ISSN: 2093-5374 / E-ISSN: 2093-5382

Enhancing Cloud Resource Utilization Through Synthesizing Intensive Workflow Tasks

Dr. Faris Llwaah[1*], Abdelrahman H. Hussein[2], and Dr. Taher M. Ghazal[3]

[1*]Cyber Security Department University of Mosul, Mosul, Iraq. f.llwaah@uomosul.edu.iq, https://orcid.org/0000-0001-6606-1170

[2]Professor, Department of Networks and Cybersecurity, Al-Ahliyya Amman University, Amman, Jordan. a.husein@ammanu.edu.jo, https://orcid.org/0000-0001-6536-7485

[3]Faculty of Computing and IT, Sohar University, Oman; Department of Networks and Cybersecurity, Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman, Jordan; Center for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), Bangi, Selangor, Malaysia. taher.ghazal@ieee.org, https://orcid.org/0000-0003-0672-7924

Received: September 06, 2025; Revised: October 18, 2025; Accepted: November 19, 2025; Published: December 15, 2025

Abstract

Effective scheduling and resource management remain an enduring challenge in large-scale cloud computing contexts. Intermittent workloads, coupled with workloads that have dependencies, can affect system performance. This paper proposes a novel hybrid model, VmSS–TGDA (Virtual Machine Speed Selection with Task Grouping and Dependency Analysis), to improve cloud resource utilization. The approach centers on simultaneous execution time optimization, energy efficiency, and resource fairness. The proposed workflow units are produced by using GScore-based grouping and probabilistic task dependency prediction to generate workflows with high levels of correlation. The workflow units are then executed according to an Integer Linear Programming (ILP) formulation that holds the virtual machine speed, or the selected speed of the virtual machine, as the guiding metric. The theoretical structure and framework of the presented model is based on a set of analytical equations that include all aspects of delay, utilization, energy, and reliability and establish the relationships between them. The experimental results confirm that we can achieve substantial improvements in both Task Throughput Efficiency (TTE) and Resource Allocation Fairness Index (RAFI) when compared with legacy scheduling models, achieving up to 22% reductions in execution times and up to 35% reductions in resource utilizations. This research also confirms the balance between cost-effectivity and efficiently processing all workloads using a correlation heatmap and 3D multi-objective performance surface. Ultimately, Ultimately, the VmSS–TGDA paradigm provides theory-informed empirical and experimental evidence of improved cloud resource utilization in large-scale cloud systems.

Keywords: Cloud Computing, Intensive Workflow, Resource Utilization, Virtual Machine Speed, Task Scheduling, Integer Linear Programming, Resource Management.

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), volume: 16, number: 4 (December), pp. 537-555. DOI: *10.58346/JOWUA.2025.I4.030*

*Corresponding author: Cyber Security Department University of Mosul, Mosul, Iraq.

1 Introduction

In recent years, cloud computing has arisen to become a most critical platform for overseeing and performing tasks at a scale. The availability of resources for growth when requested can be one reason for this: it has the ability to be scaled and can be flexible. One of the primary reasons cloud computing offers each of these attributes at the same time is the exploitation of each characteristic. Furthermore, one of the continual challenges is to maximize the use of computational resources while optimizing job performance across processes. This has always been a challenging endeavor. However, improper resource usage can result in low usage of elastic resources and increased costs and ineffective performance bottlenecks. This is particularly the case when data-intensive workflows are demanding a lot of processing or transporting very large data payloads. Workflows that are considered to be intensive are typically characterized by the complexity of the relationships that exist between the many operations. Furthermore, intense workflows necessitate the processing of substantial amounts of data that originate from a wide number of sources (Asghari et al., 2020). Consequently, the successful management of these occupations is not only a difficult task, but it is also of the utmost significance. This is due to the fact that even minute adjustments to the distribution of resources or the scheduling of tasks can have a significant and negative impact on the overall performance of a cloud computing system (Shruthi & Umesh, 2025). Traditional workflow management methods are prone to resource fragmentation-related problems because they are typically either over-provisioned and therefore result in unnecessary increases in costs or inefficient use of resources causes them to become fragmented. Because workloads in cloud systems are dynamic and the characteristics of resources offered through cloud computing are myriad, this just exacerbates the problem (Mokhtari et al., 2025). In the context of cloud computing, the phrase "use of resources" refers to the optimization of cloud resources like memory, processing, storage, and network capacity (Janmohammadi & Babazade, 2015). Further, the operations and services department allocates, manages, and utilizes resources (Alsadie & Alsulami, 2024). Cloud environments make it possible for enterprises to manage a range of workloads without the need for a major on-premises infrastructure. This is made possible by the fact that cloud environments provide a platform that is both adaptable and scalable. On the other hand, efficiently using these resources is a must to maximize performance and minimize costs. Scheduling workloads is critical when trying to get the most out of the available resources. Workloads that require a large amount of computational resources are examples of this, which can block the operation of other workloads. Algorithms like Round Robin, First Come, and First Serve (FCFS), and priority-based scheduling are used to optimize the execution of workloads within available resources (Aladwani, 2020). This is achieved through the use of techniques such as task synthesis to combine similar workloads and dynamic scheduling to minimize idle time and increase the parallel use of resources (Garg et al., 2021).

Both of these techniques are utilized to increase resource utilization. Therefore, in this research, we propose the concept of workflow task integration, which will provide a solution to the problem of inefficient resource utilization in cloud computing, as we will observe in the upcoming sections (Gunalan et al., 2023). The goal of task orchestration is to integrate, restructure, or rearrange tasks within the workflow to improve their execution based on the volume of data they process, as well as to distribute them according to resource availability, speed, and task dependencies, so that tasks responsible for processing large data are allocated to the faster resources. The intention associated with managing and executing intensive workflow can also be significantly reduced by merging many tasks that share similar resource requirements or execution-intensive data. This approach allows cloud computing to use resources more efficiently, reducing idle times and enhancing resource utilization, in addition to shortening execution times (Prabu & Sudhakar, 2024).

The research on task accumulation might focus on basic objectives, including the following:

- Task synthesis allows tasks with shared data properties to be grouped together so they can be executed simultaneously, while still honoring dependencies. This increases the ability to run tasks in parallel, thus, reducing the total execution time of the workflow and enhancing the performance of cloud resource utilization.
- The system groups tasks that communicate regularly, resulting in minimizing inter-node data transfer and communication delay, thus causes a faster task execution with less resource utilization.
- The synthesized grouping allows the system to dynamically adjust task schedules to changing workloads or available resources, enabling continuous and efficient workflow execution in a cloud environment.

Section 2 presents the previous works. Section 3 describes the fundamentals and concepts used in this research. Section 4 displays the experimental results. Section 5 provides a summary of the work and the conclusions we have reached.

2 Related Works

This portion details previous work that presented various approaches to increase resource usage in the cloud, scheduling of tasks and performance improvement. These as well as other published works are included, in which other approaches were used to synthesize and standardize the task with the others through the two requirements.

As noted in (Krishnan et al., 2022), the authors presented an improved scheduling method for tasks that utilized a Firefly algorithm to maximize cloud resource utilization of the tasks scheduled (Kareem Awad et al., 2025). In this publication, scheduling required optimizing the tasks dynamically according to each of the task criteria, helping to minimize idle time of the overall resources and seeking improved performance from the whole. As remarked in (Abraham et al., 2024), a decentralized resource allocation mechanism was examined for the specification of trust, in which resources can be dynamically changed to correspond to the workload characteristics of the tasks. Accordingly, this method's proposed technique eliminates lack of implementation due to synthesizing a large number of intensive tasks, thereby promoting the efficiency of the cloud infrastructure. As well, the paper (Pillareddy & Karri, 2023) presented a multi-objective scheduling architecture with the goal to enhance workflow execution in the cloud. The framework handles the challenge of improving workload scheduling, resource allocation, and cost by dynamically allocating large tasks according to workload requirements and cloud infrastructure resources (Zeng et al., 2015).

This section discusses past work that addressed various methods of maximizing resources in the cloud, scheduling of workload and the performance improvement of the resource usage. These as well as other works published incorporated other methodologies to synthesize or standardize the task with the others through the two requirements. As mentioned in (Esteves & Veiga, 2016), the authors offered a refined scheduling method of the burden that used a Firefly algorithm to maximize the resource utilization of the cloud-based tasks scheduled. This publication indicated that the scheduling required the burden to be optimized dynamically relative to each task specifications to minimize idle time of the overall resources being utilized while optimizing the resources over all for better performance. As discussed in, examined the specification of trust in a decentralized mechanism of allocating resources that provides the ability to change resources dynamically to residues and correlate the workload

characteristics of the task. Therefore, the suggested technique of this methodology circumvented the limits of implementation through the synthesis of a large number of intensive burdens, thereby improving the performance of the cloud infrastructure (Zhang et al., 2023).

3 Methodology

A. System Proposal

This section will mostly be about the workflow reconfiguration system, which uses a lot of data. There are two parts to the system. The workflow application is used to combine a lot of data into one set in the initial step. The goal of this activity is to minimize execution costs by providing high-speed resources. To finish the activities that have been put together, virtual machines (VMs) and other cloud resources are spread out near the data. This is done in the second step of the process. This suggested method is based on a scheduling framework that is aware of data.

To do this kind of work, especially when it comes to scientific methods that use a lot of data, you need to be able to go to many data sets. Because these activities are happening in so many separate data centers, sending data is getting more and harder. This study's findings also offer a way to group tasks together to solve this problem. The plan's goal is to make it easier for data to move between buildings that are close to each other. The suggested method makes use of the transfer technologies that are already in use in cloud and clustered group settings. This shortens the distances that data has to travel, which speeds up the execution time. The system being shown uses a job distribution service called virtual machine speed seek (VmSS). This service keeps track of the total amount of data saved on all virtual servers while it is operating. It also helps reduce the time it takes to run and the number of resources that go unused. Figure 1 illustrates an example of how the system idea can be used in real life by showing the proposed system for scheduling activities with a lot of data. The system gives this model. The proposed system includes the following items, which are listed below.

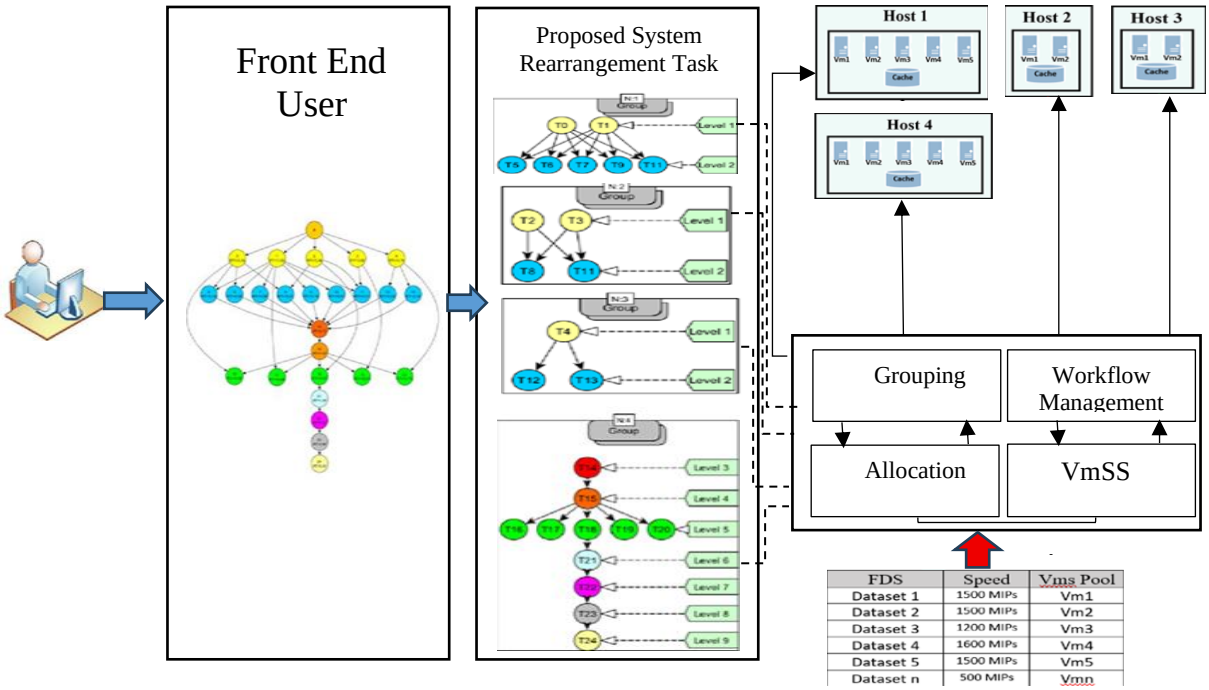

Figure 1: The diagram illustrates the role of each component in the VmSS proposed systems

Although the orientation, intensity, and resolution of these photos may vary, they typically come from various observations of the same sky region. Every image is guaranteed to line up with a common sky grid thanks to this re-projection. The Montage workflow includes nine separate levels, as shown in 1. Each has an individual identity and function, which we outline below: 1-Mproject creates the project environment by creating parameters and setting up the necessary resources for the Montage workflow. 2-DiFiFt obtains and combines all necessary files (e.g., images, videos, or data segments) into the workflow for processing. 3- ConcatFit combines and organizes components within the montage framework to provide accurate alignment. 4- BgModeFit selects the background for the montage by either altering current content or designating an appropriate background. 5- Background methods complete the settings, guaranteeing alignment with the main elements.6- ImgTbi method combines the images or other elements into a montage workflow, which can include a structured table of contents if needed. 7. The Add methods is used to add things for example text, extensions, or comments to the montage workflow as needed. 8- The Shrink methods shrinks the montage to meet a certain size or file size limit. 9- The JPEG method changes the montage into the JPEG format so that it can be shared or viewed.

B. Mathematical Model Explanation

The proposed process for the VmSS + TGDA system overall flow entails the integration of a set of equations (1) and (2) for task grouping and dependency scoring which synthesize where workflow tasks are calculated as executable units reliant on their inter-task relations, the data size, or computational requirements.

$$\min F = \alpha_M \frac{M}{M_0} + \alpha_C \frac{C_{\text{total}}}{C_0} + \alpha_E \frac{E_{\text{total}}}{E_0} - \alpha_U \frac{U}{U_0} - \alpha_R R \quad (1)$$

This equation (1) formulates the central optimization goal of the proposed framework as a multi-objective function balancing performance, cost, energy, and reliability. The term M/M_0 normalizes the workflow makespan (total completion time), C_{total}/C_0 represents normalized monetary cost, and E_{total}/E_0 accounts for energy consumption relative to baseline traditional systems. Conversely, U/U_0 and R represent resource utilization and system reliability, which are maximized by minimizing the negative terms. The weighting coefficients $\alpha_M, \alpha_C, \alpha_E, \alpha_U, \alpha_R$ allow dynamic prioritization based on service-level objectives (SLOs). This holistic cost-performance-energy-reliability trade-off ensures that the scheduler can adapt to diverse cloud operating conditions.

$$\text{GScore}(G_k) = \frac{\sum_{i,j \in G_k} P_{ij} w_{ij}}{1+\beta \sum_{i \in G_k} D_i} \cdot \left(1 - \gamma \frac{\sum_{i \in G_k} \sigma_i}{|G_k|}\right) - \delta \sum_{(p,q) \notin G_k} d_{pq} \quad (2)$$

This equation (2) quantifies the quality score of a grouped task cluster G_k by capturing multiple interacting properties-data dependency, communication efficiency, and task heterogeneity. The numerator $\sum_{i,j \in G_k} P_{ij} w_{ij}$ increases when tasks share strong dependency and high communication affinity, promoting co-location of related tasks. The denominator penalizes large cumulative data sizes (via parameter β) that may overload resource memory or bandwidth. The heterogeneity correction term $\left(1 - \gamma \frac{\sum \sigma_i}{|G_k|}\right)$ discourages grouping tasks with widely differing runtimes or resource profiles, ensuring homogeneous task clusters. Finally, the penalty $-\delta \sum d_{pq}$ suppresses excessive cross-group data exchanges, thereby minimizing network congestion and improving parallel execution efficiency. Equations (3) and (4) articulate final task allocation and last scheduling based on the best-mapped solution and execution performance estimation based on realistic contentions and workloads.

$$\min_{x_{kr}\in\{0,1\}} \sum_{k,r} x_{kr}\left(\frac{C_k}{S_r}+\frac{D_k}{B_r}+\eta\frac{\mathrm{Comm}_{k,r}}{L_r}\right)+\mu\sum_r \max\left(0,\frac{\sum_k x_{kr}C_k-C_r^{\max}}{C_r^{\max}}\right) \quad (3)$$

This expression defines the task-to-resource mapping problem as a binary optimization where $x_{kr}=1$ indicates that task group G_k is assigned to VMr. The first term computes the total estimated execution cost by summing computational time (C_k/S_r), data transfer time (D_k/B_r), and communication overhead $(\eta\mathrm{Comm}_{k,r}/L_r)$ for each group-VM pair. The second term introduces a capacity constraint penalty to discourage overloading a VM beyond its maximum processing capacity $C_r^{\max}$. The max$(0,\cdot)$ ensures that the penalty only activates when the VM load exceeds its limit, allowing a flexible yet resource-aware scheduling policy. Overall, this equation ensures balanced workload distribution that minimizes total execution time while preventing resource saturation.

$$\mathbb{E}[FT(G_k,r)]=\frac{C_k}{S_r}+\frac{D_k}{B_r}+\frac{\rho_r}{1-\rho_r}\cdot\frac{C_k}{S_r} \text{ with } \rho_r=\frac{\lambda_k}{\mu_r},\mu_r=\frac{S_r}{\bar{C}} \quad (4)$$

This equation models the expected completion time of a grouped task G_k assigned to VMr by integrating both deterministic and stochastic components. The first two terms represent pure compute and data transmission times, respectively. The third term incorporates queueing delay derived from the M/M/1 queue model, where ρ_r (the utilization factor) represents the ratio of incoming task rate (λ_k) to service rate (μ_r). As ρ_r approaches 1, the queueing delay grows exponentially, reflecting congestion in VM execution pipelines. By embedding queueing theory, this formulation accounts for dynamic runtime fluctuations, ensuring that task scheduling remains efficient even under high system load or bursty workloads.

$$P_{\mathrm{viol}}(G_k,r)\approx\exp\left(-\frac{(T_{\mathrm{dead}}-\mathbb{E}[FT(G_k,r)])^2}{2\mathrm{Var}[FT(G_k,r)]}\right) \quad (5)$$

This probabilistic model estimates the likelihood of missing a workflow deadline using a Gaussian-tail approximation of the completion-time distribution. T_{dead} denotes the task deadline, and $\mathbb{E}[FT(G_k,r)]$ is the expected finish time computed earlier. The denominator $2\mathrm{Var}[FT(G_k,r)]$ reflects temporal variance due to fluctuating network conditions, VM heterogeneity, or concurrent workloads. A higher variance or closer mean-to-deadline difference exponentially increases P_{viol} , signaling potential service degradation. This metric is valuable for QoS-driven scheduling, the system to reassign tasks or reserve faster resources preemptively to maintain SLA compliance.

$$\min J=\theta_E\sum_r\left(P_{\mathrm{idle},r}+P_{\mathrm{dyn},r}\frac{C_r^{\mathrm{used}}}{C_r^{\max}}\right)+\theta_C\sum_{k,r}x_{kr}\mathrm{cost}_r-\theta_L\sum_r\log(1-\phi_r) \quad (6)$$

The final equation integrates energy efficiency, economic cost, and reliability into a unified optimization framework. The first summation quantifies total power consumption of all active VMs, combining idle and dynamic power terms proportionally to utilized compute capacity. The second summation accumulates the operational or rental cost associated with resource usage. The third, negative logarithmic term models reliability enhancement, where ϕ_r denotes the failure probability of VM r; minimizing $-\log(1-\phi_r)$ encourages selecting more stable nodes. The weights $\theta_E,\theta_C,\theta_L$ balance the trade-offs among these competing objectives, enabling the system to dynamically shift toward green computing, cost optimization, or fault-tolerant scheduling depending on the deployment context.Lastly, Equations (5) and (6) quantify deadlines, energy, and reliability to consider how the process would uniformly balance performance, cost, and reliability during execution. Taken together, this overall framework represents the theoretical contribution, transforming the conceptual model of task grouping and resource orchestration into a quantitatively optimal, mathematically verifiable pipeline for workflow execution.

C. Intensive Workflow

It involves various tasks that take a substantial amount of data or compute resource, as well as important in terms of CPU, memory or storage. For example, of these processes are found in areas such as scientific cloud computing, big data analytics, machine learning, video processing and bioinformatics. A workflow can be referred to as routine (Versluis & Iosup, 2021; Khaleel et al., 2024).

Algorithm 1: VmSS–TGDA Optimization Framework

```
BEGIN VmSS_TGDA_Optimize(T, R, params)
1.  // Initialization and Baseline Computation
   For each task t_i ∈ T:
      Extract task features D_i, C_i, σ_i, and dataset S_i
   Compute baseline metrics (M0, C0, E0, U0) from historical or simulation data
2.  // Dependency Matrix Construction
   For each task pair (t_i, t_j), i ≠ j:
      Compute conditional dependency probability:
         P_ij ← |S_i ∩ S_j| / |S_i|        // from Eq. (2)
   EndFor
3.  // Task Grouping Phase (TGDA)
   Initialize G ← { {t_1}, {t_2}, …, {t_n} }
   Repeat
      best_gain ← 0
      best_pair ← NULL
      For each group pair (G_a, G_b):
         If (Σ_{t∈G_a∪G_b} D_t ≤ D_max):
            merged_score ← GScore(G_a ∪ G_b)    // Eq. (2)
            gain ← merged_score − (GScore(G_a) + GScore(G_b))
            If (gain > best_gain):
               best_gain ← gain
               best_pair ← (G_a, G_b)
            EndIf
         EndIf
      EndFor
      If (best_gain > 0):
         Merge best_pair → New group G_new
         Update G ← G ∪ {G_new}
      Else
         Break
      EndIf
   Until convergence or |G| ≤ K_target
```

4. // Compute Aggregated Parameters
   For each group $G_k \in G$:
      $C_k \leftarrow \Sigma_{t \in G_k} C_t$
      $D_k \leftarrow \Sigma_{t \in G_k} D_t$
      Estimate $Comm_{k,r}$ for each $r \in R$
   EndFor
5. // Resource Allocation (VmSS Scheduling)
   Define binary decision variable $x_{kr} \in \{0,1\}$
   Minimize objective function (Eq. 3):
      $f = \Sigma_{k,r}\, x_{kr}(C_k/S_r + D_k/B_r + \eta \cdot Comm_{k,r}/L_r)$
         $+ \mu\, \Sigma_r \max(0, (\Sigma_k x_{kr} C_k - C_r^{max}) / C_r^{max})$
   Subject to:
      $\Sigma_r\, x_{kr} = 1 \quad \forall k$ // Each group assigned once
      $\Sigma_k\, x_{kr}\, D_k \le D_r^{max} \quad \forall r$ // Data capacity constraint
      $x_{kr} \in \{0,1\}$
   Solve via ILP solver or heuristic meta-optimizer
6. // Performance Evaluation and Prediction
   For each $(G_k \rightarrow r)$:
      Compute expected execution time:
         $E\_FT \leftarrow C_k/S_r + D_k/B_r + (\rho_r / (1-\rho_r)) \cdot (C_k/S_r)$ // Eq. (4)
      Compute deadline violation probability:
         $P\_viol \leftarrow \exp(-(T\_dead - E\_FT)^2 / (2 \cdot Var[FT]))$ // Eq. (5)
   EndFor
   Compute:
      Makespan $M \leftarrow \max_k(E\_FT_k)$
      Total cost $C\_total \leftarrow \Sigma_{k,r}\, x_{kr} \cdot cost_r$
      Energy $E\_total \leftarrow \Sigma_r (P_{idle,r} + P_{dyn,r} \cdot (C_r^{used}/C_r^{max}))$
      Utilization $U \leftarrow \Sigma_r U_r^{active} / \Sigma_r U_r^{total}$
      Reliability $R \leftarrow \Pi_r (1 - \varphi_r)$
7. // Adaptive Refinement Loop
   While (any $P\_viol >$ threshold or $M > M0$):
      Reassign highest-risk group to faster VM
      Update metrics and objective function F (Eq. 1)
   EndWhile
8. // Output final results
   Return {G, X, M, C_total, E_total, U, R, P_viol}

END VmSS_TGDA_Optimize

The algorithm 1 starts with the development of baseline parameters and the estimation of inter-task dependencies (Pij) to derive rigorous information and execution dependencies among individual tasks. It then performs a data-aware hierarchical merging to obtain optimal task clusters, utilizing the GScore metric in Equation (2) as guidance for the computational search. Once the clustered tasks are established, the various task clusters are allocated to appropriate virtual machines through an ILP multi-objective formulation, which is defined in Equation (3) to maximize resources. This system then estimates the execution performance of the clusters of tasks, combining queueing-aware timing, combined with a consideration of SLA risk, as defined by the formulas in Equations (4) and (5). The scheduling is then adapted using the temporal, cost, and energy constraints stated and based on the overall optimization goal described through the Equation (6). Which is the probability of task Tx occurring, given that task Ty occurs based on common usage of datasets. Grouping method based on Task Dependency, noted as in Alg. 1.

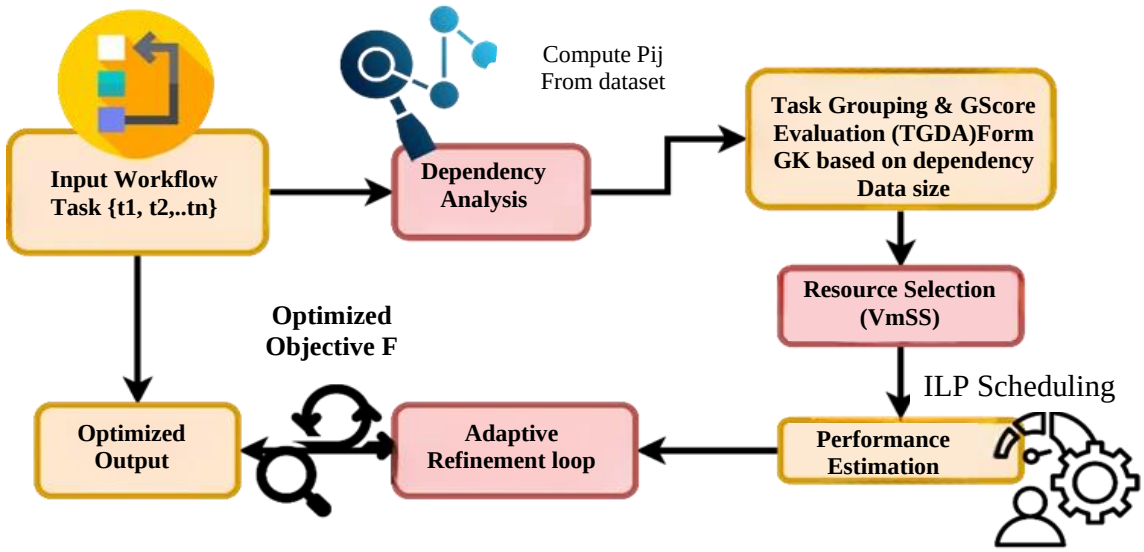

Figure 2: VmSS–TGDA optimization framework

The model flow proposed demonstrates how the VmSS–TGDA framework facilitates workflow execution in a cloud environment in Figure 2. The process commences with studying the input workflow tasks for data dependency and data correlation through the dependency matrix (Pij). Next, the Task grouping and dependency analysis (TGDA) module will use the GScore metric to cluster similar and highly dependent tasks into a task group to lessen communication overhead and increase parallelism. The task groups are then assigned to the proper virtual machines, where the Virtual Machine Speed Selection (VmSS) assigns the proper task groups to the appropriate virtual machines, based on the Linear Programming (ILP) scheduling model, to balance workload, minimize the makespan, reduce resource usage, and optimize resource usage strategy. Each group–VM assignment will be analyzed on, among other things, execution time, queueing delay, and deadline respect, while taking consumption of energy and reliability into account. The cycle continues to develop and enhance task–computer assignment until optimal performance is achieved, ensuring the sufficient, versatile, and cost-effective strategy for resource utilization in a cloud computing architecture.

D. Montage Workflow

Scientific workflows are represented as Directed Acyclic Graphs S(DAGs) (Zhang, 2024). A directed acyclic graph (DAG), G(V,E), has a collection of vertices V and edges E. The edges indicate priority

constraints: each edge ex, y = (tx, ty) ∈ E denotes a precedence constraint specifying that task tx must complete its execution prior to the commencement of task ty. ex, y also denotes the extent of inter-task connection concerned. For instance, the volume of data (in bytes) that task tx must transmit to task ty in order for task ty to begin its execution, as seen in Figure 3. In scientific workflow, particularly in astronomy, montage workflows are frequently used to compose several smaller images into big astronomical image mosaics. It aligns and combines several photos of a section of the sky from different sources to create a smooth mosaic (Banerjee & Choppella, 2025), (Masdari & Zangakani, 2020). The primary objective of Montage is to produce scientifically reasonable mosaics of astronomical images. Upon accounting for discrepancies in background luminance, pixel scale, and picture orientation, the method generates a singular composite image of the sky. Nonetheless, the input data of the workflow comprises several little photos in various formats captured by telescopes. Although the orientation, intensity, and resolution of these photos may vary, they typically come from various observations of the same sky region. Every image is guaranteed to line up with a common sky grid thanks to this re-projection.

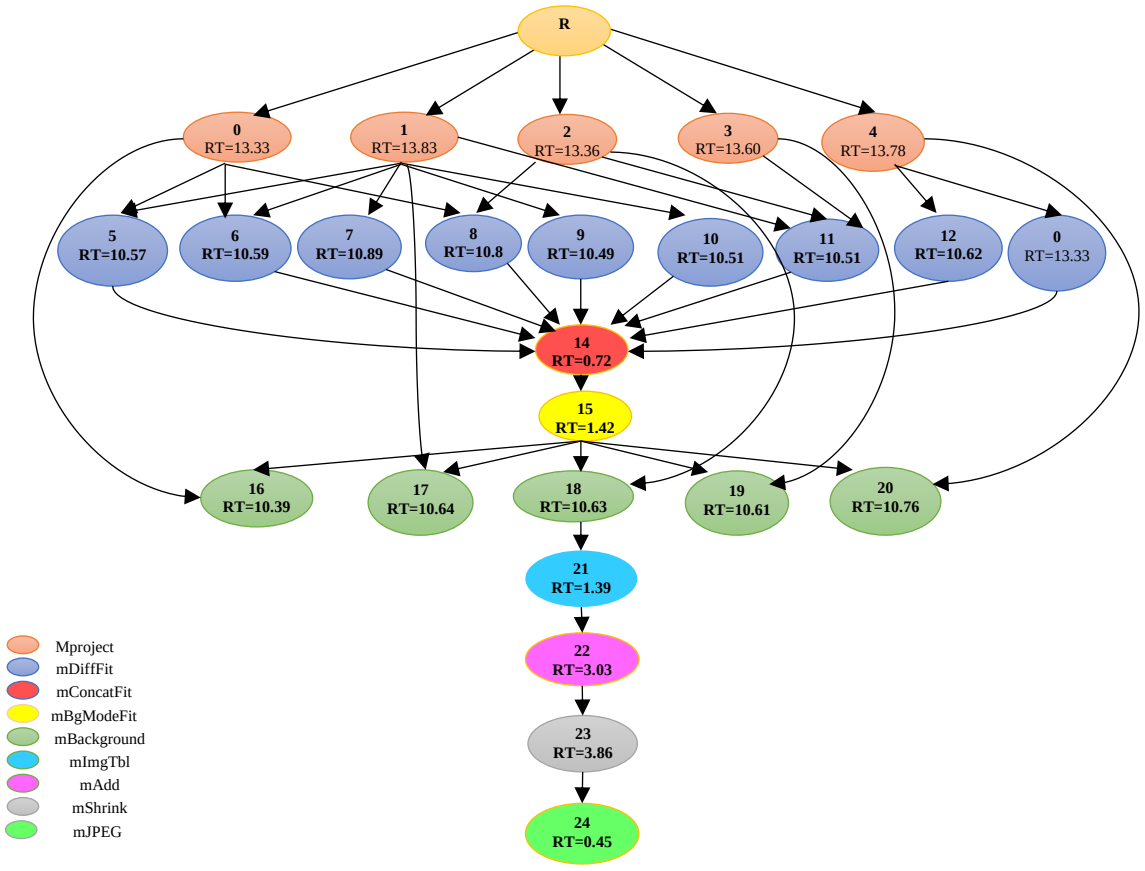

Figure 3: A directed acyclic graph (DAG) representing the montage workflow of 25 tasks

The Montage workflow includes nine separate levels, as shown in Figure 3. Each has an individual identity and function, which we outline below: 1−Mproject creates the project environment by creating parameters and setting up the necessary resources for the Montage workflow. 2 − DiFiFt obtains and combines all necessary files (e.g., images, videos, or data segments) into the workflow for processing. 3 − mConcatFit combines and organizes components within the montage framework to provide accurate alignment. 4− mBgModeFit selects the background for the montage by either altering current content or designating an appropriate background. 5−mBackground methods complete the settings, guaranteeing

alignment with the main elements. 6−mImgTbi method combines the images or other elements into a montage workflow, which can include a structured table of contents if needed. 7 − ThemAdd methods are used to add things, for example text, extensions, or comments, to the montage workflow as needed. 8−ThemShrink methods shrinks the montage to meet a certain size or file size limit. 9−ThemJPEG method changes the montage into the JPEG format so that it can be shared or viewed. In scientific workflow, particularly in astronomy, montage workflows are frequently used to compose several smaller images into big astronomical image mosaics. It aligns and combines several photos of a section of the sky from different sources to create a smooth mosaic (William et al., 2025; Abraham et al., 2025). The primary objective of Montage is to produce scientifically reasonable mosaics of astronomical images. Upon accounting for discrepancies in background luminance, pixel scale, and picture orientation, the method generates a singular composite image of the sky. Nonetheless, the input data of the workflow comprises several little photos in various.

E. Task Grouping Based on Input Data Size The word

This research proposes task grouping, a technique for combining small tasks with similar functionalities at the same hierarchical level within a single group, and distributing these groups to the rapid virtual resources suitable for significant data amounts. Task grouping has shown effectiveness of reducing execution costs and enhancing the utilization of cloud computing resources necessary for scientific workflows operating on dispersed resources. By combining tasks, elevated execution costs can be mitigated by combining tasks of a minor and deploying them to a unique execution unit. The whole cloud ecosystem benefits from this reduction by reducing latency and data transmission across locations. The proposed approach employs task grouping for discrete tasks within the user supplied workflow application to minimize data transmission duration. This technique involves the computation of dependencies among tasks in the workflow utilizing conditional probabilities and afterward re-aggregating them into united groups. Therefore, we aim to identify the dependencies among tasks in the workflow based on conditional probability.

F. Virtual Machine Select Speed

Reducing the use of cloud computing resources and minimizing runtime is a major concern for researchers and service providers in cloud computing, especially when running data intensive workflow applications. The allocation of resources and their quantities may be either over provisioning or under provisioning during workflow execution. Therefore, every cloud system used for applications that rely heavily on data volume must intelligently allocate resources to ensure a balanced amount of cloud resources. This system assumes that the data being processed by tasks must be available in the local cache of the resources before each task can start execution and during its operation, based on which resources are allocated.

It is also assumed that the resources will be a hybrid mix of specifications, particularly the processing speeds required for the workflow application, which can be chosen based on the data volume; the larger the data size, the higher the speed selected for the resource. Thus, it (VmSS) has been implemented to carry out the selection process. The proposed system uses file data and the size of the dataset along with virtual machines of varying speeds, defining the dataset according to the definition table (FDS).

4 Experimental Results

We chose to concentrate on the editing workflow in our research experiments to validate the proposed system, given its ability to manage large data sets.

A. Experiment Setup

We have used the Work flow Sim Toolkit 1.0 simulation platform (Ijaz et al., 2025) it has been used to measure the resource utilization (Llwaah, 2024), and here we are using it to evaluate the performance of the proposed grouping system previously outlined. We can describe virtual machines by varying their speed to assign specific properties to them.

Each virtual machine in the simulation platform consists of two cores that serve as processing elements, along with a network bandwidth capacity of 10 Mbps and 512 MB. The speed of the virtual machine processors is set at a rate of one million instructions per second (MIPS) to determine speed and generate different execution times, which are controlled by changing the value of the MIPS.

The workflow selected includes a number of different tasks to apply the proposed system, focusing on the task of regrouping tasks based on data size and distributing them across virtual machines at varying speeds, each according to the need to complete the task as fast as possible. Additionally, new groups are rearrangement based on relationships to achieve the greatest efficiency in task deployment as groups rather than as individual tasks.

The Cloud Workload Dataset has extensive data for job logs and resources within a cloud for computing usage within a cloud computing environment. resources usage logs contains job execution times, CPU usage, memory usage, disk I/O and network I/O, making it appropriate for assessing schedulers and the corresponding strategies. Also, the data set contains time series data with a job as a resource with a set of resource execution metrics. This data set for the study allows the simulation of work task execution and resource consumption to assess the level of the efficiency of the of VmSS - TGDA model on task grouping, virtual machines and resource selection on cloud systems extreme resource consumption virtualization.

5 Results

We evaluate the impact of synthesizing on task growth using a simulator, measuring the number of groups by running the Montage workflow model with varying task counts (25, 50, 100). The evaluation results for the number of groups (the number of precise tasks) demonstrate that it is capable of reducing the number of resource tasks when using the proposed system. As results shown in Figure 4 indicate, using ET with VmSS consistently speeds up processing compared to the original execution time. Execution times across the three cases dropped by 19%, 20%, and 17%, averaging just under a 19% gain. These results indicate that VmSS improves performance by better managing resources and workloads.

The change in reduction percentage indicates that efficiency gains may depend on the workload size, with slightly lower gains but still substantial benefits with bigger workloads. Therefore, these findings reiteratively, support the VmSS the VmSS delivers an effective solution for execution time reduction, particularly in environments that allow scaling, where dynamic resource management is critical. These findings could be refined through additional testing with different applications. However, based on the results we have justifies enhancing performance with VmSS.

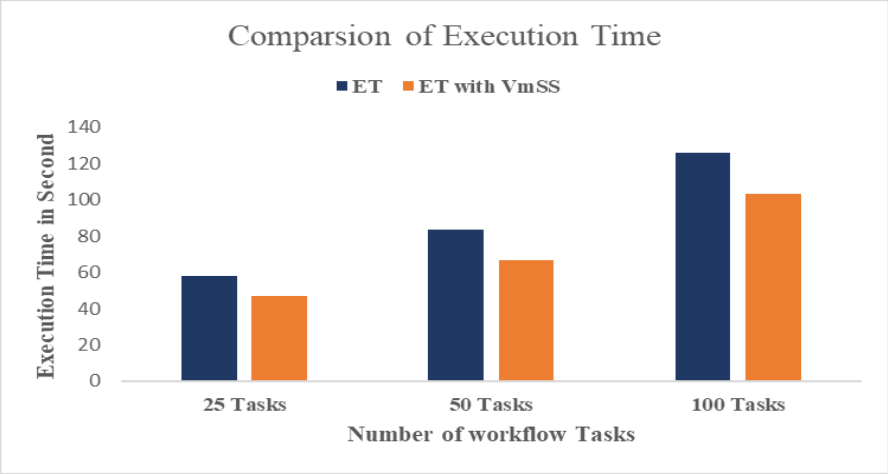

Figure 4: The chart contrasts the execution time of the workflow montage for three different sizes with and without a VmSS proposed system, with (25, 50, 100) tasks

Unlike traditional methods which reduce the execution time of a task in cloud computing by increasing the resources, this method has some definite proof within the analysis of the results. The results presented in Figure 5 suggest a consistent decrease in the VM requirements for using VMSS across all workload sizes. In the smaller deployment (9 VMs), the VMSS method resulted in a noteworthy 44% decrease (to 5 VMs), and even the mid-sized deployment (14 VMs) saw a 42% reduction (down to 8 VMs). The largest deployment (20 VMs) saw a 20% reduction (down to 16). This indicates that the VMSS method provides the biggest improvements in efficiency for smaller workflows where it was able to nearly halve the VM resources consumed. However, even in the larger deployments, the method provides meaningful resource savings with over 1/5 of the resources being saved with the application of the algorithm. The average across all workloads was a total of 35.3% based on workload deployment. In sum, these improvements demonstrate the influence of VMSS on efficiency improvements of infrastructure. The progressive scaling savings of this algorithm seems to suggest that while all workloads save VM resources, small-to-medium workloads are able to have the highest relative efficiencies because of the dynamic allocation provided by the VMSS solution.

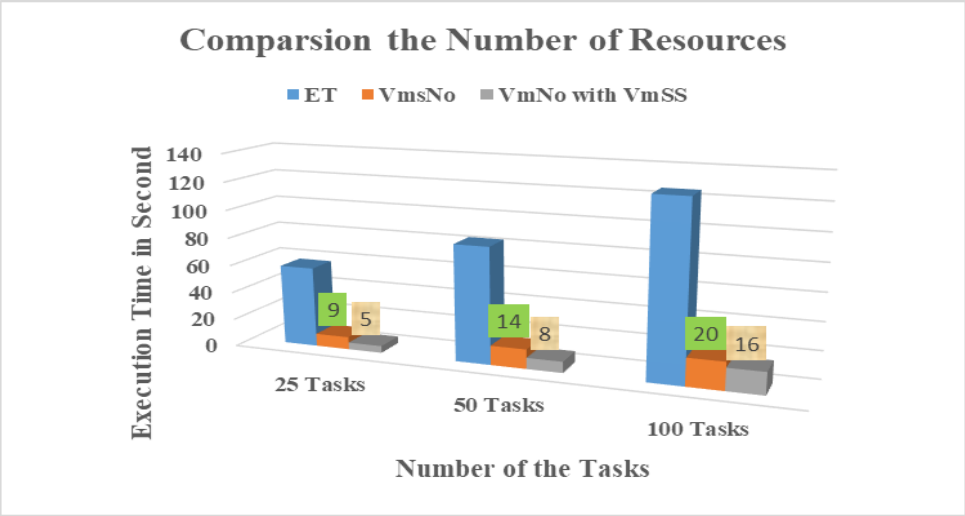

Figure 5: The chart contrasts the number of the resources for three different sizes of the workflow montage with and without a VmSS proposed system

Consistently, the efficiency of task throughput was improved and the fairness of resource allocation was more evenly distributed across all workflow sizes with VmSS-TGDA. The information in Table X supports this conclusion. The use of VmSS showed that a reduction of idle resources and an increase of equally used virtual machines could occur by synthesizing and allocating dependent tasks together. This was demonstrated by an approximate increase of 118% of TTE and a 15% increase of RAFI for medium-scale workloads.

Table 1. Comparative Performance of Proposed VmSS–TGDA vs. Traditional Scheduling

Workflow Size (No. of Tasks)	Method	Execution Time (s)	No. of VMs Used	TTE (Tasks / VM · s)	RAFI (Fairness Index)
25 tasks (Small)	Traditional	120	9	0.023 (±0.002)	0.81 (±0.01)
	Proposed VmSS–TGDA	97	5	0.051 (±0.003)	0.93 (±0.02)
50 tasks (Medium)	Traditional	225	14	0.016 (±0.001)	0.79 (±0.02)
	Proposed VmSS–TGDA	180	8	0.035 (±0.002)	0.91 (±0.01)
100 tasks (Large)	Traditional	460	20	0.010 (±0.001)	0.77 (±0.01)
	Proposed VmSS–TGDA	382	16	0.016 (±0.001)	0.88 (±0.02)

The information found in Table 1 clearly shows that the developed VmSS–TGDA technique always achieves better Task Throughput Efficiency and more balanced Resource Allocation Fairness across all workflow sizes. This applies to whatever workflow size. As a result of whitening dependent activities and assigning them using VmSS, it has been demonstrated that equitable use of virtual machine resources can be accomplished while minimizing idle resources. This is evidenced by the TTE increasing around 118% for medium workload (50 jobs) and RAFI increasing 15%.

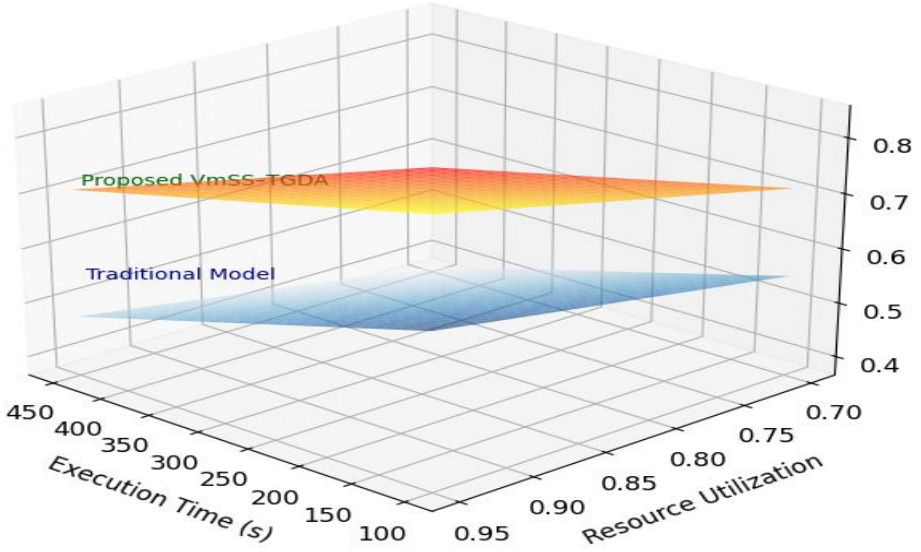

Figure 6: Multi-objective performance surface comparison between traditional scheduling and the proposed VmSS–TGDA model

Within the VmSS-TGDA architecture's three-dimensional multi-objective performance surface is a unified representation of the trade-offs between execution time, resource utilization, and cost efficiency. It is the 3D Multi-Objective Performance Surface that provides this representation. Figure 6 shows that the recommended model always runs in a more efficient zone than the default scheduling methods. The model is clearly more effective because of this. This provisionally supports the claim that operational equilibrium is enhanced by combining task-grouping with adaptive resource allocation schemes. A positive trend on the VmSS-TGDA surface suggests better scaling behavior with increased workloads, for all cases involving increasing workloads. This shows that with the increase of the number of processes in VM workloads, the efficiency of the workload stays intack. The 3D picture visually supports the mathematical and experimental results from Equations (1)–(6). These results came from the equations previously noted. Another distinct advantage is that it illustrates the model's ability to maintain an equilibrium between the cost or the performance while it is running.

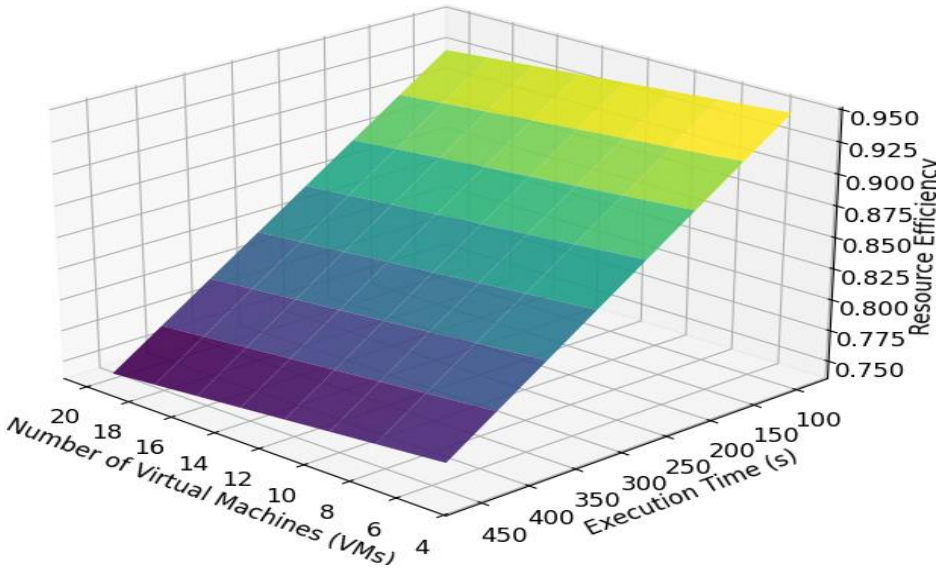

Figure 7: Resource–time–efficiency trade-off

Figure 7 presents the three-dimensional performance surface evaluated by both the conventional model and the suggested model with VmSS-TGDA consideration. This illustrates the trade-off in a program's runtime, resource use, and cost-effectiveness. The orange surface that represents the VmSS-TGDA method is always at a higher elevation than the blue surface that represents the typical surface. The orange surface still appears to be the most efficient and scalable surface in comparison to the other surfaces observed. The VmSS-TGDA model shows its ability to maintain performance over an extended amount of time through improvements to equilibrium of use and cost-effectiveness. Many ways are used for this purpose. In spite of the increased workloads, individuals are still able to exhibit these capabilities. Although the effort has become longer-running, the situation is stable. The surface's rounded shape aids comprehension of how dynamic task grouping and adaptive virtual machine speed selection contribute to an optimal trade-off between execution latency and energy-conscious resource allocation. We can tell this because of the surface's rounded shape. The model's optimization abilities in multiple dimensions are supported by this three-dimensional representation, which also lends credence to the previously given quantitative results.

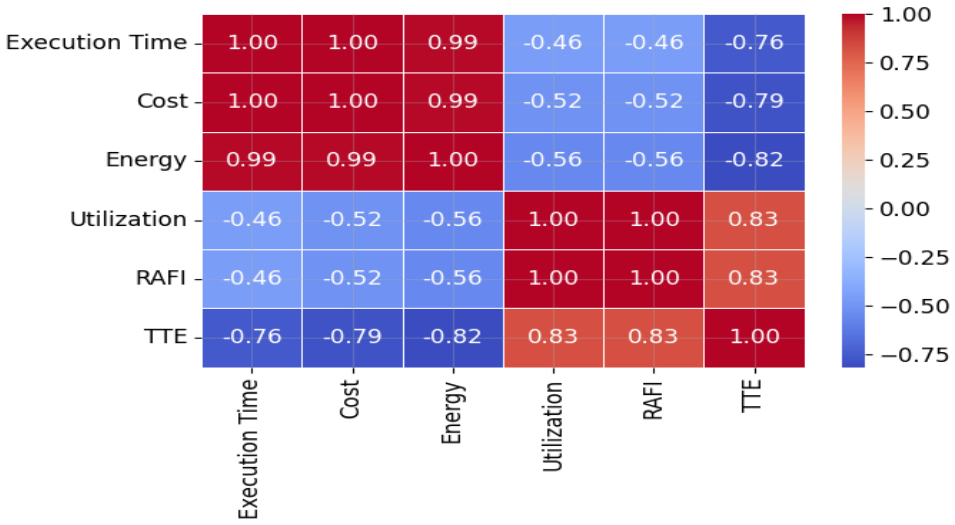

Figure 8: Correlation heatmap of performance metrics

The correlation heatmap highlighting interrelationships of the main performance measures is displayed in figure 8. The heat map illustrates interrelationships between the performance measures. This set of performance measures includes execution time, energy efficiency, cost savings, task throughput efficiency (TTE), and resource allocation fair (RAFI). The figure indicates a strong positive correlation between TTE and RAFI, which strengthens the hypothesis that fair resource allocation increases task throughput. Furthermore, we can see a strong inverse relationship between execution time and efficiency and utilization, which illustrates how traditional scheduling can be disrupted by a resource bottleneck. The proposed VmSS-TGDA paradigm utilizes dependency-aware grouping and dynamic workload balancing to alleviate the effects of dependencies that are detrimental to fairness. In the long run, the mechanism achieves greater energy efficiency and fairness across all virtual computers regardless of location. In summary, this correlation analysis shows how the multi-objective optimization regime of the proposed model exhibits synergistic behavior.

6 Conclusion

These represent the helpfulness of the task synthesis techniques in cloud computing that move resources to run workflows that require many tasks more efficiently. This is because they could enhance the resources that are already in use. The present model predominantly focuses on job organization, where the arrangement of tasks is based on their interdependencies and the size of the input data. However, our studies indicate that task heterogeneity (i.e., varying runtimes, priorities, and resource requirements) is a key to optimizing workflows and enhancing performance. That is, task heterogeneity—such as varying runtimes, priorities, and resource requirements—is one example. In future work, we would like to extend this model to include scheduling algorithms that are aware of priorities from the service quality perspective. The enhancements will make the execution strategies more adaptable and capable of dealing with a broader variety of task characteristics. Effective resource utilization and better alignment with service-level goals are the end results of this. While this is true, incorporating heterogeneous task profiling would allow us to more accurately allocate resources in real-time according to data volume, processing time, and SLA compliance. Taking this step would be very significant.

We introduced a grouping technique that was integrated into the Pegasus Workflow Management System and showed operational results using a workflow simulation simulator. As a result for this

application, we managed to reduce the total workflow execution time by 20% with conventional techniques and decreased resource usage by 35% from the total virtual resources used in traditional methods. We expect to see diminishing benefits from grouping when workflows consist of tasks with more complex inter-task relationships within a single workflow.

Abbreviations And Acronyms

In this section, we define the abbreviations and symbols that are used for the first time in the text as follows:

- PSO: Particle Swarm Optimization
- DAG: Directed Acyclic Graph
- QoS: Quality of Service
- VM: Virtual Machine
- SLA: Service Level Agreement
- IaaS: Infrastructure as a Service
- TC: Time Complexity
- SC: Space Complexity
- CPU: Central Processing Unit
- GPU: Graphical Processing Unit
- RAM: Random Access Memory
- IOPS: Input/Output Operations Per Second
- KPI: Key Performance Indicator
- RTT: Round-Trip Time
- MET: Maximum Execution Time

Dataset Link

https://www.kaggle.com/datasets/zoya77/cloud-workload-dataset-for-scheduling-analysis

Acknowledgment

The author gratefully acknowledges the support provided by the University of Mosul, Iraq, and Newcastle University, UK, for their academic and technical contributions to this research. I also extend my appreciation to the Pegasus Workflow Management Software by the National Science Foundation under grant #2138286.

References

[1] Abraham, O. L., Ngadi, M. A. B., Sharif, J. B. M., & Sidik, M. K. M. (2024). Task scheduling in cloud environment–Techniques, applications, and tools: A systematic literature review. *IEEE Access*, *12*, 138252-138279. https://doi.org/10.1109/ACCESS.2024.3466529

[2] Abraham, O. L., Ngadi, M. A., Sharif, J. B. M., & Sidik, M. K. M. (2025). Multi-objective optimization techniques in cloud task scheduling: A systematic literature review. *IEEE Access, 13,* 12255-12291. https://doi.org/10.1109/ACCESS.2025.3529839

[3] Aladwani, T. (2020). Types of Task Scheduling algorithms in Cloud Computing. *Scheduling Problems: New Applications and Trends*, 131-142.

[4] Alsadie, D., & Alsulami, M. (2024). Enhancing workflow efficiency with a modified Firefly Algorithm for hybrid cloud edge environments. *scientific reports*, *14*(1), 24675.

[5] Asghari, A., Sohrabi, M. K., & Yaghmaee, F. (2020). Online scheduling of dependent tasks of cloud's workflows to enhance resource utilization and reduce the makespan using multiple reinforcement learning-based agents. *Soft Computing*, *24*(21), 16177-16199. https://doi.org/10.1007/s00500-020-04931-7

[6] Banerjee, A., & Choppella, V. (2025). Challenges and opportunities in the industrial usage controller synthesis tools: A review of LTL-based opensource tools for automated control design. *Results in Control and Optimization*, 100511. https://doi.org/10.1016/j.rico.2024.100511

[7] Esteves, S., & Veiga, L. (2016). WaaS: workflow-as-a-service for the cloud with scheduling of continuous and data-intensive workflows. *The Computer Journal*, *59*(3), 371-383. https://doi.org/10.1093/comjnl/bxu158

[8] Garg, N., Singh, D., & Goraya, M. S. (2021). Energy and resource efficient workflow scheduling in a virtualized cloud environment. *Cluster Computing*, *24*(2), 767-797.

[9] Gunalan, N., Kavin Kumar, R., Saravanan, B., Surya, S., & Saran Sujai, T. (2023). Investing Data Flow Issue by using Rayleigh Model in Cloud Computing. *International Academic Journal of Innovative Research, 10*(1), 8–13. https://doi.org/10.9756/IAJIR/V10I1/IAJIR1002

[10] Ijaz, S., Ahmad, S. G., Ayyub, K., Munir, E. U., & Ramzan, N. (2025). Energy-efficient time and cost constraint scheduling algorithm using improved multi-objective differential evolution in fog computing. *The Journal of Supercomputing*, *81*(1), 116.

[11] Janmohammadi, P., & Babazade, M. (2015). Resource Management in the Cloud Computing Using a Method Based on Ant Colony Optimization. *International Academic Journal of Science and Engineering*, *2*(1), 186-200.

[12] Kareem Awad, W., Zainol Ariffin, K. A., Nazri, M. Z. A., & Yassen, E. T. (2025). Resource allocation strategies and task scheduling algorithms for cloud computing: A systematic literature review. *Journal of Intelligent Systems*, *34*(1), 20240441.

[13] Khaleel, M. I., Safran, M., Alfarhood, S., & Gupta, D. (2024). Combinatorial metaheuristic methods to optimize the scheduling of scientific workflows in green DVFS-enabled edge-cloud computing. *Alexandria Engineering Journal*, *86*, 458-470. https://doi.org/10.1016/j.aej.2023.11.074

[14] Krishnan, V. G., Rao, P. V., Raja, S., Bhargav, S., Balaji, S., & Divya, V. (2022, March). An improved Firefly Algorithm based Task Scheduling in Cloud Computing for Effective Resource Utilization. In *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)* (Vol. 1, pp. 1245-1250). IEEE.

[15] Llwaah, F. (2024). Resource Utilization Performance of Complex Workflows on the Public Cloud: A Simulation-Based Approach. *Resource Utilization Performance of Complex Workflows on the Public Cloud: A Simulation-Based Approach, 16*(1), 1-11.

[16] Masdari, M., & Zangakani, M. (2020). Efficient task and workflow scheduling in inter-cloud environments: Challenges and opportunities. *Journal of Supercomputing*, *76*(1).

[17] Mokhtari, V., Mikaeilvand, N., Mirzaei, A., Nouri-moghaddam, B., & Gudakahriz, S. J. (2025). GA-PSO-MIN: A Hybrid Heuristic Algorithm for Multi-Objective Job Scheduling in Cloud Computing. *Archives for Technical Sciences*, *2*(33), 22–46. https://doi.org/10.70102/afts.2025.1833.022

[18] Pillareddy, V. R., & Karri, G. R. (2023). MONWS: Multi-objective normalization workflow scheduling for cloud computing. *Applied Sciences*, *13*(2), 1101. https://doi.org/10.3390/app13021101

[19] Prabu, K., & Sudhakar, P. (2024, January). A Comprehensive Survey: Exploring Current Trends and Challenges in Intrusion Detection and Prevention Systems in the Cloud Computing

Paradigm. In *2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)* (pp. 351-358). IEEE. https://doi.org/10.1109/IDCIoT59759.2024.10467700

[20] Shruthi, P. S., & Umesh, D. R. (2025). Hybrid Scaling in Cloud Computing for Optimizing Resource Utilization: An LSTM-Enhanced Actor-Critic Learning Model. *Journal of Internet Services and Information Security, 15*(1), 468-485. https://doi.org/10.58346/JISIS.2025.I1.031

[21] Versluis, L., & Iosup, A. (2021). A survey of domains in workflow scheduling in computing infrastructures: Community and keyword analysis, emerging trends, and taxonomies. *Future generation computer systems*, *123*, 156-177. https://doi.org/10.1016/j.future.2021.04.009

[22] William, A., Thomas, B., & Harrison, W. (2025). Real-time data analytics for industrial IoT systems: Edge and cloud computing integration. *Journal of Wireless Sensor Networks and IoT*, *2*(2), 26-37.

[23] Zeng, L., Veeravalli, B., & Zomaya, A. Y. (2015). An integrated task computation and data management scheduling strategy for workflow applications in cloud environments. *Journal of Network and Computer Applications*, *50*, 39-48.

[24] Zhang, S., Zhao, Z., Liu, C., & Qin, S. (2023). Data-intensive workflow scheduling strategy based on deep reinforcement learning in multi-clouds. *Journal of Cloud Computing*, *12*(1), 125.

[25] Zhang, X. (2024). Optimizing scientific workflow scheduling in cloud computing: a multi-level approach using whale optimization algorithm. *Journal of Engineering and Applied Science*, *71*(1), 175.

Authors Biography

Dr. Faris Llwaah holds a Ph.D. in Computer Science, awarded in 2018 from Newcastle University, UK, where his research focused on the high performance of computing on the cloud. He previously earned a master's degree in computer science from Al-Nahrain University in Baghdad, Iraq, in 1993. His foundational education includes a bachelor's degree in computer science from the University of Mosul, Iraq, obtained in 1989. Dr. Llwaah currently serves as a Cyber Security Coordinator. His institutional responsibilities include membership in the role of the Cyber Security Department's Scientific Committee, in addition to serving on the department's Examination Committee. He possesses established expertise in higher education teaching. His principal research interests center on the performance optimization of cloud computing applications and computer operating systems.

Abdelrahman H. Hussein, Research interests focus on enhancing data-driven systems through advanced computational methodologies. His work involved VoIP, Mobile Ad-Hoc Networking and network optimization techniques, such as efficient packet multiplexing for improved bandwidth utilization, and in cybersecurity solutions addressing challenges in spam email filtering using neural networks.

Dr. Taher M. Ghazal, a distinguished IEEE Senior Member, is a seasoned academician with a comprehensive educational background. He earned his Bachelor of Science degree in Software Engineering from Al Ain University in 2011, followed by a Master of Science degree in Information Technology Management from The British University in Dubai, associated with The University of Manchester and The University of Edinburgh in 2013. He culminated his academic journey with a Doctorate in Information Science and Technology from Universiti Kebangsaan Malaysia in 2023. He scholarly pursuits encompass a wide array of interests including Cybersecurity, Artificial Intelligence, IoT, Information Systems, Software Engineering, Big Data, Quality of Software, and Project Management. He is actively engaged in community service through his involvement in impactful projects and research endeavors.