iJRASET
International Journal For Research in Applied Science and Engineering Technology

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 9 **Issue:** III **Month of publication:** March 2021

**DOI:** https://doi.org/10.22214/ijraset.2021.33235

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429*
*Volume 9 Issue III Mar 2021- Available at www.ijraset.com*

# Task Scheduling in Cloud Based on Three Stage Method using Historical Data

Gopika Venu[1], Vijayanand K S[2]
[1, 2]Dept. of Information Technology, Government Engineering College Bartonhill, Trivandrum, Kerala, India

***Abstract*: The on-demand availability of computer system resources such as data storage and computing power is cloud computing. Scheduling is the method of allocating jobs onto resources in time. Scheduling increases the efficiency and performance of cloud environment by maximizing the resource utilization. This scheduling process has to respect constraints given by the jobs and the cloud providers. Ordering the tasks by scheduler along with maintaining the balance between Quality of Service (QoS), fairness and efficiency of jobs is difficult. Scheduling algorithms are designed and implemented considering some parameters like latency, cost, priority, etc. This paper proposes a method for task scheduling in cloud using a three-stage method. The first stage makes use of historical scheduling data to classify tasks. Based on this VMs are created. In the second stage newly arrived tasks are considered. Based on Bayes classifier principle a matching degree is calculated to mark a task with a VM Type. In the third stage, tasks are matched with corresponding VMs. Then the tasks are scheduled using a scheduling algorithm and are executed in the host.***

***Keywords: Task scheduling, Cloud computing*.**

## I. INTRODUCTION

Cloud is a trendy term for a network or remote servers that can be accessed via an Internet connection store and manage information. Two types of models present such as deployment models and service models. The deployment model includes private, public and hybrid cloud. Private Clouds are Data center architecture owned by a single company. E.g., IBM's BlueCloud, SunCloud, WindowAzure. Public Clouds are basically the Internet. To make resource available to general people, service provider use internet. E.g., Gmail, Office 365, Dropbox. In case of hybrid clouds, for instance during peak periods individual applications or portion of applications can be migrate to the public cloud. The service model includes Software as a Service (SaaS) such as Dropbox, Google Apps, Platform as a Service (PaaS) such as Google Compute Engine, Apache Stratos and Infrastructure as a Service (IaaS) such as Virtual Machine, Storage, Servers.

Cloud provides on demand computational resources in the form of virtual machines. These Virtual Machines are deployed in a cloud provider's data center. The computational resources are shared among different cloud consumers who pay for the service accessed as per the usage. Allocation of resources and proper scheduling has a considerable impact on the performance and efficiency of the system. The main goal of cloud computing is to provide efficient access to remote and geographically distributed resources. An efficient scheduling is a key to manage the access to different resources, load balancing as well as resource allocation. Different types of resource scheduling algorithms are available in cloud computing based on certain parameters like time, cost, and performance, utilization of resources, throughput, bandwidth, resource availability, physical distances and priority.

### *A. Scheduling In Cloud Computing*

Scheduling is the allocation of various jobs to available resources. The task is a minimum computational unit to run on a node or resource. Job is a computational activity made up of several tasks that could require different processing capabilities and could have different resource requirements such as CPU, number of nodes, memory, etc. Each job may have various parameters such as desired completion time often called the deadline, required data, expected execution time, job priority, etc. A resource is something that is required to carry out an operation. For e.g., a processor for data processing, a data storage device or a network link for data transporting, etc. Scheduling problem is related to two types of users cloud consumers as well as cloud providers. Cloud consumer wants to execute their jobs for solving problem of varying size and complexity. Cloud providers will contribute resources for executing consumer jobs. Cloud consumer will benefit by selecting and aggregating resources wisely while cloud provider benefit by maximizing resource utilization. The problem of mapping tasks on resources belongs to a class of problems known as NP hard problems. These are problems for which no known algorithms are able to generate the optimal solution within polynomial time. So, solutions based on exhaustive search are impractical. Overhead of generating schedules is very high. Scheduling decisions must be made in the shortest time possible in clouds because multiple users are competing for the time and resources, so metaheuristic algorithms such as genetic algorithm, PCO etc are used.

*B. Scheduling Objectives*

An objective function is designed for scheduling algorithm. The main aim is to minimize this objective function or maximize this objective function according to the criterion specified by the user. In traditional approaches a single objective function is considered but now multicriterion objective function is created. In this, simultaneously two or more parameters are considered. On the basis of users, it can be Application Centric. For example, the parameters makespan and economic cost can be application centric. The designed algorithm should minimize either makespan or economic cost or both. On the basis of providers, it can be Resource Centric. For example, the parameters resource utilization and economic profit can be resource centric. The designed algorithm should maximize either resource utilization or economic profit or both.

Virtual Machines (VMs) are the resources for execution of tasks in cloud. Two kinds of problems present in the scheduling process. When we are assigning a task to a VM, the first problem is assigning a large task to a VM having weak processing capacity. This will lead to large processing time and sometimes the task may not be able to complete before the deadline. The whole task sequence may be interrupted. The second problem is assigning a small task to a VM having strong processing capacity. This will make other large tasks to wait for a long time for execution. This will reduce the overall throughput of the cloud. So, while scheduling, the tasks must be assigned to the most suitable VMs. Creating VMs temporarily according to the exact requirements of tasks while scheduling is a solution to this problem but this will take a lot of time. This will make the users wait for a long time to get their tasks processed.

Thus, in this method VMs are created before user submit tasks. The creation of VMs is on the basis of historical task data. A three-stage method is used for the scheduling purpose. First stage is the VM creation based on historical data. Then the submitted tasks are marked with corresponding VM type. Then tasks are matched with concrete VMs. Deadline is taken into consideration for allocating and hence time cost can be minimized.

## II. LITERATURE SURVEY

*A. Various Existing Task Scheduling Algorithms*

1) *First Come First Serve (FCFS):* The incoming task looks for the queue where the waiting time is the smallest. The queue is managed by FIFO mechanism (first come first out). In FCFS, place each incoming Task at the end of the service queue. The first task in the queue is assigned to VM when it is available. This algorithm is simple and easy to understand. The main disadvantage is large waiting time. The short jobs at the back of a queue will wait until large task in the front of queue is finished. It is based on single criterion for scheduling. This algorithm is non preemptive.
2) *Round Robin Algorithm:* Round Robin Algorithm is a pre-emptive algorithm. This algorithm distributes the jobs on the available VMs in a round form or cyclic manner, where the jobs are stored in a ring queue. Each job is given a quantum of time. If it is not completed within its turn, it would be interrupted. Then it is stored back in the tail of the queue, and wait for its next turn. The algorithm is repeated until each task in the queue is assigned to minimum one virtual machine. The pros of this algorithm are no need for a preprocessing step to fetch the nominated VM, Distribute the load equally among VMs, focuses on fairness among the scheduled tasks, Jobs are executed in turn and never waiting for previous job to finish execution (starvation free), The scheduler will not wait until all processing power of a VM is exhausted before it moves to next VM and at last, it is based on a simple rule. The major cons of this algorithm are long jobs take longer time to complete execution. Servers may be overloaded. Preemptive policies depend on the length of time slice and during short time slice this would cause many switching.
3) *Modified round robin [1]:* By dynamically taking into consideration the burst time for each incoming task entering the ready queue, modified round robin does a simple improvement on the standard round robin algorithm. The time slicing process is done on the basis of computing the average burst time of all the remaining waiting requests in the ready queue. For this purpose, two registers are there, Sreg and Areg. The total burst time of all request in the ready queue are stored in Sreg. The Average burst time by dividing the value of Sreg by the number of tasks residing in the ready queue is stored in the Areg. In the beginning the first job is allocated to virtual machine and all its burst time is taken, then the time slice of each incoming request is computed by the scheduler. When it is allocated to run on a virtual machine, each task would run for a time period that is same as the time slice granted to it by modified Round Robin, when it entered the queue. When the time slice is over the task is either removed from the ready queue or joins the ready queue again standing at the back of the queue. The scheduler adjusts the values of the registers by subtracting the burst time of the removed task from SReg and adding the value of the new joined tasks to AReg.

4) *Shortest Job First (SJF):* Shortest job first scheduling (SJN) is used for ordering a set of tasks by putting the shorter task in the front of the queue and longer tasks at the end of the queue. This algorithm reduces the waiting time of the short jobs and increases the waiting time of the long jobs thereby reducing average waiting time. This can result in starvation for longer jobs when there are a large number of small jobs.
5) *Traditional Min-Min Heuristic Algorithm [2]:* This algorithm is based on the concept of Minimum Completion Time. In Min-Min algorithm, for each task determine its minimum completion time over all machines. For all tasks find the minimum completion time. Assign the task to the machine that gives this completion time. Then iterate till all the tasks are scheduled. The main advantage of this algorithm is smaller makespan, since tasks are scheduled on the fastest machines where they are completed earlier. The algorithm is operative for the task scheduling in cloud computing. This algorithm will increase the throughput. The disadvantages of this algorithm is, it increases the total completion time of all the tasks and hence increases the makespan. The long tasks have to wait for smaller tasks to end their execution and the load is unbalanced.
6) *QoS Guided Min-Min Heuristic Algorithm [3]:* QoS guided Min-min takes bandwidth requirement of tasks into account. Jobs requiring higher bandwidth are scheduled prior to others. Hence, if bandwidth requisite of all tasks vary tremendously, QoS guided Min-min is better. In this algorithm, tasks are classified under high and low bandwidth. Task required high bandwidth are scheduled first.
7) *Traditional Max-Min Heuristic Algorithm [4]:* This algorithm is based on the concept of Maximum Completion Time. In Max-min algorithm, for each task determine its minimum completion time over all the machines. Then find the maximum completion time for all. Assign the task to the machine that gives this completion time. Iterate till all the tasks are scheduled. This can reduce the waiting time of long tasks so they never starved. This can increase the utilization and the response time is minimized. The makespan is reduced since smaller jobs are executed concurrently while other longer jobs are executed. The main disadvantage of this algorithm is as it first selects the large tasks for execution the smaller tasks are delayed. This algorithm is not effective in load balancing.
8) *Improved Max Min Algorithm [5]:* Improved Max Min is based on the expected execution time and not on the completion time. Max-Min could execute multiple short tasks simultaneously while executing larger ones. In this scenario the total makespan is decided by the execution of long task. In cases where Meta tasks contain tasks having different completion time and execution time, the makespan is not decided by one of submitted tasks. Improved max-min task scheduling algorithm tries to minimize waiting time of short jobs by assigning largest tasks to be executed by slower resources, small tasks are executed concurrently on fastest resources to complete large number of tasks during finalizing minimum one large task on slower resource. Based on these cases they proposed a substantial improvement of Max min algorithm which leads to increase of max min efficiency. The suggested improvement increases the opportunity of simultaneous execution of tasks on resources.

a) *It Works In Two Phases:* The first phase is task with maximum execution time is selected (Largest Task). The second phase is selected task is scheduled over resource with minimum completion time (Slowest Resource).

9) *Enhanced Max-min Task Scheduling Algorithm [6]:* Sometimes largest task is too large compared to other tasks in Meta-task. In those cases, overall makespan is increased because too large task is executed by slowest resource first while other tasks are executed by faster resource or when there is a huge difference among slowest and fastest sources in terms of processing speed or band width. Then the largest task is executed by the slowest resource which increases the makespan and load imbalance across resources. Therefore, rather than selecting largest task, if we select average or closest greater than average task then overall makespan is reduced and also balance load across resources.
10) *Minimum Completion Time Algorithm:* The algorithm scans the available VMs to find the most suitable machine to assign a job. The VM is selection is based on the minimum completion time. This minimum completion time is calculated by taking into consideration the processing speed and the current workload on a machine. Therefore, it is considered as successful heuristic that could be implemented in Cloud Computing. The main disadvantage is process of assigning a task to certain machine with minimum completion time is done in arbitrary order so each time a task is assigned to the fastest machine in the remaining resources pool.
11) *Suffrage heuristic:* Various steps to be performed are:

a) For each task determine the difference between its minimum and second minimum completion time over all the machines.
b) For each task determine the difference between its minimum and second minimum completion time over all the machines.
c) Over all the tasks find the maximum suffrage.
d) Assign the task to the machine that has resulted in obtaining minimum completion time.
e) Iterate till all the tasks are scheduled.

12) *Resource Awareness Scheduling Algorithm [7]:* Resource Awareness Scheduling Algorithm is a hybrid algorithm. It is a combination of Max-Min and Min- Min Algorithms. This algorithm is also known as RASA. RASA is based on the concept of Completion Time of each task. In Resource Awareness Scheduling Algorithm, for each task Expected Completion time is calculated. To schedule task, apply Max-min and Min-Min alternatively. If number of resources is even, apply Max-Min strategy first otherwise Min-Min strategy.

13) *Reliable RASA Scheduling Algorithm [8]:* To provide required QoS to user a resource reliability parameter is added to standard algorithms. Reliability plays a significant role in performance of grid. Sometimes performance in view of completion time is high but reliability of resource might be low. It is important to allocate tasks to highly reliable resources because low resource reliability indicates frequent resource failures. In this algorithm, resource reliability is provided in percentage. Resource manager sets reliability criteria. Appropriate resources which fulfill the criteria are selected. The result after applying this algorithm is risk associated with job failure reduced.

It works in three phases:

*a)* Expected Completion time is calculated for each task.
*b)* Average or closest greater than average task is selected.
*c)* Selected task is scheduled over resource with minimum completion time i.e., with slowest resource.
*d)* For each task determine the difference between its minimum and second minimum completion time over all the machines.
*e)* For each task determine the difference between its minimum and second minimum completion time over all the machines

14) *Genetic Algorithm [9]:* The Genetic Algorithm mainly works in eight phases:

*a)* *Genetic Encoding:* Two-dimension coding is the coding of the population individual.
*b)* *Genetic Decoding:* The decoding scheme of the encoded chromosomes is, the first char in genetic encode is decoded directed as the host resource. Then, the following chars are decoded as the tasks order that are scheduled on this host resource.
*c)* *Initial Population Generation:* First genetic individual of initial population is generated.
*d)* *Fitness Function:* To measure the quality of the population individual, the fitness function of the population is used. Deadline, budgets etc can be considered.
*e)* *Genetic Crossover:* Obtain new individuals in the current population by combining and rearranging parts of the existing individuals. A crossover probability is selected to bring a better population individual by combining two fittest individuals.
*f)* *Genetic Mutation:* Two genetic mutation operations such as exchange mutation and replace mutation present. To select randomly a host and select two tasks on this host to exchange, exchange mutation is used. To reallocate available hosts to tasks in the population individual, replace mutation is used.
*g)* *Genetic Selection:* The Roulette wheel is used to implement the population individual.
*h)* *Genetic Termination:* The termination condition of the genetic operation is setting the maximal iteration number.

Using GA for tasks scheduling, each job vector is represented by chromosomes and the positions in this vector are tasks. The population shows various mappings for tasks to machines and GA performs heuristic search to find the optimal solution. The fitness function measures the quality of solution. Usually, algorithm imitates the mechanism of natural selection strategy which consists of four steps Selection crossover mutation and evaluation.

The main disadvantages are complexity in computations and long-time requirement. The accuracy of algorithm can be decreased by trial/error.

15) *Particle Swarm Optimization Algorithm:* PSO is a type of meta- heuristics algorithms. This algorithm applies self-adaptive global search for optimization. It starts with random initialization for position and velocity for the practices population. Looking at the problem of task scheduling, tasks are considered as the particles and number of tasks in the workflow is the dimension of these particles. Each dimension has a value that indicates the resource where the tasks workflow is heading. So, the mapping between tasks and resources is shown by a particle in PSO. Like GA each particle is evaluated using fitness function. The traffic workload using PSO is balanced. As it could be used with any number of tasks and resources, this algorithm is scalable. It can find near optimal solutions for mapping all tasks in the workflow to the set of available resources. Less use of mathematical operators compared with GA and consequently less need for parameters tuning. This algorithm is Simple and effective. This can be used in wide applications with little computation overhead compared to GA. The major disadvantages are, it is easy to fall into local optimum in large search space and slow convergence.

16) *Cloud Task Scheduling Based on Ant Colony Optimization [10] [11]:* ACO mainly simulate the food searching behavior of ant colonies. While searching for food ants use kind of chemical for communicating with each other. This Chemical is called pheromone. In the beginning ants search for their foods randomly, once they find a food source, they leave pheromone on the path. Any ant can reach the food source by following the trail of pheromone. As this process continues, ants try to find the shortest path as there is large amount of pheromones accumulated on the way. Some of the advantages of this algorithm are they use positive feedback mechanism, inner parallelism and this algorithm is extensible. The disadvantages are stagnation phenomenon, all individuals find the same solution, searching to a certain extent, unable to search for a solution space, making the algorithm reach a local optimal solution.

ACO can be applied to any problem if it defines

*a)* The problem representation allows ants to incrementally build or modify solutions.
*b)* A constraint satisfaction method which forces the construction of possible solutions.
*c)* A pheromone updating rule has to be there which shows how to modify trails on the edges of the graph.
*d)* A transition rule of the heuristic desirability and of pheromone trail.

17) *A New Flower Pollination Based Task Scheduling Algorithm in Cloud Environment [12]:* For scheduling tasks in cloud environment here uses nature-inspired algorithm called Flower pollination Algorithm. This task scheduling approach is proposed in order to map tasks and resources in best optimized way, thus minimizing make-span as a consequence. Set of tasks of different completion times and set of resources of different processing powers are considered. In this algorithm, flowers are single solutions. An initial population is randomly generated. To decide between local pollination and global pollination a Switch probability is considered. For local pollination, the probability value is more. Global pollination and local pollination can be calculated using the single solutions. If switching probability is less than a random number generated then global pollination is applied, otherwise local pollination is applied. Then evaluate new solutions as per fitness Function. If new solutions are better than existing ones then update them in population, otherwise ignore.
18) *A Hybrid Bio-Inspired Algorithm for Scheduling and Resource Management in Cloud Environment [13]:* Cloud receives client's tasks in a random manner. Allocation of resources to these tasks must be handled in an intelligent way. In this method, first allocate the tasks to the virtual machines in an efficient manner. This is done using Modified Particle Swarm Optimization algorithm. Then allocation or management of resources, as demanded by the tasks, is handled by the Hybrid Bio-Inspired algorithm. Hybrid Bio-Inspired algorithm is a new hybrid approach. This incorporates two existing approaches such as Modified Particle Swarm Optimization (MPSO) and Modified Cat Swarm Optimization (MCSO). This approach focuses on two goals. First one is to provide efficient load balancing in task scheduling by building enhanced PSO algorithm. Second one is to provide dynamic resource allocation and resource management by building hybrid approach using MPSO and MCSO algorithms.
19) *Monkey Search Algorithm for Task Scheduling in Cloud IaaS [14]:* This Monkey Search Algorithm is a dynamic learning inspired task allocation algorithm. This algorithm is used to minimize the overall completion time and to improve resource utilization. Existing algorithms are mainly static or dynamic in nature. They may suffer from network traffic delay which leads to request failure. Network aware monkey search algorithm with fine execution time, with lease execution cost and network delay will provided better QoS than existing static, dynamic and learning based algorithms. This monkey search algorithm for task allocation is divided into 4 phases such as Initialization, Climb, Watch and Jump and Somersault. In initialization, initial population is generated using a set if tasks and VMs. In climb phase, monkey designated as cloudlet. The mountain is designated as vm. The monkey will climbs/schedule on the mountain whichever comes first in its path. In Watch and Jump phase, monkey searches in its local domain to find most optimal solution. This can be related as when a monkey gets on the top of the mountain, it will check whether other mountains around it higher than its present positions. If present, then jump to that higher mountain. In somersault phase, each monkey will find a locally maximal mountaintop around its initial point i.e., least execution cost path is found and monkey rolls down from the mountain. In future better output can be produced using this algorithm and network bandwidth can also be considered.

20) *Bacteria Foraging Based Task Scheduling Algorithm In Cloud Environment [15]:* For cloud resource scheduling, this algorithm is an optimization technique based on bacterial foraging. The scheduling parameters used are number of hosts, number of cloudlets, bandwidth and number of VMs per machine. The main steps are, a virtual machine list is obtained from the data center after the provisioning of request of the user. Then a random appropriate solution and task lists are initialized. Choosing the optimal heuristic task is started from the low level. Here each micro-organism from the large number of microorganisms represents an initial solution. Chemotaxis helps microorganisms to construct heuristic steps. At every decision point a fitness function health is computed. Health will compute and swimming process will start until bacteria have not climbed to long. This swimming process is continued until the cost has been reduced.
21) *Dynamic cloud task scheduling based on a two-stage strategy [18]: A* technique based on a two-stage strategy to reduce the non reasonable task allocation and increase the task scheduling performance in clouds. At the first stage, a job classifier motivated by a Bayes classifier's design principle is utilized to classify the tasks based on historical scheduling data. A definite number of virtual machines of the various types are accordingly created. During second stage, the tasks are assigned with various virtual machines dynamically.

All these papers deal with the ways of allocating tasks to VMs in different manners. This does not consider the time taken to create VMs during scheduling. This ignorance of VM creation time can lead to an increase in the time cost of the overall process. All these methods do not consider the priority of tasks in an efficient manner. The proposed method offers higher guarantee ratios of all types of tasks i.e., it will consider both jobs of a high paying user and regular jobs. It also gives necessary priorities to the jobs.

## III. PROBLEM STATEMENT

Scheduling virtual machines for user requests in cloud computing is a NP-hard problem. This problem is usually solved by using heuristic methods in order to reduce to polynomial complexity. In this process, the cloud computing scheduler acquire the tasks from the users and assigns them to available resources taking into consideration tasks' attributes, and requirements such as length, deadline, waiting time etc., and the resource parameters and properties. The main challenge in cloud scheduling is due to dynamic nature of cloud, heterogeneous parameters should be considered for scheduling.

## IV. PROPOSED SYSTEM

The scheduling processes are computationally cost expensive. The traditional algorithms do not consider resource utilization levels of Virtual Machines in the past. The proposed work aims to increase the task scheduling performance and reduce non reasonable task allocation in cloud. This paper considers already processing VM resource usage over time. This is done by analyzing past VM utilization levels. Neive bayes technique is used for this analysis. In general, the proposed work aims to prioritize the task list based on multiple criteria and assign an appropriate resource to the task. The objective is to propose the concept of VM scheduling according to data extracted from past VM utilization levels by using classification technique such as Neive bayes in order to classify tasks by optimizing performance. The algorithm enhances the VM selection phase by analysis of virtual and physical resources. Our aim is to increase strength of VM scheduling.

### *A. Neive Bayes Classifier*

A machine learning method that is used to distinguish different objects based on certain features is a classifier. A machine learning method that is used for the classification of tasks is a Naive Bayes classifier. The essence of the classifier is based on the Bayes theorem and it is used when the dimensionality of the input is maximum. For example, a fruit might be advised to be an orange if it is orange in color, round in shape. Even though if these attributes are depending on each other or depends upon the existence of other features of a class, a naive Bayes classifier takes these properties as independent contributions to the chance that the fruit is an orange. Using Bayes theorem, we can find the chance of a task happening based on the historical data. We can also find the matching of a task and a Virtual Machine.

### *B. Framework for Scheduling*

A set of Virtual Machines present. Each Virtual Machines has four attributes. These are CPU resources such as clock speed of CPU, memory resources, network bandwidth, and hard disk storage. A Set of task T present. Parameters of tasks are task ID, requirements of task, deadline and priority of task. The task ID is unique for all tasks. Requirements of tasks specify each task's requirements for CPU, memory, network bandwidth and hard disk storage. After scheduling, if deadline is violated, then that scheduling is considered as fail. If the task is a job of a high paying user, then it is of high priority and the value of priority is set as 1. If it is a regular job, then the value of priority is set as 0. In the scheduling process, first the tasks having high priority is selected and then regular jobs are selected. This will guarantee a high priority ratio.

A Set of hosts present in the Datacenter. These datacenters are the physical hardware infrastructure. Hosts are the physical computing nodes. Tasks are given to the hosts. VMs are generated from the hosts by virtualization. This host wills instantiate the VM scheduler and it will also manage Virtual Machines.

The function of mapping tasks to VMs is called task scheduling. After successful mapping, each task is scheduled and executed at a suitable VM. To reduce the complexity of mapping, divide tasks to fit to VM types. Each successfully created VM belongs to a VM type. The classifier is used for the mapping of tasks and Virtual Machine types. The task scheduling difficulty is reducing as the VM types number can be much smaller than that of VMs.

## V. THREE STAGES

At the first stage, classify tasks based on historical scheduling data. Then VMs of different types are created. This will save the time for creating VMs during task scheduling. At the second stage, tasks are marked with corresponding VM types. At the third stage, tasks are matched with concrete VMs of marked types and are scheduled and executed in the hosts.

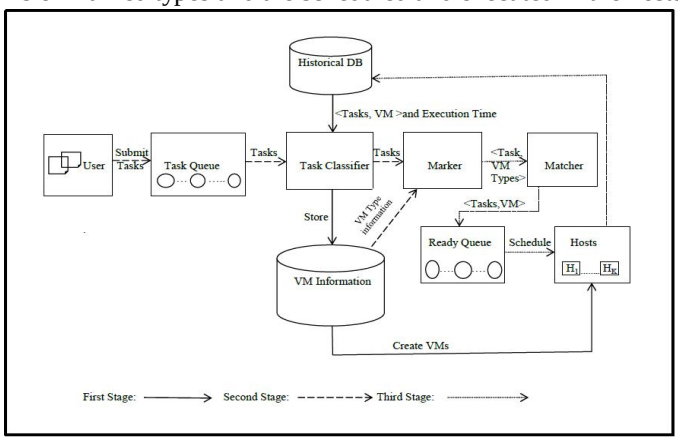



Fig 1: Task scheduling framework based on a three-stage method

### *A. First Stage*

Based on historical task scheduling information, we obtain historical data. A task classifier classifies tasks. Then create a set of VM types and create a proper number of VMs of different types at hosts.

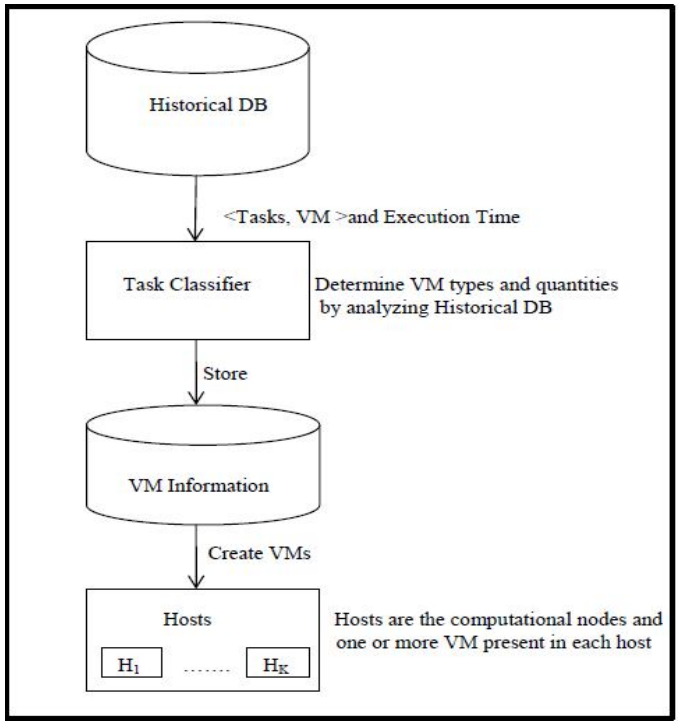



Fig 2: Task scheduling framework - First Stage

*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429*
*Volume 9 Issue III Mar 2021- Available at www.ijraset.com*

*1) Algorithm:* Create VM Types And Create VMS at Hosts

*a) Input:* Historical scheduling data.

*b) Output:* VM Types.

- *Types*← ∅;
- T ← Historical scheduling data;
- L ← Task types count of T ;
- For i =1 to L
- Compute $P(T_i)$ ;
- *Types*← create VM types;
- $v_i$ ←create VM i;
- Output *Types*;

$P(T_i)$ is calculated to get an idea about the count and range of attribute values in a task type. This is the mean value of count of different types and its attributes.

As shown in the algorithm, first we obtain historical data based on the historical task scheduling information. Then we propose a task classifier to classify tasks and store them in a database and create a set of VM types. Then we create a proper number of VMs of different types at hosts.

*B. Second Stage*

The user submits tasks to the system. The task classifier takes the task on the basis of priority. Tasks having high priority are first considered. Then the classifier fetches the VM type information that are created at the first stage. Then each task is marked with suitable VM types.

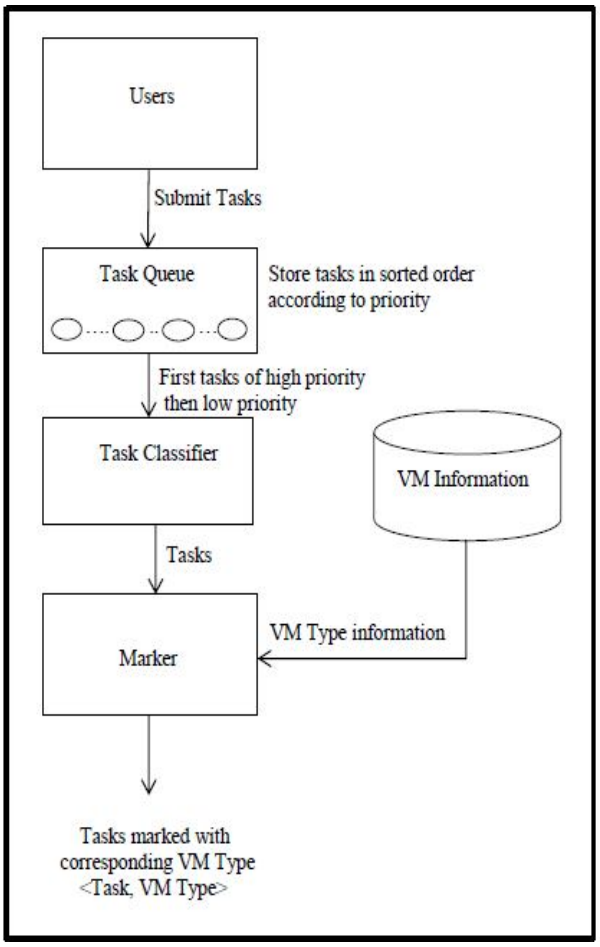



Fig 3: Task scheduling framework - Second Stage

*1) Algorithm:* Mark Task
*a) Input:* Types and initial task set, types come from algorithm 1
*b) Output:* Information of marked tasks

- Initialize information of marked tasks as null set.
- Let k be the number of types
- Rank the tasks on the basis of priority in descending order and store it in a list
- For i=1 to k, compute the matching degree value
- Assign the tasks to VMs on the basis of matching degree
- If matched VM present, set delta as 1
- Otherwise create new VM type using the values
- End
- Output: marked VM task pair

*C. Third Stage*

The marked tasks are matched with corresponding VMs of its proper type. More than one tasks may be assigned for a Virtual Machine. For scheduling Shortest Job First algorithm is used. Then the tasks are scheduled and executed at VMs. The details of scheduling are stored as historical data for future VM creation and the results are returned to the users.

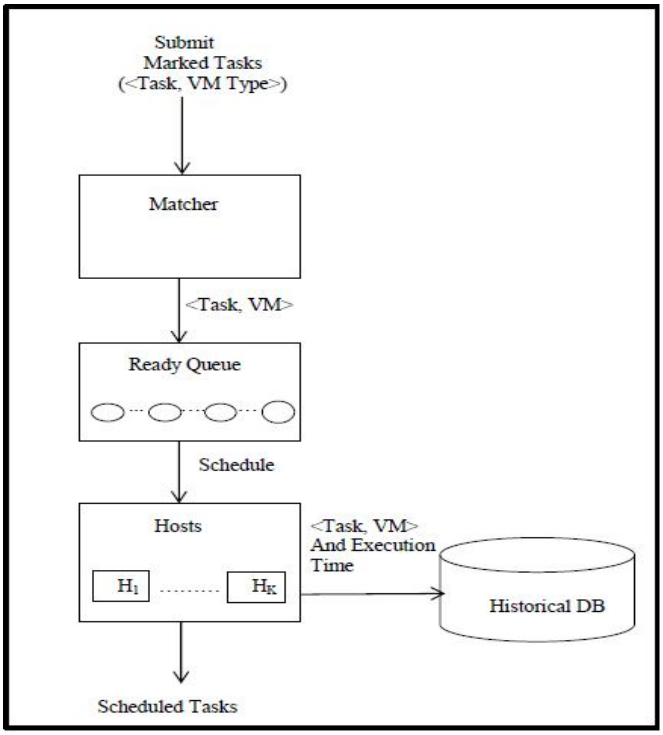



Fig 4: Task scheduling framework - Third Stage

*1) Algorithm:* Match Task
*a) Input:* Marked pair values from algorithm 2
*b) Output:* Matched vm and tasks

- If free vm present, set status as 1
- If sequence of ready not full, set ready as 1
- If status=1 and delta=1 and ready=1, add the vm, task pair to the lists
- Else create new vm
- Update task scheduling information
- End

## VI. EXPERIMENTS

*A. Experimental Setup*

Experimental environment includes CPU (AMD quad core, 2.4 GHz), memory (8.0 GB), HD (500G), Windows 10 operating system, NetBeans IDE 8.2, JDK8.0, and CloudSim. As a cloud computing simulation platform, Cloudsim is widely used. Based on it, the scheduling process can be carried out.

The implementation inherits and extends some of the existing classes, such as Vm, Datacenter Broker, Cloudlet, and Host.

*B. Experimental Results and Analysis*

*1) The Proposed Method*

*a)* Decrease makespan and average waiting time.

*b)* Failure rate is lower.

*c)* Workload of VMs of each VM type is better balanced.

*d)* The proposed method offers higher guarantee ratios of priority and ordinary tasks.

*2) Evaluation Parameters*

*a)* Makespan

- It is total time that tasks are scheduled and completed in clouds.

*b)* Average Waiting Time of Tasks

- It represents performance of overall processing capacity and throughput of the clouds.

*c)* Utilization Rate of VMs

For simulation, consider tasks data set of size 100,200,300 and 400.All these groups contains tasks of all complexities ranges from high to low. For each set of tasks, apply Max-Min, Two Stage Strategy Shortest Job First (TSS-SJF), Min-Min, Shortest Job First (SJF) and Two Stage Strategy Dynamic Cloud Task Scheduling (TSS-DCTS).

In Fig 5, we can see that makespan of Max-Min, Min-Min, Shortest Job First(SJF) and Two Stage Strategy Dynamic Cloud Task Scheduling (TSS-DCTS) are high when compared to the proposed method of two stage strategy with SJF in all cases. Makespan of the proposed method is the smallest and it is superior to the other two methods.

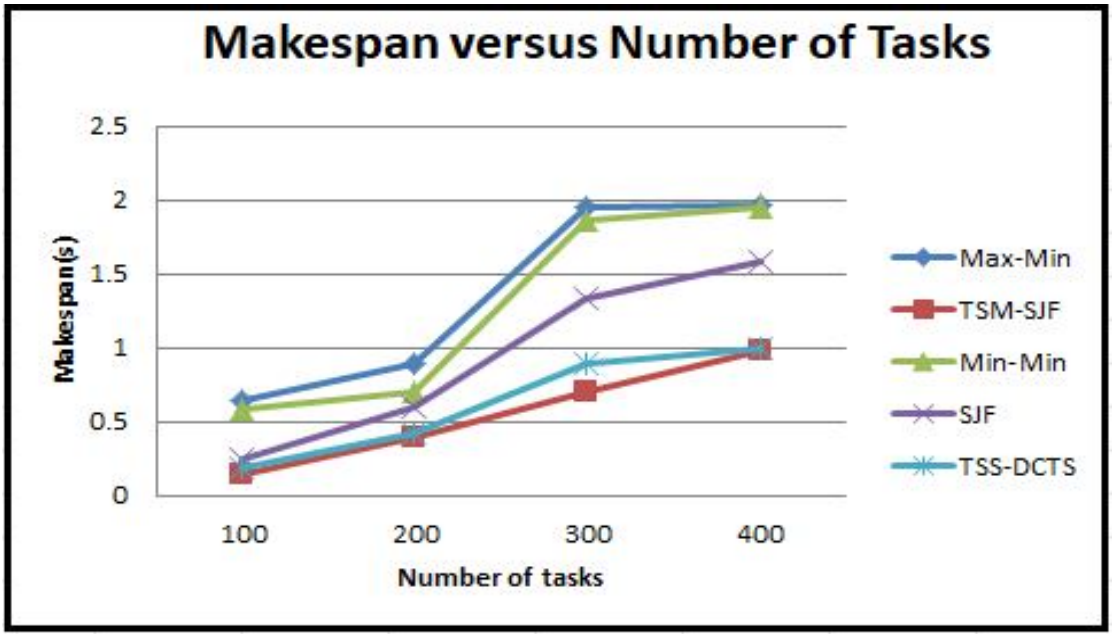



Fig 5: Makespan versus Number of Tasks

In Fig 6 all methods show their upward trend of waiting time when the number of tasks increases. Compared to other methods, the average waiting time is lower for the proposed two stage strategy SJF.

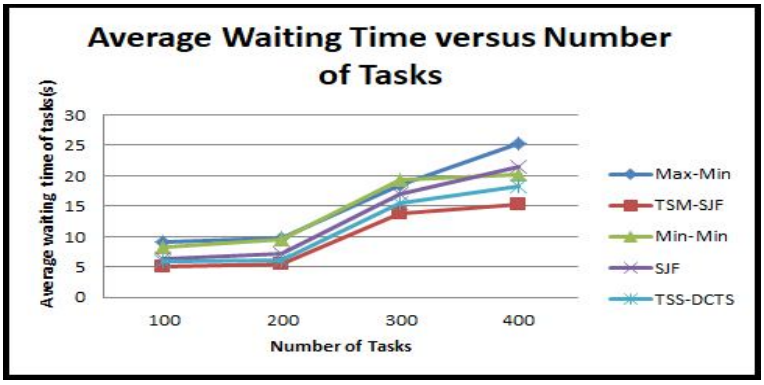



Fig 6: Average Waiting Time versus Number of Tasks

In Fig 7 the utilization rate VMs of the proposed method is basically flat. The same disadvantage of other methods is that is utilization rate fluctuates and hence workload is not balanced. When the number of tasks increases the utilization rate of VMs also increases in other methods.

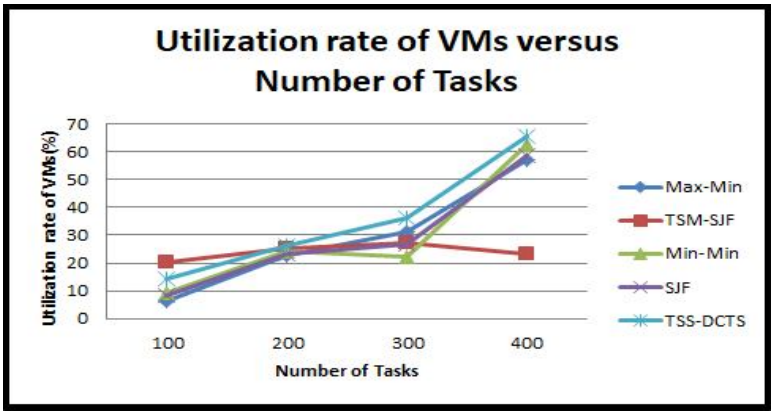



Fig 7: Utilization rate of VMs versus Number of Tasks

Task types indicate task complexity. They also represent task scale from low to high. The task data set size is 1000 in the next analysis. Divide the experimental data into ten groups. Each group's data size is 100. All the ten groups contain tasks of complexities from high to low. Apply TSS- FCFS, TSS-SJF and TSS- RR in all sets. Compute average values based on ten experimental results. Results are collected and average values are calculated. As a result, we can obtain makespan, average waiting time of tasks and utilization rate of VMs with the improved accuracy.

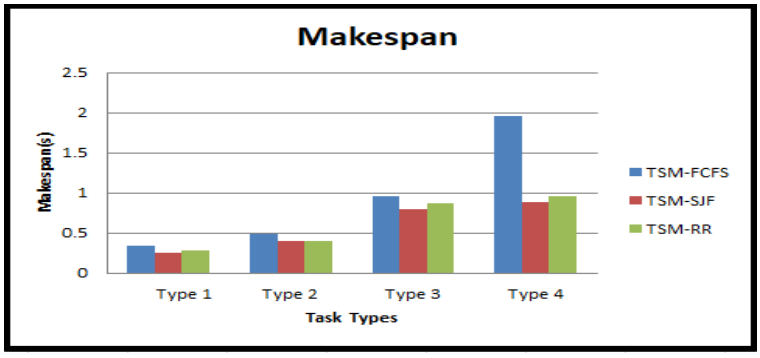



Fig 8: Makespan versus Task Types

From Figs. 8-10, we can see that the proposed technique can scale well to tasks of all types. In Fig. 8, we can conclude that: the makespans of all methods increase when task types vary from 1 to 4 and the proposed method can achieve the smallest makespan in comparison with other two methods.

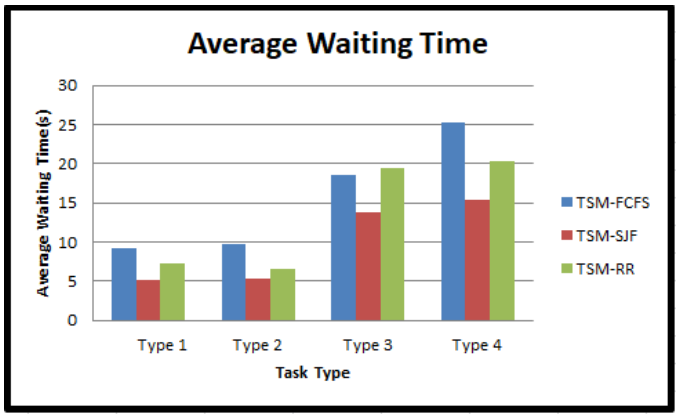



Fig 9: Average Waiting Time versus Task Types

Fig. 9 shows average waiting time obtained via three methods. The waiting time increases for TSS- FCFS and TSS- SJF methods when tasks become more complex. While a fluctuation in average waiting time occurred in case of TSS-RR when the task becomes more complex. The proposed method has the smallest average waiting time among three methods for all task types.

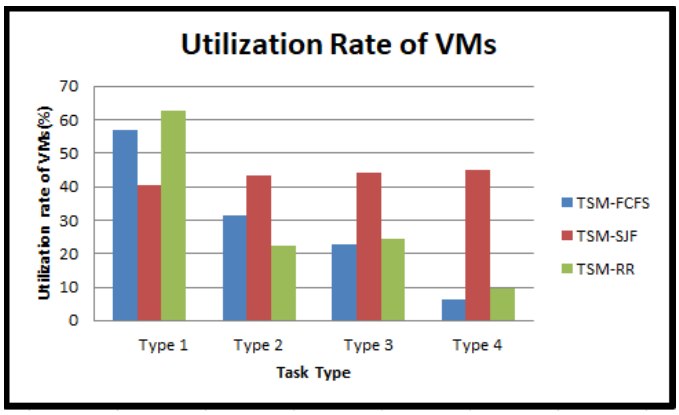



Fig 10: Utilization Rate of VMs versus Task Types

Fig. 10 shows that the utilization rate of VMs is flat when the proposed method is used, thus implying very balanced workload among all VMs. The disadvantage of other two methods is that higher type VMs have much lower utilization rate, while lower type VMs have very high utilization rate. Clearly, the workload of these two methods is not well balanced. Hence, the proposed method is superior to other two methods in terms of workload balancing.

## VII. CONCLUSION

Resources with heterogeneous characteristics are served virtually in the cloud computing environment. Efficient scheduling is very important to manage this type of heterogeneous resources in an optimized way. Various scheduling algorithms and methods are present for efficient scheduling. This paper deals with a three-stage method using Bayes classifier framework to achieve virtual machine creation and desired task classification and improves service quality of the clouds. Based on historical task scheduling information, virtual machines are precreated. This will save the time of VM creation during scheduling. In this way makespan and average waiting time can be decreased. Tasks are scheduled at the most suitable VMs on the basis of it's complexity.

## REFERENCES

[1] Ruba Abu Khurma, Heba Harahsheh and Ahmad Abdel-Aziz Sharieh, "Task Scheduling Algorithm in Cloud Computing Based on Modified Round Robin Algorithm," Journal of Theoretical and Applied Information Technology ,, 2018.

[2] S.Devipriya and C.Ramesh ,"Improved Max-Min Heuristic Model For Task Scheduling in Cloud", International Conference on Green Computing, Communication and Conservation of Energy (ICGCE),, 2013.

[3] HE XiaoShan, SUN XianHe and Gregor yon Laszewski,"QoS Guided Min-Min Heuristic for Grid Task Scheduling", Journal of Computer Science and Technology, 2003.

[4] Kobra Etminani and Prof. M. Naghibzadeh, "A Min-Min Max-Min Selective Algorithm for Grid Task Scheduing", Third IEEE/IFIP International Conference on Internet, Uzbekistan, 2007.

[5] O.M.Elzeki, M.Z.Reshad and M.A.Elsoud, "Improved Max-Min Algorithm in Cloud Computing", International Journal of Computer Applications, July, 2012.

[6] Upendra Bhoi1and Purvi N. Ramanuj, "Enhanced Max-Min Task Scheduling Algorithm in Cloud Computing,", International Journal of Application or Innovation in Engineering Management, 2013.

[7] Saeed Parsa and Reza Entezari-Maleki, "RASA:A New Grid Task Scheduling Algorithm,", International Journal of Digital Content Technology and its Applications, 2009.

[8] Sunilkumar Nakum, C. Ramakrishna and Amit Lathigara, "Reliable RASA Scheduling Algorithm for Grid Environment,",2014 IEEE International Conference on Computer Communication and Systems, 2014.

[9] Yang Cui and Zhang Xiaoqing, " Workflow Tasks Scheduling Optimization Based on Genetic Algorithm in Clouds ",, 3rd IEEE International Conference on Cloud Computing and Big Data Analysis, 2018.

[10] Airo Alonso Giraldo and M. Passino, "Honey Bee Social Foraging Algorithm for Resource Allocation", Springer Handbook of Computational Intelligence, 2015.

[11] H.W.Hashem and R. Rizk, "Honey Bee Based Load Balancing in Cloud Computing,", Transaction on Internet Information System, 2017.

[12] Jaspinder Kaur and Brahmaleen Kaur Sidhu, "A New Flower Pollination based Task Scheduling Algorithm in Cloud Environment", 4th International Conference On Signal Processing Computing And Control, 2017.

[13] Shridhar Domanal, Ram Mohana Reddy Guddeti and Rajkumar Buyya, "A A Hybrid Bio-Inspired Algorithm for Scheduling and Resource Management in Cloud Environment", IEEE TRANSACTIONS ON SERVICES COMPUTING, 2016.

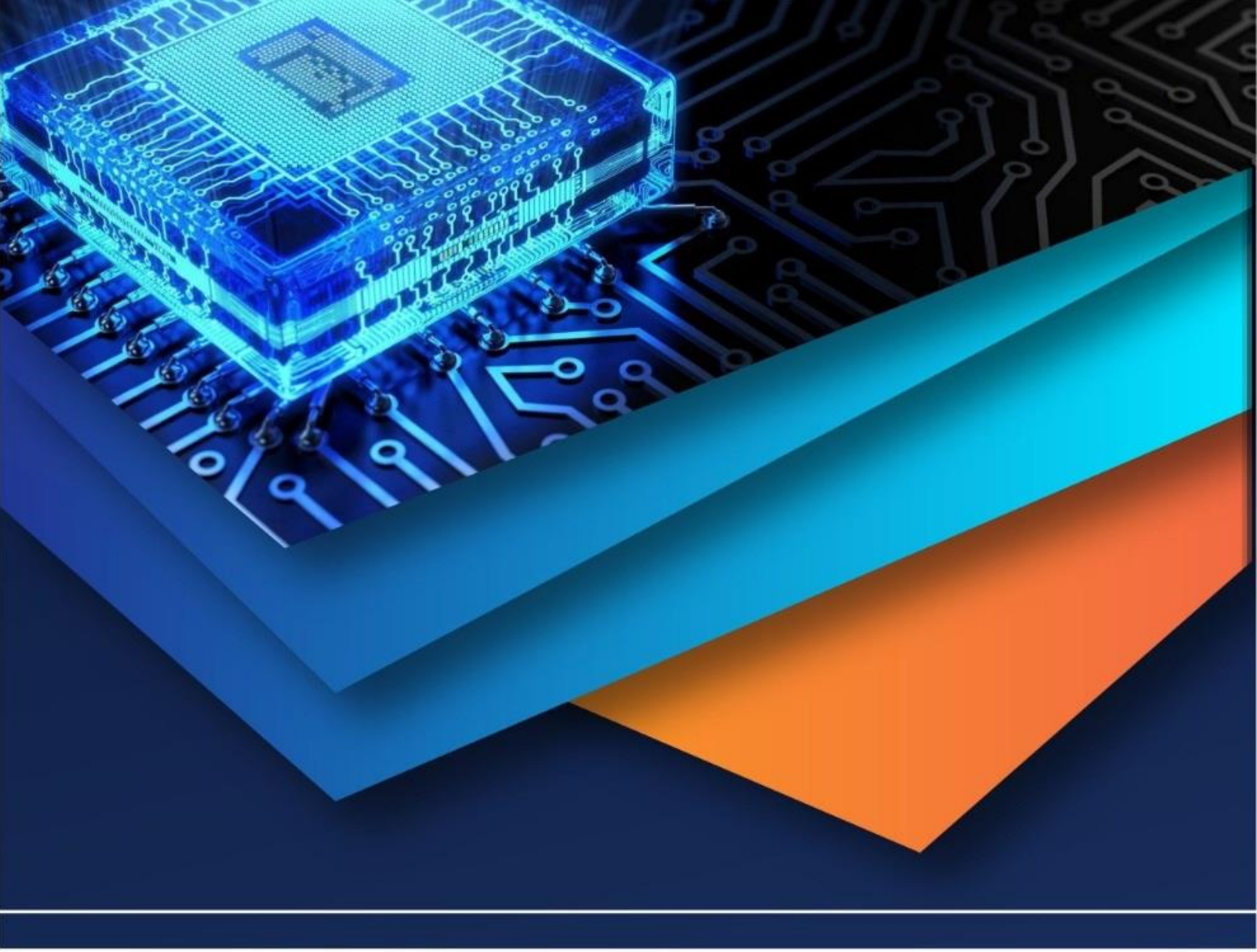
[14] Punit Gupta and Prateek Tewari, "Monkey Search Algorithm for Task Scheduling in Cloud IaaS",Fourth International Conference on Image Information Processing, 2017.

[15] Juhi Verma, Srichandan Sobhanayak and Suraj Sharma,"Bacteria Foraging Based Task Scheduling Algorithm in Cloud Computing Environment", International Conference on Communication and Automation, 2017.

[16] Sudip Roy,Sourav Banerjee,K.R.Chowdhury and Utpal Biswas, "Development and Analysis of a Three Phase Cloudlet Allocation Algorithm,", Journal of King Saud University - Computer and Information Sciences, 2017.

[17] Xiaomin Zhu, Chao Chen, Laurence T. Yang and Yang Xiang, "ANGEL: Agent-Based Scheduling for Real-Time Tasks in Virtualized Clouds", IEEE Transactions on Computers, 2015.

[18] PeiYun Zhang and and MengChu Zhou, "Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy," IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, 2018.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY