$$EET(t_i, VM_j) = \frac{l(t_i)}{total\ mips(VM_j)} \tag{1}$$

where l(ti) denotes the task's necessary processing load. Each VM's processing power is expressed in MIPS, where $total\ MIPS(VM_j) = number\ of\ cores(VM_j) * mips\ (VM_j)$. The datacenter broker creates the matrix based on the parameters of the VMs and cloudlets, and each element in the matrix represents the anticipated execution time (EETij) of task ti at VM. The equation (2) is given as follows

$$EET = \begin{bmatrix} EET_{11} & \dots & EET_{1n} \\ EET_{21} & \dots & EET_{2n} \\ \dots & \dots & \dots \\ EET_{m1} & \dots & EET_{mn} \end{bmatrix} \quad (2)$$

Each job may be done on any VM, and various VMs have varied processing capabilities, hence the cost C (ti, VMj) of processing task ti on different VMs will vary. The main goal of the task scheduling algorithm is to efficiently distribute tasks to the right resources in a way that maximizes one or more objectives. The main goal of the suggested strategy is to distribute submitted tasks across the available VMs in order to lower costs and Service Level (SL). Using π_{ij} as a binary decision variable, the scheduling issue of equation (3) may be expressed as follows:

$$\pi_{ij} = \begin{cases} 1, if\ task\ is\ submitted\ to\ VM \\ 0, Otherwise \end{cases} \quad (3)$$

The SL is computed using the following formula for every schedule solution ϕ is given as Equation (4)

$$SL(\emptyset) = \sum_{i=1}^{m} \sum_{j=1}^{n} EET(t_i, VM_j) * \pi_{ij} \quad (4)$$

One VM alone must receive each task submission. i.e. $\sum_{j=1}^{n} \pi_{ij} = 1\ \forall i \in \{1,2,3 \dots \dots, m\}$. All tasks assigned to a virtual machine (VMj) have memory requirements mem(ti) that do not exceed the virtual machine's available memory. i.e. $\sum_{i=1}^{m} mem(t_i) * \pi_{ij} \leq \left(VM_{j_{mem}}\right) \forall j \in \{1,2,3 \dots \dots, n\}$. The total MIPS of the virtual machine (VMj) must have a processing power that is lower than or equal to the required load for all jobs allocated to that VMj at once i.e. $\sum_{i=1}^{m} l(t_i) * \pi_{ij} \leq total\ mips\ \left(VM_j\right) \forall j \in \{1,2,3 \dots \dots, n\}$. Any virtual machine that fulfils the job's criteria can be used to execute any task; as a result, the VM's capabilities determine the rate at which a task can be completed. Each job must be completed without interruption since the tasks are non-preemptive. The same VM may have several tasks assigned to it at once. The next part will describe the hybrid genetic algorithm based on heuristics.

3.2. Heuristic based Hybrid Genetic Algorithm

In this section, a hybrid genetic algorithm based on heuristics is suggested for VM load balancing, which distributes resources in cloud computing. The suggested hybrid GA's input is a series of tasks that are assigned to a virtual machine, and the jobs are scheduled based on a population initialization approach using binary encoding. Then Best fit algorithm is performed for allocate the resources based on fitness function and heuristic crossover and mutation takes places to update the new chromosome based on the VM. The population initialization via binary encoding is describing in the forthcoming section.

A. Population Initialization via Binary Encoding Technique

In this section, Binary encoding and floating point number coding are the two methods that genetic algorithms can encode chromosomes. Binary encoding offers a higher search ability than floating point number coding, while having faults like continuous function discretization mapping errors and individual coding strings that might not be accurate enough right away. The discrepancy was more noticeable as the population increased. Hence, the chromosomal encoding used in this research employs binary encoding, uses a two-dimensional array for storage, and assigns tasks to rows and computational resources to columns. Each genomic locus has a value of 0 or 1. Chromosome[i][j]=1 designates the ith job to be carried out on the jth computing resource (VM). Decoding will reveal that the first task is carried out on the first computing resource(VM), on the second computing resource (VM), the second job is completed and so on and so forth the chromosome of binary value is formed. The chromosomes and EET matrix may be used to estimate the time it takes for each computing resource (such as a virtual machine) to complete all of its tasks. The total amount of time estimated above is the amount of time needed to perform all the jobs. Thus, the overall scheduling completion time is given as equation (5)

$$Complete\ time\ Scheduling = \max_{0 \leq j \leq m} \{\sum_{i=0}^{n} EET[i][j] \times chromosome[i][j]\} \quad (5)$$

Every solution has a chromosome-like representation. N genes make up each chromosome, where N is the length of the chromosome. Each schedule in workflow scheduling appears as a chromosome. With random generated population size individuals, the initial population has been generated. Each individual is made up of an encoded solution, where each task is given to a processor at random. After the jobs have been prioritized, they will be placed in the first row of genes on the chromosome, and the initial population will be produced using an appropriate resource that has been selected from a list of virtual resources with the quickest running times for each work. Then the selection process will be discussing in the below section

B. Best Fit Algorithm and Selection

As Best-Fit, this procedure is performed for all genes. In this approach, the algorithm selects the best resource for each gene's first chromosome from the virtual list's first place, but for its second chromosome from the virtual list's second position. Following the last chromosome's start from the last position in the virtual list and continuing in this manner, the best candidate-resource will be sought out using the Round Robin approach, but for resources instead of chromosomes. The value of a solution should be assessed using a fitness function with efficient parameters. According to the parameters allocation approach in GA, a solution with the best value is obtained as the minimum or maximum for the fittest solution. All solutions are subjected to the fitness function, which determines their values. Each solution's fitness value is calculated using equation (6)

$$Fitness\ F = \sum_{i=1}^{m} q_i^t \times q_i^{fr} \tag{6}$$

Where q_i^t denotes the time when resource qi completes all of the workflow tasks that are currently scheduled. The entire amount of time that each resource has been used for the present application, as measured by the makespan, is taken into account in this function.

For every individual in the population of the current generation, the selection procedure is carried out to pair them up to generate the subsequent generation. The method of choosing a roulette wheel with random acceptance is used to pick a number of chromosomes for the local search algorithm. There are two main phases in this procedure. In the first stage, a chromosome is randomly selected from the population. The selection probability (1/n) is used. The next step is to assess the chromosomal fitness value and decide if the locally selected chromosome is a member of the elite of the population. If not, the chosen chromosome is rejected, and the procedure is repeated. Finally, based on the computing time of the resources decreased makespan, the suggested Genetic based Best Fit method and selection operation is carried out. Then the Heuristic crossover technique will be discussed on the below section.

C. Heuristic Crossover Technique

In GAs, the crossover operator is crucial for changing the population chromosomes. Population evolution in GAs is likewise influenced by the crossover operator. The operator joins several chromosomes to form a new generation of chromosomes. Certain characteristics are passed down from one parent while others come from the other. The crossover procedure is performed on the remaining population to create the progeny for the following generation. Perform a single-point crossover operation on the two-parent chromosomes in order to produce new children from the residual population and determine which chromosome is more likely to be present in the upcoming generation. A single location on both parent chromosomes is chosen, and values (genes) are exchanged between the two parents at that location to produce the new child. In each iteration of the method, a pair of Physical Hosts (of VMs) from the current generation were repeatedly chosen, and the crossover operator with probability was then used to create a new VM from the selected VM. At last, the heuristic cross over technique iterated and updated the new value. Then the mutation will be described in the below section.

D. Mutation

Mutation is a genetic operator that aims to preserve genetic diversity in the population by changing a few random VMs in a physical host. Numerous chromosomes are altered using the mutation operator after the combination operator in order to avoid local optimum convergence and produce variety in the population. A new chromosome is produced by randomly selecting a mutation site and replacing the data from that gene with the data from the first independent gene (task) after it. There are two additional genes scheduled for mutational processes. It may be feasible to improve the algorithm's outcome by avoiding local minima. Additionally, a resource will be chosen at random from an online database of materials, along with a chromosome and one of its genes. If a gene's failure rate is lower than that of the previous candidate resource, it will take the place of the chosen resource. However, this situation has additional requirement. The resource under consideration is not currently under a lot of workload. The formula may be used to calculate a resource's failure factor is given as equation (7)

$$q_i^{ffr} = {q_i^{fr}}\Big/{q_i^{mips}} \tag{7}$$

where q_i^{mips} stands for the resource's processing capability expressed in terms of the frequency of machine instructions in a second and q_i^{fr} is the probability of task failure on resource qi per time unit as resource failure rate. In a few instances, the process of choosing and evaluating new candidate resources will be repeated for improved results. This mutation technique quickly finds a workable solution since the mutation process ensures that GA does not halt at the local minimum. By using crossover and mutation processes to reduce the fitness value as much as feasible, the algorithm attempts to find the solution.

Finally, the Proposed heuristic based hybrid genetic algorithm initialized the population via binary encoding, then fitness was calculated by best fit algorithm which compares the best fit with iterated fit value based on the computation

time reduced the makespan and the crossover and mutation takes place to update the current value with new value which reduced the failure rate. Hence the overall proposed method balanced the load on VM which effectively allocated the resources. The Fig.2 represents the flow Chart and the Algorithm for the proposed Heuristic based hybrid GA is given below. The section below discusses the results of the suggested technique.

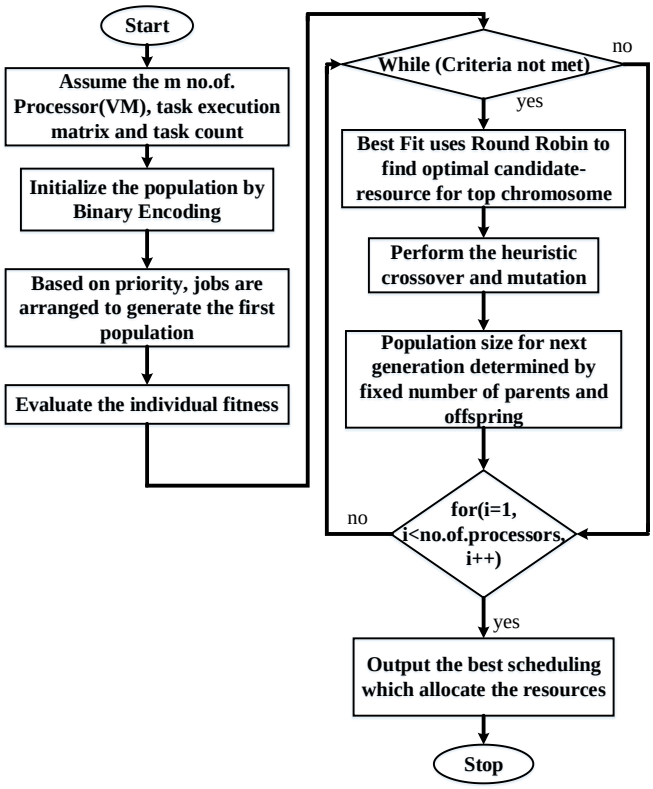

Fig.2. Flow chart for the proposed heuristic based hybrid genetic algorithm

Algorithm: Heuristic based Hybrid Genetic algorithm

Input :Task list and VM list

Output: Task Scheduling with resource allocation

Step 1: Initialize the population via Binary encoding
Step 2: Set the priority for each task
Step 3: Generate the population in the first row of Genes(tasks)
Step 4: Evaluate the individual fitness value
Step 5: While(Criteria not met)
Step 6: Based on the execution time, determine the resources that are best fit for each task.
Step 7: Select the fitness value in the current population
Step 8: New population fitness<current population fitness
Step 9: Best fit= new population fitness
Step 10: else
Step 11: Best fit=current population
Step 12: Go to step 3
Step 13: Perform heuristic crossover technique
Step 14: Select the random gene crossover
Step 15: For each offspring do
Step 16: Perform mutation
Step 17: Determine the best individual among offspring (new best)
Step 18: For new best is better than current best individual then
Step 19: Replace current best=new best
Step 20: Return best individual

4. Simulation Results and Discussion

The proposed technique outcomes, performance metrics and comparative analysis was provided in this section. The proposed heuristic based hybrid genetic algorithm was implemented by py-sim tool. The suggested solution makes the assumption that there will be 150 tasks and 16 resources (VMs) in the cloud. The task and VM were chosen because of the following reasons.

- Resource Efficiency: The assumption of 150 jobs and 16 resources (VMs) enables optimized resource allocation, guaranteeing that each task has sufficient processing power without wasting any of it.
- Predictable Performance: The system's performance may be predicted and maintained more easily with a constant task and resource count, resulting in robust and dependable cloud computing services.
- Simplified Management: System management is made simpler by using a predetermined set of activities and resources, which also makes it less difficult to scale resources dynamically in response to changing workloads.

The parameters consist of proposed heuristic based Hybrid GA utilizing the population size is 10, crossover operator rate is 75, mutation rate is 5 and the maximum number of iteration is 100. Fig.3 below shows the outcome of the suggested approach for the job assigned to the VM with processing time.

```
Table of VM tasks:
VM 0: 8 83 96 124 132 137 141
VM 1: 4 13 17 55 73 138
VM 2: 5 18 31 44 48 56 69 82 136
VM 3: 3 14 15 24 35 49 78 101 118
VM 4: 7 23 33 74 77 97 99 105 123 140 142 149
VM 5: 1 6 43 50 58 60 76 89
VM 6: 2 12 42 106 116 119 129 130
VM 7: 11 20 21 22 29 45 51 61 64 87 114 121 127
VM 8: 36 63 67 91 145 148
VM 9: 65 75 92 103 128 134
VM 10: 9 26 41 71 81 84 108 117 135
VM 11: 46 54 72 85 102 120 126 133 143
VM 12: 25 27 40 47 86 88 95 98 100 115 146
VM 13: 52 59 62 68 93 104 110 112 147
VM 14: 16 19 28 34 39 53 57 66 70 80 90 94 107 109 125 131 139
VM 15: 0 10 30 32 37 38 79 111 113 122 144
Overall processing time: 30.828000000000003s
```

Fig.3. The Proposed method output for task allocation to VM

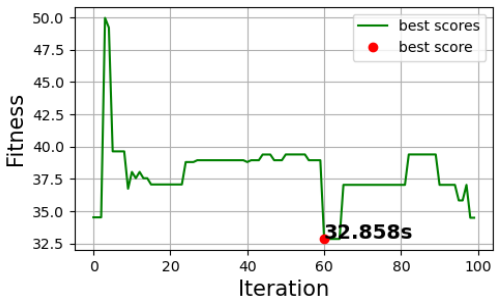

Fig.4. The proposed method output for best fit algorithm

From Fig.4 An initial assessment of fitness values within the first population is part of the proposed heuristic-based hybrid genetic algorithm. The method uses a round-robin technique to find the best fitness values over 100 iterations based on computing time. In each iteration the proposed method compares the best fitness value with previous fitness value and attained the less fitness values as 32.858s which reduced the makespan.

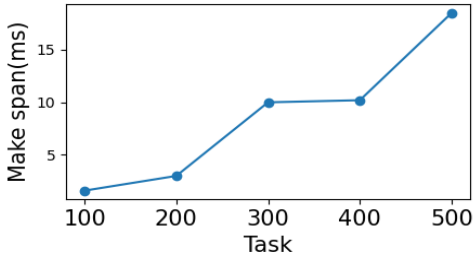

Fig.5. The proposed method output for makes pan using fitness

From Fig.5 the proposed heuristic based hybrid GA calculates the makespan using the fitness function. The approach guarantees effective task scheduling by minimizing the makespan, which results in shorter completion times and better resource use. As the number of tasks rises, so does the makespan. The makespan for 100 tasks attained 2ms, for 200 tasks attained 3.5ms, for 300 tasks attained 11.5ms, for 400 tasks attained 10ms and for 500 tasks attained 18ms.

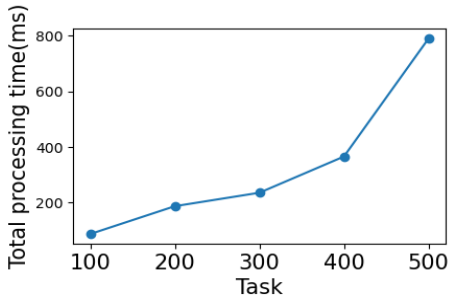

Fig.6. Total processing time output for the proposed method

Fig.6 shows that the suggested method's overall processing time for the resources used to assign the jobs to VM. The technique ensures better system performance and more efficient job execution by utilizing a combination of genetic operators and heuristics to minimize total processing time. For each task the total processing time for the proposed heuristic based hybrid GA attained the high value such as for 100 tasks attained 80ms, for 200 tasks attained 198ms, for 300 tasks attained 250ms, for 400 tasks attained 380ms and for 500 tasks attained 800ms.

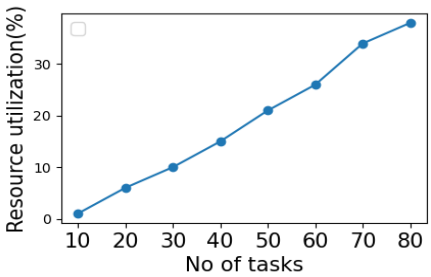

Fig.7. Resource utilization output for the proposed method

Fig.7 depicts that the resource utilization for the proposed heuristic based hybrid GA. The graph reveals that each physical host's resource use varies drastically before load balancing. It indicates that when demand is at its highest, resources with high capabilities show up more frequently. The resource fluctuations are significantly less after carrying out the load balancing using the proposed heuristic based hybrid GA. By moving virtual machines between physical hosts, the total resource utilisation becomes considerably more steady and balanced. The resource utilisation achieved for 2% for 10 tasks, 7% for 20 tasks, 11% for 30 tasks, 15% for 40 tasks, 22% for 50 tasks, 27% for 60 tasks, 34% for 70 tasks and 39% for 80 tasks. The proposed method attained the high resource utilization for increasing number of tasks in cloud computing.

The proposed heuristic-based hybrid GA technique evaluates performance measures such makespan, total processing time, failure rate, energy consumption, and resource utilization. The performance metrics are describing in below

The amount of time the system spends on a job is known as its total execution time, including the time that it spends performing run-time or system functions on its behalf.

The amount of time needed to complete all tasks in the specific order is known as the makespan.

$$makespan \geq \sum_{j \in J} q_j a_{ij}$$

Where q_j is the processing time and a_{ij} is the job assigned to the VM. The expected number of times that a job will fail over the duration of a certain amount of time is known as the failure rate.

$$Failure\ rate = {}^{R}/_{T}$$

where T is the total time and R is the number of jobs that failed.

Resource utilization is the measure of how much of the available resources are being used at any one time. Using resources efficiently is essential for cloud services to conserve energy. In a cloud environment, the term "average energy consumption" refers to the typical quantity of energy used by each resource that is assigned to a certain job.

To contrast the proposed heuristic-based hybrid GA method's QoS performance metrics with those of earlier techniques like Ant Colony Optimization (ACO) [25] and Particle Swarm Optimization (PSO) [26], Genetic algorithm (GA) [27], Hyper-heuristic [28], Node duplication Genetic Algorithm (NGA) [19], Learning based genetic Algorithm (LAGA) [20], Genetic Algorithm Based Virtual Machine Placement (GABVMP) [21], first fit [29], and Random [30].

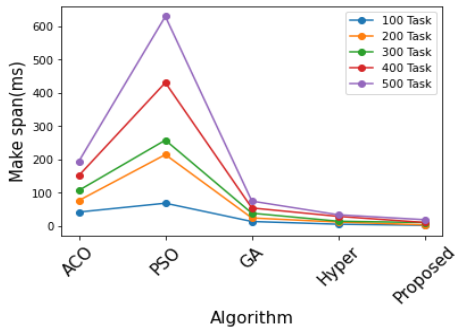

Fig.8. Comparison graph for Makespan

From Fig.8 depicts that the proposed method comparison graph for makespan. The graph reveals that the proposed method makespan attained 2ms to 18ms for 500 allocated tasks when related to the prior methods. The suggested approach reduced the makespan for 500 jobs assigned to the VM. The Table 1 shows the makespan comparison for proposed method with prior method as given below

Table 1. Makespan(ms) comparison for proposed method with prior method

Tasks	ACO	PSO	GA	Hyper Heuristic	Proposed
100	41.43	68.03	13.11	3.99	2
200	76.05	213.95	23.71	4.86	3.5
300	106.49	257.46	37.95	13.67	11.5
400	150.93	430.16	53.72	13.69	10
500	193.6	628.93	74.35	24.45	18

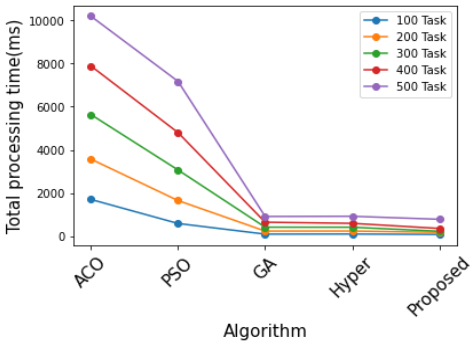

Fig.9. Comparison graph for total processing time

From Fig.9 depicts that the proposed method comparison graph for total processing time. The graph reveals that the proposed method processing time attained 80ms to 800ms for 500 allocated tasks when related to the prior methods. When compared to the previous technique, the suggested method satisfied the minimal processing time for 500 tasks assigned to the VM. The processing time comparison between the suggested approach and the earlier method is shown in Table 2 and is as follows.

Table 2. Total processing time (ms) comparison for proposed method with prior method

Tasks	ACO	PSO	GA	Hyper Heuristic	Proposed
100	1708.821	595.9126	102.8686	99.36405	80
200	3575.209	1656.824	244.5655	239.6441	198
300	5637.652	3072.283	423.9843	418.2265	250
400	7886.367	4802.7	654.2516	571.8476	380
500	10,190.08	7173.798	920.9659	904.3571	800

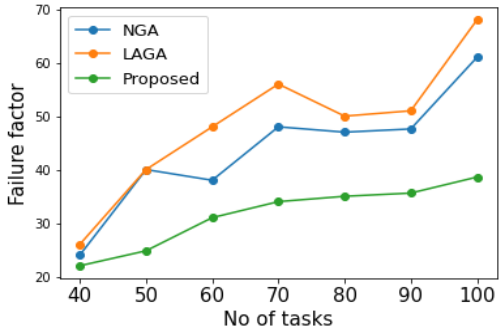

Fig.10. Comparison graph for failure factor

From Fig.10 shows that the proposed method comparison graph for the failure factor. The algorithm lessens the possibility of being stuck in local optima by adding variation and originality to the population through the use of mutation operators. The graph reveals that the proposed method failure rate attained 22 to 39 for 100 allocated tasks when associated to the prior methods. In comparison to the previous way, the suggested solution achieved a low failure rate for 100 jobs assigned to the VM. The failure rate comparison between the suggested approach and the earlier method is shown in Table 3 and is as follows.

Table 3. Failure rate comparison for proposed method with prior method

Tasks / Methods	40	50	60	70	80	90	100
NGA	23	40	35	48	46	45	62
LAGA	25	40	48	53	47	50	69
Proposed	22	25	28	31	33	36	39

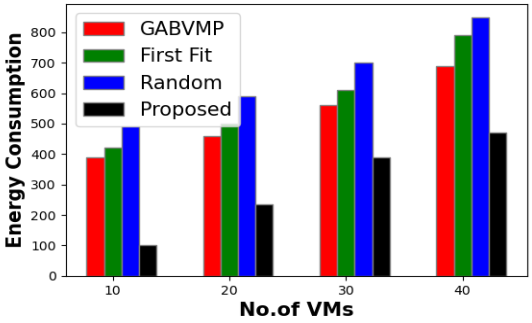

Fig.11. Comparison graph for energy consumption

From Fig.11 depicts that the proposed method comparison graph for energy consumption. The graph shows that as the number of VMs climbed, so did the resource utilization's energy usage. The method indirectly helps to reduce energy usage by making optimal use of resources and shortening makespan. The proposed technique required 80J for 10 VMs, 270J for 20 VMs, 380J for 30 VMs, and 490J for 40 VMs in terms of energy usage. Even though, compared to the prior method such as Random Placement approach, GABVMP and First Fit Placement, the proposed hybrid GA used less energy. This is due to the fact that GA required less VMs than GABVMP, Random Placement, and First Fit Placement to allocate a given number of VMs. The table 4 depicts that the energy consumption comparison for the proposed method with prior method is shown below

Table 4. Energy consumption(J) comparison for the proposed method with prior method

No.of.VMs	GABVMP	Random Placement approach	First Fit Placement	Proposed
10	390	410	500	80
20	480	510	580	270
30	560	630	700	380
40	690	790	850	490

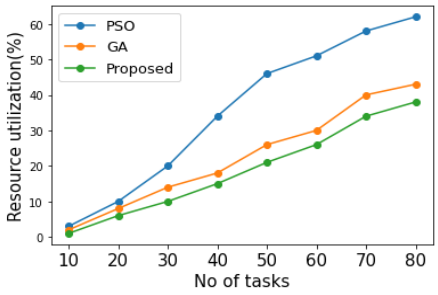

Fig.12. Comparison graph for resource utilization

Fig.12 depicts that the Proposed method comparison graph for Resource utilization. The graph reveals that the proposed method utilized 1% to 39% resources for 80 tasks allocated. The proposed method utilized less resources when compared to prior method. The table 5 depicts the proposed method comparison for resource utilization with prior method is shown below

Table 5. Comparison for resource utilization (%) of proposed method with prior method

Tasks / Methods	10	20	30	40	50	60	70	80
PSO	3	10	21	35	47	50	58	63
GA	2	8	14	19	27	32	42	45
Proposed	1	5	12	15	22	28	34	39

From the above analysis of previous method such as ACO, PSO, GA and Hyper heuristics, NGA, LAGA, GABVMP, first fit, and Random approach has issues as the high computational complexity, sensitivity to parameter settings, computational overhead, limited scalability, limited diversity, parameter tuning, inadequate optimization and ineffective resource use. Therefore, the proposed method refined the search solutions depends on the fitness function which attained the ability to handle the complex problems and improved the better Quality of services attained less makespan as 2ms, total processing time as 80ms, failure rate as 39 for 100 tasks and energy consumption of 80J for 10 tasks respectively. The proposed method attained less makespan, total processing time, failure rate and energy consumption when compared to previous method gives better QoS parameters with efficiently allocated the resources.

5. Conclusions

In this article, the number of tasks and virtual machines are assigned for the analysis, the proposed Heuristic-Based Hybrid Genetic Algorithm presents a promising approach for optimizing workflow scheduling in cloud computing environments. By leveraging genetic algorithms and employing two evaluation functions, this method addresses critical concerns such as response time improvement, load balancing, and overall resource utilization efficiency. The use of chromosomes to represent scheduling solutions, along with the Binary Tree Encoding Technique for task-resource assignment, enhances the algorithm's effectiveness. Prioritizing tasks based on their impact and utilizing heuristics for crossover and mutation enable the algorithm to strike a balance between load distribution and resource efficiency. Specifically, the outcomes of the proposed method achieved a smaller makespan of 2ms, failure rate of 39, execution time of 80ms for 100 tasks, energy consumption of 80J for 10 tasks and resource utilization obtained 1% to 39% when related to the prior technique The proposed approach holds great potential for reducing application completion times, minimizing failure rates, and enhancing the overall performance of cloud-based applications, contributing to more efficient resource allocation and management in cloud computing environments. Future research should focus on improving the algorithm's scalability, dynamically adjusting workloads, and incorporating energy-efficient tactics to reduce energy consumption and achieve performance objectives in data centers.

References

[1] Geetha, P., & Robin, C. R. (2021). RETRACTED ARTICLE: Power conserving resource allocation scheme with improved QoS to promote green cloud computing. Journal of Ambient Intelligence and Humanized Computing, 12(7), 7153-7164.

[2] Haji, L. M., Zeebaree, S., Ahmed, O. M., Sallow, A. B., Jacksi, K., & Zeabri, R. R. (2020). Dynamic resource allocation for distributed systems and cloud computing. TEST Engineering & Management, 83(May/June 2020), 22417-22426.

[3] Ajmal, M. S., Iqbal, Z., Khan, F. Z., Ahmad, M., Ahmad, I., & Gupta, B. B. (2021). Hybrid ant genetic algorithm for efficient task scheduling in cloud data centers. Computers and Electrical Engineering, 95, 107419.

[4] Kumar, T. S. (2019). Efficient resource allocation and QOS enhancements of IoT with FOG network. Journal of ISMAC, 1(02), 101-110.

[5] Thein, T., Myo, M. M., Parvin, S., & Gawanmeh, A. (2020). Reinforcement learning based methodology for energy-efficient resource allocation in cloud data centers. Journal of King Saud University-Computer and Information Sciences, 32(10), 1127-1139.
[6] Patil, K. (2022). Hybrid Genetic Algorithm and Modified-Particle Swarm Optimization Algorithm (GA-MPSO) for Predicting Scheduling Virtual Machines in Educational Cloud Platforms. International Journal of Emerging Technologies in Learning, 17(7).
[7] Wu, X., Wang, H., Wei, D., & Shi, M. (2020). ANFIS with natural language processing and gray relational analysis based cloud computing framework for real time energy efficient resource allocation. Computer communications, 150, 122-130.
[8] 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.
[9] Gao, X., Liu, R., & Kaushik, A. (2020). Hierarchical multi-agent optimization for resource allocation in cloud computing. IEEE Transactions on Parallel and Distributed Systems, 32(3), 692-707.
[10] Supreeth, S., Patil, K., Patil, S. D., Rohith, S., Vishwanath, Y., & Prasad, K. S. (2022). An Efficient Policy-Based Scheduling and Allocation of Virtual Machines in Cloud Computing Environment. Journal of Electrical and Computer Engineering, 2022.
[11] Sharma, M., Kumar, M., & Samriya, J. K. (2022). An optimistic approach for task scheduling in cloud computing. International Journal of Information Technology, 14(6), 2951-2961.
[12] Sefati, S., Mousavinasab, M., & Zareh Farkhady, R. (2022). Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: performance evaluation. The Journal of Supercomputing, 78(1), 18-42.
[13] Hormozi, E., Hu, S., Ding, Z., Tian, Y. C., Wang, Y. G., Yu, Z. G., & Zhang, W. (2022). Energy-efficient virtual machine placement in data centres via an accelerated Genetic Algorithm with improved fitness computation. Energy, 252, 123884.
[14] Hung, L. H., Wu, C. H., Tsai, C. H., & Huang, H. C. (2021). Migration-based load balance of virtual machine servers in cloud computing by load prediction using genetic-based methods. IEEE Access, 9, 49760-49773.
[15] Zhang, B., Wang, X., & Wang, H. (2021). Virtual machine placement strategy using cluster-based genetic algorithm. Neurocomputing, 428, 310-316.
[16] Kazeem Moses, A., Joseph Bamidele, A., Roseline Oluwaseun, O., Misra, S., & Abidemi Emmanuel, A. (2021). Applicability of MMRR load balancing algorithm in cloud computing. International Journal of Computer Mathematics: Computer Systems Theory, 6(1), 7-20.
[17] Narendrababu Reddy, G., & Phani Kumar, S. (2019). Modified ant colony optimization algorithm for task scheduling in cloud computing systems. In Smart Intelligent Computing and Applications: Proceedings of the Second International Conference on SCI 2018, Springer Singapore, 1, 357-365.
[18] Pirozmand, P., Hosseinabadi, A. A. R., Farrokhzad, M., Sadeghilalimi, M., Mirkamali, S., & Slowik, A. (2021). Multi-objective hybrid genetic algorithm for task scheduling problem in cloud computing. Neural computing and applications, 33, 13075-13088.
[19] Shi, L., Xu, J., Wang, L., Chen, J., Jin, Z., Ouyang, T., & Fan, Y. (2021). Multijob associated task scheduling for cloud computing based on task duplication and insertion. Wireless Communications and Mobile Computing, 2021, 1-13.
[20] Izadkhah, H. (2019). Learning based genetic algorithm for task graph scheduling. Applied Computational Intelligence and Soft Computing, 2019.
[21] Akintoye, S. B., & Bagula, A. (2019). Improving quality-of-service in cloud/fog computing through efficient resource allocation. Sensors, 19(6), 1267.
[22] Rajagopalan, A., Modale, D. R., & Senthilkumar, R. (2020). Optimal scheduling of tasks in cloud computing using hybrid firefly-genetic algorithm. In Advances in Decision Sciences, Image Processing, Security and Computer Vision: International Conference on Emerging Trends in Engineering (ICETE), Springer International Publishing, 2, 678-687.
[23] Pang, S., Li, W., He, H., Shan, Z., & Wang, X. (2019). An EDA-GA hybrid algorithm for multi-objective task scheduling in cloud computing. IEEE Access, 7, 146379-146389.
[24] Aziza, H., & Krichen, S. (2020). A hybrid genetic algorithm for scientific workflow scheduling in cloud environment. Neural Computing and Applications, 32, 15263-15278.
[25] Li, G., & Wu, Z. (2019). Ant colony optimization task scheduling algorithm for SWIM based on load balancing. Future Internet, 11(4), 90.
[26] Agarwal, M., & Srivastava, G. M. S. (2019). A PSO algorithm based task scheduling in cloud computing. International Journal of Applied Metaheuristic Computing (IJAMC), 10(4), 1-17.
[27] Pirozmand, P., Javadpour, A., Nazarian, H., Pinto, P., Mirkamali, S., & Ja'fari, F. (2022). GSAGA: A hybrid algorithm for task scheduling in cloud infrastructure. The Journal of Supercomputing, 78(15), 17423-17449.
[28] Gupta, A., Bhadauria, H. S., & Singh, A. (2021). RETRACTED ARTICLE: Load balancing based hyper heuristic algorithm for cloud task scheduling. Journal of Ambient Intelligence and Humanized Computing, 12(6), 5845-5852.
[29] Shaw, R., Howley, E., & Barrett, E. (2019). An energy efficient anti-correlated virtual machine placement algorithm using resource usage predictions. Simulation Modelling Practice and Theory, 93, 322-342.
[30] Rengasamy, R., & Chidambaram, M. (2019, March). A novel predictive resource allocation framework for cloud computing. In 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS) IEEE, 118-122.

Authors' Profiles

Ms. Shubhangi Pandurang Tidake is currently pursuing her Doctoral Programme in Sant Gadge Baba Amravati University, Amravati,Maharashtra. She holds an MSc Computer Science degree from Pune University along with NET qualified. Her field of specialization is Cloud Computing and Artificial Intelligence. She has published 3 articles in reputed peer-reviewed National and International Journals. She has presented the research papers in various Conferences at National and International level.

Dr. Pramod N. Mulkalwar is presently working as Principal in MNG Science college, Bhabhulgaon, Yawatmal, Maharashtra. He holds PhD in Computer Science with MSc and MPhil qualification. He has published 40 articles in reputed peer-reviewed National and International Journals and 3 chapters in Edited Books. He has attended/presented the research papers in various Seminars, Conferences and Workshop at National and International level. He has made significant contributions to research in the field of Cloud Computing and Data Mining.

How to cite this paper: Shubhangi Pandurang Tidake, Pramod N. Mulkalwar, "Efficient Resource Allocation to Enhance the Quality of Service in Cloud Computing", International Journal of Computer Network and Information Security(IJCNIS), Vol.17, No.1, pp.68-81, 2025. DOI:10.5815/ijcnis.2025.01.06