National College of Ireland

Optimising makespan and resource utilisation in IaaS cloud environment

MSc Research Project
Cloud Computing

Ankit Kumar
Student ID: 22113886

School of Computing
National College of Ireland

Supervisor: Dr. Shivani Jaswal

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Ankit Kumar
Student ID:	22113886
Programme:	Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Dr. Shivani Jaswal
Submission Due Date:	14/12/2023
Project Title:	Optimising makespan and resource utilisation in IaaS cloud environment
Word Count:	6363
Page Count:	23

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

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

Signature:	
Date:	14th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

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

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

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

Optimising makespan and resource utilisation in IaaS cloud environment

Ankit Kumar
22113886

Abstract

Using Infrastructure as a Service (IaaS) and other cloud services to provide scalable resources, this research report explores the revolutionary influence of cloud computing on scientific institutions. The paper acknowledges the benefits, but it concentrates on the particular difficulties faced by scientific institutions with high computing requirements. In order to improve Quality of Service (QoS) and guarantee the timely completion of crucial activities, it highlights the necessity of intelligent resource allocation in complex scientific workflows, covering domains like biology and weather forecasting. In addition, the study discusses the notable latency in task-to-task communication and highlights the significance of managing latency-related issues in order to maximize overall workflow effectiveness and save costs. Robust protocols and powerful encryption are only two examples of the robust data transfer security measures that are emphasized as being crucial to protecting sensitive data while it moves to and from the cloud. The research suggests state-of-the-art task scheduling algorithms as well as sophisticated ways to help scientific firms overcome these obstacles. In the end, a new level-based allocation model for IaaS workflows called MergingWF is presented. It incorporates task merging to minimize inter-task communication overhead and lower the number of levels. Research has shown that MergingWF performs better than well-known Directed Acyclic Graph (DAG) scheduling algorithms in a variety of contexts. This confirms that MergingWF is a beneficial asset for streamlining workflow execution times in cloud environments. The study emphasizes the necessity of ongoing adaptation to changing technological environments and workflow management techniques, enabling scientific organizations to take advantage of cloud computing opportunities while skillfully handling related challenges to foster innovation and advancement across a range of scientific fields.

1 Introduction

The rise in popularity of cloud computing has completely changed the computing environment and given businesses a glimpse into what lies ahead for technology. By enabling the effective use of shared resources and a diversified environment, cloud systems enable businesses to pay only for the particular processing power they need. Virtualization and the abstraction of resources from customers are how this is accomplished. Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and other services are provided by cloud service providers which is illustrated in Figure 1.

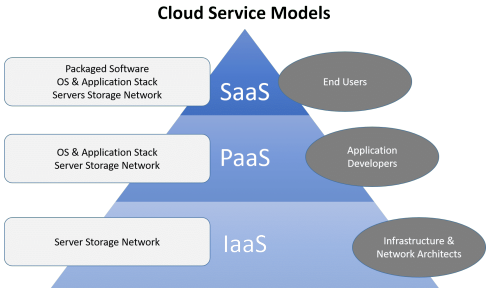

Figure 1: Cloud Service Model (Fu; 2022)

While the benefits of the cloud are undeniable, there are still some hurdles to navigate in this digital realm. It's crucial to think through things like allocating resources, organizing tasks, keeping a balance on the workload, figuring out workflows, and making sure data migration is secure (Beg et al.; 2021). When it comes to scientific organization dealing with hefty computing requirements, Infrastructure as a Service (IaaS) steps in as a helpful ally. It lets these organizations cherry-pick hardware, storage, network, and server resources that align perfectly with their unique scientific pursuits.

Workflows are like intricate roadmaps, especially in fields like biology, geology, and weather forecasting, involving some heavy-duty number crunching. So, it's super important to maximise resources utilization smartly to boost Quality of Service (QoS) and make sure those critical tasks wrap up on time. When you're sorting out workflow assignments, you run into a bunch of challenges, like making sure tasks are done in the right order to respect dependencies and avoid delays. Plus, cutting down on makespan—the total time it takes to finish the whole workflow—is a big deal for making sure resources are used efficiently (Shahid, Ashraf, Alam, Ahmad and Imran; 2021).

The considerable lag in service-to-service communication is another issue that affects workflow efficiency and raises expenses. It is critical to handle latency-related difficulties in order to guarantee smooth interactions between various process components and to optimize overall expenditures. Furthermore, data transfer security is quite important, especially when sensitive data is involved. Strong encryption techniques and reliable data transfer protocols are essential for protecting data as it moves to and from the cloud and against unwanted access attempts (Shahid, Alam, Hasan and Imran; 2021).

Scientific organizations can improve their ability to handle the challenges of cloud computing through implementing enhanced fault tolerance mechanisms, adopting efficient data management methods and implementing into effect state-of-the-art work scheduling algorithms. Such advances draw attention to the importance of modified workflow management systems in the field of cloud computing by optimizing resource use and improving cost-effectiveness.

Scientific enterprises must keep up with the newest technological developments and workflow management strategies as cloud computing continues to develop. Innovation and new research in many scientific domains can occur when cloud computing opportunities are welcomed and hazards are well managed.

1.1 Research Question

The following research question should be answered in order to complete thesis.

How can the allocation of workflow for institutions with high computational needs in Infrastructure as a Service (IaaS) be optimized to improve the Quality of Service (QoS)?

1.2 Research Objective

Following are the objectives that need to be full filled in order to answer the solution for the problem faced in task scheduling.

- Evaluate current challenges in scientific workflow allocation which uses Infrastructure as a Service (IaaS).
- Develop strategies to optimize resource allocation, reduce makespan, and enhance Quality of Service (QoS) in IaaS environments.
- Propose a state-of-the-art Merging Task Algorithm to streamline task and workflow levels, minimizing inter-task communication overhead, and enhancing overall communication efficiency.
- Compare it's efficiency with respect to other proposed policies to validate its performance.

1.3 Ethics Consideration

As per Table 1, this study does not involve human beings or private or public databases.

Table 1: **Simulation Environment**

Ethics Consideration	**Yes/No**
Involvement of human participants	**(No)**
Use of secondary dataset(s) created by the researcher	**(No)**
Use of public secondary dataset(s)	**(No)**
Use of non-public secondary dataset(s)	**(No)**
Approval letter from non-public secondary dataset(s) owner received	**(No)**

1.4 Structure of Report

The work's content overview is outlined as follows: In Section 2, the author provides an extensive literature analysis covering a range of tasks scheduling algorithms, highlighting the importance of the issue and outlining each one's drawbacks. In addition, Section 3 uses technical illustrations to explain the study methodology and the technology that was used to create the work scheduling algorithm. A thorough explanation of the algorithm is given in Section 4, paying special attention to the process used to combine tasks. Section 5 discusses how the suggested strategy is put into practice using a simulation of the

CloudSim environment. A detailed examination of the results obtained from every task scheduling technique is presented in Section 6. Finally, Section 7 encapsulates the study's findings, offering recommendations for future work, and underscores the contribution of the research to the cloud computing community.

2 Related Work

It is widely recognized that workflow allocation in an IaaS cloud context is a difficult problem that is NP-hard. An additional degree of complexity is introduced by the complex and dynamic relationships among resources, applications, and communication channels in the cloud architecture. Additionally, strict security rules must be followed by the allocation mechanism in order to guarantee the confidentiality and integrity of transferred information and to maximize the utilization of dispersed resources.

Because of the complexity of this difficult challenge, a number of process allocation methodologies have been developed by professionals and scholars and have been well documented in academic literature. To choose the best or almost best choices for process distribution inside the cloud ecosystem, these approaches use a different kind of strategies and algorithms.

Because the cloud environment is heterogeneous, meaning that its resources have varying capacities and capabilities, it is crucial to distribute workloads fairly throughout the allocation process. Some resources could be better at some activities by nature, so that's something to think about carefully. Moreover, the dynamic character of cloud computing, where resources alternate between online and offline modes, emphasizes the need for flexible and responsive allocation systems that can adjust to changing conditions.

2.1 Graphical Representations for Workflow Visualization

A workflow can be thought of as a collection of tasks connected by a web of interactions; this can be seen in graphical representations like Petri nets and Directed Acyclic Graphs (DAGs). These diagrams are useful resources for understanding the interdependencies and structural complexity of the jobs in the process (Shahid, Ashraf, Alam, Ahmad and Imran; 2021).

Within a Directed Acyclic Graph (DAG), individual tasks are symbolized as vertices, and the relationships between tasks are articulated through directed edges connecting these vertices. The acyclic attribute ensures the absence of loops or cycles in the graph, thereby mirroring the organized and sequential execution of tasks within the workflow (Shahid, Ashraf, Alam, Ahmad and Imran; 2021; Beg et al.; 2021).

For managing workflows, Hamdani and Abdelli (2020) recommend another technique called Petri nets. because of which, We now have a formal and visual way to model concurrent systems thanks to this technique. Tasks are essentially switches in this design; spots and arrows indicate how they depend on one another and use resources. When determining the relationship between jobs and resources, Petri nets come in helpful for identifying potential bottlenecks and resource conflicts.

Advanced variants like Time Petri Nets with rendezvous (RPTN) and Time Stream Petri Net are suggested by Hamdani and Abdelli (2020) for use in more complex settings. To simulate task synchronization and coordination, these modifications include time and rendezvous locations. Now including this extra dimension improves the realism of the process representation, especially where synchronization and time are critical factors.

Time Stream Petri Nets improve modeling skills by presenting the idea of time streams, making it easier to depict ongoing operations. As noted by Savdur et al. (2021), this is especially helpful for workflows that deal with real-time data streams or processes that have continuous features.

By utilizing these graphical models, practitioners and academics are able to gain a thorough grasp of how a process is structured. This improves researchers ability to perform thorough analysis, optimization, and efficient management. A clear understanding of job relationships, essential routes, and possible performance bottlenecks are made easier by the visual depiction. Additionally, these models provide as a fundamental framework for the creation of complex algorithms and optimization methods, which enhance workflow distribution, scheduling, and resource efficiency.

2.2 Workflow Scheduling Policies

Mikram et al. (2022) and Alhaidari et al. (2019) explore important facets of cloud computing in their research. Mikram et al. (2022) research compares four approaches to determine where virtual machines (VMs) should be placed. Though it requires more physical processors, the Genetic Algorithm is notable for its higher performance in processing time and migrations. Alhaidari et al. (2019), on the other hand, emphasizes the superiority of STF in space-shared policies in his work on task scheduling algorithms. With a space-shared scheduling method, every cloud job has a dedicated resource allotted to it for the duration of its runtime, such as a processor unit or core. This indicates that the resources can be used by several jobs at once. Common issues that both research face include the intricacy of the issues and the algorithms' scalability. To further improve the efficacy of these algorithms, future research will examine new variables like network capacity and dig into optimization techniques like swarm intelligence and machine learning.

The complexity of job scheduling in cloud computing is a topic that Chen et al. (2020), Bezdan et al. (2022), and Abd Elaziz et al. (2019) have addressed in their recent studies. Chen et al. (2020) use simulations to show the better performance of the Whale Optimization Algorithm (WOA) and its upgraded version (IWC) to maximize efficiency. The goal of Bezdan et al. (2022) group's hybridized bat algorithm for multi-objective task scheduling is to distribute computer resources as effectively as possible. The Moth Search with Differential Evolution (MSDE) approach for lowering makespan is a contribution by Abd Elaziz et al. (2019). Notably, they struggle with issues like juggling competing goals and the intricacy of changing cloud systems. Additionally, several cloudlets share processing elements (PEs) in a virtual machine (VM) over time under the proposed time-shared scheduling approach, with each cloudlet receiving a portion of the CPU time. Together, these methods offer a variety of approaches to the complex task scheduling problems in cloud computing, demonstrating breakthroughs in optimization methods and time-shared scheduling policy concerns.

With zero electrical resistance and room-temperature superconductivity, as well as the Meissner effect, Paulraj et al. (2023) presents the extraordinary LK-99 material in their ground-breaking discoveries. They boost task dependability with a Teaching-Learning Optimization method in their cloud computing concept, which is based on a fault tolerance-aware workflow scheduling strategy. Their claims of a room-temperature superconductor have been challenged with suspicion, but the team has provided strong data to back up their findings, demonstrating the effectiveness of their scheduling technique. The unique COMSE architecture is utilized by Wu et al. (2021), to address inefficiencies in scheduling large-scale scientific workflows on cloud platforms. By employing DAG Splitting, COMSE maximizes multi-vCPU VM efficiency while addressing the difficulties in striking a balance between process parallelism and topology. Using practical workflow trials, the researchers are able to show how COMSE may effectively reduce expenses associated with computing and communication.

The difficulties with workflow scheduling in cloud computing are discussed in great detail by Sahu et al. (2023), Jamal and Muqeem (2022) in their individual research. By pointing out problems with current heuristic techniques and suggesting hybrid optimization algorithms, Jamal and Muqeem (2022) focus on the IaaS model. Their approach shows promise in solving complicated issues such as resource heterogeneity and task interdependence. They specifically focus on multi-objective optimization and QoS parameters in an energy-efficient virtual machine placement method. A cuckoo search optimization-based technique that prioritizes and maps processes onto virtual resources is introduced by Sahu et al. (2023) in contrast, in order to address the multi-objective nature of workflow scheduling. They alleviate shortcomings in current algorithms by emphasizing task priorities and interdependence and by proposing a mechanism for determining priorities. NP-hard scheduling problems with trade-offs must be navigated, as well as dynamic cloud resource management. Sahu et al. (2023) emphasize better performance in terms of energy usage and SLA breaches, while Jamal and Muqeem (2022) emphasize scalability and flexibility. Both studies highlight advances. Future directions are suggested by both, including integrating machine learning for improved forecasts and expanding algorithms to various cloud models.

Aron and Abraham (2022), Gupta et al. (2022), Kruekaew and Kimpan (2022) and Warangkhana Kimpan have all made substantial contributions to the evolving field of cloud computing through their research work. In their investigation into the complexities of cloud scheduling, Aron and Abraham (2022) highlights the critical function that hyper-heuristic techniques play in handling dynamic problems. Aron and Abraham (2022) addresses scalability and heterogeneity issues while presenting a thorough taxonomy and criticizing current heuristics. In order to address the important problems of load balancing and task scheduling, Gupta et al. (2022) provide a Honey Bee-based BAT algorithm that works better than previous approaches and provides a reliable solution in spite of difficulties with parameter initialization and VM overload protection. The MOABCQ model is introduced by Kruekaew and Kimpan (2022), who demonstrate its superior performance in multi-objective optimization for task scheduling. Despite difficulties in managing NP-hard issues and striking a balance between exploration and exploitation behaviors, the above works offer insightful analysis and creative answers to the complex problems associated with cloud resource management.

Chandrashekar et al. (2023) present the Hybrid Weighted Ant Colony Optimization Algorithm (HWACO) as a state-of-the-art method for task scheduling effectiveness in the quickly changing cloud computing environment. They overcome the NP-hard difficulties associated with cloud task scheduling, such as resource heterogeneity, task dependencies, and quality of service requirements, by combining the Ant Colony Optimization Algorithm (ACO) with a weighted optimization technique. HWACO's superiority is demonstrated by a comparative study versus well-established algorithms, which is supported by thorough simulation experiments conducted with Cloudsim and statistical tests. The algorithm's benefits in terms of makespan and cost are highlighted by the authors, who also openly admit its drawbacks and provide the groundwork for future lines of inquiry.

Similarly, by tackling the challenges of cloud task scheduling, Kansara et al. (2023) Mean Ant Colony Optimization (MACO) method adds to the innovation. By utilizing ant colony optimization and mean value analysis, MACO is able to maximize resource usage, improve service quality, and minimize makespan and task execution costs. MACO's efficiency and scalability are demonstrated by comparisons with Min-Min, Max-Min, ACO, and PSO, supported by CloudSim simulation tests. The benefits, drawbacks, and potential directions for future study in the ever-evolving subject of cloud computing are thoughtfully discussed by the writers.

Considering cloud computing, Shahid, Ashraf and Alam (2021) introduce "A Multi-Objective Workflow Allocation Strategy in IaaS Cloud Environment." Their novel MOWA approach, focused on workflow task scheduling, is notable for simultaneously reducing makespan and flowtime while maintaining precedence restrictions via level characteristics. The study shows that MOWA outperforms MOHEFT directly in producing improved Pareto-optimal solutions, offering a good solution for IaaS cloud situations even with workflow mapping's intrinsic NP-Hard complexity.

The Multi-swarm Co-evolution based Hybrid Intelligent Optimization (MCHO) method for multiple process scheduling in the cloud by Li et al. (2021) makes a major contribution. MCHO performs very well in terms of makespan, cost, and timeliness because to its smooth integration of Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Simulated Annealing (SA). The authors provide a multi-swarm co-evolution approach to overcome difficulties with heterogeneous processes, encoding solutions, and preserving variety and convergence. MCHO surpasses current algorithms, exhibiting better solution quality, variety, and convergence. It is outfitted with global and local guiding information, a GA-based elite improvement technique, and a Metropolis acceptance criteria. The authors anticipate that future work will apply MCHO to different cloud computing settings, increase its adaptability through dynamic techniques, and expand its capabilities to suit additional goals and restrictions.

2.3 Table preview of Related works

In Table 2, related works have been compared based on their algorithm used and approach taken to solve NP - hard problem of task scheduling.

Table 2: Description of Papers and Algorithms

Paper	Description	Algorithms
Mikram et al. (2022)	compared four approaches to allocate VM	Genetic Algorithm
Alhaidari et al. (2019)	every task has a dedicated resource allocated	STF
Bezdan et al. (2022)	hybridized bat algorithm for distributing resources effectively	Hybrid BAT
Abd Elaziz et al. (2019)	proposed algorithm to lower makespan	Moth Search
Jamal and Muqeem (2022)	tried to solve resources heterogeneity and task interdependence	Cuckoo Search
Gupta et al. (2022)	addressed problems of load-balancing & task scheduling	HBBAT
Kruekaew and Kimpan (2022)	proposed multi-objective optimizer for task scheduling	MOABCQ
Chandrashekar et al. (2023)	combined Ant Colony Optimization with weighted optimization technique	HWACO
Kansara et al. (2023)	by using ACO and mean value analysis, maximized resources usage and minimized makespan and task execution cost	MACO
Shahid, Ashraf and Alam (2021)	approach focused on workflow task scheduling, reducing makespan and flowtime while maintaining precedence restriction	MOWA

3 Methodology

A methodological approach is discussed in this section with the goal of improving the processing time for several workflows that are submitted for processing in the Infrastructure-as-a-Service (IaaS) cloud computing environment. In order to minimize turnaround time and maintain the values of justice and efficient resource use, the author has suggested method that attempts to address the problem of assigning resources to several processes

in an efficient manner.The author's main area of interest is the strategic optimization of makespan and resource utilization—two important factors in cloud computing settings. The complicated dynamics of this optimization take place inside a system where users start and control jobs, workloads, or workflows that are made up of several tasks. which are then planned on the virtual machines (VMs) the datacenter's hypervisor has built. An illustration of the cloud framework can be seen in the Figure 2 below.

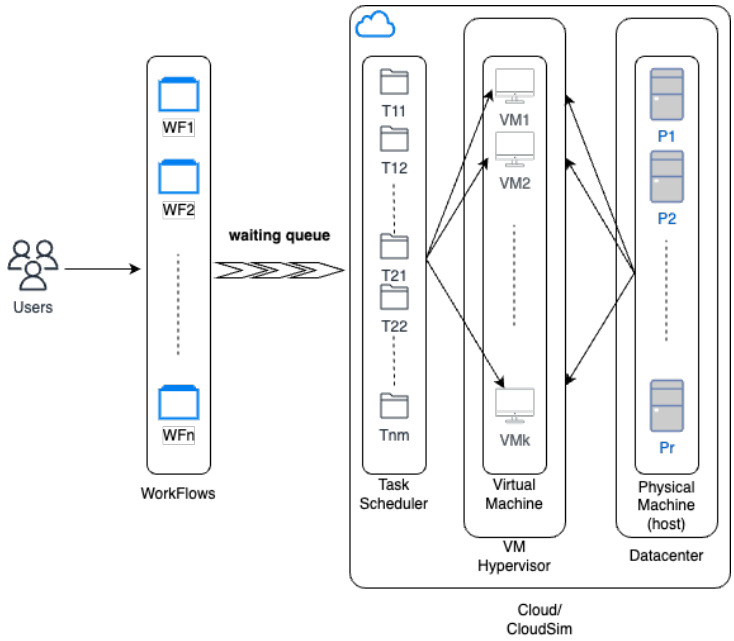

Figure 2: Cloud Framework

3.1 Workflows

Initially, users populate the cloud computing environment with extensive procedures. A series of tasks that together make up a bigger computational workload are captured by these processes. The distinct needs and features of each activity in the workflow serve as the foundation for the methods that allocate resources in a subsequent manner.

3.2 Task Manager

In order to handle the complexities of assigning tasks and using resources, a task manager is essential. In its capacity as the orchestrator, this entity receives and processes tasks from workflows that have been submitted. It is the Task Manager's job to determine the specific resource needs of every task so that Virtual Machines (VMs) may be allocated in an informed and effective manner.

3.3 Dynamic VM Allocation

The Task Manager must distribute virtual machines (VMs) effectively if the system is to function effectively. The Task Manager dynamically assigns and configures virtual

machines (VMs) inside the cloud architecture in response to each task's particular resource requirements. This assignment procedure is not only customized to meet the jobs' immediate demands, but it is also made to optimize makespan, or the total amount of time needed to do a project or process.

3.4 Datacenter and Hypervisor

Physical machines, which serve as the backbone of processing capacity, are housed within the cloud datacenter. A hypervisor orchestrates a transforming process for these physical devices. The hypervisor functions as a virtualization layer, systematically building VMs that perfectly fit with the subtle requirements of unique workloads. This virtualization approach guarantees a flexible and adaptive architecture capable of dynamically responding to the jobs' different processing demands.

The path toward optimization within a cloud computing environment is deeply entwined with user-initiated workflows, the crucial role of the Task Manager, and the virtualization that the hypervisor orchestrates. The goal isn't only to distribute resources, it's also to carefully customize virtual machines (VMs) to meet the specific requirements of each activity, which will eventually maximize makespan and resource usage in the context of cloud computing.

The current issue that is being addressed is task scheduling optimization with the goal of attaining effective resource allocation, which is a computationally challenging problem that is widely acknowledged to be NP-hard. Different scholars within the field of cloud computing have attempted to solve this complex issue by proposing algorithms, which are thoroughly reviewed in the related work section of this research paper in order to establish the foundation for further comparative study.

The algorithm proposed in this research distinguishes itself in tackling the difficult task scheduling problem by emphasizing a methodology that aims at merging tasks at different levels of hierarchy. Eliminating idle resource times will promote effective resource allocation, which is the main goal of this strategy. What makes the suggested approach noteworthy is that it aims to optimize the makespan, a crucial scheduling parameter, hence improving the overall Quality of Service (QoS) in parallel.

At the core of the proposed technique is its ability to coordinate the synchronization of tasks at various hierarchical levels, so strategically avoiding instances of resource underutilization. Through this insightful task merging method, the algorithm intends not only to minimize the makespan but also to improve resource consumption efficiency, thereby contributing to resource allocation optimization. The resulting comparative evaluation against other algorithms in the scholarly landscape will serve as a thorough assessment of the suggested method's efficiency and efficacy in tackling the inherent challenges of task scheduling for the goal of improving Quality of Service.

4 Design Specification

This research makes use of CloudSim which is a specialized simulation toolset designed for modeling and simulating cloud computing infrastructure. Without the need for a phys-

ical cloud infrastructure, this toolkit acts as a virtual platform that allows researchers to replicate and experiment with various cloud computing environments. In this research, CloudSim's capability for supporting the application and evaluation of different scheduling algorithms for tasks and resources plays a crucial role.

During this study, the author used CloudSim to model various scheduling policies. This technique was conducted with the explicit purpose of evaluating and comprehending the performance dynamics of the proposed scheduling mechanism. The study attempted to gather insights into how the suggested scheduling technique would perform under different settings and against alternative scheduling strategies by leveraging CloudSim's ability to simulate real-world cloud scenarios. This simulation-based methodology not only allowed for a controlled examination of the proposed strategy, but also demonstrated CloudSim's versatility and utility as a research tool in the field of cloud computing.

Algorithm:

1. **JobEntry Class:**
    - Define a class `JobEntry` with fields representing columns in the data (jobID, taskIndex, machineID, ram).
    - Implement the `equals`, `hashCode`, and `toString` methods for comparison and printing as CSV.
2. **Main Class - MergeWorkflowWithMaxRam:**
    (a) **Read Input:**
        - Specify input and output file paths.
        - Call `readDataFromCSV` to read job entries from the CSV file into a list.
        - Print the input data.
    (b) **Remove Duplicates:**
        - Call `removeDuplicates` to remove duplicate entries based on job ID.
    (c) **Find Max RAM for Each Job:**
        - Call `findMaxRamForJob` to find the job entry with the maximum RAM for each job.
    (d) **Print and Write Output:**
        - Print the output data.
        - Call `writeDataToCSV` to write the result entries to a new CSV file.
3. **Read Data from CSV (`readDataFromCSV`):**
    - Create an empty list of `JobEntry`.
    - Open a BufferedReader to read lines from the CSV file.
    - For each line, split the line into parts, convert to the appropriate data types, and create a `JobEntry` object.
    - Close the BufferedReader and return the list of `JobEntry` objects.

4. **Remove Duplicates (`removeDuplicates`):**
    - Create a HashSet to store unique `JobEntry` objects.
    - Add all job entries to the HashSet, removing duplicates.
    - Convert the HashSet back to a list and return it.
5. **Find Max RAM for Each Job (`findMaxRamForJob`):**
    - Create a HashMap with jobID as the key and corresponding `JobEntry` as the value.
    - Iterate through the job entries, updating the HashMap with the entry having the maximum RAM for each job.
    - Convert the values of the HashMap to a list and return it.
6. **Write Data to CSV (`writeDataToCSV`):**
    - Open a FileWriter to the specified file path.
    - Iterate through the list of `JobEntry` objects and write each entry as a CSV line.
    - Close the FileWriter.

The preceding algorithm explains its rationale in relation to the proposed methodology, however a comprehensive description is provided below. Several critical steps are included in the suggested methodological approach for improving numerous workflows in IaaS cloud computing.

1. **Partitioning and task fusion:**
    - First, the research divides the incoming cluster of many processes into more manageable divisions based on the depth levels of each workflow. A Directed Acyclic Graph (DAG) with linked tasks and parent-child interactions represents each workflow as shown in Figure 3. The goal of dividing the workflows according to depth levels is to shorten reaction times and increase completion times overall. The workflow execution procedure was more efficient when it started at the first depth level encounter because waiting times were reduced.

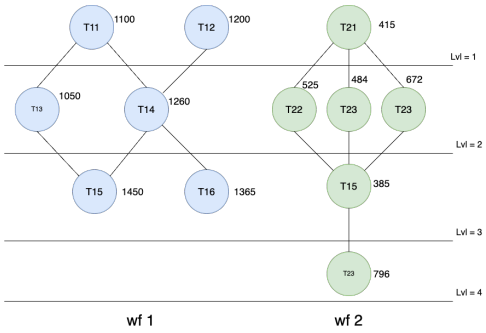

Figure 3: Sample WorkFlow

- To improve workflow performance even further, the study incorporates a task fusion strategy after the partitioning step. The overall goal of this approach is to reduce the number of depth levels and the amount of communication between tasks. This is achieved by combining multiple smaller tasks into one larger work that is then allocated to a single virtual machine as illustrated in Figure 4. The overall communication overhead is reduced by reducing the number of tasks and depth levels, which improves workflow performance and shortens workflow completion times.

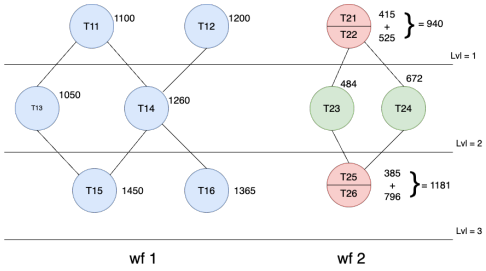

Figure 4: WorkFlow after Taks merging

- Not only does task fusion reduce intertask communication, it also contributes to a reduction in the communication costs share. The requirement for inter-machine communication is reduced by grouping tasks together and assigning them to a single virtual machine. The result is a significant decrease in the total cost of communication between jobs, which promotes more efficient resource use and improves Quality of Service (QoS) metrics.
- A single virtual machine is given merged tasks by the task fusion scheme. This allocation strategy reduces the overhead associated with task communication across several machines by ensuring that the combined tasks execute on the same virtual machine. This enhances the system's overall performance and makes better use of its resources possible.

2. **Estimating Communication Cost:**

- Tasks in a workflow often need to share data or information and depend on one another in order to be completed. The communication overhead generated by this inter-task communication may have an impact on the workflow's overall efficiency and turnaround time. The proposed approach estimates the inter-task communication cost by taking into account the communication time. Within a data center or even across many data centers, virtual machines may be distributed among different physical machines in a cloud environment. Because data transfer between distant computers may result in increased latency and communication overhead, the spatial separation between virtual machines might affect communication expenses.
- The proposed approach evaluates the impact of inter-machine communication on workflow execution by considering the closeness of virtual machines. This suggested method provides a more accurate assessment of the communication expenses related to workflow execution by accounting for both the communication time between tasks and the geographical distance between virtual

machines. With the goal of lowering communication overhead and raising system efficiency overall, this accurate assessment allows for better informed decisions to be made about resource allocation and scheduling tactics.

- Accurately assessing communication costs is necessary in order to evaluate different scheduling options and select the best schedule that minimizes communication overhead. By cutting these communication expenses, the suggested method aims to increase workflow efficiency. This would lead to quicker response times and better quality of service (QoS) for cloud users.

3. **Level Attributes for Precedence Constraints:**

   - A task's level attribute is assigned by the workflow based on its dependencies and relationships with other tasks. Dependency-free tasks have a level attribute of 0, which indicates that they can be completed early. On the other hand, actions that depend on entities with lower level attributes are designated with higher level attributes, signifying a necessity for their execution subsequent to their dependencies.
   - The suggested approach ensures the orderly completion of dependent tasks by incorporating level features. Higher level attribute tasks are only scheduled for execution following the completion of their lower level attribute dependant tasks. By following a methodical sequence, precedence limitations are avoided and the workflow is executed logically and consistently.
   - Allocating workflows optimally depends critically on the effective management of precedence restrictions through the use of level attributes. Following the right order of tasks reduces idle time and inefficient use of resources, which improves resource use and speeds up workflows.

5 Implementation

This section of the paper outlines how the proposed method will be implemented. The simulation tool for cloud computing settings, CloudSim toolkit, has been used to instantiate the research framework. An illustrated Figure 5 provides a thorough description of the implementation flow in the CloudSim environment. The author also clarifies the several scenarios that were taken into consideration and the changes that were made to the parameters. These changes are intended to make it easier to compare and evaluate how well the suggested methodology performs in relation to previous efforts. The story that follows provides a thorough examination of the complexities associated with putting the suggested design into practice inside the CloudSim framework, thereby augmenting the reader's comprehension of the research methodology.

For the purpose of clarifying the implementation flow shown in the above Figure 5, the CloudSim framework is imported into the Eclipse IDE as a Maven project. One noteworthy aspect of CloudSim is that it is hosted on GitHub and is open-source, which makes it compatible with a wide range of computer settings.

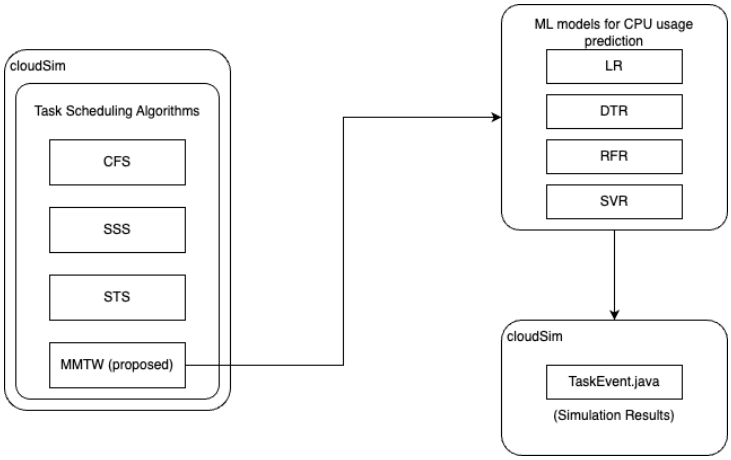

Figure 5: Implentation Flow of Research Project

The developed methodology to task scheduling is incorporated into the CloudSim framework, and the resulting outputs are then used to machine learning models that are contained in separate Python notebooks. The purpose of these models is to predict CPU consumption trends for the assigned jobs. The simulation parameters in the CloudSim environment are subsequently adjusted based on the predictive insights obtained from the machine learning models.

The simulation parameters include things like hosts, data centers, virtual machines (VMs), cloudlets (which stand in for tasks). The expected CPU use patterns predicted from the machine learning models are reflected in the simulation setups through iterative modifications. This recurrent adjustment procedure guarantees a dynamic match between the virtual environment and the expected computational demands of the tasks.

The simulation instances that have been set up in this way are compared to other task scheduling strategies. An important indicator for evaluating the effectiveness and efficiency of the suggested methodology in comparison to other well-known scheduling algorithms is this comparative evaluation.

During the evaluation stage, the variables controlling cloudlets, virtual machines, hosts, and data centers are systematically changed in order to thoroughly examine the performance features of the suggested methodology. The simulation paramters have been listed in the Table 3, while the above mentioned parameters are changed, the others remains same throughout the iterative simulation. A thorough comprehension of the proposed approach's adaptability and robustness in a range of settings and workloads is made possible by the varied experimentation.

The evaluation's findings are then carefully discussed in the section devoted to the Evaluation. This section of the research paper presents a thorough analysis of the observed outcomes and makes comparisons between the suggested methodology and other task scheduling methods. The conclusions drawn from this thorough review procedure make a significant contribution to the field of knowledge in science on task scheduling in cloud computing environments.

Table 3: **Simulation Environment**

Type	Parameter	Value
Host	Number of Host	20
	MIPS	177,730
	Bandwidth	10 GB/s
	Storage	2 TB
	RAM	16 GB
Data Center	Number of Data Center	2
	VM Scheduler	TimeShared, SpaceShared, CFS, MergingWF
Cloudlet (Task)	Length of Task	1k - 900k
	Number of Tasks	4 - 128 (can be considered more)
Virtual Machine	Number of VMs	5 - 100
	Processor Speed	3,500 - 100,000 MIPS
	Memory	1 - 4 GB
	Bandwidth	1000 - 10,000
	Cloudlet Scheduler	TimeShared, SpaceShared, CFS, MergingWF
	Number of PEs	1

6 Evaluation

This section of the research offers a thorough comparative analysis, utilizing four different bar graphs to assess the proposed algorithm's effectiveness in relation to several crucial parameters. Every category plays a crucial role in evaluating various aspects of the algorithm's efficiency in handling and streamlining computational activities in the context of cloud computing. Turnaround time, Response time, makespan (Makespan), and System utilization are the specific metrics that are being examined.

6.1 Total Turn Around Time

Figure 6 shows that when the number of processes across different scheduling strategies — MergingWF, CFS (Completely Fair Scheduler), SSS (Space Shared Scheduler), and TSS (Time Shared Scheduler)—increases, the turnaround time trajectory shows a steady upward trend. Considering that other input parameters are unchanged, this trend is in line with expectations. MergingWF is noteworthy since it exhibits the lowest value of total turn around time increase as the number of tasks are increased.

The data clearly shows that the proposed method, MergingWF, outperforms all other strategies in a range of batch sizes, showing better performance as batches get bigger. In terms of turnaround time, the following is the relative performance order: TSS (suboptimal), SSS, CFS, and MergingWF (optimal). Across batches with 4 to 128 workflows, MergingWF exhibits a performance gain ranging from 0% to 5%. The combination of merging task before allocating, a tactic that minimizes cost of communication, is responsible for this improvement in performance. As a result, when compared to other scheduling systems, this results in an improved Turnaround Time (TAT) for MergingWF.

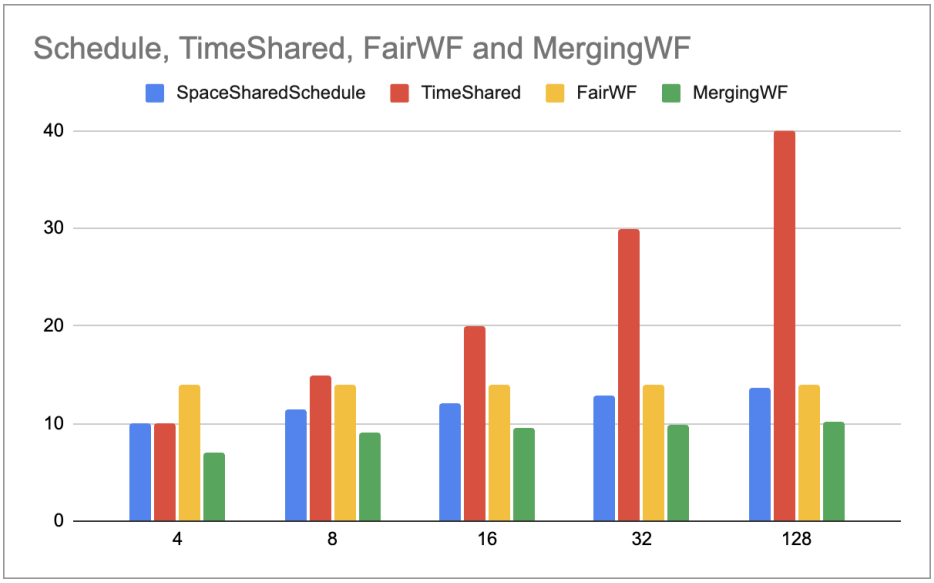

Figure 6: Turnaround Time vs Batch size

6.2 Response Time

As the batch size grows, the reaction time trend shows an ascending trajectory, as seen in Figure 7. Among other techniques, MergingWF and SSS stand out for their consistent good performance in all batch sizes tested, especially when it comes to response time. MergingWF and SSS both outperform other techniques and display similar performance levels.

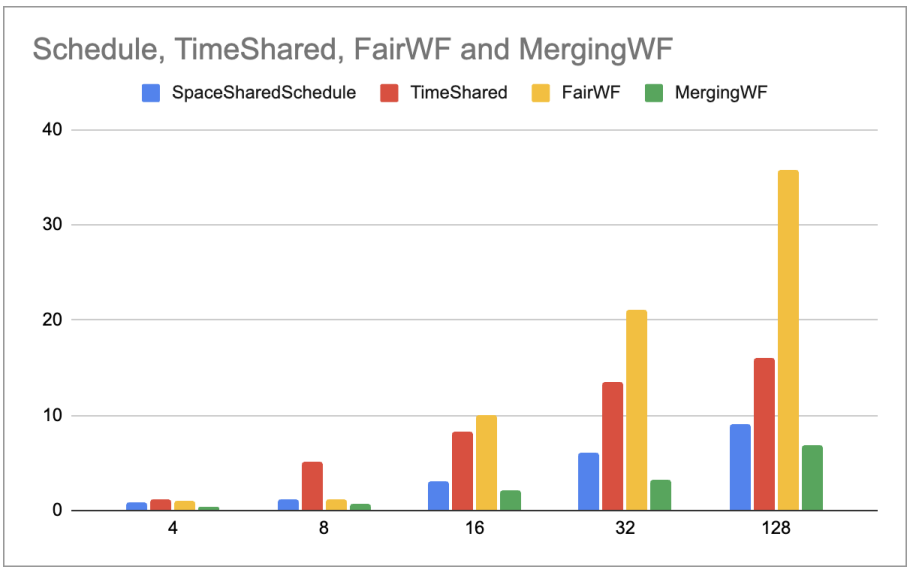

Figure 7: Response Time vs Batch size

The average start time of workflows, or response time, is positively impacted by the level-wise allocation strategy used by SSS and MergingWF. These solutions' favorable reaction times can be attributed to the fact that every task execution starts from the very beginning. One feature of MergingWF is task merging, which combines tasks from

later depth levels with those from earlier levels without affecting the 1st level tasks and so has negligible influence over response time.

In contrast, because of its sequential task assignment, CFS records the least positive performance. CFS responds more slowly than other systems in part because of this serial allocation technique. Finally, the response time trends highlight the effectiveness of level-wise allocation in SSS and MergingWF, establishing them as better options when it comes to response time optimization in the experimental setup.

6.3 Makespan

Figure 8 shows that when batch size increases, the makespan shows a rising pattern. In terms of makespan performance, MergingWF continuously outperforms all other techniques over X axis ranging from 4 to 128 representing no of tasks. Notably, for X axis ranging from 4 to 128 representing no of tasks, MergingWF exhibits improvements between 1% and 5% over SSS.

The order of performance is consistent with what was seen when it came to turnaround time, with MergingWF doing better than SSS. The previously clarified factors are responsible for the consistency in the performance order. In comparison to other approaches in the experimental situation, MergingWF is positioned as a robust and reliable solution due to its noteworthy performance across different batch sizes, which highlights its efficacy in lowering makespan.

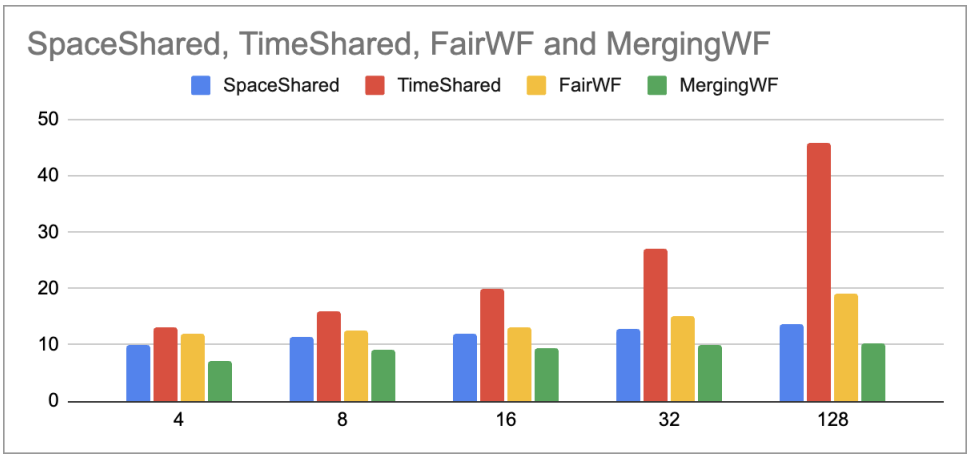

Figure 8: makespan (Makespan) vs Batch size

6.4 System Utilization

While keeping other parameters constant, Figure 9 shows that the system utilization is trending upward with an increase in batch size. MergingWF (optimal), SSS, TSS, and CFS (suboptimal) are the positions in the hierarchy of system utilization performance. System utilization is consistently higher with MergingWF than with any other strategy, even with different batch sizes.

MergingWF's high performance can be attributed to its deliberate design that takes advantage of parallelism at both the task and workflow levels. Uniform allocation on the available virtual machines (VMs) is how this is accomplished. Workload accumulation leads to greater parallelism in the batch since no of tasks are increased and are available at each priority level. Task merging is the method by which MergingWF resolves the size differences with tasks at the similar priority level. It is specifically designed to take advantage of parallelism. Because of the more efficient use of system resources that this novel approach guarantees—especially in situations when task sizes fluctuate—the system utilization performance has been seen to improve.

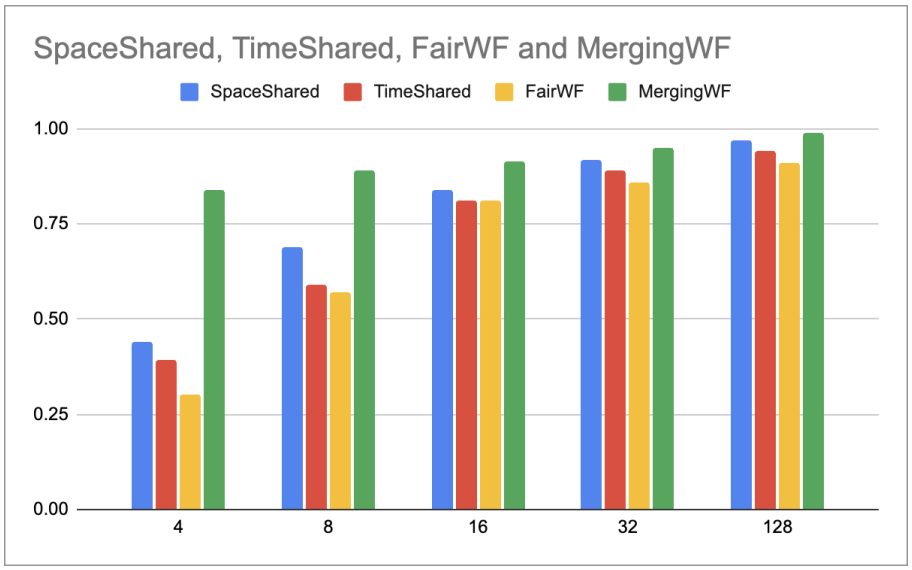

Figure 9: System Utilization vs Batch size

6.5 Discussion

An analysis of the four scheduling strategies—MergingWF, CFS, SSS, and TSS—presented here offers insightful information about how well each performs in terms of response time, turnaround time, makespan, and system usage. The results show that, for a range of batch sizes, MergingWF consistently performs better than the other techniques. Nonetheless, a critical assessment of the experiments highlights a number of areas that require additional discussion and improvement.

1. **Turnaround Time:**

   - As the number of tasks increases, the turnaround time trend is rising as expected, which is a sign of resource contention. A more thorough understanding might be possible, though, if the causes of this pattern were investigated more thoroughly.
   - Even though MergingWF shows the least increase in turnaround time, it would be useful to investigate particular situations or circumstances where alternative techniques might perform better than MergingWF in order to assist identify potential drawbacks.

2. **Response Time:**

   - Although level-wise allocation is shown to have a favorable influence on response times in SSS and MergingWF, a more thorough examination of the circumstances in which this tactic could not work as well will provide additional context to the conversation.
   - Gaining knowledge of the elements that support MergingWF and SSS's reliable and flexible operation at different batch sizes may help to explain their resilience.

3. **Makespan:**

   - The consistent superiority of MergingWF across various batch sizes is highlighted. It would improve the discussion to take a closer look at particular situations or task features that lead to this superiority.
   - A more comprehensive understanding of MergingWF's effectiveness and originality could be obtained by comparing it to the body of existing work on makespan optimization techniques.

4. **System Utilization:**

   - Although the rationale for MergingWF's high system utilization is insightful, a more nuanced understanding would be gained with a discussion of instances in which the improvement in system utilization would not be as noticeable.
   - It would be beneficial to the discussion to examine any trade-offs or difficulties related to MergingWF's parallelism at the task and workflow levels.

A comparison with related research assessing scheduling strategies like SSS, TSS, and CFS in the context of earlier research has brought the originality and contributions of the work under consideration to light. Furthermore, it's critical to identify any potential flaws in the experimental design and offer improvements. In order to evaluate the adaptability of the techniques, the reader may want to think about extending the experiment's parameters by adding more workload characteristics or changing system settings. In order to give a thorough grasp of MergingWF's advantages and disadvantages, conduct a comparative analysis with more varied scheduling methods. This discussion intends to increase the robustness and application of the results by critically analyzing the experiments and offering suggestions for improvement. This will promote a deeper comprehension of the scheduling techniques' performance in many contexts.

7 Conclusion and Future Work

Methods dealing with batches that consist of several workflows perform better by carefully taking into account parallelism at the task and workflow levels while respecting precedence limits. In order to optimise makespan and improve QoS in IaaS, this study presents a level-based, methodical allocation model for various workflows that incorporates task merging. Inspired by clustering concepts, the task merging technique seeks to minimize the overhead of communication between the task while reducing the overall no of level. This method is used in every workflow after diving the task at different level,

and allocation proceeds level-by-level from the first to the last, which helps to improve execution time overall.

Furthermore, the suggested approach demonstrates flexibility by employing simple and flexible level attributes to handle precedence limitations, offering a specified sequence of action for jobs in workflows. An empirical study contrasts MergingWF's performance with that of CFS, SSS, and TSS, three other well-known Directed Acyclic Graph (DAG) scheduling algorithms. The number of accessible nodes (hardware parallelism), the degree of workflow parallelism (depth levels), and batch size variations are among the experimental variables. The experimental study's results continuously highlight MergingWF's better performance in every scenario that was assessed, proving its effectiveness as the best option available to its competitors.

References

Abd Elaziz, M., Xiong, S., Jayasena, K. and Li, L. (2019). Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution, *Knowledge-Based Systems* **169**: 39–52.

Alhaidari, F., Balharith, T. and Eyman, A.-Y. (2019). Comparative analysis for task scheduling algorithms on cloud computing, *2019 International Conference on Computer and Information Sciences (ICCIS)*, IEEE, pp. 1–6.

Aron, R. and Abraham, A. (2022). Resource scheduling methods for cloud computing environment: The role of meta-heuristics and artificial intelligence, *Engineering Applications of Artificial Intelligence* **116**: 105345.

Beg, M. A., Alam, M. and Shahid, M. (2021). A workflow allocation strategy under precedence constraints for iaas cloud environment, *Innovations in Computational Intelligence and Computer Vision: Proceedings of ICICV 2020*, Springer, pp. 10–17.

Bezdan, T., Zivkovic, M., Bacanin, N., Strumberger, I., Tuba, E. and Tuba, M. (2022). Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm, *Journal of Intelligent & Fuzzy Systems* **42**(1): 411–423.

Chandrashekar, C., Krishnadoss, P., Kedalu Poornachary, V., Ananthakrishnan, B. and Rangasamy, K. (2023). Hwacoa scheduler: Hybrid weighted ant colony optimization algorithm for task scheduling in cloud computing, *Applied Sciences* **13**(6): 3433.

Chen, X., Cheng, L., Liu, C., Liu, Q., Liu, J., Mao, Y. and Murphy, J. (2020). A woa-based optimization approach for task scheduling in cloud computing systems, *IEEE Systems journal* **14**(3): 3117–3128.

Fu, A. (2022). 7 different types of cloud computing structures, url `https://www.uniprint.net/en/7-types-cloud-computing-structures/`. Accessed: 20 Nov, 2023.

Gupta, A., Pauri, U. and Bhadauria, H. (2022). Honey bee based improvised bat algorithm for cloud task scheduling.

Hamdani, A. and Abdelli, A. (2020). Towards modelling and analyzing timed workflow systems with complex synchronizations, *Journal of King Saud University-Computer and Information Sciences* **32**(4): 491–504.

Jamal, M. K. and Muqeem, M. (2022). Optimization algorithms for efficient workflow scheduling in iaas cloud, *2022 2nd International Conference on Emerging Frontiers in Electrical and Electronic Technologies (ICEFEET)*, IEEE, pp. 1–6.

Kansara, A. S., Gohil, B. N. and Verma, N. (2023). Task scheduling in cloud computing using mean ant colony optimization, *2023 6th International Conference on Information Systems and Computer Networks (ISCON)*, IEEE, pp. 1–8.

Kruekaew, B. and Kimpan, W. (2022). Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning, *IEEE Access* **10**: 17803–17818.

Li, H., Wang, D., Zhou, M., Fan, Y. and Xia, Y. (2021). Multi-swarm co-evolution based hybrid intelligent optimization for bi-objective multi-workflow scheduling in the cloud, *IEEE Transactions on Parallel and Distributed Systems* **33**(9): 2183–2197.

Mikram, H., El Kafhali, S. and Saadi, Y. (2022). Processing time performance analysis of scheduling algorithms for virtual machines placement in cloud computing environment, *International Conference On Big Data and Internet of Things*, Springer, pp. 200–211.

Paulraj, D. et al. (2023). Reactive fault tolerance aware workflow scheduling technique for cloud computing using teaching learning optimization algorithm, *2023 Fifth International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, IEEE, pp. 1–7.

Sahu, B., Swain, S. K., Mangalampalli, S., Mishra, S. et al. (2023). Multiobjective prioritized workflow scheduling in cloud computing using cuckoo search algorithm, *Applied Bionics and Biomechanics* **2023**.

Savdur, S. N., Khamatshaleeva, G. A., Stepanova, G. S., Maslennikova, N. N. and Stepanova, J. V. (2021). Stream modeling of an online store based on modified petri nets in consumer cooperation, *Frontier Information Technology and Systems Research in Cooperative Economics*, Springer, pp. 787–796.

Shahid, M., Alam, M., Hasan, F. and Imran, M. (2021). Security-aware workflow allocation strategy for iaas cloud environment, *Proceedings of International Conference on Communication and Computational Technologies: ICCCT-2019*, Springer, pp. 241–252.

Shahid, M., Ashraf, Z. and Alam, M. (2021). A multi-objective workflow allocation strategyin iaas cloud environment, *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, IEEE, pp. 308–313.

Shahid, M., Ashraf, Z., Alam, M., Ahmad, F. and Imran, M. (2021). A multi-objective workflow allocation strategyin iaas cloud environment, *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, IEEE, pp. 308–313.

Wu, H., Chen, X., Song, X., Zhang, C. and Guo, H. (2021). Scheduling large-scale scientific workflow on virtual machines with different numbers of vcpus, *The Journal of Supercomputing* **77**: 679–710.