National College of Ireland

Energy Efficient Task Scheduling Approach in Cloud Environments towards Green Cloud Computing

MSc Research Project
Cloud Computing

Shiva Ram Raja Gandhi Raja
Student ID: x21219095

School of Computing
National College of Ireland

Supervisor: Mr. Rashid Mijumbi

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Shiva Ram Raja Gandhi Raja
Student ID:	x21219095
Programme:	Cloud Computing
Year:	2022
Module:	MSc Research Project
Supervisor:	Mr. Rashid Mijumbi
Submission Due Date:	14/8/2023
Project Title:	Energy Efficient Task Scheduling Approach in Cloud Environments towards Green Cloud Computing
Word Count:	5537
Page Count:	19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Shiva Ram Raja Gandhi Raja
Date:	17th September 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Energy Efficient Task Scheduling Approach in Cloud Environments towards Green Cloud

Shiva Ram Raja Gandhi Raja
x21219095

Abstract

Assigning and distributing computing resources in an environment of the cloud is known as task allocation or scheduling. Despite the efficiency and effectiveness in an environment of cloud computing, task allocation or scheduling is still a challenge for the same. While task scheduling is responsible for the distribution of load evenly during the mapping of resources, there are multiple challenges faced during the same. While there have been multiple studies on cloud computing and task scheduling, the existing studies have less focus on the effective utilization of resources that is compute, storage, and network capacities. And therefore, failed to focus on the implementation of task scheduling algorithms. The goal of the current thesis is to concentrate on a hybrid GA (Genetic Algorithm) and PSO (Particle Swarm Optimization) which is a GA-PSO work scheduling technique. This GA-PSO scheduling technique efficiently distributes the jobs across the resources. The suggested technique incorporates characteristics of modified GA-PSO algorithms to shorten the makespan, shorten the execution and turnaround time, and to decrease the communication and execution cost. Using the CloudSim framework, the efficiency of the GA-PSO approach will be compared to that of the conventional PSO algorithm. The results show that by conserving time, money, and resources, the Hybrid GA-PSO technique speeds up the convergence to an ideal solution. The proposed technique has overall enhancement and its performance in terms of execution cost as well as makespan.

1 Introduction

A ground-breaking paradigm for providing customers with on-demand computing services as well as resources over the internet, the environment of cloud computing has evolved. The inbuilt scalability and flexibility of cloud computing have helped it become extremely popular across a wide range of businesses. Balancing cost containment with efficient utilization of cloud resources is still a difficult task for cloud service providers. Creating a task scheduling approach that effectively distributes computer resources to workloads while reducing operational costs and time is one of the crucial issues in cloud computing. As cloud-service providers work to offer premium services at reasonable costs, cost-effective scheduling is crucial.

Task scheduling in cloud computing has its own challenges the reason being the dynamic and distributed nature of the environment of cloud computing. Key challenges during task allocation or scheduling in cloud computing environment are addressed as

below:

- Various computer resources with a range of capabilities and performance characteristics are often found in cloud data centers. Scheduling jobs to the best available resources while taking CPU, memory, storage, and network bandwidth into account presents a difficult task.
- Workloads in the cloud are extremely dynamic, with variable rates of task arrival and resource requirements. For schedulers to efficiently manage changing workloads, they must be able to adjust in real-time.
- Cloud infrastructures are made to be extremely scalable so that many users and processes can run at once. In order to address the growing scale of cloud deployments, scheduling algorithms must be scalable. M.S. Sudheer (2019)
- Multiple users are served simultaneously by cloud data centers, which cause resource conflict between various tasks. Making sure that resources are distributed fairly and preventing performance deterioration because of multi-tenancy is a complex task.
- Different requirements of QOS (Quality of Service), such as throughput, response time and data consistency, apply to various jobs and applications. User pleasure depends on meeting these QoS standards while scheduling tasks. Sangwan and Dhanda (2019)
- Scheduling tasks can have an impact on energy usage, which is a big energy user in cloud data centers. It can be difficult to optimize scheduling choices for lower energy consumption without sacrificing performance. Dong and Rojas-Cessa (2015)
- Failures in cloud infrastructures are common, including node breakdowns and network outages. For high availability and dependability, task scheduling must be fault-tolerant, allowing jobs to be redelegated to alternate nodes in the event of a failure. Varanasi (2020)
- Cloud service companies strive to match user needs while reducing operational expenses. A difficult trade-off is finding a balance between resource use, energy use, and infrastructure maintenance expenses.

Cloud data centers offer Infra resources as a service. The physical systems servers, storage, network systems for the data center, and software programs, operating systems, and management tools that the data center provides as Infrastructure as a Service (IaaS) and Platform as a Service (PaaS), respectively. Programs like web search, social media, computation, etc. are offered as Software as a Service (SaaS) by means of cloud computing data centers. Calheiros et al. (2011) These applications operate on virtual machines, which are virtualized IT resources offered by IaaS and PaaS. Cloud service providers fulfill requests with resources, such as various kinds of VMs, based on the request. Madni et al. (2016)

Figure 1 describes how a cloud-based computing infrastructure typically operates,

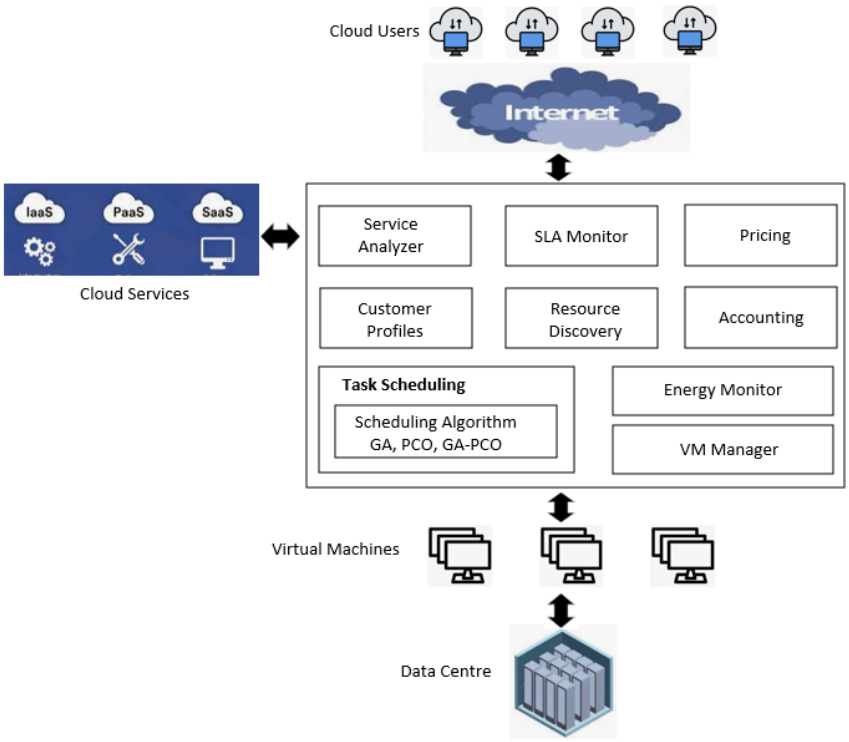

Figure 1: Cloud Infrastructure

Cloud infrastructure is the backbone of cloud computing services, providing physical and virtual resources to support various applications and services. It comprises data centers, virtualization, resource pooling, orchestration and automation, self-service portals, networking, security measures, scalability, monitoring and performance optimization, and redundancy and disaster recovery. Data centers house physical hardware components, virtualization technology enables multiple virtual instances on a single server, resource pooling allows for better resource utilization, orchestration and automation tools manage resources efficiently, self-service portals enable users to select and provision computing resources on-demand, and networking components facilitate communication between elements. Security measures, scalability, monitoring, and redundancy and disaster recovery mechanisms ensure data resilience and high availability. Overall, cloud infrastructure revolutionizes the way businesses and individuals access and utilize computing power, storage, and applications, enabling cost-effective and efficient solutions for various use cases. IaaS resource scheduling optimization strategies are used to either improve operating system performance or customer satisfaction. Certain types and hybrid categories of clouds, which are frequently handled throughout the resource management process in the figure below, can be used to explain task scheduling schemes. These systems are further separated according on the kind of algorithms employed.

The Figure 2 shows various categories that highlight the key elements of the scheduling process. Each group tries to improve the outcomes by planning the allocation of activities and resources. The chart shows that the first two scheduling classes put an emphasis on energy and utilization, which tends to reduce energy consumption and increase the effective use of cloud resources. The primary focus is on minimizing workload,

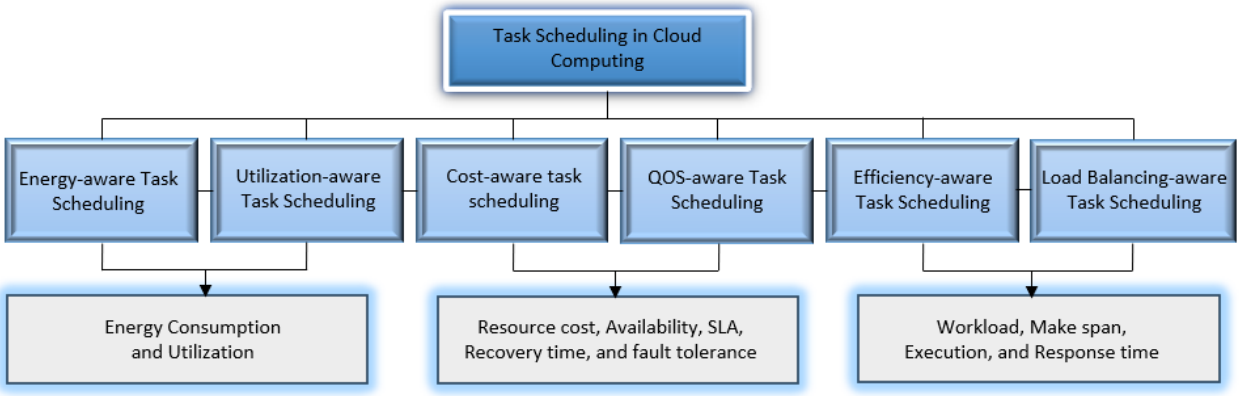

Figure 2: Task Scheduling in Cloud Computing

makespan, execution time, resource expenses, user costs, and response times. The connections between the aforementioned classes enable effective resource management.

We shall talk about scheduling classes that emphasize time and money issues in this study because they show how effectively a cloud provider's facilities work. Implementation time and makespan should be decreased while speed or bandwidth is enhanced to get this competency. Therefore, the main aim is to create a task allocation or scheduling strategy that effectively uses the resources while taking into account the cost of execution, cost of communication, makespan, time for execution, completion time, and service level agreement-bound turnaround time. This is done by using an algorithm that utilizes the resources more promptly and in a cost-efficient manner.Akhter and Othman (2016)

1.1 Need for the Research

Resource allocation, standard compliance, customer QOS, and enhancing the advantages of providers of cloud are all established through management of resources and task scheduling. Resources must be assigned and dispersed to users in the fair, fierce, and productive manner. According to service providers, a cost-effective scheduling method is currently a crucial requirement for data centers. There are numerous methods available for task scheduling, including First FCFS, ACO, LJF and others. Each of these algorithms has various benefits as well as restrictions. Mustafa et al. (2015)

Additionally, in IaaS of cloud, the technique of resource organization is the main area of worry, where appropriate resource optimization and task completion in a shorter amount of time are still regarded as the main issues Arunarani et al. (2019), Given the objectives mentioned above, this field necessitates a novel task scheduling technique to control the total cost, unexpected workload, and minimize reaction and execution times for the efficient management of massive resources. Because of this, our main objective is to enhance the scheduling algorithm by using current methods to enhance the QoS operation.

The major goal of using an effective integrated scheduling technique, such as the Genetic Algorithm that is GA technique and Particle Swarm Optimization that is PSO technique, is to reduce the optimization problem.

1.2 Research questions

"How can the hybrid approach using Genetic Algorithm -Particle Swarm Optimization (GA-PSO) techniques for task allocation/scheduling strategies maximize efficient optimization of resources with reduced costs in the cloud computing environment?"

The major goal of this research project is to find a solution to the problem of task scheduling and enhance resource utilization in a timely and cost-effective way in order to improve system effectiveness and decrease the makespan, execute time, and turnaround time. We employ an enhanced a hybrid genetic algorithm and a particle-swarm optimization technique to do this. The proposed method runs faster and produces the best results when the GA and PSO algorithms have less iterations. This approach initializes different encoded solutions to a random population and then uses the GA-PSO to provide the best possible solution.

2 Related Work

Our literature review is segregated into related sections as indicated below in order to provide a thorough understanding of this research field. Section 2.1. defines theories on existing algorithms studied by researchers. Section 2.2 describes theories on the proposed system where the heuristic scheduling strategy elaborates on the significance of a few algorithms that are inherently problem-dependent while the scheduling strategies meta-heuristic and hybrid elaborate on the significance of algorithms that are, by definition, independent of problems.

2.1 Theories on pre-existing algorithms

The term "cloud scheduling" refers to a technique that distributes jobs over a number of virtual computers or charges virtual machines with using the available resources to meet user needs. Resource management and job scheduling have received extensive research and survey attention in recent years. Kumar et al. (2013)

IaaS task and resource scheduling issues are explored and several hybrid scheduling strategies, their algorithm benefits, and drawbacks are covered by Syed Hamid Hussain in his research. The difficulties around efficient resource use have received a lot of attention from researchers working on infrastructure as a service. Singh and Chana (2016)

2.2 Theories on proposed system

The heuristic scheduling algorithm is based on the idea that individual tasks should begin as soon as their predecessor tasks are finished, provided that there are enough resources available. If there are insufficient resources, such as when a individual is needed for an activity but is already working on another, the activity is added to a list of ones

that can be scheduled as soon as more resources are available. This list is checked to see if an activity is waiting for a resource that has been released after an activity completes and releases its resources. Several activities will frequently be vying for the same resource. The activities are given resources and begun in decreasing order according to their "urgency" in this example. These algorithms are categorized as problem-specific techniques, and their capacity to resolve issues depends on the intricacy of the situation at hand. Such algorithms have the capacity to often and accurately offer solutions for specific types of difficulties over a regular time span. To address the associated problems associated with workflow applications and scheduling concerns, a wide variety of heuristic techniques are created in cloud computing. Akhter and Othman (2016)

FCFS scheduling method was created by Li and Shi (2009) and operates on the principle that priority is given to the queues that arrived first, followed by those still awaiting service. However, the method was unable to balance the burden among many VMs (virtual machines).

Mr. Buyya et al. (2018) presented the Weighted Round Robin (WRR) algorithm which divides the jobs using time slices in a circular sequence. The method distributes the jobs to the optimal virtual machine based on its information, including processing power, load, and the scope of the work assigned as per its preference. In some cases, the task may take longer than expected to finish the computations at the time of execution owing to the execution of a large amount of series on the standard instructions. The primary factor for Weighted Round Robin is QOS, which says that response time is its ideal metric. The performance evaluation and test results concluded that Weighted Round Robin is perfectly viable for both homogeneous and heterogeneous workloads. Although this algorithm increased the ratio of resources consumed and reaction times, for some reason it did not properly balance the demand on the resources. To improve energy consumption techniques, a staggering amount of problems are brought to light. The author has also discussed the limitations of current scheduling methods that center on energy and created a strategy known as a load detection policy. The load detection technique and the cost of electricity are highlighted in detail in. In order to identify the over loaded and under loaded hosts and reduce energy consumption and service level agreement, this technique measures the standard deviation in addition to the median. It fell short, nevertheless, in looking at the deadline restriction. The author then came up with the concept of static scheduling made up of some reactive strategies to replace dynamic scheduling; he called this technique the energy effective reactive programming Heuristic algorithm. This method has the ability to distribute real-time jobs across virtual machines in so as to ultimately preserve cloud energy. The ERECT algorithm's SLA violation when migrating VMs for energy conservation is a drawback, though. In an effort to reduce energy use, the scheduling algorithms inadvertently missed the SLA breach.

Time management, resource allocation, and task optimization are just a few of the optimization problems that have been solved by metaheuristic algorithms. Some of the most popular meta-heuristic algorithms include GA-Based, ACO-Based, PSO-Based, and other evolutionary metaheuristic algorithms. The term "metaheuristic" describes more advanced heuristics that were created to offer solutions for a variety of optimization issues. Hence the other algorithms accumulated enormous demand over time and are now frequently used for problem independent as well as nondeterministic solutions to NP-hard optimization-related problems. GA, introduced in 1975, has been used to schedule

workflows and improve resource allocation, while ACO, introduced by Marco Dorigo, is used for optimizing energy consumption and increasing profits for service providers. Ant Colony Optimization (ACO) is an inconsistent search algorithm designed to solve computational problems in cloud computing. It enhances consistency of pheromones, allowing persistent solutions to issues like abrupt loads. Pradhan et al. (2022)

Other evolutionary meta-heuristic algorithms include the Bees Life algorithm, Krill Herd (KH), binary grey wolf optimizer (QI-BGWO) inspired by quantum, cuckoo search, and flower pollination algorithms, and the improved whale optimization algorithm (IWOA). These algorithms have shown high accuracy and effectiveness in solving optimization problems and achieving high-quality solutions. Several meta-heuristic algorithms have been put forth for computing performance in cyber-physical social systems (CPSS), in addition to GA and ACO. These algorithms have been tested in various scenarios, achieving better performance and reducing energy consumption. Opposition-based learning has also been used to optimize PSO (OPSO) for task scheduling in a cloud computing environment, improving convergence, energy consumption, and makespan. Overall, meta-heuristic algorithms have been developed to address various optimization problems and improve overall performance in various domains. Ghumman and Kaur (2015)

Ghumman and Kaur (2015) proposed a modified genetic algorithm that mitigates the usage of power and with the use of paralleling, can be enhanced. Context switching optimization minimizes overhead and enhances system responsiveness. Implementations and approaches may vary based on the system's requirements, the nature of task, and available resources. This set of algorithms was created by combining various task and resource scheduling heuristics and meta-heuristics to address scheduling issues in the context of computing a cloud. To reduce the makespan, cost waiting time, Ghumman and Kaur (2015) also proposed a hybrid algorithm that combines the ACO and MaxMin algorithm. By using task priority as a constraint, this technique raises QoS parameters even more.

2.3 Purpose of the study

This paper presents an enhanced scheduling algorithm based on PSO-GA. The hybrid algorithm combines the benefits of both the above algorithms, increasing resource utilization and convergent solutions quicker. GA is an advanced algorithm but slow for large-scale problems. The algorithm uses GA and PSO techniques to reduce iterations and perform swapping to achieve optimal results. Cloud Simulation is used to evaluate the strategy in comparison to more established methods.

3 Methodology

To build the suggested model, we identified and examined a number of scheduling algorithms in the previous section, including particle swarm optimizer, first come first serve, genetic algorithm, shortest job first. The suggested method aims to solve the issue of work scheduling and improve resource utilization in a time and money efficient way. To acquire the fittest value, the numbers of iterations for the genetic algorithm and particle swarm optimization algorithms are reduced in this article, and certain operators are added to accomplish swapping. As a result, the suggested algorithm will run more quickly to reach the ideal answer. GA is a more sophisticated algorithm than PSO as

a result it takes more time to provide results. PSO, however, produces findings more quickly, but they are less accurate. The proposed system will be improved and combined with various algorithms to yield the best result. These hybrid algorithms are tested in both heterogeneous as well as homogeneous situations by the Cloudsim toolbox.

3.1 Proposed System Complexity

The GA is used in the proposed method to generate a random population (chromosomes), which is then used to establish a standard for a specific number of repeats. 'n' represents the number of iterations repeated by half as often that is 'n/2' to reduce complexity because the GA is very sophisticated and requires some time to discover the most suitable potential answer. In Genetic Algorithm, chromosomes refer to a collection of genes that together for providing a solution to a certain issue. Every chromosome in this algorithm contains many genes that represent the host machine's virtual machines. The assessment of the function of fitness is indicated in Algorithm 1 shown in Figure 3 determines the kind of chromosome to be chosen. Besides; selection, crossover, and mutation operators help these chromosomes get better with each iteration.

```
The Tournament Selection Method in Genetic Algorithm
-----------------------------------------------------
Input: VM (the chromosomes)
Output: VMfitness (fitnesschromosome)
Step1: Set the tournamentSize = n
For i = 0 to tournamentSize
id = Math.random()* chromosome/VM.size( ) // Random Selection of VM
tournament[i] = get VM (id)
End For
fitness←tournament.getFitnest( ) //return fitness value.
```

Figure 3: Algorithm 1

For the selection of the best collection of Virtual Machines from the sample population, tournament selection or the selection operator is used on the population (chromosomes), as given in Algorithm 1. When referring to solutions to multiple task problems in GA, the term "population" refers to how the work is divided up among the available VMs for each task problem. For the purpose of ensuring that the picked ids appropriately reflect the index of the selected chromosome, the option picking operators create a random id after performing numerous matches between a few VMs. The operator who crossovers on the group of VMs with the best combination of tasks is likewise chosen based on its fitness value.

The crossover operator on the group of VMs with the best combination of tasks is also chosen based on its fitness value. A crossover operator seeks to create an entirely novel set of chromosomes by transferring the virtual machines, or VMs, that exist in each pair of VMs. A random number is selected from among the available VMs in this. The crossover operator that produces the population with the best fitness value is changed by a mutation operator. This operator's operation on the population created by the selection process depends on how quickly mutations arise. The way the responsibilities are

divided among the readily available VMs is plainly demonstrated by the fact that if the two virtual machines are identical, their places are switched to create a new population.

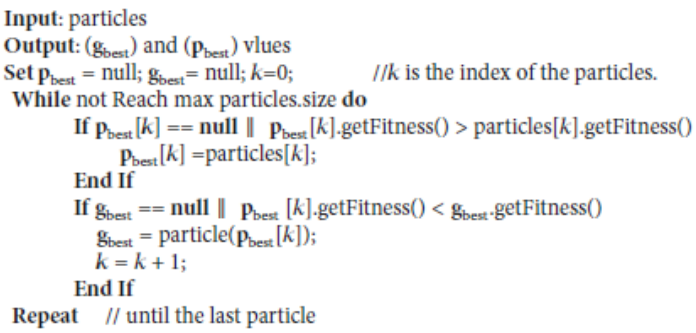

Figure 4: Algorithm 2

Create new Virtual Machine combinations (particles) using the position and speed of earlier iterations. The particles' pbest as well as gbest values are evolved during this step. With each iteration, the values of pbest and gbest, which alter, determine how the particle positions and velocities change. In PSO, the initials pbest and gbest stand for a particle's best personal location and the best position for the overall number of particles, or swarm. These parameters are used in accordance with Algorithm 2 in Figure 4, in which the first iteration's pbest (s) equal the GA algorithm's output, where (s) separates each result from the others. In order to compare earlier results, gbest is further defined as the result with its lowest fitness value.

Updating of the velocity position matrix: Raising particle velocity's objective is to create a new production from a group of VM sites that is more fit than its predecessor (each particle refers to a VM. In PSO, the process of velocity helps particles evolve or modify their places in pursuit of optimal solutions. Using the particle's g-best and p-best values, the velocity is calculated based on the swarm's best fitness value. In this way, the enhanced velocity measurements are used to modify the position of each particle's VMs.

3.2 Algorithm Proposed

This section precisely describes the whole GA-PSO algorithm using its virtual code, which is displayed in Algorithm 3. The method begins by initializing the jobs provided to the broker, and then, as shown in step 2, GA algorithm is applied over the randomly generated population (chromosomes), with the genes on each chromosome standing in for a number of VMs. The three GA operators of selection, crossover, and mutation are then applied n/2 times, and the result is an optimal solution. As seen in step 3, the solution (chromosome) generated by GA serves as a population (particles) for the Particle Swarm Optimization algorithm. Each particle in this illustration represents a VM, and each VM's index indicates a task. PSO calculates the particle's pbest and gbest values, updates its velocity, and repeats the process until it reaches the best solution and meets the stopping requirement. The outcomes demonstrate the best value with the lowest calculation, communication, and execution costs as well as the shortest possible execution time. Additionally, the GA-PSO results are contrasted with the performance of the existing PSO algorithm and a variety of jobs and VMs. The population describes

many approaches to the task problem, where each approach involves distributing all work among the available VMs.

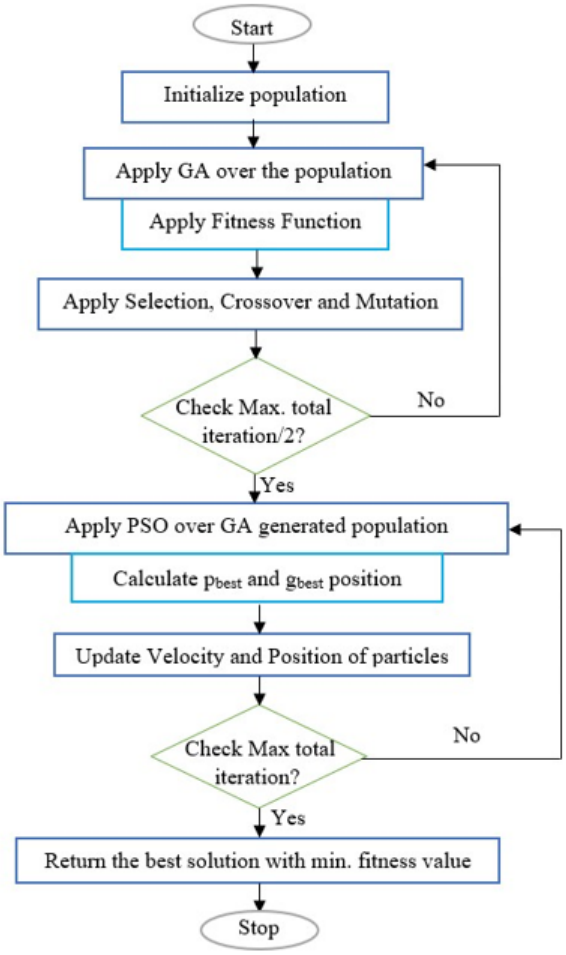

Figure 5: Proposed System Flowchart

A flowchart shown in Figure 5 shows an in-depth analysis of the strategy for implementation of the proposed system. Tasks will be distributed to virtual machines (VMs) in this manner, with the initial algorithm being the Genetic Algorithm, with operators for selection, crossover, and mutation. The p-best and g-best values will also be calculated using the PSO technique by updating the particle's position and velocity utilizing chromosomes provided by the GA's solution. In the end, the suggested algorithm will analyze many factors with varying numbers of tasks like makespan, completion time, average reaction time, etc. The following are the steps for the architecture of the proposed system,

- Apply the algorithm for genetics to the initialized encoded population for n/2 times as many iterations, or merely half as many.
- Determine the fitness value using the tournament selection method and choose the chromosomes with the greatest fitness value with the help of the selection operator.

- To achieve the best result, switch and reorder the chromosome positions using the crossover and mutation operations.
- Apply the proposed model-generated solution (chromosomes).
- Determine the GA chromosomes' pbest which is personal best and global best that is gbest positions.
- Up until the minimal fitness value is returned, PSO updates the velocity and position using the values (pbest, gbest).

4 Design Specification

The GA is one of the most complex algorithms, although it is slower to solve complicated issues. The results from PSO, on the other hand, aren't much better than GA, despite being rapid. The recommended GA- PSO techniques are altered to acquire the best population by lowering the amount of iterations as well as including certain operators with cutting-edge techniques to carry out transferring, so that the suggested approach will arrive at the ideal solution more quickly. The suggested method places a strong emphasis on lowering execution costs, makespan, and assessing various workloads to deliver the best possible set of virtual machines (VMs) to complete the tasks. To achieve the best possible outcome, different encoded solutions will be subjected to PSO-GA. Using cloud simulation, the obtained strategy result of is compared to the results of other approaches.

4.1 Architecture Proposed:

Even though numerous algorithms have been developed recently, there are still certain issues with the job scheduling process. It is proposed to use the GA-PSO approach to reduce down on wait times, turnaround times, completion times, and execution costs. It focuses on producing outcomes with the solution generated by Genetic Algorithm while consuming lesser time as well as less money. The goal of genetic algorithms is to produce optimal values that have the highest fitness value.

The design shown in Figure 6 concisely explains the suggested method by grouping it into the following three major categories.

- User-side tasks: This displays the requests made by cloud end users. Each task has its own specific preferences for resources, duration, and other elements.
- Cloud Broker: Cloud brokers are important in the scheduling process since they are the organization in charge of using an algorithm to assign jobs to the resources that are available. It also monitors the secure data transit between cloud customers and different cloud providers
- Cloud Provider: A cloud service provider creates resources in response to demands from cloud end users, and the created resource may or may not meet the same specifications.

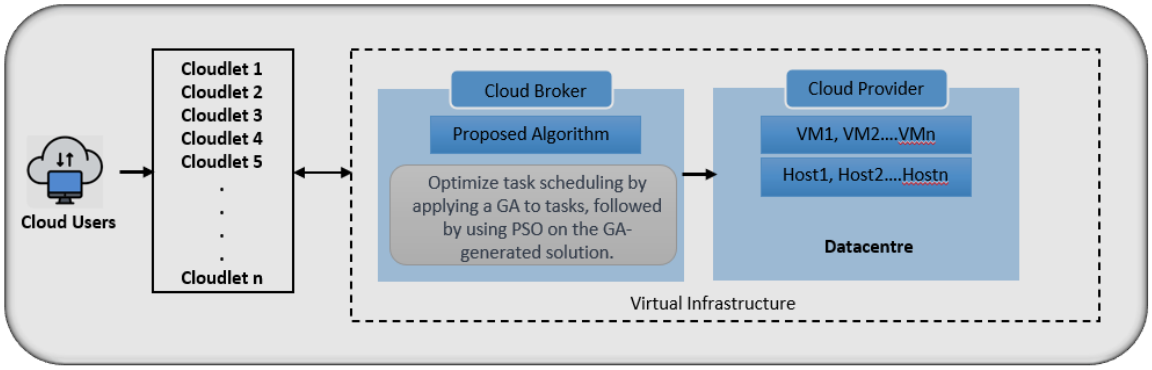

Figure 6: Architecture Proposed for allocation of tasks to VMs

5 Implementation

CloudSim Simulation Toolkit is a strong and popular simulator for conducting research on the cloud environment. CloudSim makes it possible to simulate, model, and test cloud-based applications and infrastructure services in real time. This simulation toolkit is used by researchers as well as cloud developers to assess the performance of proposed cloud applications before setting up the environment. Modelling and simulating large-scale cloud environments like data centres in addition to a single hardware node like a desktop, server, or laptop are some of CloudSim's main tasks. It can switch between allocating processor cores to virtual resources in a time- as well as and space-shared fashion. Manasrah and Ali (2018)

5.1 CloudSim's Architecture and Design

There are several layers that make up CloudSim's parts. The end-user or developer decides the requirements for the simulation in the end-user code layers, including those for virtual machines (VMs), host and their characteristics, as well as applications and their specifications. Virtualized datacenter simulating and modelling are provided by the layers that fall in the Cloudsim group name.

The class diagram in Figure 7 more clearly displays the fundamental elements of the simulation system. An illustration of a key hierarchy that starts the simulation is the CloudSim core. Regarding cloudlets, datacenter broker policies, and other relevant entities, this structure necessitates a sequence of iterative actions.

A CloudSim class is the primary one in charge of managing event queues. These occurrences are regarded as SimEvent class instances. The generated events are then assigned to a future queue and sorted according to time constraints. However, as the events' scheduled times arrive, the aforementioned created events are switched from future queues to postponed queues. SimEntity is referred to as an abstract class where action is taken based on events in a deferred queue via event handling methods. There is also a collection of tags for occurrences in CloudSimTags. Both the startup and shut entities are activated during the starting and stopping of a simulation entity.

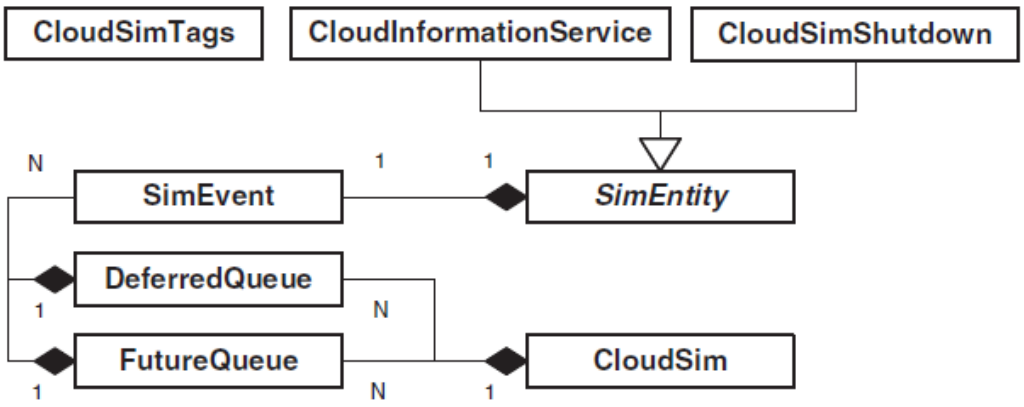

Figure 7: Hierarchy of classes that enables the CloudSim Simulation Toolkit

5.2 Set Up of the Proposed Model

The suggested model employs a modified hybrid GA-PSO algorithm to efficiently as well as quickly resolve the issue of work scheduling and enhance resource utilization. This shortens the makespan, execution time, and turnaround time while improving system efficiency.

A few simulation parameters which are mentioned in In Table 1 are used to assess the effect of the GA-PSO algorithm on the work scheduling problem. A CloudSim simulator is used to put the suggested method into practice. The virtual computers used in the experiment and its duties are defined by these settings. To minimize execution time and increase execution cost, different task counts are used. The table shows that there are 20 jobs totaling ten virtual machines. Million instructions per second i.e. MIPS which is two hundred per second.

Table 1: Simulation Parameters of GA-PSO

Parameter	**Value**
No. of Tasks	20
No. of VMs	10
MIPS	200
RAM (MB)	512
BW (mbps)	1000
Process Speed	10000
No. of Processors	1

GA is used to 20 randomly selected solutions (populations) in order to determine the optimal chromosomes with the aid of the selection, crossover, and mutation operators. While the rate of mutation in the mutation stage is 0.30, the crossover rate is 0.95 in the crossover stage as shown in Table 2

Table 2: GA-PSO Parameters

Parameter	Value
No. of Tasks	20
Crossover	0.95
No. of Iterations	20(n/2)
No. of Executions	100
Mutation Rate	0.30

5.3 Implementation of the Proposed Model

There are a few basic prerequisites that must be met for CloudSim to be available on desktop, and these versions are all available on GitHub. The required .zip format version must first be downloaded. Since the entire CloudSim source code is written in Java, Java JDK3 must be installed on the computer where the code will be executed. Additionally, Java IDE can be downloaded from the Eclipse foundation and deployed locally for greater feasibilityfootnote4. Cloudsim provides a significant benefit for resolving complex computations. They necessitate the use of a math library function, which may be obtained from the Apache website, in order to take advantage of this functionality.

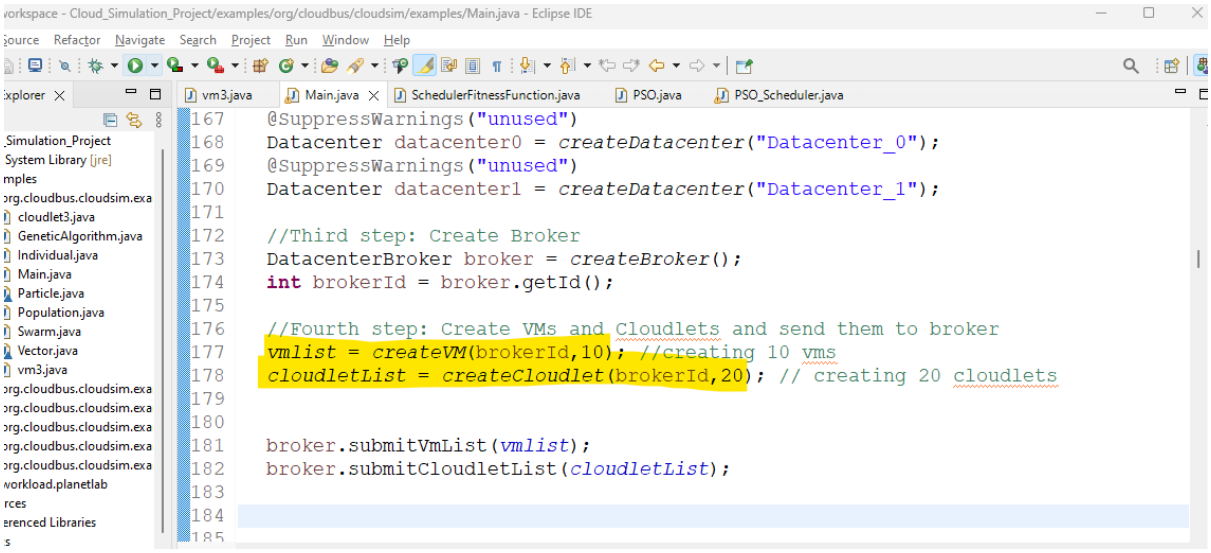

Figure 8: GA-PSO with 10 VMs configuration setting

Two experiments will be run on CloudSim. Experiment no. 1 uses 10 VMs of various size as mentioned in Figure 8, and experiment no. 2 uses the conventional PSO algorithm and 10 VMs of all permuted configurations as described in Figure 9. Test results obtained from the trials are then compared with each other. The suggested strategy asserts that it takes less time and money to attain the outcome than the traditional PSO method.

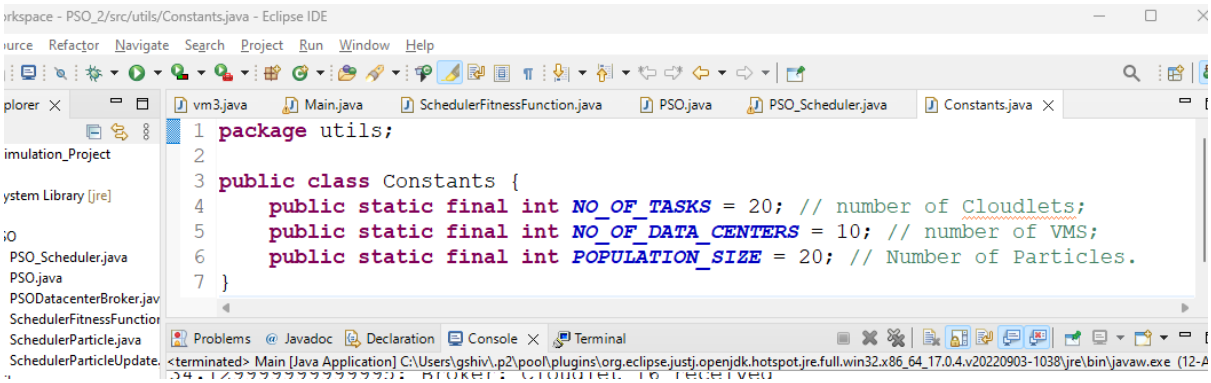

Figure 9: Traditional PSO with 10 VMs Configuration setting

6 Evaluation

The GA-PSO hybrid technique is constructed by use of the CloudSim simulator in order to assess the suggested algorithm. A combination of different virtual machines is chosen from the already available virtual machines, taking into account factors such as average turnaround time, average cost, and an average completion time. Outcomes of the methods proposed are contrasted with those of the conventional PSO algorithm. To obtain the findings using CloudSim, 20 jobs and 10 VMs are taken into account in both algorithms. This section includes charts and graphs that help you better comprehend the outcomes of using the proposed algorithm.

6.1 Performance Evaluation

The main goal of the evaluation is to improve the particle swarm optimization by incorporating the GA technique to raise the makespan, completion time, execution cost and turnaround time characteristics. In order to demonstrate the advantages of the suggested algorithm, the outcomes of the enhanced version of PSO model are also compared with those of the classic Particle Swarm Optimization method.

Figure 10 shows that the time of completion of the proposed method is smaller than that of the PSO algorithm. The proposed algorithm's least time recorded is 6.13ms, in comparison to the PSO algorithm's recorded time which is 52.3m, the least, which is a big difference. Additionally, 205.04ms is the PSO's maximum completion time, roughly three times as long as the suggested algorithm's maximum completion time.

Table 3: PSO vs Proposed Algorithm MakeSpan and Execution Cost.

Algorithm	**MakeSpan (ms)**	**Cost ($)**
PSO	1202.305	21.2
Proposed Hybrid Algorithm	201.51	15.1

The Particle Swarm Optimization approach with 20 tasks and 10 virtual machines and the proposed algorithm approach using 20 tasks and 10 virtual machines for the exact same algorithm are all used in the above table. The main goal of the above comparison shown in Table 3 and Figure 11 is to understand the makespan and the execution costs

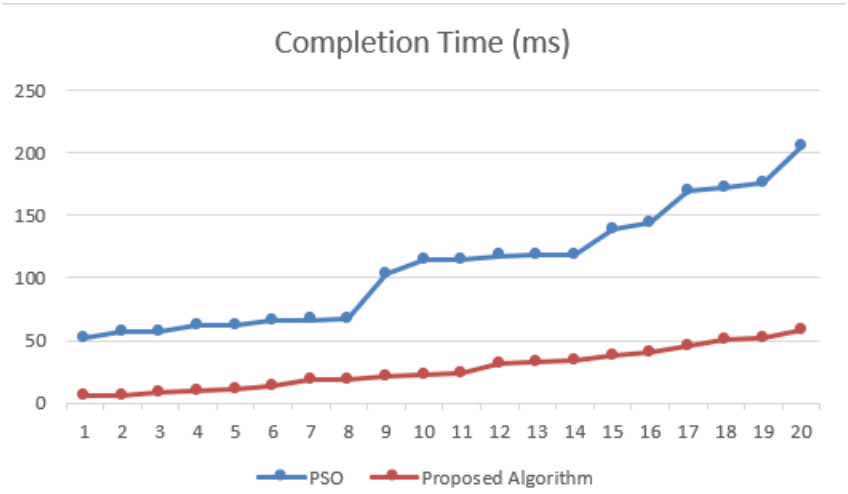

Figure 10: GA-PSO vs PSO Completion time (ms)

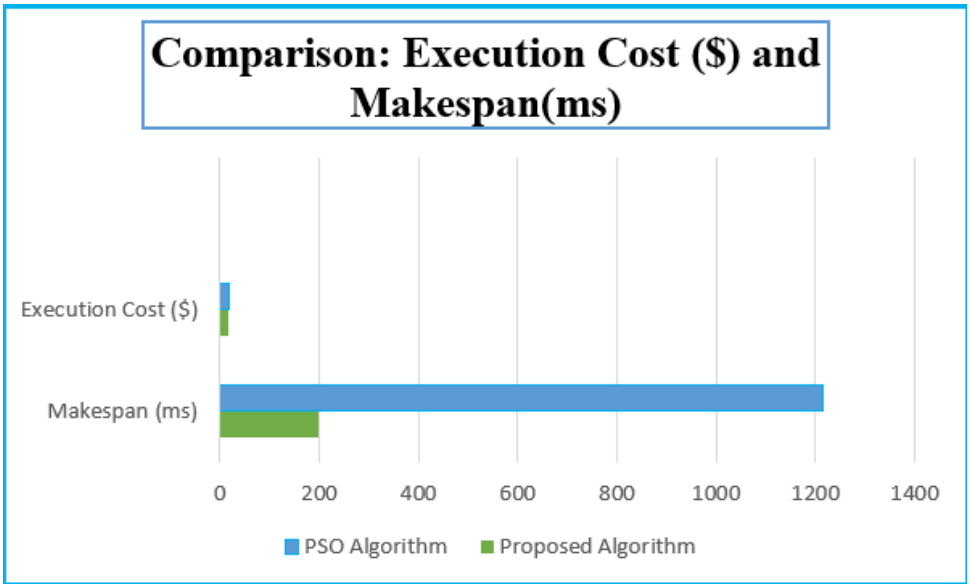

Figure 11: PSO vs Hybrid model main results

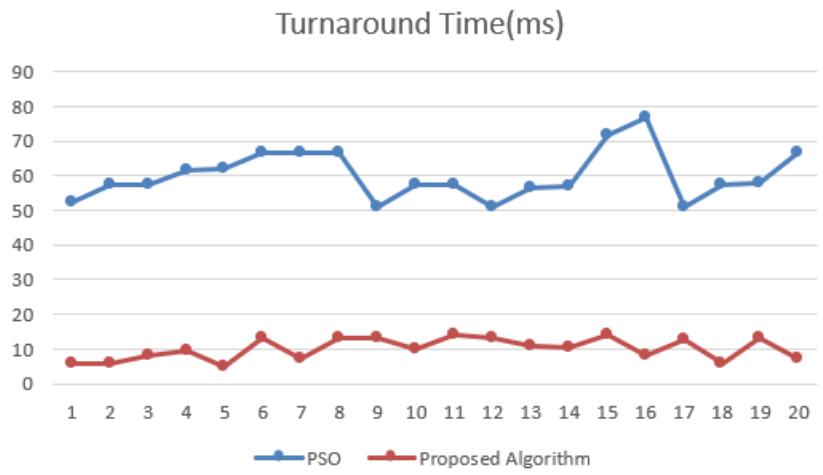

Figure 12: Turnaround Time for PSO vs proposed model

of the two situations so that the Particle Swarm Optimization algorithm's makespan and execution costs are greater than the proposed algorithm. With this output, we will attain efficiency.

The calculation of turnaround time is mentioned in Figure 12 by subtracting the arrival time from the completion time. This calculation shows that the minimum turnaround time of every task of the proposed model; its turnaround time being 5.93, or about 3 percent less than that of the PSO technique. It can be inferred that the proposed model has produced more better outcomes in both of the aforementioned parameters.

6.2 Discussion

We designed an effective scheduling method in this paper which will enhance the cost as well as time parameters to a point where, after analyzing the literature, we offered a hybrid technique. The plotted graphs in sub section explain turnaround and completion time outcomes, and timeline with the execution cost. The proposed algorithm succeeded admirably in each of the outcomes listed in the report's purpose. The GA-PSO method enhanced the time and cost factors efficiently that were the focus of this paper, as shown by the displayed graphs that compared the proposed and standard PSO methods.

7 Conclusion and Future Work

Thus, our research reviewed the task allocation and resource allocation algorithms from previous related research papers, outlining their main drawbacks and benefits. This information allowed us to propose an efficient algorithm by combining modified GA-PSO used in solving scheduling problems including time and money-related variables. This study contrasts the GA-PSO method with the traditional PSO technique to demonstrate how the proposed technique performed better for each metric, including execution cost, makespan, turnaround, and completion time. The outcome demonstrates the suggested model's efficiency in terms of time and cost as well as its capacity to identify rapid fixes

for issues. The suggested technique can be expanded in the future to support additional virtual machines (VMs) by employing numerous data centers in a heterogeneous domain, allowing jobs to be allocated to available VMs in each data center based on their speed and size. Additionally, this will improve cloud scheduling and provide consumers with more freedom.

References

Akhter, N. and Othman, M. (2016). "energy aware resource allocation of cloud data center: review and open issues. cluster compute ", *ONLINE* **12**: 1163–1182.

Arunarani, A., Manjula, D. and Sugumaran, V. (2019). Task scheduling techniques in cloud computing: A literature survey, *Future Generation Computer Systems* **91**: 407–415.
URL: *https://www.sciencedirect.com/science/article/pii/S0167739X17321519*

Buyya, R., Srirama, S., Casale, G., Calheiros, R., Simmhan, Y., Varghese, B., Gelenbe, E., Javadi, B., Vaquero, L., Netto, M., Toosi, A., Rodríguez, M., Llorente, I., Vimercati, S., Samarati, P., Milojicic, D., Varela, C., Bahsoon, R., Assuncao, M. and Shen, H. (2018). A manifesto for future generation cloud computing: Research directions for the next decade, *ACM Computing Surveys* **51**.

Calheiros, R. N., Ranjan, R. and Buyya, R. (2011). Virtual machine provisioning based on analytical performance and qos in cloud computing environments, *2011 International Conference on Parallel Processing*, pp. 295–304.

Dong, Z., L. N. and Rojas-Cessa, R. (2015). "greedy scheduling of tasks with time constraints for energy-efficient cloud-computing data centers", *SpringerOpen* **12**.

Ghumman, N. S. and Kaur, R. (2015). Dynamic combination of improved max-min and ant colony algorithm for load balancing in cloud system, *2015 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–5.

Kumar, Y. R., Priya, M. M. and Chatrapati, K. S. (2013). Effective distributed dynamic load balancing for the clouds.

Li, W. and Shi, H. (2009). Dynamic load balancing algorithm based on fcfs, *2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC)*, pp. 1528–1531.

Madni, S. H. H., Latiff, M. S. A., Coulibaly, Y. and Abdulhamid, S. M. (2016). Resource scheduling for infrastructure as a service (iaas) in cloud computing: Challenges and opportunities, *Journal of Network and Computer Applications* **68**: 173–200.
URL: *https://www.sciencedirect.com/science/article/pii/S1084804516300674*

Manasrah, A. and Ali, H. (2018). Workflow scheduling using hybrid ga-pso algorithm in cloud computing (workflow simulation), *Wireless Communications and Mobile Computing* **2018**: 1–16.

M.S. Sudheer, D. V. K. (2019). "dynamic pso for task scheduling", *International Journal of Recent Technology* **12**: 2277–3878.

Mustafa, S., Nazir, B., Hayat, A., ur Rehman Khan, A. and Madani, S. A. (2015). Resource management in cloud computing: Taxonomy, prospects, and challenges, *Computers Electrical Engineering* **47**: 186–203.
URL: *https://www.sciencedirect.com/science/article/pii/S004579061500275X*

Pradhan, A., Bisoy, S. K. and Das, A. (2022). A survey on pso based meta-heuristic scheduling mechanism in cloud computing environment, *Journal of King Saud University - Computer and Information Sciences* **34**(8, Part A): 4888–4901.
URL: *https://www.sciencedirect.com/science/article/pii/S1319157821000033*

Sangwan, O. and Dhanda, M. (2019). Qos based scheduling techniques in cloud computing: Systematic review, *ResearchGate* .

Singh, S. and Chana, I. (2016). "cloud resource provisioning: survey, status and future research directions", *ONLINE* p. 1005–1069.

Varanasi, P. (2020). "understanding fault tolerance in cloud computing and its significance. cloudcodes", *Online* .
URL: *https://www.cloudcodes.com/blog/fault-tolerance-in-cloud-computing.html*