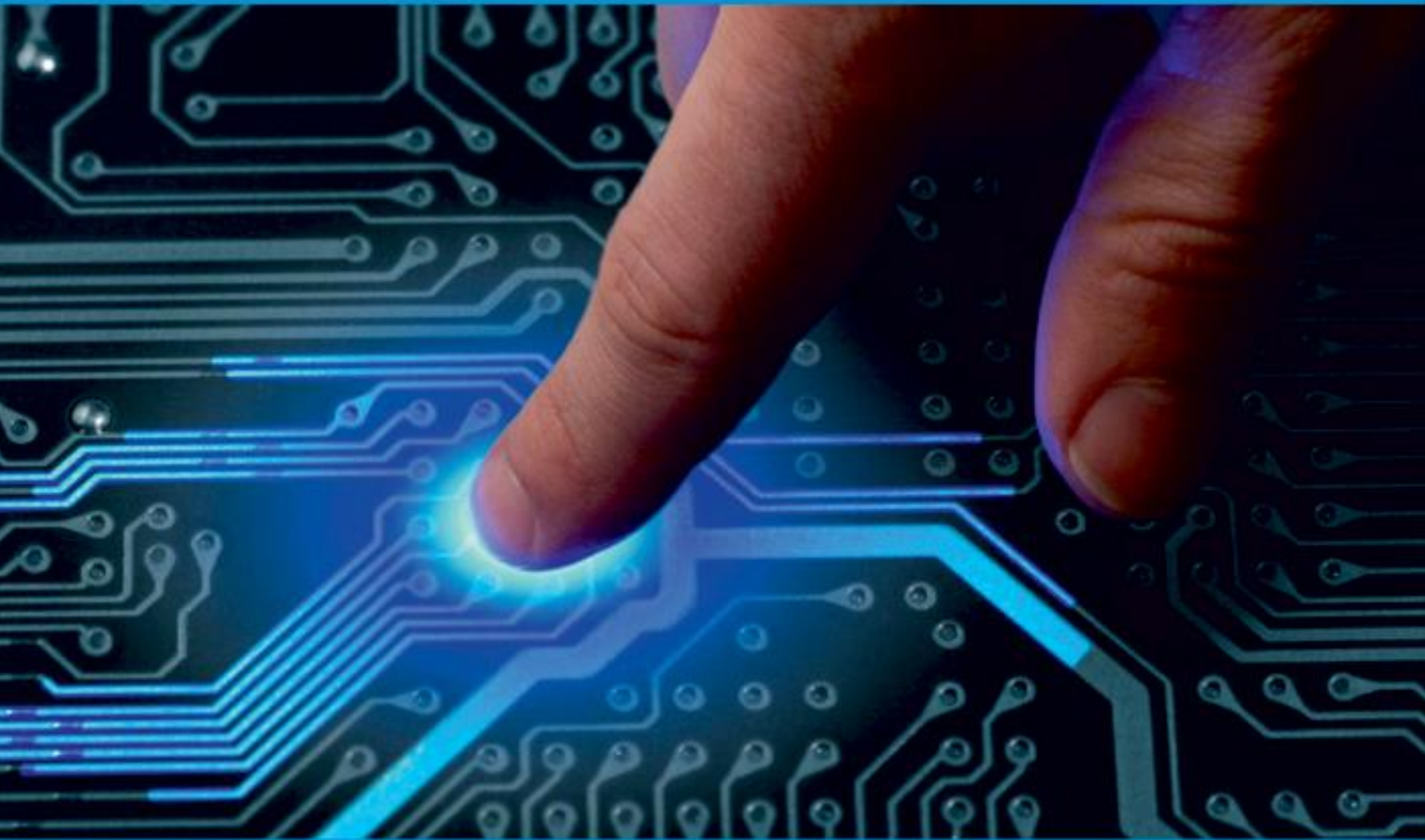
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 12, Issue 4, April 2024

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.379

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204151 |

DCCM : Deadline Constrained Cost Minimization For Cloud Computing Environment

Dr. V. V. S. S. S. Balaram, Akula Rahul, Bollampally Sai Kiran Reddy, Guttula Manognya

Professor, Department of CSE, Anurag University, Hyderabad, India

Department of CSE, Anurag University, Hyderabad, India

Department of CSE, Anurag University, Hyderabad, India

Department of CSE, Anurag University, Hyderabad, India

ABSTRACT: In recent years, there has been a surge in interest in using commercial cloud computing resources for scientific computations. However, efficiently scheduling multiple workflows in cloud computing can be quite challenging due to various constraints and complex resource requirements. Many existing scheduling algorithms rely on search-based approaches, which often come with high computational overhead and lengthy execution times. The paper introduces the Deadline Constrained Cost Minimization (DCCM) algorithm tailored for scheduling cloud resources, offering original solutions to efficiency challenges in cloud computing environments. The key focus of DCCM is to minimize costs while ensuring that tasks meet their specified deadlines. The algorithm achieves this by grouping tasks based on their deadline constraints and data dependencies. Unlike some other approaches, DCCM divides tasks into different priority levels to prioritize meeting user-defined deadlines. Simulation results demonstrate that DCCM outperforms state-of-the-art approaches in terms of success rates. This means that DCCM is more effective at ensuring tasks are completed within their specified deadlines compared to other existing algorithms

KEYWORDS: Cloud computing, deadline constraints, resource scheduling, scientific workflow, optimization.

I. INTRODUCTION

In recent years, there has been a notable surge in interest in utilizing commercial cloud computing resources for scientific computations. However, effectively scheduling multiple workflows in cloud computing poses challenges due to various constraints and complex resource needs Several scheduling algorithms proposed in academic literature utilize search-based methodologies, often resulting in high computational overhead and lengthy execution times.. In response to these challenges, a novel algorithm called Deadline Constrained Cost Minimization (DCCM) is introduced in this paper for efficiently scheduling resources in cloud computing environments. The key innovation of DCCM lies in its focus on meeting user-defined deadlines by categorizing tasks based on their scheduling deadline constraints and data dependencies. Unlike traditional approaches, DCCM subdivides tasks into different priority levels to ensure timely completion. Additionally, as functions are increasingly moving towards the network edge, there's a growing need for cloud scheduling and management solutions. To address this, a dynamic resource scheduling strategy for cloud computing is proposed. This strategy involves grouping identical tasks based on their priorities and constraints to minimize processing costs and reduce queuing overheads. Furthermore, the deadline of the entire workflow is sub-distributed across multiple groups of tasks. The proposed resource scheduling strategy for Virtual Machines (VMs) is dynamic and scalable, aiming to optimize performance. The proposed scheme's effectiveness was evaluated through comprehensive simulations to ensure rigorous assessment.. Results indicate that the proposed algorithm outperforms existing task scheduling algorithms in terms of success rate in meeting deadlines and mean load management.

II. RELATED WORK

Understanding the Challenge

The text delves into the complexities of scheduling scientific workflows, particularly within cloud computing frameworks. It emphasizes the importance of strategies such as heuristics, meta-heuristics, and search-based techniques to optimize resource provisioning and scheduling, aiming to minimize workflow execution time while considering cost and deadlines.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204151 |

Exploring Scheduling Strategies

Particle Swarm Optimization (PSO) emerges as a notable approach, leveraging particles' social behavior to tackle task allocation and resource scheduling problems. Researchers have proposed PSO-based algorithms to address various concerns like cost minimization, execution time reduction, and communication cost optimization, all while accommodating budget and deadline constraints.

Limitations of PSO-Based Algorithms

Despite their effectiveness, PSO-based algorithms encounter challenges in managing heterogeneous resources and overlooking crucial factors like task execution order and resource types, particularly within Infrastructure as a Service (IaaS) setups. Static PSO algorithms may assign tasks randomly to instances, disregarding workflow characteristics.

Introducing IC-PCP

To address these limitations, the IaaS Cloud Partial Critical Paths (IC-PCP) strategy is introduced. IC-PCP schedules tasks on the same VM instance to meet user-defined deadlines without introducing additional overhead to critical paths. However, it overlooks considerations such as VM boot times.

Enhancing IC-PCP with EIPR

To overcome IC-PCP's shortcomings, the Enhanced IC-PCP Algorithm with Replication (EIPR) is proposed. EIPR replicates tasks using idle instances and surplus budgets, aiming to meet deadlines more effectively. While successful in mitigating performance variability, EIPR may lead to increased computational overhead, especially as task execution approaches the billing period.

III. PROPOSED SYSTEM

A. PARTITIONING OF WORKFLOW

In the workflow partitioning phase, the main objective is to prioritize and group tasks based on their dependencies and functionalities. This phase involves several steps:

1. Task Prioritization based on Dependencies:

Tasks are prioritized according to their dependencies using the Upward Rank (UR) metric in Equation 1

(1)

UR considers the Estimated Time to Complete (ETC) for each task and the maximum deadline among its predecessors. This prioritization helps identify the critical path length from the entry task to the exit task in the Directed Acyclic Graph (DAG).

2. Computation of Dependency Degree:

(2)

The degree of dependency for each task is computed using the below formula in equation

$$UR_{t_i} = ETC_{t_i} + \max_{t_i \in predt_j} DT_{ij} + UR_{t_j}$$

$$L_{t_i} = \sum_{t_i \in pred\,t_j} E^i_{t_i} + \sum_{t_i \in pred\,t_j} E^o_{t_i}$$

This equation sums up the internal and external edges, providing insight into the task dependencies.
It helps in understanding the relationships between tasks and their impact on the workflow.

3. Grouping of Tasks based on Similar Functionalities:

Tasks are grouped based on their functionalities and dependencies. Each group may contain single or multiple tasks that
share similar characteristics. This grouping facilitates efficient workflow management by organizing related tasks together.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204151 |

B. TASK SELECTION

In the proposed scheme, the task selection process involves managing tasks within the execution queue while considering their dependencies and deadlines. This phase includes:

1. Execution Queue Management:

Tasks remain in the execution queue until their predecessors are executed. Tasks grouped can be executed simultaneously, streamlining the workflow process.

2. Deadline Distribution:

Before task execution, the deadline distribution ensures that the global deadline of tasks is met by distributing deadlines among different levels. Data transfer time (DTij) is computed using available Virtual Machine (VM) instances . in

$$DT_{ij} = \frac{M_{t_i}}{\beta_{VM}}$$

equation 3

(3)

3. Computation of Available Time:

To ensure the overall deadline is met, sub-level deadlines for each sub-task are implemented. The available time (AT) is computed as the elapsed time between the workflow's deadline and the estimated makespan in equation 4

(4)

Available time for group-level tasks (ATGL) is also computed to meet the overall deadline in equation 5

(5)

4. Sub-Deadline Calculation:

The sub-deadline of each task (D_{tisub}) is determined to ensure timely execution. The sub-deadline calculation considers data transfer time, task duration, and available time for group-level tasks in equation 6

$$D_{t_i sub} = \max_{t_i \in pred\, t_n} (DT_{ij} + ETC + AT_{GL})$$

$$AT_W = D_W - TC_{ms}$$

(6)

C. GROUPING OF TASKS

$$AT_{GL} = \frac{ETC_{t_n}^{GL}}{ETC_{t_n}^{W}} \times AT_W$$

In the grouping of tasks phase, tasks are organized into groups using a horizontal grouping strategy, as described by the authors in [6]. This phase involves:

1. Horizontal Grouping Strategy:

Tasks with similar capabilities and depths are integrated into the same group to enable parallel processing. This strategy ensures that tasks within a group can be executed simultaneously, improving overall workflow efficiency.

2. Homogeneity within Task Groups:

Tasks within the same group share homogeneous characteristics. A single immediate processor is shared among tasks with similar attributes, such as data size, input/output size, computing cost, data distribution pattern, and type.

3. Characteristics of Task Groups:

Each group may consist of either a single task or multiple tasks. Tasks within a group exhibit similar attributes, enabling efficient utilization of resources and parallel processing.

4. Horizontal Grouping Implementation:

Rodriguez and Buyya et. al. [6] described the implementation of horizontal grouping by assigning tasks with the same characteristics and the same color. This visual representation aids in identifying and grouping tasks with similar

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204151 |

attributes effectively.

D. DEADLINE DISTRIBUTION
The deadline distribution strategy ensures that tasks are scheduled effectively within the specified deadline constraints. This phase involves:

***1. Selection of VM Instances*:**
Algorithm 1 selects the cheapest VM instance from the available VM instances. This selection ensures that the estimated makespan for each task is within the user-defined deadline.

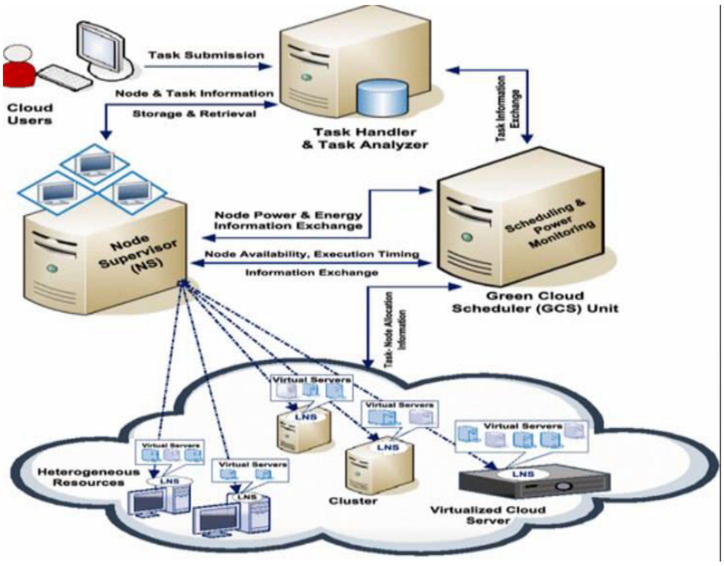
2. Proportional Distribution of Available Time :
(ATW): Available time (ATW) is distributed proportionally across all levels of tasks based on their durations, using ATGL. The allocation guarantees tasks receive sufficient time to meet deadlines, ensuring efficient time management.

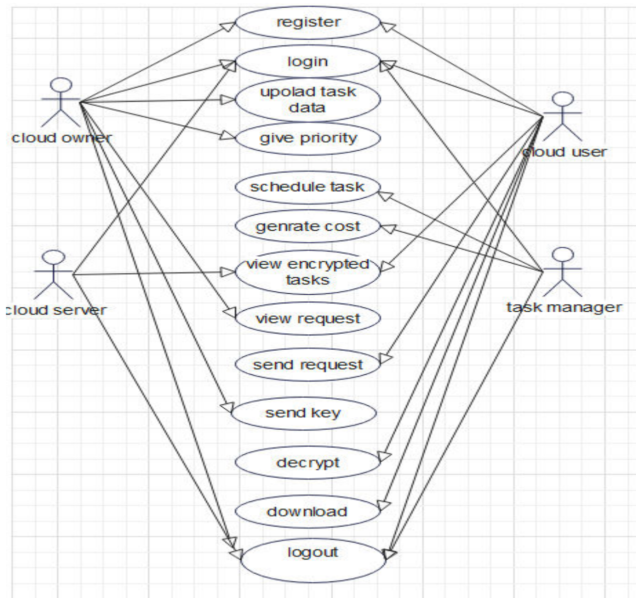
3. Computation of Sub-Deadlines:
The estimated sub-deadline (Dtisub) is computed for each task using the allocated available time and task dependencies. Tasks within a group are assigned the same sub-deadline, ensuring synchronization and efficient workflow execution.

4. Dynamic Scheduling of Task Groups:
Algorithm 2 outlines the dynamic task grouping scheduling employing sub-deadline distribution, offering an original approach to optimize resource allocation. Tasks are sorted based on the descending order of priorities for each level, considering upward rank and degree of dependency. Once task groups are identified, they are placed in a priority queue and sorted based on relevance before being placed in the execution queue.

Figure 2 *Use Case Diagram*

Figure 1 *Architecture Diagram*





| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204151 |

IV. RESULTS

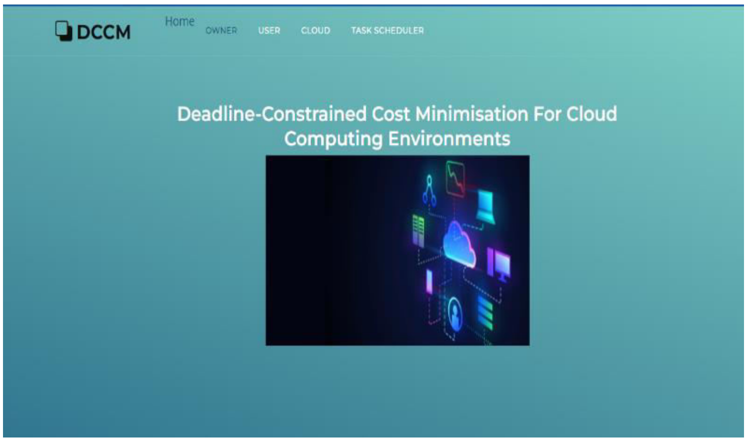


Figure 3 Home Page Of DCCM

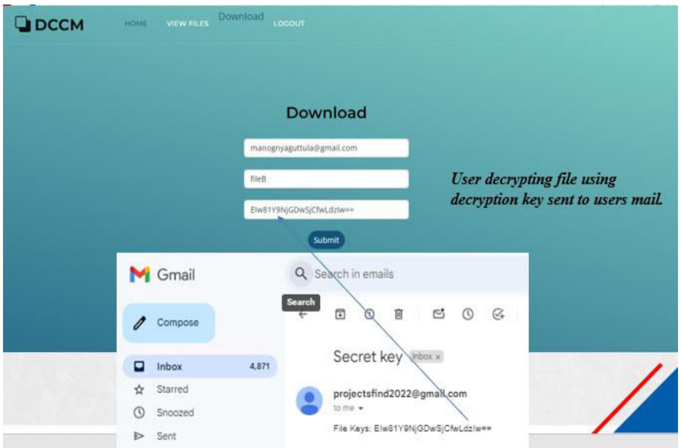


Figure 5 User Entering Key And Downloading Files

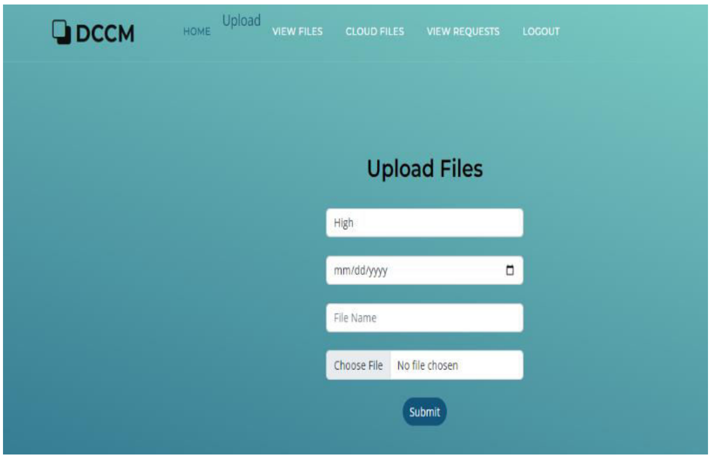


Figure 4 Owner Uploading Files

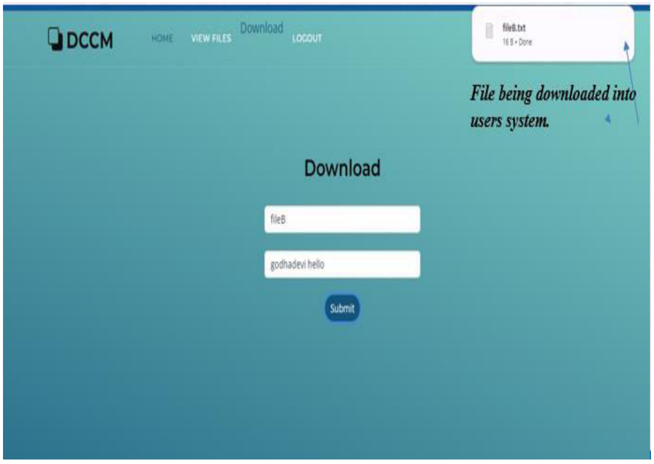


Figure 6 File Downloading Into User System

COMPARISON WITH PREVIOUS METHODS:

The text outlines the development and methodology of the Deadline Constrained Cost Minimization (DCCM) algorithm for efficiently scheduling cloud resources, especially in the context of scientific computations. Let's compare DCCM with the existing methods discussed earlier.

Approach:

Existing methods primarily rely on search-based approaches, which often lead to high computational overhead and lengthy execution times.DCCM introduces a novel algorithm tailored for scheduling cloud resources, emphasizing the minimization of costs while ensuring tasks meet specified deadlines.

Focus:

Existing methods focus on resource provisioning and scheduling, aiming to reduce execution time while considering factors like monetary cost and deadline constraints.DCCM's key focus lies in minimizing costs while prioritizing meeting user-defined deadlines. It achieves this by grouping tasks based on their deadline constraints and data dependencies.

Task Prioritization and Grouping:

Existing methods may prioritize tasks based on various factors, but may not categorize tasks into different priority levels to ensure timely completion.DCCM subdivides tasks into different priority levels based on dependencies and

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204151 |

functionalities, facilitating efficient workflow management.

Execution Strategy:
Existing methods may manage task execution without specifically considering deadline constraints and dependencies.DCCM's task selection process includes managing tasks within the execution queue while considering their dependencies and deadlines, ensuring efficient workflow execution.

Deadline Distribution:
Existing methods may distribute deadlines among tasks without a clear strategy for meeting global deadlines effectively.DCCM implements a deadline distribution strategy to ensure tasks are scheduled effectively within specified deadline constraints, considering factors like available time and task dependencies.

Dynamic Scheduling:
Existing methods may lack dynamic scheduling capabilities, potentially leading to suboptimal resource allocation.DCCM proposes a dynamic task grouping scheduling approach, employing sub-deadline distribution and priority-based sorting to optimize resource allocation effectively.

V. CONCLUSION

This paper presents DCCM, a novel algorithm tailored for scheduling resources in cloud computing setups, with a key emphasis on reducing costs while ensuring tasks meet user-defined deadlines. The effectiveness of DCCM is evaluated against four well-known scientific workflows commonly discussed in academic literature. Results from the experiments demonstrate that DCCM surpasses existing approaches by up to 25% in terms of both planning success rate and the ratio of makespan cost. Looking forward, the authors aim to further enhance their research by developing a dual-objective scheduling model. This model will focus on minimizing both the makespan (the duration of the workflow) and the overall monetary service cost simultaneously. This future research endeavor seeks to provide a more comprehensive solution that balances both efficiency and cost-effectiveness in cloud resource scheduling.

REFERENCES

[1] Y.-K. Kwok, ‘‘Parallel program execution on a heterogeneous PC cluster using task duplication,’’ in Proc. 9th Heterogeneous Comput. Workshop (HCW), 2000, pp. 364–374.
[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, ‘‘Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,’’ Future Generat. Comput. Syst., vol. 25, no. 6, pp. 599–616, 2009.
[3] H. Topcuoglu, S. Hariri, and M.-Y. Wu, ‘‘Performance-effective and lowcomplexity task scheduling for heterogeneous computing,’’ IEEE Trans. Figure 14 User Entering Decryption Key sent via Mail to download the file
Parallel Distrib. Syst., vol. 13, no. 3, pp. 260–274, Mar. 2002.
[4] J. J. Durillo and R. Prodan, ‘‘Multi-objective workflow scheduling in Amazon EC2,’’ Cluster Comput., vol. 17, no. 2, pp. 169–189, Jun. 2014.
[5] M. Malawski, K. Figiela, M. Bubak, E. Deelman, and J. Nabrzyski,‘‘Scheduling multilevel deadline-constrained scientific workflows on clouds based on cost optimization,’’ Sci. Program., vol. 2015, pp. 1–13, Jan. 2015.
[6] M. A. Rodriguez and R. Buyya, ‘‘Budget-driven scheduling of scientific workflows in IaaS clouds with fine-grained billing periods,’’ ACM Trans. Auto. Adapt. Syst., vol. 12, no. 2, pp. 1–22, May 2017.

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING