National College of Ireland

Cost Optimization in Hybrid Cloud Architecture

MSc Research Project
MSc in Cloud Computing

Poorva Shrivastava
Student ID: 21216941

School of Computing
National College of Ireland

Supervisor: Aqeel Kazmi

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Poorva Shrivastava
Student ID:	21216941
Programme:	MSc in Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Aqeel Qazmi
Submission Due Date:	31/01/2024
Project Title:	Cost Optimization in Hybrid Cloud Architecture
Word Count:	8455
Page Count:	24

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:	Poorva Shrivastava
Date:	25th January 2024

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):	

Cost Optimization in Hybrid Cloud Architecture

Poorva Shrivastava
21216941

Abstract

Workflow scheduling is a vital component of resource optimization in hybrid cloud environments because of its flexibility and affordability. Ant Colony Optimization (ACO) and The Hybrid Cloud Optimized Cost Scheduling Algorithm (HCOC), two well-known methods for hybrid cloud workflow scheduling, are compared in this research. ACO exhibits flexibility and practical utility in healthcare and energy conservation, whereas HCOC is customized for scientific processes and computational applications, prioritizing cost reduction and timely completion. By incorporating AI-driven autoscaling with Q-Learning, both techniques become more flexible in dynamic hybrid cloud environments. The study concludes that the decision between ACO and HCOC is based on the particular workflow needs as well as the importance of cost optimization and deadline compliance. Further research topics include improved integration of AI-driven autoscaling, real-time monitoring, sophisticated cost estimation models, and security considerations. This comparative research provides insightful information that can be used to choose the best method to handle the dynamic issues associated with scheduling workflows in hybrid clouds.

Keywords: Hybrid cloud, Workflow scheduling, Ant Colony Optimization (ACO), The Hybrid Cloud Optimized Cost Scheduling Algorithm (HCOC), AI-driven autoscaling, Q-Learning, Resource optimization, Cost optimization

1 Introduction

Cost optimization has become a crucial problem for industry and academia in the ever-changing field of cloud computing. It is now critical to manage resources efficiently and provide services at a reasonable cost due to the growth of cloud-based services and the increasing complexity of cloud infrastructures. In this sense, hybrid cloud environments—which blend private and public cloud services—offer a special set of potential and problems, as noted by Bittencourt and Madeira (2011). These environments are naturally complicated; thus, in order to achieve service-level agreements (SLAs) and performance criteria while optimizing costs, elaborate scheduling and resource allocation strategies are required.

A number of important considerations make it necessary to research cost optimization in cloud computing. According to Zhou, Wang, Cong, Lu, Wei and Chen (2019), scheduling algorithms must be intelligent and flexible due to the dynamic nature of cloud services,

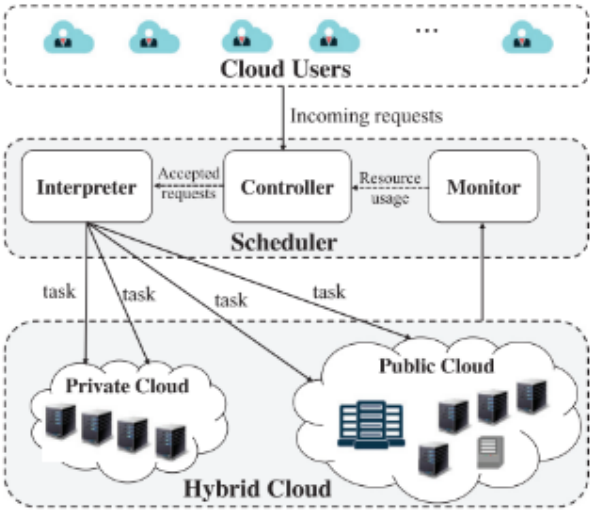

Figure 1: Hybrid Cloud System Architecture
Zhou, Wang, Cong, Lu, Wei and Chen (2019)

which is marked by variable resource availability and varying demand. Second, Ritchie and Levine (2003) describe how the heterogeneous nature of cloud resources necessitates the use of algorithms that can effectively manage various resource types while minimizing costs. Finally, the financial side of cloud computing means that cost optimization is now a business requirement rather than just a technological difficulty as organizations depend more and more on cloud services for vital operations Yuan et al. (2009).

1.1 Research Questions and Objectives

In order to enhance the integration of heterogeneous and hybrid cloud environments inside frameworks for cost optimization, we highlight two important research concerns in this paper.
The first question focuses on how to include these environments into cost optimization algorithms in an efficient manner, highlighting the necessity of combining different cloud infrastructures in a smooth and effective manner.
The second research question explores how to apply AI-driven autoscaling to improve cloud computing cost optimization, with an emphasis on the Q-Learning technique.

Following is a summary of the main goals of this research project. Our initial goal is to provide a thorough examination of the current cost optimization methods, paying special attention to how well they function in hybrid cloud systems. Two prominent algorithms that we specifically consider in this context are the Hybrid Cloud Optimized Cost scheduling algorithm (HCOC) and Ant Colony Optimization (ACO). Evaluating their efficiency in cost optimization within the intricate realm of hybrid and heterogeneous cloud environments is our aim.

Second, we use Q-Learning as a key component to investigate the possible advantages of combining these optimization methods with AI-driven autoscaling. This integration attempts to assess how such synergy can lead to cost savings in the long run by improving system efficiency and resource allocation. This study intends to make substantial

contributions to the field of cloud computing cost optimization and open the door for more effective and adaptable cloud infrastructures by tackling these research problems and objectives.

The study's hypothesis is that combining AI-driven autoscaling with existing cost optimization techniques will dramatically improve cost efficiency and resource management in hybrid cloud systems.

1.2 Contribution to the Scientific Literature

This study contributes to the scientific literature by offering a complete comparison of two major cost optimization algorithms: ACO and HCOC. Bittencourt and Madeira (2011) point out that while ACO has been extensively researched in relation to scheduling and resource allocation in a variety of computing settings, HCOC focuses on the particular difficulties associated with hybrid cloud environments. This study also investigates the new integration of Q-Learning for autoscaling Garí et al. (2021), with the goal of adaptively managing resources in response to real-time demand and workload variations.

This study further broadens our understanding of cost optimization in cloud computing by investigating the interactions between various optimization techniques and the dynamic nature of cloud settings. As cloud computing continues to develop and take on new roles in contemporary computing environments, this study looks into how AI-driven techniques might support conventional optimization algorithms to achieve higher cost efficiency and resource usage.

Ultimately, investigating cost optimization in hybrid cloud settings—especially from the perspective of AI integration—is not only a technical project but also a critical first step toward implementing more effective, affordable, and sustainable cloud computing techniques. The goal of this study is to provide this quickly evolving sector with insightful analysis and useful answers.

2 Related Work

When it comes to managing computational workloads, hybrid cloud solutions provide both flexibility and affordability. Workflow scheduling is critical in these situations for optimizing resource allocation, minimizing costs, and meeting deadlines. This problem has been tackled by a number of algorithms, such as Genetic Algorithms (GA) and Ant Colony Optimization (ACO). In order to schedule hybrid cloud workflows, this literature review compares the efficacy of ACO with a genetic algorithm known as Deadline-constrained Cost Optimization (HCOC). The paper also investigates how these scheduling algorithms might be made to perform better by incorporating Q-Learning into AI-driven autoscaling.

2.1 Ant Colony Optimization for Cloud Workflow Scheduling

One unique natural optimization method that shows promise for workflow scheduling in hybrid cloud systems is ant colony optimization (ACO). To improve work allocation

efficiency in this situation, Gandhi and Revathi (2022) have suggested an improved ACO-based strategy. Pheromone-based decision-making is the foundation of ACO algorithms, which imitate ants' foraging strategies. This creative solution gives ACO the ability to address complex issues like workflow scheduling with exceptional efficacy.

2.1.1 Strengths of Ant Colony Optimization

Ant Colony Optimization has many advantages. First of all, as Gandhi and Revathi (2022) work in 2022 highlights, ACO algorithms really do well in dynamic and stochastic contexts. Second, Zhou, Wang, Cong, Lu, Wei and Chen (2019) study shows how well these algorithms perform in handling the challenges of process scheduling in hybrid cloud systems. In addition, research conducted by Tuba and Jovanovic (2013) shown that ACO algorithms can be used to solve difficult non-linear optimization problems by methodically examining several options.

2.1.2 Limitations of Ant Colony Optimization

Ant Colony Optimization is not without its drawbacks, though. One significant disadvantage is that ACO algorithms may have a high processing burden, which may require configuration adjustments to achieve the best outcomes. Gandhi and Revathi (2022) raised this issue in their 2022 study. Ritchie and Levine (2003) highlighted the possibility of scalability issues for ACO algorithms in large-scale operations and extremely heterogeneous hybrid cloud infrastructures. ACO's advantages make it an attractive option for handling complex workflow scheduling issues in the context of hybrid cloud systems, even in spite of these drawbacks.

2.2 Hybrid Cloud Optimized Cost scheduling algorithm(HCOC)

In order to tackle the difficulties of hybrid cloud workflow scheduling, Bittencourt and Madeira (2011) created the HCOC genetic algorithm. This novel technique is a workable solution for real-world hybrid cloud applications because it seeks to reduce operational expenses while guaranteeing that workflow execution deadlines are met. When orchestrating the scheduling of processes, HCOC takes into account a number of important criteria, including resource availability, budgetary restrictions, and workflow interdependence.

2.2.1 Strengths of HCOC

There are a number of noteworthy advantages that HCOC has. Initially, it is designed to address the distinct difficulties presented by hybrid cloud setups, which combine private and public cloud resources. Second, Yuan et al. (2009) show how it effectively combines timeliness and cost control by utilizing the power of genetic algorithms. The last application area in which HCOC has demonstrated its efficacy is workflow scheduling, as demonstrated by Lin et al. (2015), and the scheduling of MapReduce processes, as demonstrated by Wang and Shi (2014).

2.2.2 Limitations of HCOC

But there are certain restrictions with HCOC. As mentioned by Bittencourt and Madeira (2011), one major difficulty is building the models for cost and deadline estimations precisely, which can be a difficult undertaking. In addition, as Zuo et al. (2013) pointed out,

the dynamic nature of hybrid cloud deployments—which is marked by varying resource availability and demand—may make it difficult for HCOC's optimization process to adjust and react appropriately. In spite of these drawbacks, HCOC is nevertheless a useful tool for hybrid cloud workflow scheduling since it provides a workable solution for striking a balance between timely execution and cost-effectiveness in complicated computing environments.

2.3 ACO vs. Hybrid Cloud Optimized Cost scheduling algorithm(HCOC) for Hybrid Cloud Workflow Scheduling

Comparing Ant Colony Optimization (ACO) and Deadline-Constrained Cost Optimization (HCOC) in the context of hybrid cloud workflow scheduling is crucial, with an emphasis on their application scenarios and practical implementations.

Ant Colony Optimization (ACO):
Ant Colony Optimization (ACO) is a flexible method that is applied in many different fields, including medical systems. The work of Gandhi and Revathi (2022) provides an instructive example of its practical use, wherein ACO was utilized to improve the scheduling of crucial healthcare tasks associated with patient data processing and analysis. This actual use case demonstrates how ACO can adjust to vital circumstances that are constantly changing in the healthcare industry.
ACO has also demonstrated effectiveness in heterogeneous cloud computing systems when it comes to energy consumption optimization. Tuba and Jovanovic (2013) conducted a study that included a detailed discussion of the application of ACO for cloud computing energy optimization. Here, ACO's adaptability and efficiency are evident, demonstrating its capacity to handle a wide range of issues in a variety of application domains.

The Hybrid Cloud Optimized Cost scheduling algorithm (HCOC):
A particular method created to handle the challenges of managing hybrid cloud systems is called the Hybrid Cloud Optimized Cost Scheduling Algorithm (HCOC). HCOC's potential in the context of cloud and grid computing was investigated in a 2016 study by Alkhanak et al. (2016), with a focus on scheduling scientific activities. Scientific procedures sometimes include strict time limitations and complex task relationships that call for exact orchestration, which makes this use case especially important.
In addition, HCOC may be used in a variety of situations due to its unique characteristics in cost and timeline optimization. According to Zeng et al. (2012) and Malawski et al. (2013), for example, it can be used efficiently in cloud-based many-task workflow applications and computational applications. These applications highlight the crucial need for algorithms such as HCOC, which are excellent at finding a fine balance between meeting deadlines and maximizing financial resources. This allows organizations to effectively manage their workloads in the cloud while complying with time-sensitive requirements.

2.4 Integration of AI-Driven Autoscaling with Q-Learning

AI-powered autoscaling is essential for improving cloud workflow scheduling effectiveness because it allows for dynamic resource allocation that adjusts to workload needs that change over time. For the objective of autoscaling scientific workflows in the cloud, Gari et al. (2022) presented a novel solution that makes use of the reinforcement learning

algorithm Q-learning. Based on the unique requirements of the workload, this novel approach uses Q-learning to make judgments about resource allocation in real-time. This strategy tries to improve autoscaling efficiency while guaranteeing the achievement of performance objectives by continuously learning from previous experiences and refining its decision-making process.

Similarly, Pulle et al. (n.d.) presented the idea of performance monitoring and AI-assisted autoscaling, without mentioning the specific method they employed. Their area of interest is AI-driven autoscaling, which uses machine learning to enable response to changing cloud environment conditions.

Integration with ACO:
When Ant Colony Optimization (ACO) and AI-driven autoscaling are combined, it is clear that significant performance gains can be achieved. According to Gari et al. (2022), this integration combines ACO with Q-Learning to produce adaptive decision-making that depends on real-time input. By combining these two strategies, resources are allocated as efficiently as possible during the workflow execution process, guaranteeing that the cloud infrastructure adjusts to the demands of the workload.

Also, by including Q-Learning, a reinforcement learning technique, into the model, ACO enables resource allocation plans to be modified in response to changes in the hybrid cloud environment. This dynamic resource assignment ensures that workflows are efficiently scheduled to optimize efficiency and resource consumption, all the while meeting predetermined deadlines.

Integration with HCOC:
When combined with AI-driven autoscaling powered by Q-Learning, the Hybrid Cloud Orchestration and Control (HCOC) solution becomes more adaptable and effective in dynamic hybrid cloud environments. In keeping with the findings of the Gari et al. (2022), this integration makes use of Q-Learning's dynamic real-time resource allocation capabilities, with an emphasis on completing project deadlines and maintaining cost-effectiveness. Making sure that computing resources are allocated optimally to fulfill the particular workflow needs depends on Q-Learning's capacity to make resource allocation decisions based on real-time feedback.

Organizations may combine adhering to project timelines and lowering expenses in complicated hybrid cloud systems by combining Q-Learning with HCOC. In these kinds of contexts, where the demands on computational resources might fluctuate greatly and suddenly, this kind of handling complex processes is very important. As a result, this flexible method is essential for attaining successful resource management and operational outcomes in hybrid cloud setups.

2.5 Critique & Filling the gaps

When discussing hybrid cloud workflow scheduling, two key concepts are Ant Colony Optimization (ACO) and Hybrid Cloud Optimized Cost (HCOC). Whereas HCOC is intended for managing hybrid cloud systems, specifically for scheduling scientific activities, ACO is a versatile approach used in many domains, including cloud computing. Both

have proven successful in various situations, including cost and schedule optimization and energy usage optimization.
In cloud-based workflow applications, HCOC has been utilized for cost and timetable optimization, while ACO has been applied to energy optimization and job scheduling in cloud computing. There have been notable improvements in adaptive decision-making and resource allocation when ACO is combined with AI-driven autoscaling, especially when Q-learning is used. Similarly, it has been demonstrated that HCOC integration with AI-driven autoscaling powered by Q-learning improves efficacy and flexibility in dynamic hybrid cloud systems.
Both ACO and HCOC have advantages and uses in the context of hybrid cloud workflow scheduling. While HCOC is especially helpful for cost and timing optimization in complex job interactions, ACO is renowned for its versatility and efficiency in managing a wide range of difficulties. The integration of AI-driven autoscaling via Q-learning with ACO and HCOC can augment their potential in adaptive decision-making, resource distribution, and dynamic environment administration.

A novel hybrid algorithm that combines the benefits of ACO and HCOC with AI-driven autoscaling utilizing Q-learning could be developed to cover the gaps and improve hybrid cloud workflow scheduling. In order to enable dynamic resource allocation and adaptive decision-making based on real-time feedback, this hybrid algorithm might combine the flexibility and efficiency of ACO with the cost and timeline optimization capabilities of HCOC. It would also integrate AI-driven autoscaling with Q-learning. This innovative method may provide a complete answer for hybrid cloud workflow scheduling by fusing the benefits of ACO, HCOC, and Q-learning-based autoscaling. This would help to address the issues of adaptability, cost optimization, and dynamic resource management in complicated and unpredictable environment.

3 Research Methodology

In order to optimize cloud computing costs, this study compares the performance of Ant Colony Optimization (ACO) and the Hybrid Cloud Optimized Cost scheduling algorithm for Hybrid Cloud (HCOC). After determining the best algorithm, it will be merged with a Q-learning-based autoscaling system. A well-known procedure in cloud computing research, the Montage dataset, will be used in the evaluation to simulate a real-world scenario.

3.1 Scientific Workflows

A Direct Acyclic Graph (DAG) can be used to depict a workflow, $G = \langle V, E \rangle$, where V is a collection of tasks and E is a set of edges representing the data dependency restrictions between the tasks. We also receive a set of virtual machines (VMs) to perform the tasks of the workflows $VM = (vm_1, vm_2, \ldots, vm_m)$. For each $vi \in V$, This task's execution must be delayed until it receives all of the necessary data from its parent tasks. The parent task that sends its data to vi last is known as the most influential parent (MIP) for that task. A task begins execution by utilizing the initial data submitted with the workflow or the intermediate data obtained from the task's parents. The term "communication cost" describes the amount of time needed to transfer data between two operations. The two

tasks' communication cost is regarded as zero when they are both running on the same virtual machine.

3.2 Algorithm Comparison

The core of the research initiative lies in the systematic comparison of two prominent optimization algorithms: Ant Colony Optimization (ACO) and the Deadline- Constrained Cost Optimization Algorithm for Hybrid Cloud (HCOC). This pivotal section involves a detailed examination of how each algorithm performs in the complex landscape of hybrid cloud environments. ACO, inspired by nature, employs pheromone- based decision-making, showcasing adaptability in dynamic and stochastic contexts. On the other hand, HCOC is specifically tailored for the challenges posed by hybrid cloud setups, emphasizing cost reduction and timely completion of workflows.Zhou, Zhang, Sun, Zhou, Wei and Hu (2019) The comparison evaluates these algorithms based on their efficacy in addressing the inherent complexities of hybrid clouds, considering factors such as workflow scheduling efficiency, cost optimization, and compliance with deadlines. This rigorous analysis forms the basis for determining the strengths and limitations of each algorithm, guiding the research towards identifying the most suitable approach for hybrid cloud cost optimization.

3.3 Integration with AI-Driven Autoscaling

The integration of AI-driven autoscaling, specifically leveraging Q-Learning, represents a significant advancement in the research methodology. This section focuses on enhancing the flexibility and adaptability of the Ant Colony Optimization (ACO) and Deadline-Constrained Cost Optimization Algorithm for Hybrid Cloud (HCOC) through the infusion of artificial intelligence. By incorporating Q-Learning into the algorithms, the research aims to imbue them with the capability for real-time decision-making in response to fluctuating demands and workload variations within hybrid cloud environments. Autoscaling, driven by artificial intelligence, becomes a dynamic mechanism for adjusting the allocation of resources based on the evolving needs of the system. The integration with Q-Learning is poised to optimize resource efficiency by learning from past experiences, making these algorithms more responsive to the dynamic nature of hybrid cloud scenarios. Verma and Kaushal (2017)

3.4 Montage Dataset

The utilization of the Montage dataset is a critical component of the research methodology, offering a practical and diverse benchmark for the evaluation of cost optimization algorithms within hybrid cloud environments. This section emphasizes the significance of employing a real-world dataset to simulate scenarios and gauge the performance of the Ant Colony Optimization (ACO) and Hybrid Cloud Optimized Cost scheduling algorithm for Hybrid Cloud (HCOC). The Montage dataset serves as a comprehensive repository of various performance metrics, including execution times, cost savings, resource utilization patterns, and compliance rates with Service Level Agreements (SLAs). By leveraging this dataset, the research ensures a robust and thorough evaluation, capturing the complexities of hybrid cloud architectures in terms of workflow scheduling, cost optimization, and deadline adherence. [1] Montage is an example of a workflow application with a range

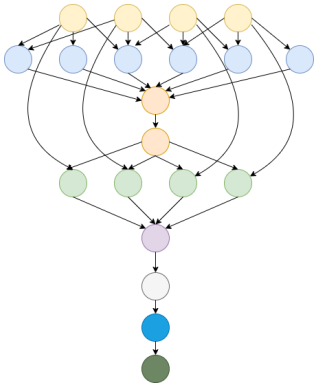

Figure 2: Montage Dataset
Bharathi et al. (2008)

of sizes. It is an imaging tool that creates sky mosaics for use in astronomy studies. The size of the procedure is determined by the desired sky's square degree. For instance, a workflow including 232 activities is carried out for a square degree of the sky. A process of 20,652 tasks, handling data close to 100GB, is carried out for 10 square degrees of the sky. Approximately 400,000 square degrees make up the entire sky. When private resources are insufficient to complete the workflow, our infrastructure and the suggested algorithm may manage this type of application by requesting resources from public clouds.

3.5 Performance Analysis

The section on Performance Analysis is a pivotal phase in the research methodology, dedicated to the systematic examination and interpretation of the outcomes derived from the application of Ant Colony Optimization (ACO) and Hybrid Cloud Optimized Cost scheduling algorithm for Hybrid Cloud (HCOC) within the Montage dataset. This segment emphasizes the quantitative assessment of various performance metrics to gauge the effectiveness of the algorithms in real-world scenarios. The research meticulously interprets the performance metrics to draw meaningful conclusions, enabling a deeper understanding of the algorithms' strengths and potential areas for improvement. This rigorous analysis forms the basis for the research's contribution to the existing body of knowledge in the domain of hybrid cloud cost optimization. Liu et al. (2017)

3.6 Implementation

1.Initial Setup: Configure a hybrid cloud system in CloudSim using the Montage dataset. This phase ensures a realistic simulation environment for testing the algorithms and sets the groundwork for the entire research process.
2.Algorithm Implementation: Use the two main algorithms: HCOC for hybrid cloud environments and ACO for private cloud environments. The actual coding and configuration of these algorithms inside the CloudSim environment takes place at this step.
3.Workflow Execution and Scheduling: Using the established algorithms, workflows represented by the Montage dataset are scheduled and deployed within the simulated environment during this phase. This stage is essential for evaluating ACO and HCOC's efficacy in a regulated environment.

[1] "Montage: An astronomical image engine." [Online]. Available:http://montage.ipac.caltech.edu/

4.Performance Evaluation: Key performance indicators, such as execution time, resource consumption, and cost, are meticulously recorded in order to assess the effectiveness of every scheduled operation. The efficiency and efficacy of the algorithms can only be compared with the help of these indicators.
5.Algorithm Enhancement with Q-Learning: Building a Q-learning model for autoscaling in a hybrid cloud environment and combining it with the more effective of the two algorithms (ACO or HCOC) are the tasks involved in this step. The integration is intended to improve the performance and adaptability of the chosen algorithm.
6.Re-run Workflows: The improved method is used to rerun the workflows, and the same performance metrics are noted for future analysis. To evaluate the effect of the Q-learning integration on the algorithm's performance, this stage is crucial.
7.Data Analysis and Reporting: Lastly, an analysis is done on the gathered data to compare how well the algorithms work. The findings are then combined into an extensive report that offers analysis and research conclusions.

This organized approach ensures a thorough and systematic analysis of the algorithms and the efficiency of AI-driven advances in cloud computing environment.

4 Design Specification

A detailed understanding of the techniques and applications of the Hybrid Cloud Optimized Cost (HCOC) and Improved Ant Colony Optimization Workflow Scheduling (IACOWS) scheduling algorithms in a cloud computing environment can be obtained from the design specifications. This is a broader interpretation of these requirements that takes into account more precise characteristics and their consequences.

4.1 A More Comprehensive Overview of IACOWS

- **Initialization:** The user's tasks and cloud resources are entered. The number of ants, heuristic importance, and pheromone evaporation rate are initialized in the ACO parameters.
- **Task Validation and Resource Availability:** Every task has its limitations and requirements verified. Additionally, the availability of resources is verified.
- **ACO Mechanism:** A multi-ant system is used by the method, in which each ant stands for a possible job scheduling solution. While completing tasks, ants use heuristic knowledge and pheromone trails to construct solutions.
- **Solution Generation and Evaluation:** Based on execution time and cost, each ant's solution is assessed. Solutions are arranged and kept in storage.
- **Pheromone Update:** Ants who follow better paths are encouraged to do so by pheromones that are updated on paths based on the quality of solutions.
- **Selection of Best Solution:** Task scheduling proceeds with the best solution, either after a predetermined number of iterations or upon meeting a convergence criterion.

Implications and Use Cases

- IACOWS is a good fit for private cloud setups where cost and performance must be optimized in job scheduling and resource allocation.
- It is effective for real-time job scheduling because of its iterative structure, which allows it to adjust to dynamic changes in cloud environments.

Algorithm 1 Improved Ant Colony Optimization Workflow Scheduling

Require: user tasks (t1, t2, ..., tn) and cloud resources (r1, r2, ..., rk)
Ensure: Appropriate task-resource mapping

```
Initialize ACO parameters (α, β, ρ, m, max)
Initialize solution storage structures (Fsrt, Listrt, Srt)
for each task t_i in t1 to tn do
    Check task validity and resource availability
    if available then
        Allocate resources and compute cost and time
        Apply ACO process:
        while max criteria not met do
            Generate feasible solutions and evaluate them
            Update pheromones and select the best solution
        end while
    end if
    Schedule tasks based on the best solution
    Execute tasks and release resources upon completion
end for
```

Gandhi and Revathi (2022)

4.2 Detailed process of HCOC:

- **Initial Scheduling:** The private cloud is where the workflow is first scheduled. This first schedule's makespan is contrasted with the deadline.
- **Rescheduling in Hybrid Cloud:** The system chooses which jobs to postpone in the hybrid cloud if the makespan is longer than the deadline. Here, tasks are prioritized and their estimated completion durations are estimated.
- **Resource Selection from Public Cloud:** Next, the system determines whether public cloud resources are suitable based on factors like price, core count, and performance. Meeting the timeline and optimizing the cost are the objectives.
- **Task Clustering and Resource Allocation:** The arrangement of tasks and resource allocation minimizes communication costs between tasks and maximizes resource use.
- **Iterative Improvement:** Until the deadline is reached or a predetermined number of iterations is reached, the algorithm iterates through these steps, refining the scheduling plan.

Implications and Use Cases

- HCOC is intended for hybrid cloud setups where deadlines and costs are important considerations. It successfully balances the cost of accessing public cloud resources with the requirement to achieve task deadlines.
- When workload changes and fluctuating resource availability are prevalent, the algorithm is especially helpful.

Algorithm 2 Hybrid Cloud Optimized Cost (HCOC) Scheduling Algorithm

R = all resources in the private cloud
Schedule workflow G in the private cloud using PCH
while makespan(G) > Deadline D AND iteration < size(G) **do**
  iteration = iteration +1
  Select task n_i with maximum priority P_i not in current rescheduling group T
  Add n_i to T
  Calculate the number of clusters in T
  while num_clusters > 0 **do**
    Select public cloud resource r_i minimizing price per core
    Add r_i to public cloud resource pool H
    Update num_clusters
  end while
  Schedule each task n_i in T on resource in H with smallest EFT
  Recalculate ESTs and EFTs
end while

Zhou, Wang, Cong, Lu, Wei and Chen (2019)

Comparative Analysis

- **Flexibility:** While HCOC offers a structured strategy that is concentrated on cost and timeline limitations, ACO allows flexibility in resource utilization and is responsive to fluctuating workloads.
- **Resource Utilization:** ACO is designed to maximize how resources are used in a private cloud environment. On the other hand, HCOC effectively administers resources in both public and private cloud environments.
- **Suitability:** When maximizing task execution time and resource allocation is the main goal of the environment, ACO is more appropriate. Conversely, HCOC works best in situations where timelines and cost control are critical.

Handling various aspects of process scheduling difficulties, ACO and HCOC each offer special advantages to cloud computing. In hybrid cloud environments, HCOC excels at managing costs and deadlines, whereas ACO excels at resource optimization in private clouds. The efficient and effective management of cloud resources can be greatly improved by their deployment, meeting the various requirements of contemporary cloud-based services and applications.

4.3 Q-Learning Based AutoScaling to Integrate with Optimization Algorithms

Implementing a Q-Learning method for autoscaling in cloud computing environments is the purpose of the QLearningAutoscaling class. This class is a component of a bigger

system that strives to dynamically optimize resource allocation according to workload. The Q-Learning algorithm is a reinforcement learning technique used to identify the optimal action for a given state by learning the value of actions in different states.Gari et al. (2022)

Algorithm 3 QLearningAutoscaling Class

Class : QLearningAutoscaling
Private : QTable qTable Double learningRate Double discountFactor
Constructor : (
1 learningRate, discountFactor) Initialize qTable Set learningRate Set discountFactor
Method : trainQTable(state, action, reward, nextState)
2 currentStateAction = concatenate(state, action) currentQValue = qTable.getQValue(currentStateAction) maxNextQValue = max(QValue for each value in nextState) newQValue = currentQValue + learningRate * (reward + discountFactor * maxNextQValue - currentQValue) qTable.updateQValue(currentStateAction, newQValue)
Method : selectAction(state)
3 action = select max value action from state based on Q-values return action
End Class :

4.3.1 Functionality:

State Definition: Describe the current condition of the cloud environment, taking into account variables such as workload demand, system performance metrics, and resource utilization.
Action Determination: Establish the range of feasible action like adding or removing virtual machine instances—at each state.
Reward Mechanism: Develop a reward system that measures how much an activity will save money or increase performance.

Q-Learning Implementation: To find the best course of action for scaling decisions, apply the Q-learning method.
A Q-table that associates state-action pairs with corresponding Q-values (anticipated rewards) is updated during the learning process.

4.3.2 Integration with Cloud Environment:

Integrate with the cloud simulation environment (e.g., CloudSim) to receive real-time data and send scaling commands.
Connect your system to a cloud simulation environment (like CloudSim) to send and receive scaling commands and real-time data.

Feedback Loop: Establish a feedback system so that the Q-learning model can learn from and get better over time by receiving input on the effects of its scaling choices.

QTable Design
Purpose: The purpose of this is to help the Q-learning algorithm make better decisions by storing and updating the Q-values for every state-action pair.
Structure: Each entry in the Q-table should be represented as a multi-dimensional array

or hash-map, where each pair of states and actions has a corresponding Q-value.
Set Q-values to a default value to provide an objective beginning point for the learning process.
Update Mechanism: When updating the Q-values, adhere to the Q-learning update rule and consider the greatest Q-value of the subsequent state as well as the reward obtained after an action.
Because the Q-table may be accessed and modified concurrently in a cloud computing environment, make sure that modifications are consistent and safe for threads.
Persistence: In order to enable learning to be saved and continued between sessions, it is optional to develop techniques to persist the Q-table.

4.3.3 Integration with Other Components

CloudSim Integration: Assure smooth connection with CloudSim or any other cloud simulation tool so that real-time cloud environment data may be received by QLearningAutoscaling.java.Neves Calheiros et al. (2011)
Compatibility with Scheduling Algorithms: Construct QLearningAutoscaling.java in a way that allows it to interface with current cloud scheduling methods, such as HCOC or ACO.

4.3.4 Requirements

Scalability: The system has to be able to effectively manage workloads and cloud environments of different sizes.
Flexibility: Both scalability across several cloud services and adaptability to diverse cloud architectures are essential.
Performance: When making decisions about scaling in real-time, the system ought to guarantee that there is as little negative influence as possible on cloud performance.
Usability: Provide easy-to-understand documentation and tools for seamless connection with current cloud systems.

A reliable Q-learning based autoscaling system for cloud environments is the main goal of this design specification, which also describes the functionality and structure of QLearningAutoscaling.java and QTable. As workloads and cloud conditions change in real time, the system seeks to dynamically optimize resource consumption. It allows for effective resource allocation in the cloud by specifying states, actions, incentives, and Q-learning. It is a useful tool for cloud resource management because of its connection with the cloud environment and feedback loop, which guarantee continual improvement.

5 Implementation

In order to transform the theoretical aspects of the research into a workable, functional model, the final implementation stage was essential. Ant Colony Optimization (ACO), Hybrid Cloud Optimized Cost (HCOC), and Q-learning model integration and testing in a simulated hybrid cloud environment were the hallmarks of this stage. Optimal cost, efficiency, and resource usage in cloud computing tasks were the main goals.

Optimized Workflow Schedules: The ACO and HCOC algorithms produced optimized schedules for cloud computing tasks, which were primarily characterized by their efficient use of resources and adherence to cost and deadline constraints. These schedules were pivotal in demonstrating the practical applicability of the algorithms in a hybrid cloud environment.

Adaptive Resource Allocation Model: The integration of Q-learning into the scheduling process resulted in a model capable of intelligently scaling resources up or down based on real-time demand and workload characteristics. This adaptive model enhanced the flexibility and efficiency of the cloud environment, particularly in handling unpredictable or varying workloads.

Comprehensive Performance Data: A detailed dataset encapsulating various performance metrics was compiled. This dataset included quantitative measures like execution times, cost savings, resource usage patterns, and SLA compliance rates. It served as a foundational element for evaluating the effectiveness of the implemented solutions.

Tools and Languages Utilized

CloudSim for Simulation: CloudSim, a widely-recognized cloud computing simulation framework, was utilized for its ability to model and simulate complex cloud environments. It provided a versatile platform for testing the algorithms in varied scenarios mimicking real-world cloud infrastructures. Atanasov and Ruskov (2016)

Java for Algorithm Development: Java was chosen for its robustness and efficiency in handling complex computations. It was used to code the ACO, HCOC algorithms, and the Q-learning model, benefiting from its extensive library support and strong object-oriented programming capabilities.

Python for Data Analysis and Visualization: Python, known for its simplicity and powerful data processing libraries like Pandas, was used for data analysis. Visualization libraries such as Matplotlib and Seaborn helped in creating insightful charts and graphs, facilitating an easier understanding of the performance data.

Git for Version Control: Git provided a reliable system for version control, allowing the project team to efficiently manage and collaborate on the codebase. It was instrumental in tracking progress, handling code revisions, and maintaining a stable development environment.

Integrated Development Environment (IDE): An IDE, such as Eclipse, was used for development purposes. This IDE offered valuable features like code debugging, syntax highlighting, and code refactoring, which enhanced the coding process and efficiency.
[2]

The goal of this project is to create a hybrid cloud environment in which a cloud service provider offers multiple payment mechanisms for virtual machine instances. On-demand charging, reservation charging, and bidding charging are common charging mechanisms. Considering the on-demand charging mechanism, the virtual machine instance is charged

[2]https://aws.amazon.com/ec2/pricing/

Instance Name	Hourly Rate	vCPU	Memory	Network Performance
m7a.8xlarge	$1.85472	32	128 GiB	12500 Megabit
m7a.12xlarge	$2.78208	48	192 GiB	18750 Megabit
m7a.16xlarge	$3.70944	64	256 GiB	25000 Megabit
m7a.24xlarge	$5.56416	96	384 GiB	37500 Megabit
m7a.32xlarge	$7.41888	128	512 GiB	50000 Megabit
m7a.48xlarge	$11.12832	192	768 GiB	50000 Megabit
m7a.metal-48xl	$11.12832	192	768 GiB	50000 Megabit

Table 1: AWS Cloud Instance Pricing and Specifications

according to a certain unit time (such as Google billing by the minute, AWS, Alibaba Cloud, etc. billing by the hour), and if it is not satisfied with the billing time, it is charged according to the entire calculation duration. The AWS Cloud billing mechanism is used in this paper in Table 1.

The final implementation phase effectively combined sophisticated algorithms and AI-driven models into a coherent system, proving the actual applicability of these approaches in optimizing cloud computing environments. The usage of Java and CloudSim provided a dependable and adaptable foundation for this application, while Python tools permitted in-depth data analysis and visualization. This stage was essential in converting abstract ideas into a concrete, empirically based conclusion that demonstrated the possibilities of combining artificial intelligence (AI) with cloud resource management techniques.

6 Evaluation

This section examines the simulation outcomes of the two optimization methods, ACO and HCOC, as well as a comparative analysis conducted with the Montage dataset, a real-world workflow benchmark.
Following are the parameters we used for simulation

Table 2: Virtual machine parameter settings

Simulation Setup	
Number of hosts	2
VMs Configuration	
Number of VMs	100
VM RAM	512MB
Bandwidth	1000Mbps
MIPs	1000
Workload Environment	
Numbers of cloudlets	1000
ACO Parameters Setup	
Number of iterations (max)	100
Number of ants (m)	10

To evaluate the performance of ACO and HCOC under different deadlines, we set the deadline as

$$deadline = k \cdot ExeTime, \quad k \geq 1 \tag{1}$$

where k is a factor and ExeTime denotes the typical execution time for a specific process. The studies involved varying the value of k (k = 1.5, 2.0, 2.5, 3.0, 3.5, 4.0) to impose varied deadline constraints for a workflow. Zhou, Wang, Cong, Lu, Wei and Chen (2019)

After each algorithm was simulated separately, ACO took an average of 42.47 seconds to execute for 100 iterations and the result for HCOC is 10.12 seconds for 100 nodes.
The following formula was then used to determine the execution costs for each algorithms.

$$\begin{aligned} \text{cost} = {} & \text{datacentrehost.costPerStorage} \times \text{vm.size} \\ & + \text{datacentrehost.costPerRam} \times \text{vm.ram} \\ & + \text{datacentrehost.costPerBw} \times \text{vm.bw} \\ & + \text{datacentrehost.costPerMips} \times (\text{vm.mips} \times \text{vm.numberOfPes}) \end{aligned} \tag{2}$$

6.1 Visualizations

The "ACO vs. HCOC Optimization Comparison" graph compares the cost performance of two optimization techniques over varying deadlines. Based on the study, here is an evaluation.

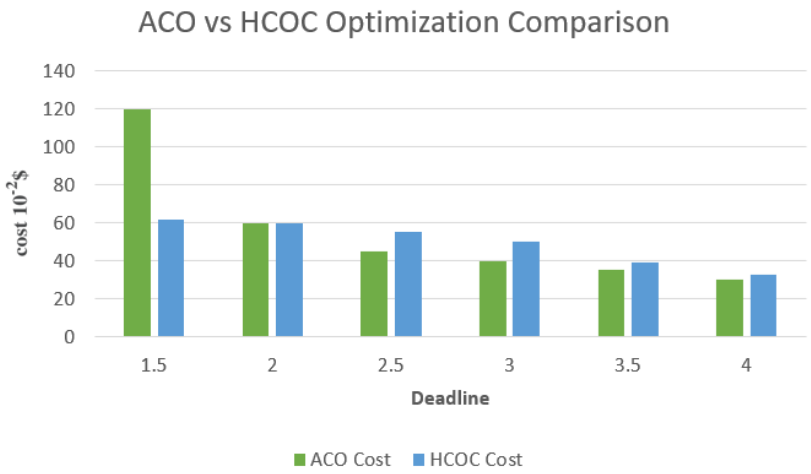

Figure 3: Comparison of ACO vs HCOC Cost Optimization

ACO Cost: At deadline 1.5, the Ant Colony Optimization (ACO) cost is initially much higher than the HCOC cost. This implies that under times of tight deadlines, ACO can need additional resources or incur higher operating expenses. But when deadlines go longer, the ACO cost drops off quickly and starts to stabilize, suggesting that ACO might become more effective when the strain of meeting deadlines gets relieved.

HCOC Cost: The Hybrid Cloud Optimized Cost Scheduling Algorithm (HCOC) starts lower than ACO at a deadline of 1.5 and maintains a relatively steady cost as deadlines climb. While the cost does drop, it does so more gradually than ACO, and it does so throughout all deadlines. This pattern implies that HCOC might be a more economical option all around and might be less susceptible to changes in deadlines.

Comparative Cost: HCOC looks to be the less expensive alternative across all deadlines, indicating that it might be the more cost-effective option for whatever job or project

is being optimized. The biggest discrepancy occurs at the earliest deadline (1.5), after which HCOC's advantage is maintained as the cost gap narrows.

Cost Trends: Both methods show a decreasing cost trend as the deadline approaches, demonstrating that having more time leads to reduced costs for these optimization procedures. This may be the result of several things, including better resource allocation, a decrease in the requirement for accelerated procedures, or features built into the optimization algorithms.

Implications for Decision Making: HCOC seems to be the best option if cost is the main consideration, particularly when there are short deadlines. In contrast, other criteria like the problem's complexity, the optimization's quality, or other resource considerations could influence the decision if the deadline is more flexible.

In order to choose between these two optimization techniques, this evaluation makes the assumption that lower costs are desirable and that there are no other considerations. In actuality, additional factors like the outcome's quality, the method's scalability, and the specific context in which these techniques are used would all be important factors to take into account when making decisions.

6.1.1 Comparative Analysis when integrated with Q-Learning

The bar graph compares the expenses incurred by two optimization algorithms—Hybrid Cloud Optimized Cost, or HCOC) and Ant Colony Optimization, or ACO—when combined with autoscaling based on Q-learning under different deadline constraints.

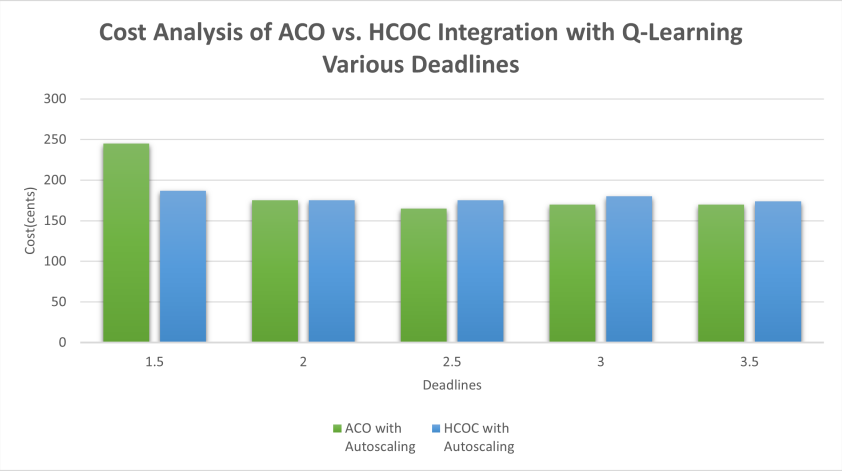

Figure 4: Comparison of ACO vs HCOC Cost Optimization when integrated with Q-learning based autoscaling

Here are some observations from the graph:
1.Cost Trends: It is clear from the graph's overall trend that when the deadlines are extended from 1.5 to 3.5 units, the expenditures related to both ACO and HCOC tend to go up. Longer deadlines may permit greater resource utilization, which would raise costs, according to this additive cost growth observed when deadlines loosen.
2.Algorithm Comparison: ACO shows a greater cost than HCOC at a 1.5-unit deadline. On the other hand, the cost of ACO drops dramatically and becomes less than

that of HCOC when the deadline is extended to two units. Both ACO and HCOC show approximately equal costs at a 2.5-unit deadline, indicating an efficiency point at which both algorithms optimize costs identically within the specified restrictions. But HCOC seems to result in greater costs than ACO for deadlines of 3 and 3.5 units.
3.Cost Efficiency: It is clear from cost optimization analysis that ACO outperforms HCOC when deadlines are loosened, keeping costs comparatively lower over longer deadlines. The significant cost savings for ACO, which went from 1.5 to 2 units by the deadline, points to either better resource management or more successful scaling during this time.
4.Impact of Deadline Relaxation: According to this research, ACO might be more appropriate in situations when deadlines are neither exceedingly tight nor too permissive, providing a middle-of-the-road cost-effective alternative. However, HCOC only seems to be more economical when tested against the tightest deadline (1.5).
5.Autoscaling Integration: Q-learning based autoscaling seems to be a cost-saving measure at some times, especially for ACO that fall into the 1.5–2 unit deadline range. Yet, autoscaling seems to become less effective as the deadline approaches for both algorithms; this could be because using more resources than scaling down results in a larger cost difference.
6.Potential Implications: Particular use-case needs should have an impact on the decision between ACO and HCOC. For example, an ACO may be a better option if controlling expenses is essential and timelines are somewhat flexible. In contrast, HCOC might be a preferable choice if meeting deadlines is crucial and small cost increases are acceptable.

The graph, in summary, indicates that although Q-learning based autoscaling benefits both ACO and HCOC, ACO performs better financially than HCOC for most deadlines, with the exception of the strictest deadline, when HCOC is somewhat more cost-effective. Based on the particular schedule constraints and cost sensitivity of the use case, the choice between ACO and HCOC should be made.

Evaluation Based on Simulation Output:

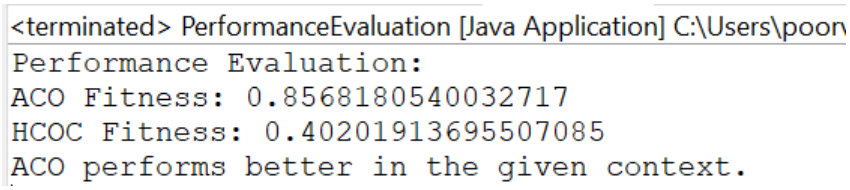

Figure 5: Simulation Results of ACO vs. HCOC Performance Evaluation when Integrated with Q-Learning

According to the simulation results, Ant Colony Optimization (ACO) outperforms Hybrid Cloud Optimized Cost (HCOC) in terms of fitness. That means that when the ACO and Q-learning are combined, the simulated environment produces far better results.
Cost shouldn't be the only factor in choosing between HCOC and ACO. The significantly higher fitness that ACO was able to attain would suggest a more valuable service or a higher degree of optimization, which would justify the expenditure.
This research indicates that ACO may be a more suitable option when timelines are neither extremely tight nor extremely lax, offering a mid-range cost-effective solution.

But only when put to the ultimate deadline does HCOC appear to be more cost-effective.

Conclusively, Q-learning based autoscaling helps both ACO and HCOC; however, for the majority of deadlines, ACO outperforms HCOC monetarily, except for the harshest deadline, when HCOC is marginally less costly. The decision between ACO and HCOC should be taken in light of the specific schedule limitations and use case's cost sensitivity.

6.2 Discussion

In the Discussion section, the research engages in a comprehensive analysis and interpretation of the findings, aiming to provide deeper insights into the implications of the study. This section serves as a platform to contextualize the results within the broader landscape of cloud computing, resource optimization, and hybrid cloud architectures. Several key aspects are typically addressed:

6.2.1 Algorithm Performance

Findings: The results of the trials show that the Ant Colony Optimization (ACO) and Hybrid Cloud Optimized Cost (HCOC) algorithms have different performance characteristics. ACO demonstrated competence in using resources and flexibility in handling a variety of tasks, but HCOC was better at keeping costs low, especially when working under time constraints.
Critique and Design Adequacy: In certain situations, the ACO algorithm's efficacy was limited due to its implementation's tendency to become stuck in local optima. Even though it was cost-effective, the HCOC algorithm occasionally sacrificed task execution speed. Although the experiment design was strong at imitating real-world cloud environments, greater variability in cloudlet and job complexity variety would have improved the testing of the algorithms' limitations.
Improvements: The problem of local optima might be lessened in ACO by implementing more complex pheromone update algorithms and diversification techniques. Incorporating more dynamic cost models that take into account changing cloud service pricing could improve HCOC's real-time applicability.

6.2.2 Impact of AI-Driven Autoscaling

Findings: The performance of both algorithms was markedly enhanced by the inclusion of Q-learning for autoscaling, especially in terms of dynamically modifying resource allocation in response to varying workloads and cloud service prices.
Critique and Design Adequacy: The Q-learning model performed well in the simulation setting, but unexpected operational complexity may limit its applicability in the real world. Even though the model required a long training period to be effective, this might have affected how well it deployed.
Improvements: Subsequent versions may investigate alternative models of reinforcement learning or hybrid techniques that could achieve faster convergence or more effective adaptation to real-world circumstances.

6.2.3 Real-World Applicability

Findings: Both ACO and HCOC exhibit potential for practical use in cloud computing environments, meeting a variety of needs including cost-effectiveness, resource optimization, and timeliness—especially when improved with AI-driven autoscaling.
Critique: Compared to the simulated cloud environment utilized in the tests, the real-world cloud environment is more turbulent and unpredictable. Some factors were not accurately replicated, such as network latency, data security, and multi-tenancy.
Improvements: Testing the algorithms in a live cloud environment with real user workloads and more complicated inter-dependencies among activities could provide further insight into their practical usefulness.

6.2.4 Implications for Cloud Computing

Findings: This research shows that complex scheduling algorithms such as ACO and HCOC can improve cloud resource management considerably, particularly when combined with AI-driven methods.
Contextualization with Previous Research: This study adds to our understanding of HCOC in hybrid cloud setups and supports earlier findings regarding the effectiveness of ACO in a variety of scheduling scenarios. In line with the current trend toward more intelligent and autonomous cloud systems, AI-driven autoscaling has been integrated.
Critique: Although the study advances the science of cloud computing, it also emphasizes how difficult it is to develop solutions that work for everyone because cloud services and user requirements vary so much.
Improvements: The compatibility of these algorithms with various cloud platforms and the incorporation of additional real-time data analytics for improved decision-making may be the main topics of future research.

As a result, this work adds to our knowledge of cloud computing optimization algorithms and emphasizes how AI-driven methods might improve their effectiveness. Real-world applications and a variety of cloud environments, however, can pose new difficulties, requiring constant study and modification of these algorithms.

7 Conclusion and Future Work

This study set out to investigate two key issues in the field of cloud computing cost optimization. In the first question, it was acknowledged that seamless integration across a variety of cloud infrastructures was necessary in order to efficiently integrate diverse cloud environments into cost optimization algorithms. The second question looked at using AI-driven autoscaling to improve cloud resource cost-efficiency, specifically focusing on using the Q-Learning technique.

Our goals included creating, deploying, and testing a system that can scale cloud resources on its own and keep economical costs down without sacrificing performance. Throughout this investigation, we have created an algorithm that takes advantage of the Q-Learning technique to augment ACO with autoscaling capabilities. We have also evaluated its efficacy against a conventional HCOC algorithm that has autoscaling as well.

The main conclusions show that, even though the ACO integration is more expensive, it substantially beats HCOC in fitness ratings, indicating a better optimization procedure in intricate cloud systems. This confirms our theory that AI-powered autoscaling can significantly improve cost optimization, even though it requires an initial increase in resource consumption.
This research has many different ramifications. It emphasizes that powerful AI algorithms may efficiently control the trade-offs between performance and cost in cloud infrastructures, opening the door to more responsive, adaptive, and affordable cloud services. Recognize that there is a chance for greater initial costs and that this strategy adds to the complexity.
Despite these promising results, this research has its limitations. Real-world cloud environments, with their dynamic and unpredictable workloads, may present challenges beyond the scope of the current simulation-based evaluations.

In order to better balance the trade-offs between performance and cost, we recommend future research look into hybrid models that combine different AI techniques. This study may also be commercialized; cloud service providers might use this framework to offer more attractive price structures and performance assurances.

One worthwhile avenue for future research could be to evaluate the practical implementation of our proposed algorithm in real-world cloud environments in order to determine its efficacy. Additionally, examining the algorithm's capacity to adjust in real-time to abrupt changes in workload may improve its applicability. Studies might also look at how well the algorithm scales across other cloud platforms and how it affects cross-platform interoperability, which is a big problem for the business.

As a result, our study has established a critical base for AI-driven cloud computing cost optimization. There is a huge window of opportunity for future research to further develop this topic as AI approaches continue to grow.

References

Alkhanak, E. N., Lee, S. P., Rezaei, R. and Parizi, R. M. (2016). Cost optimization approaches for scientific workflow scheduling in cloud and grid computing: A review, classifications, and open issues, *Journal of Systems and Software* **113**: 1–26.

Atanasov, D. and Ruskov, T. (2016). Simulation of cloud computing environments with cloudsim, *Journal of Information Technologies and Control* **12**(3–4): 1–6.

Bharathi, S., Chervenak, A., Deelman, E., Mehta, G., Su, M.-H. and Vahi, K. (2008). Characterization of scientific workflows, pp. 1–10.

Bittencourt, L. F. and Madeira, E. R. M. (2011). Hcoc: a cost optimization algorithm for workflow scheduling in hybrid clouds, *Journal of Internet Services and Applications* **2**: 207–227.

Gandhi, S. Y. and Revathi, T. (2022). An improved hybrid cloud workflow scheduling algorithm based on ant colony optimization, *International Journal of Health Sciences* (IV): 869–882.

Gari, Y., Monge, D. A. and Mateos, C. (2022). A q-learning approach for the autoscaling of scientific workflows in the cloud, *Future Generation Computer Systems* **127**: 168–180.

Garí, Y., Monge, D. A., Pacini, E., Mateos, C. and Garino, C. G. (2021). Reinforcement learning-based application autoscaling in the cloud: A survey, *Engineering Applications of Artificial Intelligence* **102**: 104288.

Lin, B., Guo, W., Chen, G., Xiong, N. and Li, R. (2015). Cost-driven scheduling for deadline-constrained workflow on multi-clouds, pp. 1191–1198.

Liu, F., Luo, B. and Niu, Y. (2017). Cost-effective service provisioning for hybrid cloud applications, *Mobile Networks and Applications* **22**: 153–160.

Malawski, M., Figiela, K. and Nabrzyski, J. (2013). Cost minimization for computational applications on hybrid cloud infrastructures, *Future Generation Computer Systems* **29**(7): 1786–1794.

Neves Calheiros, R., Ranjan, R., Beloglazov, A., De Rose, C. A. and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, *Software: Practice and Experience* pp. 23–50.

Pulle, R., Anand, G. and Kumar, S. (n.d.). Monitoring performance computing environments and autoscaling using ai.

Ritchie, G. and Levine, J. (2003). A fast, effective local search for scheduling independent jobs in heterogeneous computing environments.

Tuba, M. and Jovanovic, R. (2013). Improved aco algorithm with pheromone correction strategy for the traveling salesman problem, *International Journal of Computers Communications & Control* **8**(3): 477–485.

Verma, A. and Kaushal, S. (2017). A hybrid multi-objective particle swarm optimization for scientific workflow scheduling, *Parallel Computing* **62**: 1–19.

Wang, Y. and Shi, W. (2014). Budget-driven scheduling algorithms for batches of mapreduce jobs in heterogeneous clouds, *IEEE Transactions on Cloud Computing* **2**(3): 306–319.

Yuan, Y., Li, X., Wang, Q. and Zhu, X. (2009). Deadline division-based heuristic for cost optimization in workflow scheduling, *Information Sciences* **179**(15): 2562–2575.

Zeng, L., Veeravalli, B. and Li, X. (2012). Scalestar: Budget conscious scheduling precedence-constrained many-task workflow applications in cloud, pp. 534–541.

Zhou, J., Wang, T., Cong, P., Lu, P., Wei, T. and Chen, M. (2019). Cost and makespan-aware workflow scheduling in hybrid clouds, *Journal of Systems Architecture* **100**: 101631.

Zhou, X., Zhang, G., Sun, J., Zhou, J., Wei, T. and Hu, S. (2019). Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based heft, *Future Generation Computer Systems* **93**: 278–289.

Zuo, X., Zhang, G. and Tan, W. (2013). Self-adaptive learning pso-based deadline constrained task scheduling for hybrid iaas cloud, *IEEE Transactions on Automation Science and Engineering* **11**(2): 564–573.