International Journal of Computing and Digital Systems

Improving The Performance Of An Advanced Algorithm For Taskflow Scheduling In Cloud Computing

Narayana N Gourav[1], Dr. Dayananda R B[2] and Akshatha Kamath[3]

[1,2,3] Computer Science and Engineering, M S Ramaiah Institute of Technology, Bangalore, India

E-mail address: narayana.n.gourav@gmail.com, dayanandarb@msrit.edu, akshathakamath@msrit.edu

Abstract: The domain of cloud computing with an emphasis on taskflow scheduling optimisation. It emphasises the benefits that cloud resources provide for the both business and research endeavours. The study revolves around these scheduling algorithms, especially the Max-Min algorithm and its improved version, which are essential for optimising resource use and reducing work runtime. It can minimize the makespan, throughput, average waiting time and average response time of workflow executions. These methods optimise resource allocation and improve overall performance by dynamically allocating jobs based the on current workload requirements and real-time data. The algorithm is tested out using CloudSim, the results shown that the Improved Max-Min algorithm has achieved better in almost of the cases considered. Smart scheduling methods can make a big difference in how well cloud resources are used. This leads to cloud networks that are more efficient and can handle the ever-changing needs of today's IT systems. The Improved Max-Min algorithm's ability to prioritize important tasks and change how resources are allocated can make the system run much better and improve its performance. This research adds to our understanding of how to make the most of cloud resources. Studies show that the Improved Max-Min algorithm works well. This study shows how important it is to use advanced scheduling algorithms in cloud computing. These algorithms help use cloud resources better and improve taskflow management.

Keywords: Cloud Computing, Taskflow, Scheduling Algorithm, Improved-Max-Min, Max-Min, FCFS, SJF, Round-Robin, Makespan, Throughput, Average Waiting Time, Average Response Time.

1. INTRODUCTION

Cloud computing has emerged as one of the crucial component of IT system in recent years. It effectively supplies scientific and commercial users with IT resources and is essential to the management of scientific workflow applications across dispersed platforms. Companies can easily lease cloud services for storage and processing, which results in significant cost savings. The cloud computing resources are very helpful for scientific applications that need to process large amounts of data, such image processing and seismic simulation.

Achieving the best possible mapping of workflow activities to cloud resources requires the creation of scheduling algorithms, which is a major topic in cloud computing. Although there are number of algorithms, such as the Max-Min and other algorithms, have been presented, there is still opportunity for development in order to reduce the execution time of workflows.

Based on the Max-Min algorithm, an enhanced version of this algorithm has been studied. It typically outperforms Max-Min because it increases resource utilisation while reducing the overall job execution time (makespan) of a cloud process.

Advanced algorithms are included into cloud computing workflow scheduling systems to improve performance and make cloud infrastructures more efficient and economical overall. Through dynamic task allocation determined by workload demands and real-time data, these algorithms are able to adjust to changing circumstances and guarantee the best possible resource utilisation and responsiveness. The algorithm prioritises key tasks and allocates resources more efficiently since it can take into account variables like load balancing, task dependencies, and resource availability. As a result, process execution is faster and more dependable. It also cuts down on the possibility of resource bottlenecks and contention.

Research demonstrates how crucial intelligent scheduling is to optimising the advantages of cloud resources and laying the groundwork for cloud infrastructures that are

E-mail:narayana.n.gourav@gmail.com, dayanandarb@msrit.edu, akshathakamath@msrit.edu

more responsive and efficient and can keep up with the changing needs of contemporary IT environments.

Without a question, cloud computing has become a fundamental component of contemporary IT infrastructure, providing a revolutionary method of allocating and overseeing resources for both academic and business purposes. The administration of scientific workflow applications has been completely transformed by its capacity to deploy cloud computing resources across distributed platforms in an efficient manner. This is especially true for applications that require considerable data processing, such seismic modelling and image analysis.

Mapping workflow operations to available computer resources effectively is one of the key issues in making the most of cloud resources. This calls for the creation of complex scheduling algorithms, a crucial area for cloud computing research. While already-existing algorithms, such as Max-Min, have set the groundwork, further workflow execution time reduction requires constant innovation and refining.

Extensive research has been conducted on improved variants of this Max-Min algorithm, which has demonstrated better performance through improved resource utilization and reduced the job execution times. These developments highlight how cloud computing is always evolving, mostly due to the incorporation of these sophisticated scheduling algorithms into workflow management systems.

These sophisticated algorithms make use of dynamic task allocation techniques to react instantly to changes in data and workload needs. They reduce bottlenecks and conflict by giving priority to important activities and allocating resources optimally, which results in quicker and more dependable process execution.

Empirical studies validate the critical function of intelligent scheduling in fully utilising cloud resources, establishing the foundation for adaptable and effective cloud infrastructures that can satisfy the ever-changing needs of modern IT settings.

To sum up, the development of scheduling algorithms in cloud computing not only improves the efficiency with which tasks are completed and the use of resources, but it also serves to highlight the role that cloud technology plays in fostering innovation in a variety of fields. The promise of cloud computing as a disruptive force in IT is still very much alive, as research into optimisation continues to push the envelope.

A. *Task Scheduling In Cloud Environment*

The set of rules controlling how tasks are executed in a computer system's order is referred to as scheduling. Within cloud data centres, tasks are scheduled according to a set of guidelines and standards that determine which jobs, from a pool of available tasks, will be carried out at any particular moment. In cloud data centres, the work assigned to the task scheduling algorithm is to distribute the jobs submitted by user from the available resources.

Increasing system throughput and improving computing speed are the main goals of task scheduling methods. Optimising the resource utilisation is made possible by efficient scheduling, which also makes efficient control of CPU and memory resources possible. For the system to operate as efficiently as possible, strong scheduling policies must be put in place.

As an intermediary between users and providers, the Data Centre Broker receives jobs from consumers in the cloud computing industry. Task scheduling for Virtual Machines (VMs) in the Data Centre is managed by this broker. The Data Centre is a virtual infrastructure made up of many hosts. In order to allocate jobs to virtual machines (VMs), the Data Centre Broker works directly with the Cloud controller to identify various policies that govern task scheduling. There are several different types of scheduling, including the centralised or the distributed, instantaneous or batch, static or dynamic, and primitive or nonprimitive. In accordance with user needs, task scheduling seeks to minimise completion times and maximise resource utilization.

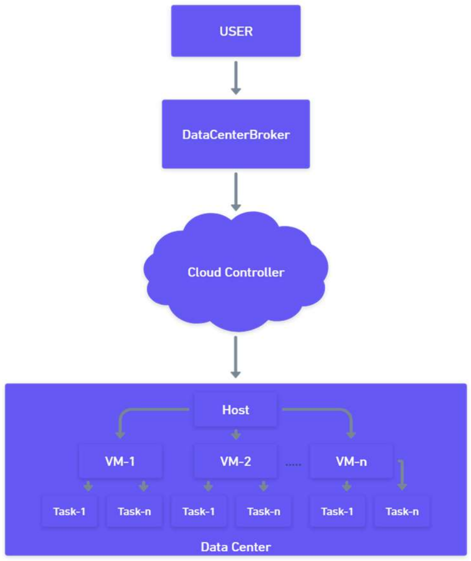

Figure 1. Task Scheduling Structure.

When discussing or considering of cloud systems, scheduling refers to the collection of guidelines controlling the sequence in which tasks are completed. Task scheduling in cloud data centres means following a predetermined set of rules and specifications that determine which jobs are performed at any given time from a pool of tasks that are available. In cloud data centres, the task scheduling algorithm is essential to the way user-submitted workloads are distributed among the resources.

The main goals of task scheduling techniques in cloud environments are to increase processing speed and system throughput. Efficient scheduling makes it easier to achieve optimal resource utilisation, which in turn makes it possible to manage CPU and memory resources effectively. Strong scheduling policies need to be put in place to guarantee the best possible system performance.

In the cloud computing sector, the Data Centre Broker plays a crucial function as a go-between for customers and service providers. This broker coordinates the scheduling of tasks for Virtual Machines (VMs) in the data centre. With its many hosts, the data centre functions as a virtual network infrastructure. The Cloud controller and the Data Centre Broker work closely together to set up different policies controlling task scheduling and assign jobs to virtual machines (VMs).

There are several different kinds of scheduling methods, such as distributed or centralised, batch or instantaneous, static or dynamic, and primitive or non-primitive. Whatever the kind, the general goal of task scheduling is always the same: to reduce project completion times and optimise resource use while keeping user requirements in mind.

To put it simply, efficient task scheduling systems are critical to coordinating the effective use of cloud resources, guaranteeing peak system performance, and satisfying the various demands of users in the ever-changing cloud computing environment.

2. RELATED WORK AND BACKGROUND

A. Task Scheduling Strategies

Cloud computing is a quickly developing technology that keeps getting better every day. Even though cloud computing's scheduling issue presents difficulties, continued study and innovation lead to the creation of increasingly effective algorithms.

In general, no algorithm can find an optimal solution to the scheduling problem in polynomial time. On the other hand, some algorithms can yield good results and operate in polynomial time. One approach that can reduce the makespan, for instance, is the Max-Min algorithm.

Because cloud computing is dynamic, new scheduling techniques must be constantly explored in order to meet changing needs. Although reaching optimality in polynomial time is still a challenge, the search for effective algorithms propels this area forward. One example of how scheduling effectiveness might be improved in cloud systems is the Max-Min algorithm.

Many scheduling algorithms have been developed over time to optimise different jobs in this field. We'll discuss some of these task scheduling strategies below.

Max-Min: In distributed computing environments, especially in grid and cloud computing systems, Max-Min is a task scheduling technique. With a fair distribution of resources, Max-Min seeks to maximise the minimum utility attained by each activity in order to optimise resource allocation. Tasks given resources in Max-Min scheduling are determined by their utility values; tasks with higher utility have priority access to resources. This guarantees that the most resource-demanding tasks are properly completed, which enhances system performance as a whole. Fairness is further enhanced via max-min scheduling, which keeps lower utility jobs from experiencing resource depletion. Max-Min scheduling, however, may have more of these computational overhead than other scheduling methods since it necessitates precise estimation of job utility values, which can be difficult to ascertain. Max-Min scheduling has a lot to offer in terms of resource optimisation and fairness in distributed computing settings, despite these difficulties.

First Come First Served (FCFS): The First Come First Served (FCFS) scheduling algorithm is a fundamental approach used in various computing systems, including operating systems and cloud computing environments. It works by ranking the jobs according to the time they arrive, making sure that the task that is arrived at the earliest is completed first, and then the other tasks in the order that they arrive. In contrast to preemptive algorithms, FCFS permits tasks to start and run continuously until they are finished. Because of its simplicity, FCFS is a desirable choice because it is simple to understand and apply. Work is arranged in a queue, and tasks are processed in the order that they arrive. Fairness is one of FCFS's main benefits since it guarantees that jobs are carried out openly and without favouring any one activity or user over another. Because it does not require prior knowledge of job priorities or durations, it is especially well-suited for cases where task execution timeframes are uncertain. The simplicity of FCFS, however, can also have negative effects, such as inefficient use of resources and increased wait times, particularly when shorter tasks are backed up by longer-running ones in the queue. Therefore, even while FCFS is transparent and easy to use, it might not always be the best option in complicated computing settings where resource allocation optimisation is essential.

Shortest Job First (SJF): Shortest Job First (SJF) is a task scheduling algorithm widely used in computing systems, particularly in operating systems and cloud computing environments. SJF executes the shortest job first and ranks tasks according to their burst or execution time, in contrast to First Come First Served (FCFS). This reduces waiting times and maximises resource use by processing activities with the shortest execution times before those with longer ones. Preemptive SJF allows shorter jobs to come later and interrupts longer tasks that are presently being executed. Non-preemptive SJF operates in a different way. Conversely, non-preemptive SJF works continuously through tasks until they are finished. SJF has benefits in terms of reducing waiting times and maximising resource use, but it also necessitates precise task execution time estimation, which isn't always possible in real-world situations. SJF is a valuable technique for decreasing wait times and optimising resource use, but its efficacy depends on accurate task execution time estimation. Accurately calculating task execution times is sometimes difficult in real-world circumstances, which can affect how well the algorithm performs.

Round Robin (RR): In operating systems, networking systems, and other computing contexts, Round Robin (RR) is a popular preemptive job scheduling technique. Each job in a queue is given a fixed time quantum by RR, and tasks are then carried out in a cyclical fashion. A task is preempted and the subsequent task in the queue is chosen for execution when its time quantum expires. In order to wait for their next turn, preempted jobs are positioned at the end of the queue. As each task has an equal chance to use system resources, RR guarantees task execution fairness. Furthermore, as jobs receive regular CPU time slices, RR offers responsiveness, especially in time-sharing systems. However, RR might not be the most effective option for activities with variable execution periods because it could result in higher context switching overhead, particularly with smaller time quantum values. Even while RR is responsive and fair, it could not work as well for activities that have varying execution durations. This is due to the possibility that RR scheduling will result in greater context switching overhead, especially when time quantum values are less, which will affect system performance as a whole. As a result, even if RR works well in some situations, it might not be the best choice for jobs with inconsistent execution times.

B. Related Work

Researchers' interest in workflow scheduling has grown as it is a crucial component of cloud computing and distributed environment applications.

As data quantities and computational tasks rise exponentially, workflow scheduling becomes increasingly important for maximising resource utilisation, reducing processing times, and guaranteeing smooth task execution in dispersed contexts.

Particularly, cloud computing platforms have enormous scalability and flexibility, which makes them invaluable for managing a wide range of workloads in a variety of businesses. However, in order to effectively coordinate the distribution of computing jobs, advanced workflow scheduling techniques are needed to fully use cloud resources. Furthermore, efficient workflow scheduling is even more essential in distributed environments—where computer resources are divided among several nodes—in order to guarantee fault tolerance, workload balancing, and overall system stability.

Additionally, the increasing popularity of cutting-edge technology like machine learning, big data analytics, and Internet of Things (IoT) applications has highlighted how crucial effective workflow scheduling is. These systems frequently have extensive task relationships and convoluted data processing pipelines, which makes sophisticated scheduling algorithms necessary to meet strict performance requirements and maximise resource allocation.

Essentially, the growing body of research on scheduling workflows reflects its pivotal role in facilitating the smooth integration and effective use of computing in the cloud along with distributed environment resources. Researchers hope to fully realise the promise of these technologies and enable businesses to get the most out of their computational resources by tackling the difficulties related to task orchestration.

[1], This study provides the Enhanced Max-Min Algorithm (EMM) appears as a way to streamline these procedures. Based on the effective framework of the Max-Min algorithm, the EMM performs well while performing small-scale activities in parallel and allocating resources appropriately, which reduces completion time. The EMM further improves resource utilisation and shortens the makespan by utilising the ideas of the Max-Min algorithm, which offers the priority of allocating these resources to the respective activities, with the maximum completion durations. This effectiveness provide highlights how important the EMM is for improving resource allocation and job scheduling in cloud systems, and it fits in perfectly with the ever-changing needs of contemporary cloud computing technology.

[2], This study provides a hybrid strategy incorporating three essential algorithms like Min-Min scheduling, Max-Min scheduling, and evolutionary algorithms, that showcase the persistent problem of task and resource scheduling within cloud environment. Reduced Makespan and improved resource load balancing are the goals. Makespan and resource usage are used as metrics in the study to assess how effective this strategy is. The suggested hybrid approach's efficiency and effectiveness are demonstrated by comparison with the Min-Min scheduling and Max-Min scheduling algorithm, which yields superior results in terms of lowering Makespan and efficiently employing existing resources.

[3], Effective scheduling algorithms are needed as cloud computing becomes more widely used, especially in data centres where energy efficiency is crucial. This study, a

unique scheduling method called Maxmin Fast Track (MXFT) is presented, which is customised for the Cloud Computing environment. By improving it to better handle smaller jobs with more stringent Service Level Agreements (SLAs), MXFT expands upon the Max-min algorithm. Motivated by supermarket checkout lines, MXFT prioritises these kinds of tasks by adding a fast lane. The results show that the MXFT algorithm outperforms Min-min in the overall makespan and also outperforms Max-min in the task execution times. The study also presents "ScheduleSim," a simulator that makes it easier to validate scheduling algorithms before applying them to particular Cloud Computing applications. using its ability to streamline studies using well-known simulators like CloudSim, this application offers a useful forerunner.

[4], In order to effectively serve numerous users, cloud computing allows on-demand access to shared computer resources over the Web or Internet. Nevertheless, cloud service providers frequently face difficulties in satisfying users' resource requirements. In order to address this, the Max-Min (MM) algorithm allocates resources based on the different needs of the user, yet it has long waiting and completion times. An approach called Improved Max-Min (IMM) is presented to alleviate these shortcomings. By optimising the distribution of resources in cloud environments, IMM lowers average wait and completion times and increases total resource utilisation.

[5], Virtualization and scalable resources are essential to the delivery of services in cloud computing, where customers pay for the use of resources. Because it is heuristic, many workflow scheduling strategies, such as the well-known Max-min approach, strive for efficiency and almost-optimal results. Research is still being done to improve its functionality even more. In order to achieve optimal scheduling, this study suggests a strategy that minimises the overall execution time while taking workflow task lengths into account. The suggested method is superior to the usual Max-Min scheduling scheme, as demonstrated by a comparative study using two real-world scientific workflows as test data for simulation validation.

[7], Cloud computing, well-known for its pay-as-you-go on-demand services over the Internet, has gained a lot of popularity in both business and academics. Task scheduling is one of a difficult issue in cloud environments that has drawn a lot of attention, especially in multi-cloud settings. Since it is NP-Hard, a lot of heuristics have been developed to deal with it. The Normalised Multi-Objective Min-Min Max-Min Scheduling (NMOMXS) algorithm which is proposed in this research, utilising the widely used Min-Max normalisation method from data mining. Through simulation, NMOMXS outperforms two of the well known best algorithms in terms of the makespan and resource usage, showcasing its effectiveness in handling the challenges associated with task scheduling with in the cloud environment.

[10], This study provides a modified Max-Min algorithm was presented, that determines how long each machine's planned tasks should take to complete on average. The slowest machine with the lowest overall completion time is then given the task that takes longer to complete than the average execution time.

[12], In cloud computing, load balancing is crucial for maximising resource usage and avoiding bottlenecks. There are many different load balancing strategies, but not all of them provide the elasticity required in cloud settings. In order to balance load in elastic clouds, a new task-scheduling technique called Max-Min is presented in this study. In order to facilitate the proper workload distribution and the load balancing amongst nodes, this suggested approach makes use of the task status database to calculate the real-time virtual machine load and the job completion timings. The Max-Min task-scheduling technique has been shown through extensive studies to optimise performance in cloud environments by reducing task response times and increasing resource utilisation.

[19], Effective task scheduling systems for large-scale cloud data transfer are crucial, but have not received as much attention as they should from the application point of view. Inadequate scheduling can result in hardware malfunctions, energy inefficiency, and traffic congestion. Large volumes of data are produced by the spread of IoT devices, which necessitate efficient management and planning in order to extract value. In order to overcome these obstacles, we suggest an approach called Hyper Min Max Task Scheduling (HMMTS) that optimises task allocation by utilising Cascade Shrink Priority (CSP). The load balancing method is implemented by the algorithm through the use of a Changeover Load Balancer (CLB) and a Preemptive Flow Manager (PFM), which enhances work allocation. And distributing the burden to improve reaction time. Experiments in both phases and the random uniform propagation scenarios have shown improved load balancing, lower power usage, and the lower time consumption rates. The efficiency of the proposed approach is demonstrated by the reduced data processing time and improved load neutralisation shown by the simulation performance.

[21], There is an urgent demand for quick ways to decrease user and task waiting times because of the growing dependence on cloud computing and the desire to maximise resource utilisation while meeting users' pay-as-you-go needs. In this paper, a new approach called Task-Oriented Min-Min Max-Min Prioritisation (TOMMP) is presented. It incorporates task prioritisation while utilising the advantages of the classic Min-Min and Max-Min algorithms. Even with today's abundance of these scheduling algorithms, job prioritisation is still often disregarded. To close this gap, TOMMP uses a prioritisation algorithm to rank activities according to

importance. Then, it uses the median number to determine whether approach is Min-Min or Max-Min should be used. Test findings highlight TOMMP's superiority in task scheduling efficiency by showing that it drastically cuts waiting times when compared to current methods.

[25], The pay-as-use model of cloud computing has revolutionised, providing of the IT resources over the internet. Applications for scientific workflow have a lot to gain from using cloud resources. Still, workflow scheduling algorithm optimisation is a challenging task to any researcher, that requires more investigation. To minimize the makespan of workflow execution and optimise resource utilisation, an improved version of the Max-Min algorithm is presented in this study. WorkflowSim is used to assess the algorithm using five workflows taken from the Pegasus workflow management system. The results showcase the efficacy of the suggested algorithm by improving workflow scheduling efficiency in cloud computing environments, as it regularly outperforms the conventional Max-Min strategy in the majority of scenarios.

[26], One emerging technology that provides customers with access to a range of features, including pay-per-use models, scalable resources, and on-demand self-services, is cloud computing. Many load balancing methods have been developed as a result of the importance of load balancing in cloud computing architecture. We present an improvement on the classic Max-Min method by introducing the Improved Max-Min Ant Colony Algorithm, which prioritises jobs according to execution time instead of completion time. The main goal is to minimise the entire makespan and balance the cloud system's overall load. We evaluate the performance of Improved Max-Min with a novel hybrid Improved Max-Min Ant method through simulation using the CloudSim toolkit, with an emphasis on total processing time and processing cost. These results provide a great insight about this suggested algorithm's effectiveness in optimising resource allocation and improving overall system efficiency in cloud environments.

[27], Cloud computing provides a very high-performance calculation and storage capabilities over large distances by utilising the network-delivered computing resources. According to this usage paradigm, maximising the resource utilisation, reducing completion times, and improving performance all depend on the availability resources by executing multiple number of tasks simultaneously. It is essential to create and apply effective scheduling algorithms in order to achieve these goals. Inspired by both the RASA algorithm and the Max-min approach, this work presents an updated version of the Max-min method. By giving predicted execution time precedence over completion time, the presented Improved Max-min algorithm that is provided here enhances the RASA scheduling procedure. It is shown through simulation that the Improved Max-min algorithm, when used for the task scheduling in a cloud computing environment, may achieve a lesser makespan than the original Max-min strategy, improving overall performance.

[30], Cloud computing enables easy access to remote computational resources and is being hailed as a game-changing tool for future computing paradigms. In this field, task scheduling is a focus of ongoing study with the goal of reducing response times and maximising resource use. In this research, a new task scheduling method combining the best features of the Enhanced MaxMin and MaxMin algorithms is presented. The choice of tasks is dependent on the availability of resources, which are divided into two categories according on MIPS speed. A work with a length somewhat longer than average is picked when the selected resource is part of the first set; if not, the job with the longest length is chosen. Workflowsim toolkit experimentation shows the effectiveness of the suggested algorithm with notable gains in performance.

3. PROPOSED ALGORITHM

Every task's completion time on every virtual machine is determined on the fundamentals Max-Min algorithm. Tasks will be prioritized based on execution time, with those requiring the longest time being scheduled before those with shorter execution times. It gives the first fastest VM out of all the VMs the first longest task out of all the tasks. The second quickest VM is then given the second-longest task. Until all tasks are scheduled, the same process is repeated. The possibility exists that a slower machine may be given a larger job when workflow scheduling with Max-Min, increasing the entire execution time. To prevent this problem, a better Max-Min algorithm is required.

The proposed Improved-Max-Min algorithm focuses on refining task allocation and resource management that are within cloud computing environments. It introduces enhancements like task prioritization, where tasks are assigned to VMs based on their execution requirements and the VMs' processing capabilities.

Moreover, it aims to optimize resource usage for critical workloads by assigning high-priority tasks to VMs with lower current workloads or faster processing capabilities. Additionally, dynamic VM provisioning adjusts the number of active VMs based on workload fluctuations to maintain optimal performance and resource efficiency.

This improved algorithm prioritizes minimizing the energy consumption, used by consolidating tasks onto a minimal number of active VMs and efficiently managing VM power states. While not explicitly incorporating predictive analytics, fault tolerance mechanisms, multi-objective optimization, or the hybrid approaches, the algorithm offers significant improvements in task scheduling efficiency and the resource utilization within cloud environments.

The proposed Improved-Max-Min algorithm evaluates four key parameters: Makespan, Throughput, Average Waiting Time, and Average Response Time. It demonstrates superior performance across all these parameters when compared against the other scheduling algorithms such as First Come First Served (FCFS), Shortest Job First (SJF), and Round Robin. Time measurements are standardized in seconds for evaluation consistency.

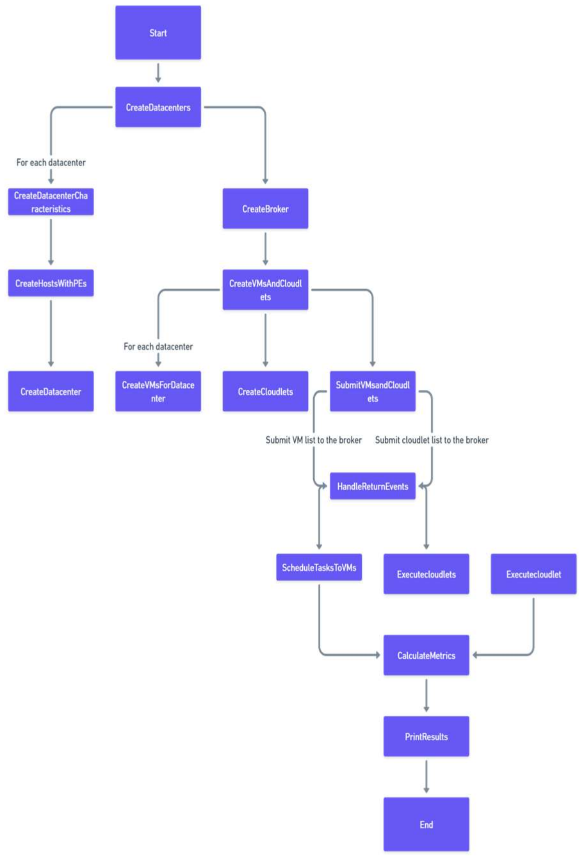

Figure 2. Flow Chart Of The Proposed Algorithm.

The content presented in this paper, particularly the Improved-Max-Min algorithm, is entirely original and does not draw from previously published research or literature. Our investigation has resulted in enhancements to job allocation and resource management, specifically tailored for cloud computing systems.

A. Algorithm

The task set is initialised and the initial completion durations for each task on all available resources are determined at the start of the algorithm. The job with the longest execution time is then assigned to the resource where it will finish the earliest. The method then recalculates the completion times for all resources and modifies the completion time of the resource that was assigned. Until all tasks are assigned, this process is repeated recursively.

```
1. Initialize Meta-task set with all submitted tasks Ti.
2. Initialize resource completion times Cij for all resources Rj.
3. For all submitted tasks in Meta-task; Ti
   3.1. For all resources; Rj
      3.1.1. Calculate completion time Cij = Eij + rj.
4. Find task Tk with maximum execution time (Largest Task).
5. Assign task Tk to resource Rj which gives minimum completion time (Slowest resource).
6. Remove task Tk from Meta-tasks set.
7. Update resource completion time rj for selected Rj.
8. Update completion times Cij for all j.
9. While Meta-task not Empty
   9.1. Find task Tk with maximum completion time.
   9.2. Assign task Tk to resource Rj which gives minimum execution time (Faster Resource).
   9.3. Remove Task Tk from Meta-tasks set.
   9.4. Update resource completion time rj for selected Rj.
   9.5. Update completion times Cij for all j.
```

In each iteration, the task, that is with the longest completion time is identified, it is assigned to the resource where it will execute the fastest, and the resource completion times of these tasks are updated correspondingly. The algorithm ends when every task has been allocated and carried out satisfactorily. In general, the goal of this technique is to minimise completion time and maximise resource utilisation when allocating tasks in a cloud computing environment.

4. DESIGN IMPLEMENTATION

A. High Level Design

The Improved MaxMin scheduling technique serves as the cornerstone of the high-level design, orchestrating the simulation of a cloud computing environment with precision and efficiency. Initially, the simulation parameters, such as the number of users and calendar instance, are configured to establish the CloudSim environment seamlessly. This setup process is facilitated by Constants, which defines essential constants like the number of tasks and data centers, ensuring consistency throughout the simulation.

The DatacenterCreator class takes center stage, responsible for creating data centers by generating hosts and virtual machines tailored to the specifications of each data center. This pivotal step lays the foundation for the simulation, providing the necessary infrastructure to support cloudlet execution and job scheduling.

To manage the workings of the simulation, including job scheduling and cloudlet execution, a specialized broker named DatacenterBroker is instantiated. Acting as the orchestrator, the DatacenterBroker oversees the allocation of cloudlets to virtual machines, ensuring efficient resource utilization and optimal task execution.

The Scheduler class plays a crucial role in generating both virtual machines and cloudlets, essential components of the simulation environment. Cloudlets are meticulously assigned to data centers based on communication and execution matrices derived from GenerateMatrices, while virtual machines are constructed considering factors such as MIPS and RAM, ensuring compatibility with the workload requirements.

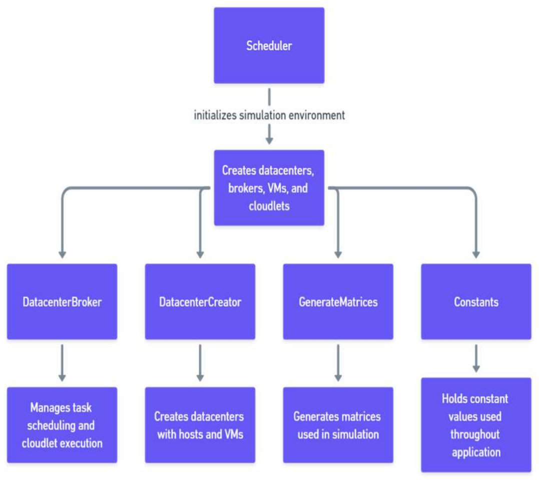
Scheduler
initializes simulation environment
Creates datacenters, brokers, VMs, and cloudlets
DatacenterBroker
DatacenterCreator
GenerateMatrices
Constants
Manages task scheduling and cloudlet execution
Creates datacenters with hosts and VMs
Generates matrices used in simulation
Holds constant values used throughout application

Figure 3. High Level Design For The Algorithm.

As the simulation unfolds, the broker's MaxMin scheduling method comes into play, efficiently allocating cloudlets to virtual machines to optimize performance and resource utilization. Finally, performance metrics including makespan, throughput, waiting time, and response time are computed and printed, providing valuable insights into the effectiveness and efficiency of the simulated cloud computing environment.

B. Related Work

The design of the Improved-Max-Min algorithm is centred on a few main classes and how they interact at a basic level. The primary orchestrator, the Scheduler, is in charge of setting up the simulation, constructing its components, and carrying it out.

It includes instructions for setting up cloudlets and virtual machines (VMs), initiating the simulation, and computing performance metrics. Using the MaxMin algorithm, the DatacenterBroker oversees cloudlet execution and job scheduling, guaranteeing effective resource allocation.

While GenerateMatrices generates or reads the communication and execution matrices needed for the simulation, DatacenterCreator builds datacenters with hosts and virtual machines. These Constants defines constants like the quantity of tasks and data centres.

By executing cloudlets on virtual machines and producing performance metrics to assess the effectiveness of the MaxMin scheduling technique, these classes together imitate the cloud computing environment.

Furthermore, the application's modular and structured design improves its readability, maintainability, and extensibility. By adding these essential classes, the simulation procedure is expedited and uniform resource distribution and performance assessment are guaranteed. This focus on design principles makes the algorithm easier to understand and provides a strong base for future improvements and the smooth addition of new features. This increases the algorithm's overall resilience and longevity in handling changing cloud computing challenges.

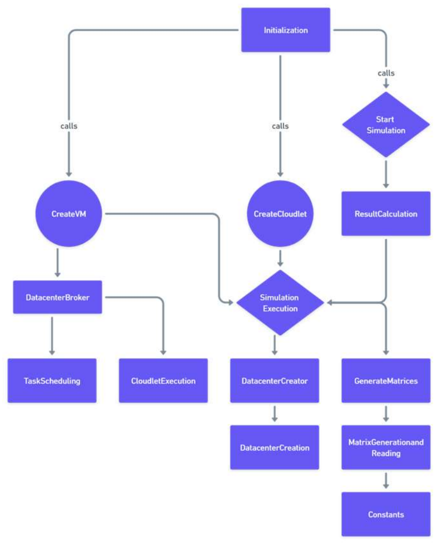

Figure 4. Low Level Design For The Algorithm.

To structured design not only facilitates ease of comprehension for researchers and developers but also enhances the algorithm's overall robustness and by fostering a clear understanding of the Improved-Max-Min algorithm is well-positioned to meet the evolving needs of cloud computing environments and drive innovation in the field.

5. PROPOSED ALGORITHM

When assessing the effectiveness and performance of scheduling algorithms in cloud computing settings, comparative analysis is essential.

The result of the proposed Improved-Max-Min algorithm is compared with the output of FCFS, SJF, Round Robin in order to assess the effectiveness of the suggested algorithm. This comparative analysis offers insightful information about the relative advantages and disadvantages of the recommended algorithm compared to accepted approaches.

The CloudSim framework, a popular simulation tool for modelling and simulating cloud computing infrastructures, is used in the evaluation process. The suggested approach was replicated using the CloudSim framework with the version - 3.0.3.

When the suggested algorithm is tested using the following critical parameters: Makespan, Throughput, Average Waiting Time, and Average Response Time, it produces results that are superior in every way.

After a thorough testing and analysis, the outcomes clearly show that the suggested Improved-MAX-MIN algorithm exhibits greater performance across all evaluated metrics. It notably outperforms existing scheduling algorithms in aspects such as Makespan, Throughput, Average Waiting Time, and Average Response Time, demonstrating its effectiveness in maximising resource use, reducing task completion times, and improving system responsiveness as a whole.

The table compares four different scheduling algorithms like FCFS, SJF, RR, and Improved-MAX-MIN across various performance metrics such as Makespan, Throughput, Average Waiting Time, and Average Response Time. Each metric is measured in seconds, except for Throughput, which is unitless. The Improved-MAX-MIN algorithm shows the best performance in terms of the lowest Makespan and Average Waiting Time, while FCFS has the lowest Average Response Time. The SJF algorithm leads in Throughput.

Algorithms / Time [sec]	Algorithms			
Time Taken [sec]	FCFS	SJF	RR	Improved-MAX-MIN
Makespan	5176	5864	6803	2635
Throughput	0.00162	0.00151	0.00128	0.00118
Average Waiting Time	6572	6602	7662	4360
Average Response Time	8985	8911	10231	5900

TABLE I. PERFORMANCE METRICS OF SCHEDULING ALGORITHMS

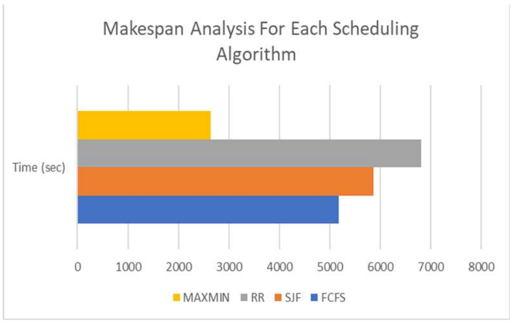

Figure 5. Makespan Comparison For Improved-Max-Min, FCFS, SJF And RR.

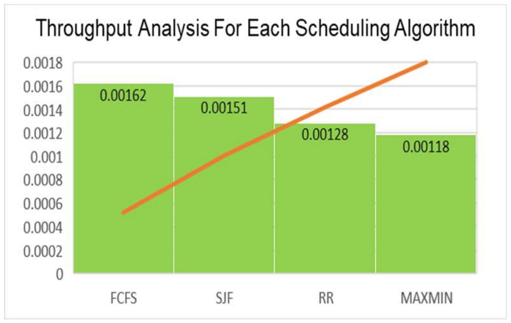

Figure 6. Throughput Comparison For Improved-Max-Min, FCFS, SJF And RR.

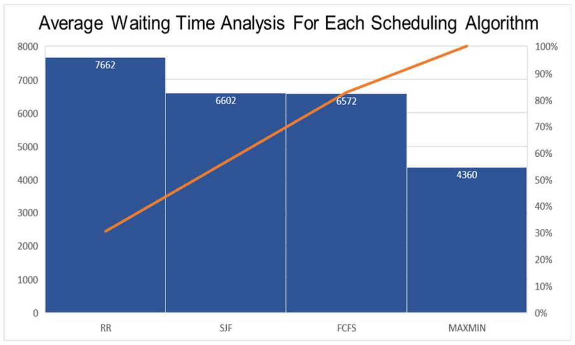

Figure 7. Average Waiting Time Comparison For Improved-Max-Min, FCFS, SJF And RR.

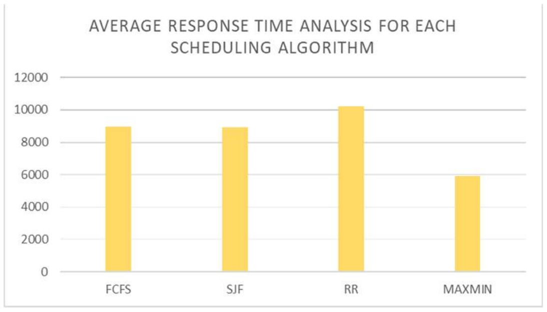

Figure 8. Average Response Time Comparison For Improved-Max-Min, FCFS, SJF And RR.

6. FUTURE WORK AND CONCLUSION

Taskflow scheduling for cloud computing may involve in more investigating hybrid approaches that combine various algorithms for better performance, improving fault tolerance mechanisms for increased system reliability, and investigating multi-objective optimisation techniques to for the balance competing objectives. These developments are intended to improve system flexibility, scheduling effectiveness, and resource usage in cloud environments.

The proposed Improved-Max-Min algorithm represents a significant advancement in the field of task scheduling within cloud computing environments. It is essential to acknowledge that there is still ample room for further refinement and improvement to achieve even better results.

In conclusion, effective taskflow scheduling is critical in cloud computing environment to optimise resource use and reduce job execution time. This proposed Improved-Max-Min algorithm and its improvements when tested performed well than current scheduling methods in terms of Makespan, Throughput, Average Waiting Time, and Average Response Time, have showcased significant progress and better results when compared with multiple other task flow algorithms like FCFS, SJF and Round Robin and was able to achieve the better results. The study demonstrated the value of real-time data utilisation and dynamic job allocation in allowing for flexible response to changing conditions and efficient use of available resources. All things considered, are only to make cloud computing workflow scheduling to be efficient.

REFERENCES

[1] Mala, K., S. Priyadharshini, and R. Madhumathi. "Resource Allocation in Cloud Using Enhanced Max-Min Algorithm." 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT). IEEE, 2021.

[2] Aref, Ismael Salih, Juliet Kadum, and Amaal Kadum. "Optimization of max-min and min-min task scheduling algorithms using ga in cloud computing." 2022 5th International Conference on Engineering Technology and its Applications (IICETA). IEEE, 2022.

[3] Moggridge, Paul, et al. "Revising max-min for scheduling in a cloud computing context." 2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE). IEEE, 2017.

[4] Pradhan, Pandaba, Prafulla Ku Behera, and B. N. B. Ray. "Improved max-min algorithm for resource allocation in cloud computing." 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC). IEEE, 2020.

[5] Auna, Shuaibu Yau, Faruku Umar Ambursa, and Abdulhakeem Ibrahim. "A New Modified Max-min Workflow Scheduling Algorithm for Cloud Environment." 2019 15th International Conference on Electronics, Computer and Computation (ICECCO). IEEE, 2019.

[6] Khatavkar, Bharat, and Prabadevi Boopathy. "Efficient WMaxMin static algorithm for load balancing in cloud computation." 2017 Innovations in Power and Advanced Computing Technologies (i-PACT). IEEE, 2017.

[7] Gajera, Vatsal, Rishabh Gupta, and Prasanta K. Jana. "An effective multi-objective task scheduling algorithm using min-max normalization in cloud computing." 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT). IEEE, 2016.

[8] Lakshmi, R. Durga, and N. Srinivasu. "A dynamic approach to task scheduling in cloud computing using genetic algorithm." Journal of Theoretical & Applied Information Technology 85.2 (2016).

[9] Gandhi, S. Yuvaraj, and T. Revathi. "An improved hybrid cloud workflow scheduling algorithm based on ant colony optimization." International Journal of Health Sciences IV (2022): 869-882.

[10] Bhoi, Upendra, and Purvi N. Ramanuj. "Enhanced max-min task scheduling algorithm in cloud computing." International Journal of Application or Innovation in Engineering and Management (IJAIEM) 2.4 (2013): 259-264.

[11] Hamad, Safwat A., and Fatma A. Omara. "Genetic-based task scheduling algorithm in cloud computing environment." Int. J. Adv. Comput. Sci. Appl 7.4 (2016): 550-556.

[12] Li, Xiaofang, et al. "An improved max-min task-scheduling algorithm for elastic cloud." 2014 International Symposium on Computer, Consumer and Control. IEEE, 2014.

[13] Haladu, Mubarak, and Joshua Samual. "Optimizing task scheduling and resource allocation in cloud data center, using enhanced min-min algorithm." IOSR J. Comput. Eng.(IOSR-JCE) 18.4 (2016): 18-25.
[14] Guo, Lizheng, et al. "Task scheduling optimization in cloud computing based on heuristic algorithm." Journal of networks 7.3 (2012): 547.

[15] Al-maamari, Ali, and Fatma A. Omara. "Task scheduling using hybrid algorithm in cloud computing environments." Journal of Computer Engineering (IOSR-JCE) 17.3 (2015): 96-106.

[16] Kumar, M. Sathish, et al. "A binary Bird Swarm Optimization technique for cloud computing task scheduling and load balancing." 2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES). IEEE, 2022.

[17] Agarwal, Mohit, and Shikha Gupta. "An Adaptive Genetic Algorithm-Based Load Balancing-Aware Task Scheduling Technique for Cloud Computing." Computers, Materials & Continua 73.3 (2022).

[18] Ahmed, Sara, and Fatma A. Omara. "An Enhanced Workflow Scheduling Algorithm for Cloud Computing Environment." International Journal of Intelligent Engineering & Systems 15.6 (2022).

[19] Gnanaprakasam, D., et al. "Efficient Task Scheduling in Cloud Environment Based on Hyper Min Max Task Scheduling." 2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE). IEEE, 2023.

[20] Mansouri, Najme, and Reyhane Ghafari. "Cost-Efficient Task Scheduling Algorithm for Reducing Energy Consumption and Makespan of Cloud Computing." Computer and Knowledge Engineering 5.1 (2022): 1-12.

[21] Derakhshan, Majid, and Zohreh Bateni. "Optimization of tasks in cloud computing based on MAX-MIN, MIN-MIN and priority." 2018 4th International Conference on Web Research (ICWR). IEEE, 2018.

[22] Emara, Farouk A., et al. "Genetic-Based Multi-objective Task Scheduling Algorithm in Cloud Computing Environment." International Journal of Intelligent Engineering & Systems 14.5 (2021).

[23] Gad-Elrab, AHMED AA, et al. "Genetic-Based Task Scheduling Algorithm with Dynamic Virtual Machine Generation in Cloud Computing." International Journal of Computing 20.2 (2021): 165-174.
[24] Kodli, Shilpa, and Sujata Terdal. "Hybrid Max-Min Genetic Algorithm for Load Balancing and Task Scheduling in Cloud Environment." International Journal of Intelligent Engineering & Systems 14.1 (2021).

[25] Al-Haboobi, A. S. "Improving max-min scheduling algorithm for reducing the makespan of workflow execution in the cloud." Int J Comput Appl 975 (2022): 8887.

[26] Ghumman, Navtej Singh, and Rajwinder Kaur. "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). IEEE, 2015.

[27] Devipriya, S., and C. Ramesh. "Improved max-min heuristic model for task scheduling in cloud." 2013 international conference on green computing, communication and conservation of energy (ICGCE). IEEE, 2013.

[28] Murad, SALLAR SALAM, et al. "Optimized Min-Min task scheduling algorithm for scientific workflows in a cloud environment." J. Theor. Appl. Inf. Technol 100.2 (2022): 480-506.

[29] Biswas, Dipto, et al. "Optimized Round Robin Scheduling Algorithm Using Dynamic Time Quantum Approach in Cloud Computing Environment." Int. J. Intell. Syst. Appl 15 (2023): 22-34.

[30] Santhosh, B., and D. H. Manjaiah. "A hybrid AvgTask-Min and Max-Min algorithm for scheduling tasks in cloud computing." 2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT). IEEE, 2015.

Narayana N Gourav Narayana N Gourav, did B.Tech in Computer Science and Engineering from Presidency University, Bangalore, India in 2022. M.Tech in Computer Science and Engineering from M S Ramaiah Institute of Technology in 2024 from and current research interests include Cloud Computing, Cybersecurity.

Akshatha Kamath Akshatha Kamath is working as an Assistant Professor in Computer Science Department of Ramaiah Institute of Technology. Her areas of interest include AI and ML, Deep Learning, Data Science, Algorithms and Data Structure.

Dr. Dayananda R. B. Dr. Dayananda R. B Associate Professor in Department of Computer Science and Engineering, MSRIT, Bengaluru. Under his guidance Ph.D. degree is awarded to research scholar from VTU on machine learning in the year 2021. His research mainly focuses on Design and Development of a Cloud Computing Architecture for data security and machine learning.