IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 9 July 2025, accepted 4 September 2025, date of publication 1 October 2025, date of current version 16 October 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3616511

RESEARCH ARTICLE

Predictive State-Aware Deep Reinforcement Learning With Hyper-Heuristic for Resolving Conflicting Objectives in Scientific Workflow Scheduling

AWADH SALEM BAJAHER[1], NOR ASILAH WATI ABDUL HAMID[1,2], (Senior Member, IEEE), IDAWATY AHMAD[1], AND ZURINA MOHD HANAPI[1], (Senior Member, IEEE)

[1]Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang 43400, Malaysia
[2]Institute for Mathematical Research, Universiti Putra Malaysia, Serdang 43400, Malaysia

Corresponding authors: Awadh Salem Bajaher (bajaher92@gmail.com) and Nor Asilah Wati Abdul Hamid (asila@upm.edu.my)

This work was supported by Universiti Putra Malaysia (UPM) through Geran Putra under Grant GP/2020/9693400.

ABSTRACT Scientific workflows in cloud environments are highly complex and dynamic, necessitating intelligent and flexible scheduling solutions to handle important factors, including resource heterogeneity, budget constraints, deadlines, and ever-changing workload requirements. In contrast to traditional scheduling methods, a model with predictive ability and adaptability is required to manage these complex challenges effectively. Hence, this work intends to address the challenges associated with scheduling scientific workflows, balancing conflicting objectives, such as cost and deadline, managing intricate inter-task workflow dependencies, and dynamic resource availability. To accomplish this goal, this work presents a predictive state-aware deep reinforcement learning with a hyper-heuristic for deadline and budget-aware workflow scheduling optimization. Initially, the proposed system applies a Multihead Graph Attention Network (MGAN) to describe complex interactions between workflow tasks and cloud resources in order to predict the state for modeling an accurate environment. Moreover, the design of hyper-heuristic generation with Deep Q-Network (DQN) improves deadline and budget-aware decision-making in the uncertain workflow scheduling environment. Experiments show that the proposed method outperforms state-of-the-art approaches in terms of deadline and cost-effectiveness, providing a reliable and intelligent strategy for scheduling scientific workflows.

INDEX TERMS Cloud computing, scientific workflow tasks, workflow scheduling, deadline, budget, hyper-heuristic generation, reinforcement learning, q-learning, predictive state, multi-head graph attention network (MGAN).

I. INTRODUCTION

Cloud computing provides dynamically scalable services over the Internet by utilizing virtual servers. Its significant features, including elasticity and flexibility, make it a leading technology for storage and computation services [1]. This trend motivates cloud platforms to become a dominant environment for executing complex scientific applications. Large-scale scientific workflows, such as astronomy, bioinformatics, and climate modeling, are executed in the cloud. Within each scientific workflow, large sets of interdependent tasks are connected by dependency constraints, necessitating successor tasks to begin only after the completion of their predecessors' tasks. A directed acyclic graph (DAG) represents complex workflows in which nodes indicate individual tasks and directed edges indicate dependencies among the tasks. These workflows have different resource requirements, including storage and CPU, and several constraints that need

The associate editor coordinating the review of this manuscript and approving it for publication was Nitin Gupta.

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

to be considered during task execution [2]. The effective execution of scientific workflows is based on utilizing available resources efficiently, enabling researchers to develop effective approaches, known as workflow scheduling, for allocating workflow tasks to available computing resources [3]. Efficient resource provisioning improves resource utilization, whereas workflow scheduling algorithms allocate the resources needed by tasks during execution [4]. Workflow scheduling must ensure successful workflow execution without affecting Quality of Service (QoS) parameters, including deadline adherence, energy efficiency, and cost efficiency [5]. To this end, QoS is assessed based on specific criteria such as computational cost, reliability, and processing time, which may directly impact user satisfaction. Thus, researchers have focused on developing near-optimal solutions for scheduling problems due to the NP-completeness. Notably, balancing large data volumes, complex processing, and multiple criteria in scientific workflows is challenging. Therefore, research works focus on optimizing scientific workflows by balancing multiple conflicting QoS factors. Balancing conflicting objectives like meeting deadlines and minimizing costs is difficult since faster processing requires more powerful resources, and lower-cost resources may increase task completion times. Hence, efficient strategies are essential to provide tradeoffs between deadlines and budget constraints [6]. Deep learning (DL) has become vital for scientific workflow scheduling because of its ability to handle complex and nonlinear patterns and the dynamic and multi-objective nature. Among DL models, Deep Reinforcement Learning (DRL) is highly suitable for adapting to dynamic environments, including varying execution times, resource availability, and workflow priorities. Through reward-driven optimization, DRL efficiently overcomes the challenges of balancing multiple conflicting objectives within scientific workflows. DRL dynamically adapts policies depending on the current state to learn optimal scheduling approaches by considering variations in resource availability and workflow constraints. Hyper-heuristics (HH) are crucial for scientific workflow scheduling since they offer a flexible, adaptive approach to addressing the complexities and variability in large-scale scientific workflows. HH utilizes evolutionary algorithms and machine learning models to generate or select heuristics for a class of problems instead of directly solving a specific problem [7]. Unlike conventional methods that function within the solution space in which HH algorithms search within the space of heuristics, allowing the system to create new heuristics personalized for similar types of problems [8]. DRL-based HH generation solves this limitation by enabling the system to learn and improve scheduling approaches through continuous experience [9]. DRL enables the system to dynamically generate and fine-tune heuristics, efficiently handling different workflows with multiple conflicting objectives [10]. This study overcomes the limitations in traditional workflow scheduling, including the inability to handle dynamic workloads and heterogeneous resources, limited predictive machine-learning models for optimizing scheduling, and insufficient multi-objective optimization approaches to balance deadlines and budgets. Existing hyper-heuristic algorithms lack adaptability in dynamic cloud environments, with limited heuristic diversity and poor handling of uncertainty. The absence of fault-tolerant and multi-objective hyper-heuristic frameworks exacerbates these challenges, often leading to missed deadlines and budget overruns. Addressing these issues requires integrating predictive models, adaptive strategies, and multi-objective optimization frameworks. Hence, this work designs a novel workflow scheduling architecture using Deep Reinforcement Learning and Hyper-heuristics models to enable prediction-aware decision-making.

The primary contributions of this work are presented as follows.

- This work introduces a predictive state modeling by a Multihead Graph Attention Network (MGAN) to capture intricate relationships among cloud resources and task dependencies, enabling accurate prediction of dynamic states in the workflow scheduling environment.
- Also, the adoption of Graph Neural Networks (GNNs) for task prioritization effectively manages inter-task dependencies. It improves the overall efficiency of scheduling in complex scientific workflows and leverages time-efficient workflow scheduling.
- By designing a heuristic generation within the hyper-heuristics framework, the proposed approach dynamically fine-tunes scheduling strategies in DRL by incorporating execution priority, cost, and deadline factors to optimize performance in varying conditions through a reinforcement agent, accomplished by the combination of Genetic Programming (GP) and metaheuristics.
- In DRL, the agent generates deadline and budget-aware actions for workflow scheduling based on the fitness score obtained from the hyperheuristic generation model and Q-value-based adaptive reward calculation by the DQN model, enhancing the system's ability to generate optimal scheduling decisions across uncertainty and resource variability in the cloud.

II. LITERATURE REVIEW

This section reviews conventional workflow scheduling research involving budget, deadline-aware scheduling, and hyper-heuristic-assisted models.

A. COST AND DEADLINE-AWARE WORKFLOW SCHEDULING APPROACHES

Workflow scheduling strategies consider cost and deadline objectives to balance meeting deadlines and execution costs for enhancing task execution in the cloud. Several workflow scheduling methods for business and scientific applications ensure timely workflow completion while minimizing resource costs. In order to maintain cost and deadline tradeoffs, several existing strategies applied

heuristic, meta-heuristic, and machine learning approaches, highlighting developments such as efficient scheduling and multi-objective optimization for dynamic cloud resource management. A two-stage preference-driven Multi-Objective Evolutionary Algorithm (tsp-MOEA) is presented in the work [11], addressing the constraint of scheduling workflows with multiple roles in the cloud. The enhanced ACO algorithm with Deadline and Budget constraints, called DB-ACO, is designed in work [12] for cloud workflow scheduling. The DB-ACO investigates and optimizes the workflow scheduling while considering execution costs and budget by dynamically balancing the priorities in the cloud environment. The work in [13] applies mini-batch sampling for dynamic scheduling with improved generalization; it lacks adaptive multi-objective decision-making and dynamic outcome analysis during execution, even when paired with hyper-heuristic algorithms. The work in [14] presents metaheuristic methods, such as L-Ant Colony Optimization (L-ACO), which provides a deadline-aware cost-minimized scheduling solution based on the pheromone trail and probabilistic upward rank, and ProLiS provides a deadline-aware cost-minimized scheduling solution based on the probabilistic upward rank with two-step list scheduling. An adaptive approach presented in [15] effectively schedules workflow ensembles in IaaS clouds, maximizing resource utilization while adhering to stringent timelines and cost limitations. Considering priorities, it employs three methods: FastestScheduling, SchedulingWithMinCost, and GapRate analysis, enabling flexible decision-making and optimal scheduling that guarantees cost-effective execution, prompt execution, and improved resource management for various workflows. The work in [16] presented a cloud-based approach for scheduling scientific workflows: Budget-Deadline Constrained Workflow Scheduling (BDCWS). In order to mitigate execution costs, the BDCWS selects the cost-effective virtual machines and builds an optimistic available budget based on the slowest virtual machine, thereby balancing cost and time. To maximize time management and execution costs, it also presents a balancing factor and selection technique. BDCWS improves efficiency and cost-effectiveness for large-scale scientific applications by dynamically assigning resources depending on workflow requirements, ensuring workflows are completed within budget and deadline constraints. The work in [17] presented a Self-Detection and Comprehensive Learning-Based Budget and Resource Optimization (BRO) solution for workflow scheduling in the cloud. To enhance the global search exploration, it integrates the re-spawn mechanism in the BRO and then detects the local optimum based on the count of similar soldiers. Also, to increase the search diversity, the solution is enhanced with an elite enhancement strategy, and thus, it maintains the optimal balance between exploration and exploitation with minimal makespan and budget. A multi-objective scheduling approach is presented in [18] to optimize the cloud's resource utilization and energy consumption. To accomplish this, security is initially ensured by the design of Universal Unique Identification- Blake (UUID-Blake). It predicts the resource availability using the Manhattan Distance-Partition around algorithm (MD-PAM) and Anova-Recurrent Neural Network (A-RNN) algorithms. Finally, the Linear Scaling-Crow Search Optimization (LS-CSO) algorithm schedules the workflows using the appropriate cloud resources. The work in [19] presented a hybrid multi-objective optimization method that combines the Seagull Optimization Algorithm (SOA) and Grasshopper Optimization method (GOA), namely HGSOA-GOA. Balancing exploitation and exploration improves convergence with the aid of chaotic maps. Furthermore, the workflow scheduling is based on a knee-point approach that selects the best answer from the Pareto front in dynamic multi-cloud systems. This results in increased scheduling efficiency and cost-effectiveness. A cost-driven smart scheduling method was developed in [20] for IoT workflow applications to manage deadline constraints. In order to minimize scheduling costs and satisfy deadlines, this approach designs the Ant Colony Optimization (ACO) algorithm with cost constraints. The algorithm applies a deadline distribution approach to determine task sub-deadlines further and allow for dynamic modifications for unscheduled workflow tasks. Also, an adaptive ACO-based task prioritization method improves the execution of scheduling sequences, speeds up convergence, and reduces redundancy in the search space. Thus, the cost-driven scheduling approach assigns workflow tasks to virtual machines with minimal execution costs and time, greatly improving cost-effectiveness and timely completion. The work in [21] designed a cost-aware workflow scheduling in the cloud that combines Simulated Annealing (SA) and Deep Reinforcement Learning (DRL). In particular, the Deep Q-Network (DQN) is integrated with SA, namely SA-DQN, in which the SA algorithm assists the neural network learning by optimally selecting the subtasks to execute on cloud servers. Moreover, the application of DRL determines the best scheduling policies for dynamic workloads. To address the uncertainty in cloud environments, the work in [22] presented cost-effective and deadline-constrained scheduling for multi-workflow by improving the task allocation with uncertainty modeling. F-MWSA is a composition of fuzzy task scheduling and fuzzy deadline distribution in the cloud. Cost-effective task scheduling in the cloud resources drastically mitigates scheduling costs by fuzzy task scheduling. In contrast, the fuzzy deadline distribution phase assigns a sub-deadline for each workflow task to achieve the task execution before the deadline due to the dynamic environment. To efficiently schedule the workflows to resources in the data center, the work in [23] developed the Hybrid HEFT-PSO-GA, namely HEPGA. By minimizing makespan, HEPGA optimizes task scheduling in cloud environments with the assistance of PSO and GA, optimizing resource utilization through parallel processing. However, research on cost- and deadline-aware workflow

scheduling focuses on handling task dependencies in different workflow applications and multi-objective optimization in complex workflows. In addition, predicting the resource availability, cost, and task execution time poses challenges to traditional heuristic-based algorithms, limiting optimal decision-making among dynamic cloud resources.

B. HYPER-HEURISTIC ALGORITHM-ASSISTED WORKFLOW SCHEDULING APPROACHES

To improve diverse or different workflow scheduling performance, the existing workflow scheduling research works have increasingly adopted hyper-heuristic algorithms to either select or generate heuristics in cloud environments. With the design of a genetic programming approach with a multi-tree representation, the work in [24] presented a modified discrete event-driven dynamic workflow scheduling simulator. By optimizing scheduling decisions using a multi-tree structure, it adaptively handles the dynamic workflows in multi-cloud systems. To improve generalization in Genetic Programming Hyper-Heuristics (GPHH), the work in [13] presented mini-batch sampling methodologies for dynamic workflow scheduling. To examine the impact of six sampling methods, such as three mini-batch sampling, two hybrid strategies, and one rotation strategy, the GPHH is applied with a sampling strategy that enhances hyper-heuristic performance and flexibility in a dynamic environment. Mini-batch sampling enables the scheduling algorithm to effectively manage changes in workflow tasks and resource availability, thereby improving resource management in dynamic situations and increasing scheduling efficiency with higher generalization. The hyper-heuristic-based workflow scheduling algorithm in [25] presented three algorithms, involving two algorithms for makespan and cost decisions, respectively, and the third coevolves priority rules for both the decisions in the workflow tasks. By improving solution space exploration, the cooperative coevolutionary genetic programming hyper-heuristic maximizes multi-objective makespan and cost. The work in [26] employs Genetic Programming Hyper-Heuristics to construct energy-efficient scheduling heuristics, emphasizing dynamic workflow scheduling in cloud environments. It effectively balances resource utilization and energy efficiency, which allows adaptive selection of the best scheduling techniques for a wide range of workflow requirements. Although workflow scheduling with the help of hyper-heuristic algorithms improves task scheduling by generating or selecting heuristics dynamically to accomplish the objectives in cloud environments, these methods need to improve adaptability and scalability by the low-level heuristics to address complex scheduling problems. Even though rule-based and evolutionary-driven hyper-heuristics address execution speed, cost minimization, deadline, and resource variability, developing adaptive multi-objective optimization for real-time scheduling is essential for hyper-heuristics in a dynamic cloud environment.

III. PROBLEM FORMULATION AND SYSTEM DESIGN

With the heterogeneous nature of workloads, different task dependencies, and the dynamic nature of resources, workflow scheduling in the cloud encounters several challenges. Hence, this work presents a novel architecture to solve the problem of heterogeneous workflow scheduling while being aware of deadlines and budgets. The proposed workflow scheduling framework combines DRL with hyper-heuristics and DL-based state prediction for heterogeneous workflows and multi-objective optimization in a dynamic cloud.

A. PROBLEM FORMULATION

In cloud computing, the scientific workflow scheduling problem is modeled as a Task Dependency Graph (TDG) that comprises tasks and their dependencies, which is a Direct Acyclic Graph (DAG) structure. TDG is represented as $G = (V, E)$, in which edge $e_{ij} \in E$ is the precedence dependency between tasksT_i and T_j, and each node $v_i \in V$ is for task ti. Each task T_i, where i = 1, 2, 3,…, n, is mapped to a resource r_j where j = 1, 2, …., m. In the cloud, workflow scheduling decision y_{ij} is formulated in equation (1) for the 'p' ordered sequence of tasks based on their execution dependencies.

$$y_{ij} = \begin{cases} 1, & \text{if task } T_i \text{ is assigned to resource } r_j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Each T_i has a predefined execution time as d_{ij} for r_j and is stored in workflow matrix 'D' with the dimension of $n \times m$ in which d_{ij} refers to the element of D. Also, the weight w_{ij} is assigned to e_{ij}, indicating the communication cost between T_i and T_j. The proposed scientific workflow scheduling approach aims to minimize the overall budget (CostT) while meeting deadline constraints 'K'. The objective function is given by [27],

$$\min\left(\left[\sum_{i=1}^{n}(d_{ij}+L_j)\right]\left[\sum_{i,j}\left(\frac{S_{ij}}{\min(b_{j1},b_{j2})}+\text{Cost}_S\right)\right]\right)$$
$$\text{subject to: } F_{\text{exit}} \le K \tag{2}$$

As formulated in equation (2), S_{ij} is the transferred data size, L_j is the latency produced by resource r_j to execute task i, and b_{j1} and b_{j2} are the bandwidths of resources r_{j1} and r_{j2}, respectively. d_{ij} denotes the execution time of i^{th} task associated with its j neighbors and Cost_S refers to the scheduling cost. If two tasks are assigned to the same resource ($r_{j1} = r_{j2} = r_a$), the communication cost will be 0. Hence, to address workflow deadline constraints, the entire workflow tasks need to be completed within K, indicating:$F_{\text{exit}} \le K$where F_{exit} is the finish time of the last task (T_n) in the workflow application.

B. SYSTEM MODEL

The system design of cloud workflow scheduling architecture incorporates the cloud infrastructure, workflow, and task model.

TABLE 1. Comparison of Cloud Workflow Scheduling Approaches.

Reference	Objectives	Algorithm	Workflows	Limitations
Xie et al. (2024) [11]	• Cost • Execution time	• Two-stage preference-driven MOEA (tsp-MOEA)	• Epigenomics • SIPHT • Inspiral • Montage • CyberShake	Prioritization is challenging for diverse workflows
DB-ACO et al. (2023) [12]	• Budget • Deadline	• Ant Colony Optimization (ACO)	• CyberShake • Epigenomics • Montage • LIGO	Fails to schedule heterogeneous workflows
Yang et al. (2024) [13]	• Resource Utilization • Execution time	• Hyper-heuristic with mini-batch sampling	• Epigenomics • LIGO • CyberShake • Montage	Lack of handling dynamic execution time changes
Shafinezhad et al. (2023) [15]	• Budget • Deadline	• FastestScheduling • SchedulingWithMinCost • GapRate Analysis	• Montage • LIGO	Fails to handle diverse workflow characteristics
Zhou et al. (2023) [16]	• Budget • Deadline	• Budget-Deadline Constrained Workflow Scheduling (BDCWS)	• Epigenomics • Montage • LIGO	Lack of focus on dynamic resource availability
Tian et al. (2023) [17]	• Budget	Self-detection and Comprehensive Learning-based Battle Royale Optimization (SCLBRO)	• CyberShake • Epigenomics • Montage • Inspiral	Lack of focus on dynamic resource availability
Reddy & Reddy (2023) [18]	• Makespan • Resource Utilization • Throughput	• UUID-Blake • MD-PAM • LS-CSO • Anova-RNN	• Cybershake • Broad Band • Montage • LIGO • Epigenomics	Complex workflow scheduling is challenging
Mohammadzadeh & Masdari (2023) [19]	• Makespan • Cost • Energy • Throughput	• Seagull Optimization (SOA) • Grasshopper Optimization (GOA)	• CyberShake • Epigenomics • Montage • LIGO	Large-scale workflow is challenging
Ye et al. (2024) [20]	• Cost • Deadline	• Ant Colony Optimization (ACO)	• Epigenomics • Montage • LIGO • SIPHT	Dynamic IoT workload is challenging

TABLE 1. *(Continued.)* **Comparison of Cloud Workflow Scheduling Approaches.**

Gu et al. (2024) [21]	• Cost	• Deep Reinforcement Learning • Simulated Annealing	• CyberShake • Montage • LIGO	Fails to schedule heterogeneous workflows
Ye et al. (2024) [22]	• Cost • Deadline	Fuzzy-Multi-Workflow Scheduling (F-MWSA)	• Epigenomics • SIPHT • Inspiral • GeneSeq	Handling large-scale complex workflow is difficult
Mikram et al. (2024) [23]	• Budget • Deadline • Resource Utilization • Execution time	Hybrid Enhanced Particle Genetic Algorithm (HEPGA)	• Epigenomics • LIGO • CyberShake • Montage	Lack of balance between cost and execution time
Sun et al. (2024a) [24]	• Resource Utilization • Execution time	• Hyper-heuristic • Genetic programming (multi-tree)	• Epigenomics • LIGO • CyberShake • Montage	Lack of generalization
Zaki et al. (2024) [25]	• Cost • Makespan • Deadline • Resource Utilization	• Cooperative Coevolutionary GP • Hyper-heuristic	• Montage • LIGO • CyberShake • Bioinformatics	Lack of scalability
Sun et al. (2024b) [26]	• Budget • Energy • Deadline • Resource Utilization	• Genetic Programming • Hyper-heuristic	• Montage • LIGO • CyberShake	Limited adaptability to dynamic workloads

1) CLOUD INFRASTRUCTURE MODEL

In the cloud, heterogeneous physical servers are deployed to efficiently manage scientific workflow execution by considering deadline and budget constraints. Each host h_i is denoted as:

$h_i = \{P_i, C_i, R_i, B_i, S_i\}$where '$i$' is the index of the host (i = 1, 2,...m), P_i is the number of processing elements in h_i, C_i is the CPU capacity of h_i in a million instructions per second (MIPS), R_i is the RAM capacity of h_i in MB, Bi is the network bandwidth capacity of hi, and Si is the storage capacity of h_i. C_i is distributed across the availableP_i. For each h_i, k number of VMs are deployed to execute scientific workflow tasks, which are dynamic and adaptive based on computational requirements of the workflow for balanced cloud execution. Figure 1 shows the system model of the proposed workflow scheduling framework.

2) WORKFLOW MODEL

A scientific workflow is a structured collection of interdependent tasks with a specific goal, which are executed in a predefined order to solve complex scientific problems efficiently. The proposed system considers heterogeneous scientific workflows with complex computational processes that require different computational resources and execution environments. Balancing budget and deadlines ensures optimized task execution. In this model, scientific workflows are represented as TDG with $G = (V, E)$, where V is the set of tasks and E is their dependencies. As depicted in Figure 1, entry task (T_{en}) and exit task (T_{ex}) are the initial tasks with no predecessors or dependencies, and the final task without successors, respectively, belongs to one or more entry or exit tasks in each workflow application. Virtual entry task (T_{env}) and virtual exit task (T_{exv}) are modeled without data

transfer requirements and zero execution time, ensuring each TDG has an input and an output.Each T_i is characterized by the computational resources required and the size of data transferred from T_i to T_j. Hence, a workflow scheduling mechanism needs to satisfy tasks' deadlines to maintain workflow efficiency. Task execution is limited by resource availability and time constraints, and tasks are dynamically scheduled on available VMs with appropriate resources based on task precedence relationships.

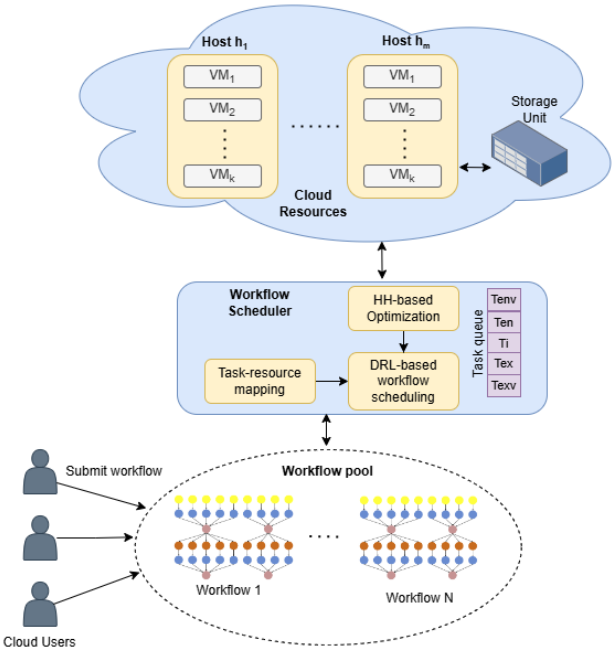


FIGURE 1. System Model.

Depending on resource availability, task dependencies, and execution priorities, task start and finish times vary in the cloud environment based on workflow scheduling.

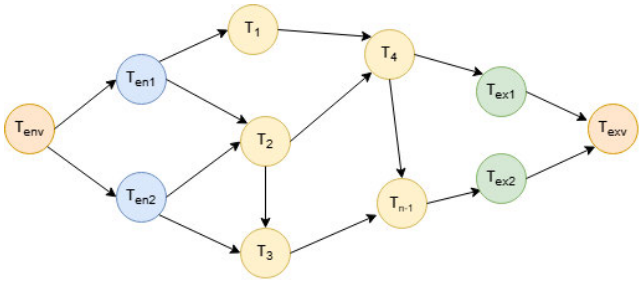


FIGURE 2. TDG Graph for Scientific Workflow Tasks.

Figure 2 shows an illustration of the TDG Graph for scientific workflow tasks. T_{en}^{v} acts as the starting point of the workflow for initializing the application. However, virtual tasks do not belong to computation. T_{en1} and T_{en2} are independent entry tasks with no dependencies. $\{T_1$ to $T_{n-1}\}$ are the core workflow tasks representing tasks with data dependencies. Tasks are scheduled based on the budget and deadline factors. T_1 utilizes output from T_{en1} to perform preprocessing. T_2 combines output data from T_{en1} and T_{en2}. Similarly, all the tasks are executed. T_{n-1} is the predecessor of exit task T_{ex}^{2}, aggregating results from T_3 and T_4 for the final processing step. T_{ex1} and T_{ex2} are the exit tasks to complete the workflow in which results from all dependencies are combined, which are further provided to T_{exv}, which acts as the end of the workflow.

IV. PROPOSED METHODOLOGY

In cloud environments, several traditional scheduling methods for scientific workflows are reactive. They are unable to adapt to changing resource availability, deadlines, execution costs, and task dependencies due to their static heuristics and lack of predictive capabilities. In addition, the existing approaches lack strong task prioritization and adaptive strategies, resulting in sub-optimal performance and resource utilization. Hence, the lack of balance between objective factors, such as deadline time and cost, emphasizes the necessity of an intelligent adaptive framework to satisfy the complex requirements of cloud-based tasks. Figure 3 shows the proposed workflow scheduling architecture.

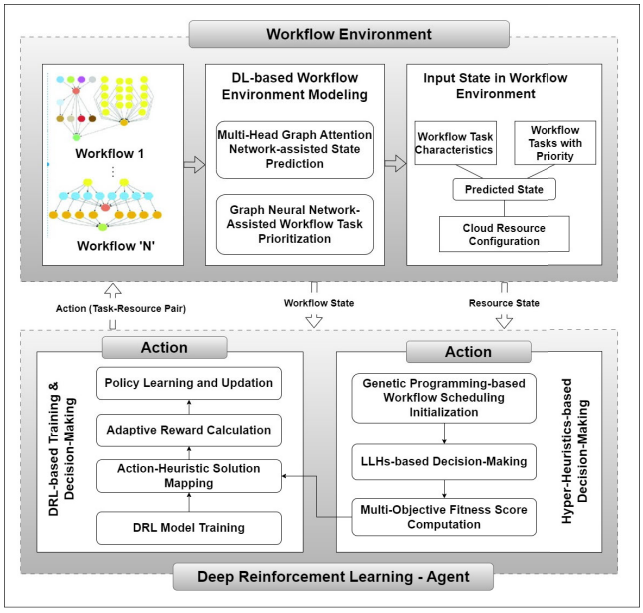


FIGURE 3. Proposed Workflow Scheduling Architecture.

The proposed workflow scheduling mechanism learns the input from diverse workflows with different resource requirements. To enhance the performance of the DRL system for workflow scheduling in a dynamic cloud environment, the proposed approach applies a Multihead Graph Attention Network (MGAN) model-based state prediction model that predicts resource availability and task completion time. If the new state is predicted, the DRL agent generates an action for adaptive decisions based on workload fluctuations, resource constraints, or execution delays. The predictive

ability allows the scheduling algorithm to precisely fulfill deadlines and execute within budget constraints, especially under dynamic and uncertain conditions. According to the action, the workflow tasks are assigned to resources by the policy and adaptive reward. The process is iterated if the objective tradeoff is needed until the conflicting objectives are balanced. If multiple objectives are balanced, task-resource mapping is performed, and subsequently, the workflow scheduler executes the tasks in a dynamic environment. Thus, an effective solution for adaptive workflow scheduling in a cloud system is accomplished by integrating improved state prediction, heuristic optimization, and the DRL method.

1) MULTIHEAD GRAPH ATTENTION NETWORK MODEL FOR STATE PREDICTION

For an effective DRL-based workflow scheduling, accurate prediction of states based on the variations in available resources and uncertainties in the cloud is crucial. In RL, state prediction is essential for enabling agents to predict future changes in their environment and perform precise decision-making because the RL model can proactively modify scheduling strategies based on dynamic factors, such as resource availability and workload fluctuations. Hence, the predictive ability guarantees effective resource allocation, minimizes delays, and maximizes cost utilization. In particular, by anticipating state changes, the scheduling model prioritizes workflows, avoids resource shortages, and ensures balanced budget and deadline factors in an unpredictable and changing cloud environment. Traditional state prediction is performed through the LSTM model [28]; however, LSTM's sequential nature provides the predictions for one timestep, leading to state prediction of the current task after completion of the preceding timestep, resulting in slower training. In addition, LSTM failed to effectively learn very long-term dependencies due to constraints imposed by training dynamics and memory capacity. Hence, it is not suitable for complex scientific workflow scheduling. To overcome this constraint, the proposed approach applies the Multihead Graph Attention Network (MGAN) model that utilizes the multi-head attention mechanism for assigning dynamic weights to neighbors of each task in graph 'G', allowing the model to concentrate on the relevant dependencies for predicting the state within DRL. In order to represent intricate relationships in structured data, such as workflows or cloud settings, MGANs become potential solutions for state prediction. MGAN can accurately predict the reinforcement state by utilizing multiple attention heads that gather a variety of patterns and interactions among workflow tasks, enabling proactive resource allocation by learning the dependencies in workflow scheduling. The multihead mechanism facilitates concurrent attention to diverse perspectives of a graph, such as resource constraints, task priority, and execution delays, to improve the robustness of predictions and ensure better adaptation to dynamic environments. The MGAN model aids in predicting states based on workload fluctuations, resource constraints, or execution delays in a dynamic environment. Incorporating an MGAN-based state prediction model into a DRL framework helps in adaptive decision-making and focusing on multiple relevant aspects of the environment to improve the ability of agents to handle complex scientific workflow scheduling that requires parallel examination of multiple factors and task dependencies. For example, the Cybershake workflow comprises multiple interdependent tasks, such as zipPSA (Pseudo-Spectral Acceleration), Xipseis, seismograms, Peakvalcalcokaya, and Extract Seismogram Green's Tensors (SGTs), each of which has a distinct function in seismic hazard analysis. Extract SGT facilitates effective ground motion simulations by computing SGTs. Source modeling and SGT integration are handled by Xipseis, which also manages earthquake rupture simulations. Seismograms utilize precomputed SGTs and rupture scenarios to create synthetic time-series data of ground motion. ZipPSA further analyzes the data to compute pseudo-spectral acceleration values for seismic hazard maps after Peakvalcalcokaya computes decisive factors from the seismograms, such as peak ground acceleration or velocity. These tasks in the Cybershake are interconnected, and the results of one task influence other tasks based on the workflow dependency, creating an organized workflow that is necessary for probabilistic seismic hazard analysis. For instance, 'Seismograms' depends on the completion of 'Extract SGT' and 'Xipseis', highlighting the significance of effective scheduling and resource allocation. Also, different tasks have different computational requirements, such as Extract SGT and Xipseis consume higher computation costs, whereas 'PSA' and 'Peakvalcalcokaya' consume minimal computational costs. However, these tasks rely on input data from previous phases. In order to comprehend and reduce the risks of earthquakes, probabilistic seismic hazard analysis relies on the creation of hazard curves and maps, which are produced by the combined outputs of these tasks.

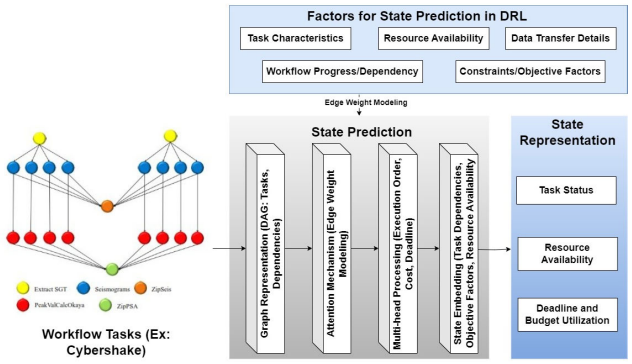


FIGURE 4. Workflow Task Modeling for State Prediction.

According to the workflow characteristics in this example, the proposed approach applies the MGAN model to predict the state of the Cybershake workflow, depicted in Figure 4. Initially, the CyberShake workflow is represented as a DAG

consisting of a number of interrelated tasks that are essential for seismic hazard analysis. A task, such as creating seismic waveforms, simulating ground motion, or calculating danger curves, is represented by each node in the DAG, and the edges expose data relationships between these activities. To ensure resource efficiency and reduce execution time, these tasks are efficiently mapped to cloud resources. For example, high-performance computing nodes are assigned to computationally demanding tasks, such as ground motion simulation. In contrast, resources near storage nodes are assigned to data-intensive tasks that require frequent access to seismic data in order to reduce data transfer latency. Additionally, by prioritizing the workflow's vital path, the mapping method guarantees that tasks have an impact on the overall completion time. To further improve the task mapping process, a multi-head graph attention network is included for state prediction, locating task relationships, resource states, and execution time by examining the CyberShake DAG in order to dynamically schedule the optimal task-resource pair based on the predicted state representation. To reduce latency, for instance, resources with higher network bandwidth are assigned to activities that require a lot of inter-task communication. In order to achieve cost-effectiveness and time compliance while preserving the accuracy of seismic hazard calculations, this predictive mapping ensures that the workflow scheduling is adaptive to the changing conditions of the cloud environment.In the MGAN-based model, each workflow task and cloud resource availability is represented as a node. The edges are task dependencies that form a bipartite graph, where each task node is connected to one or more resource nodes, indicating allocation possibilities. To model the precise representation of workflow as a graph structure, the proposed approach enables position encoding based on the dependency of the workflow tasks as the nodes in the graph. Each node contains task-specific information, such as task size, estimated execution time, and priority, whereas edges indicate dependence constraints. Each task(Ti) is represented by its feature vector, $T_i \in \mathbb{R}$ where 'M' indicates the number of features indicating the deadline, budget, and resource availability. In the proposed system, MGAN applies multiple attention heads [29] to process this graph. Each head focuses on the task's feature vectors from different perspectives, involving workflow task requirements (deadline), resource availability, and task dependencies with a unique identification of the task ID. For each Ti, feature representation is updated by information that is featured from its neighbors and weighted using the attention coefficients. The outputs from these heads are combined to observe detailed contextual and relational details about the workflow tasks. In the state prediction, the concatenated feature representation of each task is based on its neighbors across the multiple heads with the integration of the output projection matrix. To resolve the constraint in the LSTM-based model that is lacking in learning the very long-term dependencies, the proposed approach intends to enrich the feature representation in a temporal context.

$$Y_{T_i}^{AH'} = \sum_{H=1}^{AH} \text{LeakyReLU}\Bigg(\gamma^H \Bigg(\sum_{T_j \in \mathcal{N}(T_i)} \beta_{T_i,T_j}^H \cdot W_H Y_{T_j} + \mu^H \cdot T_{\Delta T} \Bigg)\Bigg) \tag{3}$$

As formulated in equation (3), the feature representation of each task is enhanced by temporal encoding ($\mu^H \cdot T_{\Delta T}$) with a temporal discount factor (γ^H) for future state prediction in the cloud environment. LeakyReLU is a nonlinear activation function, and attention coefficients are further normalized with the help of the softmax function, $\beta_{T_i,T_j}^H = \text{softmax}\left(\sigma_{T_i,T_j}\right)^H$ n long-term dependency modeling of scientific workflows, the scaling factor (γ^H) maintains the significance of timesteps appropriately and weight matrix (μ^H)) in each head modulates a learned temporal embedding ($T_{\Delta T}$) based on the time difference (ΔT) between tasks (T). The final concatenated output represents an enriched feature vector for each task 'Ti' by capturing task-specific information, such as deadline, budget, and resource availability. This output is fed into the dense layer to predict future states based on task-specific information and resource availability. The dense layer generates a predicted state,$\widehat{s}_{t+1}$, as states based on resource availability and task-specific information, which maps the features to action probabilities. Finally, the state prediction combines the concatenated output of each task, $Y_{T_i}^{AH'}$ and output projection matrix in multiple heads (W_{AH}), resulting $\widehat{S}_{t+1} = \left(Y_{T_i}^{AH'} \cdot W_{AH}\right)$.Accordingly, the MGAN model predicts the state with minimal loss between the actual and the predicted state as the action in the DRL. Subsequently, the reinforcement learning agent exploits the predicted state representation for proactive scheduling decisions. In the proposed system, the state prediction is the estimation of task execution time based on the resource constraints and workflow dependency in a dynamic environment. Thus, the DRL agent proactively predicts future states, resulting in optimal workflow scheduling in terms of reallocating the workflow tasks to different resources or updating task priorities, leveraging the DRL policy to adapt to dynamic heterogeneous workflows based on resource availability while satisfying deadlines and budget constraints.

A. HYPER-HEURISTICS AND ADAPTIVE REWARD MODELING IN DRL

In subsequent state prediction, the DRL agent generated a new action for the predicted state based on the deadline and budget factors with the assistance of heuristic generation in hyper-heuristics. To implement hyper-heuristics for workflow scheduling, the proposed approach initially prioritizes the tasks using the graph-based model. Then, it selects the action for the predicted state that acts as the current state for improved DQN-based workflow scheduling. Finally, the heuristics are generated for each workflow based on the Q-learning in the DRL model.

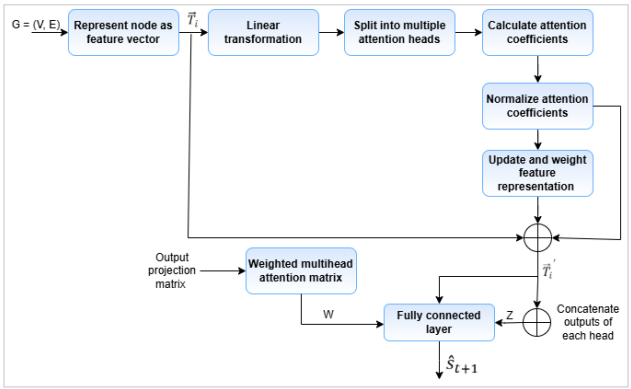


FIGURE 5. Proposed MGAN-based State Prediction Model.

Algorithm 1 : MGAN-Based State Prediction

```
Input  : Scientific Workflow Tasks
Output: Predicted State for DRL Implementation
for all workflow tasks and resources do
    for each iteration/episode in scheduling do
        Initialize workflow tasks and resources;
        Apply DRL for scheduling decision;
        for each task T_i in workflow tasks do
            Apply MGAN to predict state based on
            neighbors with position encoding;
            for workflow graph structure do
                Learn task requirements and resource
                availability with deadline and budget;
                Compute execution time for each task as
                state with task-resource pair;
                Predict state by MGAN;
            end
            predicted_state_{T_i} ← MGAN(T_i, neighbors);
            Update state representation, Y_{T_i};
            for each task's representation do
                Integrate temporal factor over timesteps
                between task transition;
                Update state representation;
            end
        end
    end
end
```

1) TASK PRIORITIZATION USING GRAPH NEURAL NETWORKS

The proposed approach adopts the GNN for workflow task prioritization due to its capability to accurately represent and comprehend the intricate relationships and interactions between tasks in a workflow. In this context, tasks are modeled as nodes in a graph, and edges are represented as task dependencies in the workflow. GNNs analyze the graph structures to determine the relationships between tasks and provide priorities according to deadlines, budget, and resource availability. By combining information from nearby nodes and edges, GNNs are able to collect both local and global context to prioritize the workflow tasks, enabling dynamic priority modeling of workflow tasks based on the changing environments, such as task dependencies and objective factors. In GNN, each task is trained with an estimated execution time with the target label as priority scores or execution order based on the deadline and budget without compromising dependency constraints. For each 'Ti' in the scientific workflow application, GNN captures the inter-dependent relationships in the graph structure to assign priorities and generate task sequences based on the objective factors (deadline, budget) from the predicted state with the estimated execution time of each task (i).w_E, w_D, and w_B are the constant weights associated with objective factors that influence its significance on task priority.

Algorithm 2 GNN-Assisted Task Prioritization

```
Input  : Scientific Workflow Tasks
Output: Prioritized Tasks
for all workflow tasks and Resources do
    for each iteration/episode in scheduling do
        for each task T_i in workflow tasks do
            Apply GNN to prioritize tasks based on
            neighbors' features;
            for workflow graph structure do
                Compute Priority score for each task
                based on deadline and budget;
                if deadline & Budget satisfies with
                modeled priority then
                    Assign priority score as High in
                    {workflow tasks' priority};
                end
                Model sequence of tasks with priority as
                target labels;
                for workflow task-modeled GNN do
                    Train the model with various
                    prioritized workflow sequences;
                    Predict the priority for each
                    workflow;
                end
                Assign the prioritized workflow tasks;
            end
        end
    end
end
```

$$\text{Label}_P \leftarrow \left\{ w_E \times \text{Execution time}_i; \left[w_D \times \text{Deadline}_i,\ w_B \times \text{Budget}_i \right] \right\} \quad (4)$$

To examine the input data in a graph structure, GNN utilizes the feature vectors in MGAN-based state prediction to generate output task sequences by processing task features and dependencies. A new representation is assigned to each Ti

that highlights its features and dependent tasks. It is updated by accumulating information from neighboring nodes based on the dependencies, allowing the model to prioritize tasks with higher dependencies. Moreover, GNN updates the task representation based on the information passing through its neighboring nodes to generate the priority score for each task. The proposed scheduling strategy applies topological sorting to generate a baseline order of tasks that considers all task dependencies, which avoids task scheduling before the completion of its predecessor tasks in the dependent workflow tasks. Thus, the final set of task sequences is generated by utilizing the updated task priorities.

2) IMPROVED DQN–BASED HYPER HEURISTICS MODELING

In hyper-heuristics, heuristics generation dynamically optimizes task scheduling based on workflow characteristics and cloud resource availability, which is an effective solution for heterogeneous workflow scheduling. In particular, the proposed framework effectively manages the varying workloads of cloud systems by constantly updating the hyper-heuristic algorithm. In contrast to the static heuristic technique, this hybrid approach enables a flexible and generalizable scheduling solution for the heterogeneous workflow tasks in the cloud. The hyper-heuristic technique allows a wider search space by creating new heuristic strategies in real time that are adapted to the workflow constraints and characteristics, including task dependencies, budget, and deadlines. The workflow scheduling problem is addressed by utilizing a generation-based constructive HH approach integrated with established algorithms from the optimization domain. An improved DQN-based module is considered a high-level heuristic, and it is designed to adjust to dynamic environments.

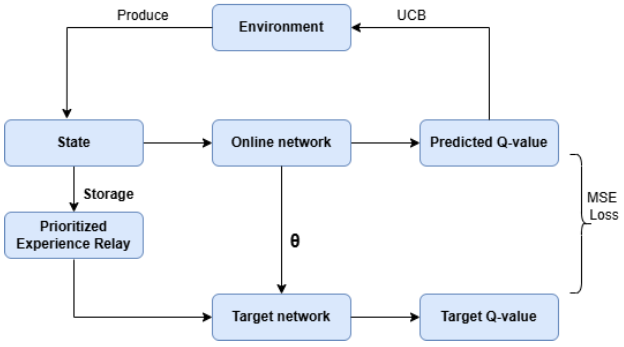


FIGURE 6. Improved DQN-based Hyper-Heuristics.

During training, workflow tasks are presented sequentially after task prioritization until all tasks in the workflow are completed, which is indicated as an episode. At each episode, the Low-Level Heuristics (LLH) generation module generates a new LLH from a predefined set by transforming or combining the algorithms to construct real-time solutions. Although LLHs are problem-specific, the High-Level Heuristic (HLH) is modeled to be generalized. It applies to several problems other than workflow scheduling problems due to its domain-agnostic nature. The structure and modeling of the proposed improved DQN-based HH are illustrated in Figure 5. The high-level heuristic is applied as a neural network, which is initialized with parameters θ. The scheduling decision relies on the neural network during training and is stored in a prioritized experience replay for updating the network weights. Instead of applying uniform sampling, each transition is assigned a priority depending on its importance. Transitions that have higher priorities are often sampled for training. Thus, an improved DQN-based trained model as the high-level heuristic leverages the proposed system to learn from states, decisions, and rewards obtained from the environment to iteratively improve its performance.

3) ADAPTIVE REWARD CALCULATION

In the DRL, the reward function is often static and predefined and fails to completely examine the dynamicity of objective factors during workflow scheduling decision-making. Hence, an adaptive reward function is utilized to evaluate the quality of task allocation on the basis of objective factors, such as budget and deadline. This function adjusts to evolving goals as training progresses, aiding in effectively attaining complex, multi-objective optimization. It helps the reinforcement policy 'π' to improve over time by learning from feedback. The adaptive reward function is designed to calculate deadline and cost efficiency under 'π' in a given state space 'S' that is parameterized by 'θ'. From the inspiration of work in [30], the expected adaptive reward function is modeled based on objectives in equation (5). $R(S, \pi)$ is the adaptive reward function that dynamically adjusts on the basis of current task scheduling outcomes and multi-objective factors, such as budget and deadline.

$$R_{\text{objective}}(S, \pi) = \delta_1 \cdot \text{CostEfficiency} + \delta_2 \cdot \text{Deadline} \tag{5}$$

In equation (5), δ_1 and δ_2 are constant weights for balancing objectives during workflow scheduling decision-making. Dynamic scaling adaptively increases rewards for complex tasks during high-priority stages. In adaptive reward, $P(\pi \mid S)$ is the adaptive penalty provided when objectives are not met, which is mentioned in equation (6) with the constant weights τ_1 and $\tau_2$for the objective factors.

$$P(\pi \mid S) = \tau_1 \times \text{Cost}_{\text{inefficiency}} + \tau_2 \times \text{Deadline}_{\text{violation}} \tag{6}$$

For dynamic workflow scheduling, the RL agent continuously interacts with the environment by exploring and exploiting to generate new LLHs that optimize the scheduling decision. The model learns the best action to optimize the objective function over time. If the tradeoff between budget and deadline is satisfied during the scheduling decision, tasks are mapped to corresponding resources. Otherwise, the policy

will be updated, and the process will be repeated. If task-resource mapping is completed, the workflow scheduler adaptively executes the tasks for dynamic environmental variables, including resource availability or task execution delays, which are constantly assessed and updated during execution. With the help of this dynamic generation process, the proposed approach ensures effective task-resource mappings, balancing multiple factors, such as dynamic variations in the cloud environment, cost, and deadline objectives.

B. Q-LEARNING-BASED HEURISTIC GENERATION AND WORKFLOW SCHEDULING

In the DRL-based proposed system, to dynamically generate heuristics for optimizing workflow scheduling tasks, the traditional heuristic evolution method is enriched by heuristic generation, which is adaptive optimization. Evolutionary algorithms are computationally expensive and often struggle for local or global convergence, particularly for highly complex scientific workflow-based optimization problems. Moreover, evolutionary algorithms are limited in their ability to cope with problems with multiple objectives simultaneously. In contrast to traditional evolutionary methods [23], [24], [25], [26], [27] the hyper-heuristic algorithm is an effective solution for complex scientific workflow tasks due to adaptive learning of multiple objectives through training on state-action sequences in RL scenarios. In HH, the high-level heuristic is modeled as a neural network with weights' θ' that dynamically generates heuristics based on LLH by learning from interactions with the environment. In the proposed system, to generate hyper-heuristics, Genetic Programming (GP) is represented as perturbative heuristics in which the heuristic solutions are modeled as trees. In GP, trees comprise nodes that represent composite mathematical operators and leaves that are modeled with the results of LLHs with their parameters, involving Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Simulated Annealing (SA), referring to the parent heuristics. The proposed approach adopts the PSO, ACO, and SA models for scientific workflow scheduling due to their specific features, which are beneficial for cloud-based workflow scheduling that considers deadlines and budgets. In the LLH set, each heuristic algorithm is responsible for computing the fitness score for the entire workflow scheduling solution based on the workflow stages, as mentioned in Figure 6. In the LLH modeling, PSO utilizes swarm intelligence to dynamically distribute workflow tasks on resources with balanced performance and cost while adjusting to changes in cloud environments. In addition to its cost-effectiveness, PSO plays a vital role in rapid reallocation, which is beneficial for deadline violation constraints during execution. By imitating ant behavior, ACO effectively manages task dependencies and optimizes execution paths, enabling the workflow scheduling algorithm to fulfill deadlines and reduce data transmission times, thereby offering cost-effective mappings. Also, SA provides a robust solution to avert the local optima when allocating resources for the prioritized tasks, ensuring an effective and time-efficient task execution in the cloud, even for complex workflow tasks. In order to explore and utilize the advantages of the solution space, the proposed workflow scheduling decision-making employs LLHs as building elements. In HH, the genetic process consists of selection, crossover, and mutation wherein offspring heuristics are created by combining parent heuristics that are selected based on their performance in terms of Q-value and fitness score belonging to deadline and budget factors. In crossover operation, subtrees representing various LLHs are switched between parents to produce a varied collection of offspring, whereas mutation necessitates changes in the heuristic parameters. By iteratively improving the heuristic trees according to the fitness of the generated solutions, the proposed approach maximizes the overall effectiveness of workflow scheduling in the cloud. In the proposed hyperheuristic modeling, the switch-over procedure of LLHs for deadline and budget-aware workflow task scheduling includes a dynamic transition between PSO, ACO, and SA according to task requirements and real-time performance. To accomplish this, the proposed approach implements the workflow scheduling algorithm for LLH-based task-resource mapping based on the workflow stages, such as initial task-resource mapping as stage 1, dependency-aware mapping as stage 2, and refinement as stage 3. Among the three LLHs, PSO is initially used in the initial mapping stage for all the tasks in the scientific workflow to produce extensive task-resource mappings quickly, gaining the advantage of its tendency to handle changing resource availability and rapid convergence. Subsequently, for the stringent dependency constraints between the workflow tasks, ACO utilizes pheromone-based reinforcement to minimize execution costs and optimize task allocations with complicated dependencies. Furthermore, SA is triggered to explore and refine solutions globally during the refinement stage, addressing negligible budget or deadline violations from local optima through its exploration characteristics. During the LLHs execution, monitoring fitness scores, such as achieving deadlines, maintaining budgets, and fulfilling priority constraints, initiates the transitions between these heuristics. Even though each LLH computes the fitness score based on both the deadline and budget, when the improvement degrades or violations occur, the heuristics are switched at the refinement stage. For instance, ACO optimizes costs when budgets are at risk for a complete workflow, and PSO quickly reallocates to resolve deadline violations. In order to achieve the dual goals of deadline and budget factors, this adaptive, staged, and criteria-based switch-over approach guarantees effective, balanced workflow task scheduling in the cloud.

In a dynamic environment, at each time 't', a high-level heuristic generates a new LLH that is suitable for the current state to provide the workflow scheduling solution in real-time, ensuring that LLHs are contextually optimized for the problem instead of relying on the predefined state of task-resource characteristics. During training, this framework interacts with the environment to gather states, actions, and

Algorithm 3 Heuristic Generation in HH and Scheduled Workflow Execution

```
Input  : Scientific Workflow Tasks
Output: Deadline and Budget-aware Scheduled Workflow Tasks
for all workflow tasks and Resources do
    for each iteration/episode in scheduling do
        for all workflow tasks with requirements do
            Generate heuristics based on objective factors and priority of tasks using Q-learning;
            Divide workflow stage for LLH switch over process;
            LLH(T_i) = Generate_heuristic(  priority_Ti, cost_Ti,
             deadline_Ti)
            for all tasks in workflow do
                Apply genetic programming (crossover and mutation) to LLHs;
                for each heuristic tree do
                    Apply PSO for initial task-resource mappings based on deadline & budget;
                    for initially scheduled workflow tasks do
                        Apply ACO for strict dependency-aware scheduling during execution;
                        for all workflow tasks do
                            Apply SA for refinement with deadline-budget-aware fitness score;
                            if deadline violates for each workflow then
                                Trigger PSO re-execution;
                            end
                            if budget violates for each workflow then
                                Trigger ACO re-execution;
                            end
                        end
                    end
                end
                Compute final fitness score for the workflow with generated heuristic tree and solution;
            end
            Action(T_i)Select_action(Q, T_i, LLH(T_i), Reward);
        end
        for all tasks do
            Execute RL agent's action:
            Action = T_i(predicted state, generated heuristic,
            priority, and objective factors);
            Execute tasks on resources;
        end
        for all the scheduled tasks do
            Compute new state and reward for current state;
            if reward ≠ objective factors then
                Execute adaptive reward function for all the tasks;
            end
            Deadline and budget-aware optimized workflow task execution;
        end
    end
end
```

rewards based on objective factors, deadlines, and budget. For periodic weight updates, these interactions are stored in an experience memory. Improved DQN is utilized to learn an optimal policy to generate LLHs under dynamic conditions. In DQN and Double DQN (DDQN), the Q-values are directly computed as a cumulative reward of action for a particular state without separating the advantage component and value component. Also, combining dueling architecture with DDQN enables it to effectively concentrate on the state value with Q-value approximation, particularly when the actions minimally influence the outcome. Consequently, the agent effectively learns the environment where the state is vital for objective-aware scheduling decision-making, which is crucial in heuristic generation. In improved DQN, policy

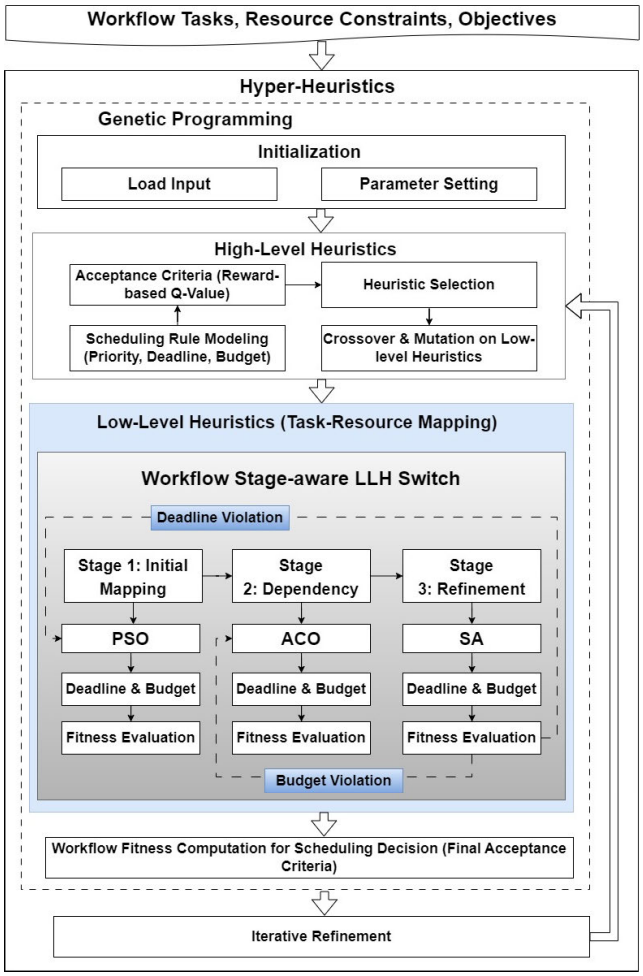


FIGURE 7. Hyper-Heuristics Modeling in Proposed Workflow Scheduling.

'π' maps predicted $\widehat{s}_{t+1}$ to action, At to decide the best action to balance the scheduling objectives. Each action At related to generating LLHs is optimized for cost efficiency and deadline adherence. The policy maximizes the long-term return that aggregates rewards through a sequence of steps. Q-value functions help to assess the quality of actions. In improved DQN, the optimal Q-value function is the maximum expected return attainable from St based on the optimal policy π, modeled in equation (7).

$$Q'\left(\widehat{s}_{t+1}, A_t\right) = \max_{\pi} Q_{\pi}\left(\widehat{s}_{t+1}, A_t\right) \tag{7}$$

In equation (7), $Q\left(\widehat{s}_{t+1}, A_t\right)$ indicates a state-action value function that evaluates the expected return from the action, At for state $\widehat{s}_{t+1}$. It enables the DRL agent to make optimal decisions by selecting optimal action (At), which maximizes long-term returns. The Bellman Equation represents the relationship between the Q-value of a state-action pair and the succeeding state. Bellman's principle of optimality is applied to transform the problem into iterative sub-problems, allowing the optimal solution to be derived step-by-step from the final state. A neural network with parameter 'θ' is considered the approximator of the optimal predicted Q-value function. The predicted Q-value (Q_P) after applying LLH for $\widehat{s}_{t+1}$ is given by equation (8), which is a target Q-value (Q_T) to train the reinforcement learning model for workflow scheduling optimization.

$$Q_P\left(\widehat{s}_{t+1}, \text{LLH}; \theta\right) \approx R\left(S, \pi\right) + \phi \cdot \left(\max_{\text{LLH} \in A_N} Q\left(\widehat{s}_{t+2}, \text{LLH}; \theta\right)\right) \tag{8}$$

For Q-learning, the target Q-value in equation (8) is the combination of the reward and prediction for the next state with hyper-heuristics, generating stable and reliable target values for training to prevent variations in the learning process. $R\left(S, \pi\right)$denotes the immediate reward for the state transition based on the policy (π), leverages the action selection. 'ϕ' signifies the importance of reward for future reward modeling over all the actions (A_N) at the state after the next ($\widehat{s}_{t+2}$) in which scheduling decisions are based on LLH. To achieve minimal loss, the proposed approach updates the weights. It utilizes the optimally predicted Q-values based on the backpropagation, resulting in the DRL learning the predicted optimal Q-values. In DRL, the proposed approach applies an Upper Confidence Bound (UCB) that balances exploration and exploitation dynamically by ensuring continuous exploration of actions in an uncertain environment. Hence, the UCB method [31] is utilized to balance exploration and exploitation to select actions with maximum Q-values in the proposed workflow scheduling context. Figure 7 shows the steps involved in the proposed methodology as a flowchart.

1) HYPERHEURISTICS-BASED WORKFLOW SCHEDULING

In the proposed system, the DRL agent interacts with the cloud environment to develop an optimal solution involving the multi-objective-aware heuristic generation in hyper-heuristics and workflow scheduling, which is the key process of heterogeneous workflow scheduling. Integrating a hyper-heuristic mechanism with the DRL model improves the agent's decision-making process while allocating the workflow tasks to dynamic cloud resources. During the decision-making, the proposed approach designs the adaptive reward function based on the prediction outcome and multi-objective factors based on the policy update of previous state-action pairs in the dynamic environment.

V. EXPERIMENTAL EVALUATION

The experimental model implements the proposed algorithm using Workflowsim to experiment with real-world workflow datasets, such as Cybershake, Epigenomics, Inspiral, and Montage. The performance of the proposed cloud workflow scheduling framework is examined with the experimental setup modeled in Table 2. The hardware configurations are Ubuntu 18.04 with an x86 processor, which includes scalable host and virtual machine configurations designed for effective workflow execution. With 100 GB of storage,

10,000 MIPS of processing power, 64 GB of RAM, and 1 Gbps of bandwidth, hosts provide performance for handling numerous processes. With 2500 MIPS, 4 GB RAM, and 100 Mbps bandwidth, virtual machines with Xen hypervisor provide an optimal trade-off between cost and resource availability. The evaluation highlights the efficacy of the scheduling approach in minimizing makespan and cost while resolving deadline and budget constraints, highlighting its scalability and applicability for real-world workflows, such as CyberShake, Epigenomics, Inspiral, and Montage. The results demonstrate the effectiveness of the workflow scheduling algorithm in addressing large-scale, deadline-sensitive, and budget-constrained scheduling challenges in cloud environments.

Dataset Modeling: To implement the scientific workflows in the Workflowsim for deadline and budget-aware workflow scheduling algorithm,

this work unifies the scientific workflows as datasets with task length, task size, workflow dependency, and deadline-budget factors, as mentioned in Table 3.

Table 4 provides the application and its real-time data source details for the evaluated scientific workflow datasets, such as Cybershake, Epigenomics, Inspiral, and Montage.

A. IMPLEMENTATION STEPS

This section outlines the implementation steps to map the proposed workflow scheduling algorithm with different scientific workflows in the Workflowsim.

TABLE 2. Workflowsim Simulation Configuration.

Resource	Attributes	Configuration/Values
System/Platform	Architecture	×86
	Operating System	Ubuntu 18.04
Cost	Process	$0.02 per second
	Memory	$0.01 /GB per Hour
	Storage	$0.1 /GB/Month
	Bandwidth	$0.05/GB
Host Configuration	Storage	100 GB
	MIPS	10000
	RAM	64 GB
	Bandwidth	1 Gbps
VM Configuration	VMM Name	Xen
	Storage/Image Size	10 GB
	MIPS	2500
	RAM	4 GB
	Bandwidth	100 Mbps
	Processing Elements	4

Step 1: Run Workflowsim.
Step 2: Select workflow type (Ex: Cybershake) from Workflowsim.
Step 3: Define task (size, deadline) and resource (CPU, memory, bandwidth, cost) characteristics.

FIGURE 8. Flowchart of the Proposed Workflow Scheduling.

TABLE 3. Dataset Modeling in Workflowsim.

Workflow	In WorkflowSim	Values	Customization
Task Length	Yes	Predefined value in Millions of Instructions (MI)	Enabled
Task Size Input Data Size Output Data Size	Yes	Predefined value in Bytes	Enabled
	Yes	Predefined value in Bytes	Enabled
Workflow Dependency	Yes	Predefined Workflow structure	No
4*Deadline	No	To be configured for each task/workflow	Enabled
4*Budget	No	To be configured for each task/workflow	Enabled

Step 4: Implement MGATN model on workflow for state prediction.
Step 4.1: Model DRL scenario as mentioned in Table 5.
Step 5: GNN for task prioritization.
Step 5.1: Model graph structure for GNN as shown in Table 6.
Step 5.2: Node embeddings based priority score prediction for each task by GNN.
Step 5.3: Train GNN with labeled priority score to ensure minimal loss.
Step 6: DQN-based Hyper-Heuristic generation.

TABLE 4. Workflow Dataset Details.

Workflow Type	Application	Dataset Source
Cybershake	Earthquake – Disaster Prediction	Southern California Earthquake Center (SCEC)
Epigenomics	Bioinformatics – DNA Sequencing	UCSC Epigenome
Inspiral	Astrophysics – Gravitational Wave	Gravitational Wave Open Science Center
Montage	Astronomy – Image Mosaicking	NASA Infrared Science Archive

TABLE 5. DRL Modeling Scenario.

State	Task status (Completed/pending/in progress), Resource availability, Objective Factor
Action	Earliest deadline first based on priority and objective factors.
Reward	Positive reward – For improving workflow execution efficiency Negative reward – For higher makespan and cost
Policy	DQN maps the state actions to improve cumulative rewards based on LLH

Step 6.1: Implement LLH for workflow scheduling and resource allocation as modeled in Table 7.
Step 7: Adaptive Reward calculation based on actions related to objective factors.
Step 8: Q-learning-based workflow scheduling.

B. RESULTS DISCUSSION

To validate the performance of the workflow scheduling algorithm, this work utilizes the normalized cost and Makespan as the key metrics, along with a novel metric of Deadline-Budget Adherence Score (DBAS).

1) NORMALIZED COST

It is the ratio between the execution cost of the algorithm for each workflow and the low-priced cost for the corresponding workflow, which measures the cost efficiency under budget constraints. For the modeling of low-priced cost, all the tasks in the workflow are scheduled for only low-priced VM.

$$\text{Normalized Cost} = \frac{\text{Total Execution Cost}}{\text{Low-priced Cost}} \tag{9}$$

2) MAKESPAN

It is the total time taken to complete all the workflow tasks from the initiation of the first task to completion of the end task, including communication time, and measures the time efficiency under the deadline constraints. Makespan is the time difference between the end task's completion time (T_{End}) and the first task's initiation time (T_{Start}) in which lower makespan indicates effective scheduling performance.

$$\text{Makespan} = T_{\text{End}} - T_{\text{Start}} \tag{10}$$

TABLE 6. Graph Modeling for GNN.

Node	Tasks
Edges	Dependencies in workflow
Node Features	Execution time, resource demand, dependency level (based on the number of incoming and outgoing edges) tasks
Edge Features	Data transfer size

TABLE 7. Hyperheuristics Modeling.

Hyper-heuristic Generation	Heuristics type
Perturbative Heuristics	Heuristic Representation using Genetic Programming
Heuristic Representation	In tree representation: Nodes are composite mathematical operators; Leaves are outputs of PSO, ACO, and SA with parameters
LLHs	Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Simulated Annealing (SA)
Selection, Crossover, Mutation	Subtree combination of parent heuristics to generate offspring based on objective factors in fitness score and Q-value

3) DEADLINE-BUDGET ADHERENCE SCORE (DBAS)

It is the measure of a balanced compliance score for deadline and budget-aware workflow scheduling algorithm while computing for each workflow, ranging between 0 and 1, in which a higher score indicates an effective scheduling solution.

$$\text{DBAS} = \alpha \cdot \left(1 - \frac{\text{Makespan} - \text{Deadline}}{\text{Deadline}}\right) + \beta \cdot \left(1 - \frac{\text{Total Cost} - \text{Budget}}{\text{Budget}}\right) \tag{11}$$

In DBAS, α and β refer to the constant weighting parameters, $\alpha + \beta = 1$ The deadline and budget indicate the maximum allowable time and cost for the corresponding workflow execution.

4) DEADLINE VIOLATION RATIO

It is the measure of the feasibility of workflow scheduling solution, which is the ratio between the workflow finish time, that is, makespan, and the workflow deadline constraint.

$$\text{Deadline Violation Ratio} = \frac{\text{Makespan}}{\text{Deadline}} \tag{12}$$

If the deadline violation ratio is greater than 1, the workflow scheduling solution fails to ensure the deadline constraint. Hence, it is not feasible; otherwise, a workflow scheduling solution is feasible.

5) COMPARATIVE PERFORMANCE ON EXISTING MODELS

The performance of the proposed system is assessed with different existing workflow scheduling models [11], [12], [13],

[23], [25], [26], in which the research work [12] is compared with the proposed system due to similar workflow models and experimental scenarios. To implement the workflow scheduling algorithms, this work employs the Cybershake (30,50,100,1000), Epigenomics (24,46,100,997), Inspiral (30,50,100,1000), and Montage (25,50,100,1000) workflow sizes. By iteratively executing the different workflow sizes 10 times, average results are obtained for each workflow type, proving the capability of the workflow scheduling model on heterogeneous workflows. In the result evaluation, deadline and budget factors refer to the degree of looseness of deadline and budget, respectively, in which a large deadline factor indicates strict budget constraints and looser deadline constraints during workflow scheduling. With an advanced tradeoff mechanism, the work in [11] introduced a two-stage preference-driven multi-objective evolutionary algorithm for cloud workflow scheduling that effectively balances makespan and cost. It confronts adaptability in scheduling decision-making to changing cloud conditions because it concentrates on static environmental conditions in the workflow and cloud. However, in a variety of uncertain situations, the proposed method outperforms the static framework by guaranteeing adaptability to state prediction and optimal decision-making. The DB-ACO approach [12], which is designed for deadline and budget constraints, provides an effective solution in moderate environmental settings; however, it confronts an extremely dynamic cloud execution environment because it depends on pre-recognized heuristics. In contrast, the proposed approach outperforms DB-ACO by adding hyper-heuristics and predictive modeling, guaranteeing better performance and a higher success rate in demanding and dynamic scheduling problems. On the other hand, a genetic programming hyper-heuristic method that is effective at heuristic selection by mini-batch sampling is presented in [13]. However, it does not have predictive adaptability for dynamic settings. However, it is restricted to static or moderately dynamic situations. Similarly, the work in [23] optimized cost and makespan by combining evolutionary and genetic methods. A cooperative coevolutionary genetic programming paradigm that is effective for large-scale workflows, however, consumes high computational costs and fails in environmental adaptation [25]. Furthermore, energy-efficient scheduling [26] achieved energy efficiency. However, it was confronted with higher costs and makespan due to a lack of tradeoff between the time and budget constraints in a dynamic cloud environment.

TABLE 8. Normalized Cost Performance with Budget and Deadline Factor in the range [0.005, 0.05].

Comparative Works	Normalized Cost Performance on Different Workflows							
	Budget Factor				Deadline Factor			
	Cybershake	Epigenomics	Inspiral	Montage	Cybershake	Epigenomics	Inspiral	Montage
ProLiS [14]	0.280	3.875	–	0.364	6.926	5.960	–	12.267
L-ACO [14]	0.280	3.785	–	0.240	7.762	5.799	–	12.296
DB-ACO [12]	0.240	3.708	–	0.230	6.078	6.061	–	12.362
Proposed	0.180	3.285	3.019	0.212	5.141	5.432	3.123	10.561

To assess the performance improvement of the proposed hyperheuristic tree-based deadline-budget aware workflow scheduling algorithm, the experimental model evaluates the existing ProLiS [14] that offers reliable deadline distribution methods. To further improve workflow scheduling under deadline restrictions and cost optimization goals, LACO incorporates ACO, building on ProLiS. To help ants create an ideal task ordering, LACO maintains a pheromone trail, which is coupled with the probabilistic upward rank to provide precise job priority. In the subsequent task orders, ProLiS assigns each task to a Virtual Machine (VM) service, guaranteeing that deadlines are fulfilled and overall workflow costs are maintained to a minimum level. Even though ProLiS and L-ACO ensure timely execution by allocating the workflow deadlines across tasks efficiently, the adaptive reward computation based on the hyperheuristic-fitness score greatly mitigates the normalized cost by efficient and deadline-budget-aware scheduling in the proposed approach.

TABLE 9. Normalized Cost Performance with Budget factor in range [0.05, 0.5].

Comparative Works	Normalized Cost Performance on Different Workflows							
	Budget Factor				Deadline Factor			
	Cybershake	Epigenomics	Inspiral	Montage	Cybershake	Epigenomics	Inspiral	Montage
ProLiS [14]	0.296	3.890	–	0.488	1.189	4.196	–	1.847
L-ACO [14]	0.280	3.788	–	0.296	0.909	4.069	–	1.577
DB-ACO [12]	0.240	3.790	–	0.284	0.908	4.072	–	1.647
Proposed	0.175	3.149	3.014	0.235	0.719	3.836	3.089	1.118

For various deadline and budget factors in the cloud environment for workflow tasks, the experimental model compares the performance of normalized cost, depicted in Tables 8 and 9. As mentioned in Table 9, by contemplating the deadline and cost factors during both the adaptive reward calculation and fitness score computation in hyperheuristics, the proposed approach overcomes the drawbacks in the existing research. It allows for dynamic scheduling and real-time flexibility. Consequently, it minimizes computing overhead, optimizes workflow scheduling across a variety of goals, and dynamically predicts changes in the workflow environment. Compared to the work [12], which is shown in Tables 8-9, the comprehensive optimization strategy in the proposed system, particularly hyperheuristic generation and predictive state-based decision-making by DRL, guarantees balanced performance in terms of cost and makespan that satisfies deadline, ensuring reliable and effective scheduling solution for dynamic cloud environments. To measure the average runtime performance with the increase in workflow size, the experimental model utilizes the different workflow sizes of Cybershake workflows with a deadline factor of 0.2. For example, for 200 workflow sizes, the experimental model exploits two Cybershake-100 workflows. Similarly, the linearly increased workflow sizes are evaluated. In Figure 8, even though the runtime of all methods increases with workflow size, the proposed DRL-based adaptive workflow scheduling model performs better than DB-ACO [12],

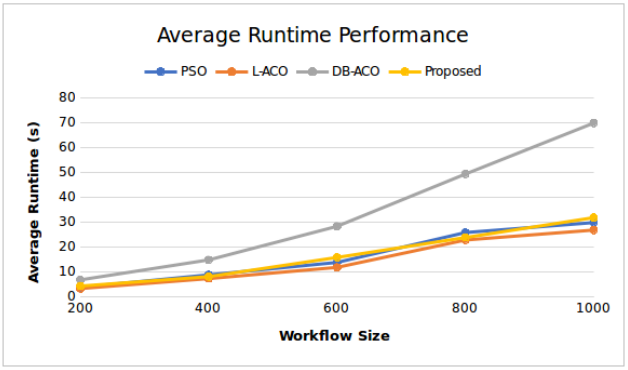
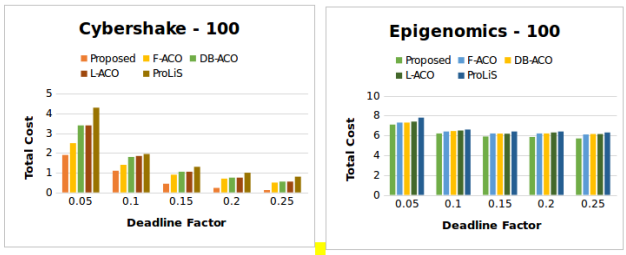


FIGURE 9. Runtime Performance on Different Workflow Size.

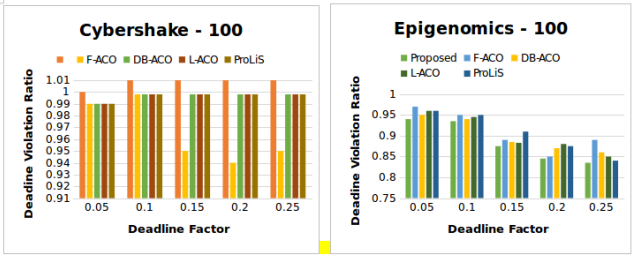
PSO, and L-ACO [14] models. The efficiency of DB-ACO decreases with larger-scale processing due to heuristic modeling, which is resolved by the hyper-heuristic design-based workflow scheduling in the proposed system. The DRL-based adaptive scheduling model streamlines decision-making by learning the changing environment from a predictive state through a multi-head graph attention network, which eliminates the irrelevant task-resource scheduling. Moreover, the proposed method reduces the effect of workflow size on runtime by dynamically updating the scheduling decisions via adaptive rewards and fitness scores. Thus, by the combination of deep reinforcement learning and hyper-heuristic generation, the proposed model manages task-resource mapping with higher efficiency and adaptability. Large-scale, deadline- and budget-aware workflows are ideally appropriate for the DRL model due to its environment condition-based decision-making. It produces a highly scalable and effective scheduling solution in the cloud. For Cybershake and Epigenomics workflows with 100 tasks,



(a) Cybershake -100 (b) Epigenomics -100

FIGURE 10. (a-b)-Deadline Violation Ratio against Deadline Factors.

Figure 9(a-b) compares the proposed hyperheuristic-based workflow scheduling approach against the F-ACO, DB-ACO, L-ACO, and ProLiS models, emphasizing the changes in total cost with the increase of deadline factors from 0.05 to 0.25. From the analysis of Figures 9 (a) and (b), it is recognized that the overall cost consumed by all five algorithms degrades as the deadline factor rises, resulting in loosened deadlines on cost efficiency. In comparison to F-ACO, DB-ACO, L-ACO, and ProLiS, the proposed approach accomplishes minimal cost, especially over different deadline constraints. The proposed scheduling strategy demonstrates its overall superiority across different workflow types, preserving competitive cost-reduction performance while effectively handling budget and deadline restrictions. Furthermore, the efficacy of the proposed scheduling algorithm under various deadline factors proved against the four scheduling algorithms that comply with workflow deadline limits.



(a) Cybershake -100 (b) Epigenomics -100

FIGURE 11. (a-b)-Total Cost Performance against Deadline Factors.

In comparison to F-ACO, DB-ACO, L-ACO, and ProLiS across Cybershake and Epigenomics workflows under various deadline factors, the deadline violation ratio of the proposed workflow scheduling algorithm is shown in Figure 10(a-b). Figure 10 depicts that the proposed approach constantly performs better than the other four algorithms in minimizing the deadline violation ratio for cyber shake and epigenomics processes, especially when deadlines are strict and flexible, indicating higher time efficiency. With a task size of 100, Figure 10(a-b) validates that all algorithms are able to guarantee deadline compliance as the deadline factors increase to 0.25, highlighting their viability and robustness for different workflow task operations. Figure 10(a) demonstrates that all five methods reliably accomplish the specified schedule limits in the context of the cybershake workflow. Even under strict deadline constraints, the proposed method shows competitive performance in guaranteeing few deadline violations against larger violations in the other algorithms. Furthermore, the consistent meeting of deadlines in a variety of deadline situations confirms the applicability of these scheduling algorithms in real-time settings, guaranteeing workflow scheduling that is reliable and efficient.

6) COMPARATIVE PERFORMANCE ON DIFFERENT WORKFLOWS

To assess the performance of the workflow scheduling algorithm, the experimental model evaluates different complex scientific workflows in cloud systems such as CyberShake, Epigenomics, Inspiral, and Montage. Moreover, the proposed system dynamically generates the scheduling heuristics by incorporating hyper-heuristics, considering resource availability and workflow states at the time. In consequence, it is essential to handle the different data dependencies and computing demands in processing, such as Montage and CyberShake workflows. The proposed

approach prioritizes satisfying time constraints in the context of strict deadlines for workflow tasks, ensuring effective workflow scheduling with minimal execution cost. This is because several existing scheduling strategies that focus on quick resource provisioning are confronted with increased expenses. As experimented in Table 10, the proposed approach is evaluated with different budget and deadline factors to examine the performance of workflow scheduling algorithms in different scenarios. This comparative evaluation enables the experimental framework to systematically assess the capability of the proposed algorithm on each workflow to produce effective solutions under different levels of constraints. By an incremental step size of 0.1, budget and deadline factors changed around [0.1, 0.5]. Figure 11 illustrates that as the budget factor increases, the normalized cost on all the heterogeneous workflows increases dramatically based on workflow dependency and computation complexity. The performance of the proposed approach varies depending on the workflow and budget factors. The accomplishment of normalized cost is higher for epigenomics and inspiratory, whereas Cyberhske and Montage consume comparatively minimal cost due to their workflow characteristics. Figure 12 shows the average normalized cost of the proposed algorithm, assessed for different deadline factor values between [0.05 and 0.5]. The proposed approach achieves a lower normalized cost on the Cybershake and Montage workflows when the deadline factor is 0.05 compared to other workflows and deadline factors. Among heterogeneous workflows, the proposed approach performs prominently better in terms of cost reduction than DB-ACO [12] when the deadline is 0.5, as compared in Tables 8 and 9. The results of the proposed approach outperform baseline scheduling strategies and DB-ACO for deadline factors between 0.05 and 0.5, with comparatively better enhancements on the CyberShake workflow than the variations in budget factors.

TABLE 10. Normalized Cost Performance of Proposed Algorithm.

Workflow Types	Normalized Cost									
	Budget Factor					Deadline Factor				
	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Cybershake	0.22	0.18	0.17	0.17	0.165	1.72	0.85	0.48	0.32	0.21
Epigenomics	3.52	3.40	3.45	3.44	3.20	4.72	3.96	3.74	3.56	3.20
Inspiral	2.55	2.37	2.31	2.34	2.33	2.09	2.06	2.01	1.13	1.10
Montage	0.21	0.16	0.20	0.22	0.24	3.10	1.54	1.12	0.89	0.64

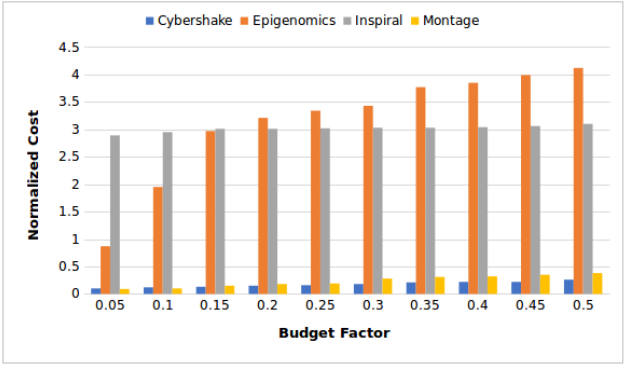


FIGURE 12. Normalized Cost Performance of Proposed Algorithm on Varying Budget Factors.

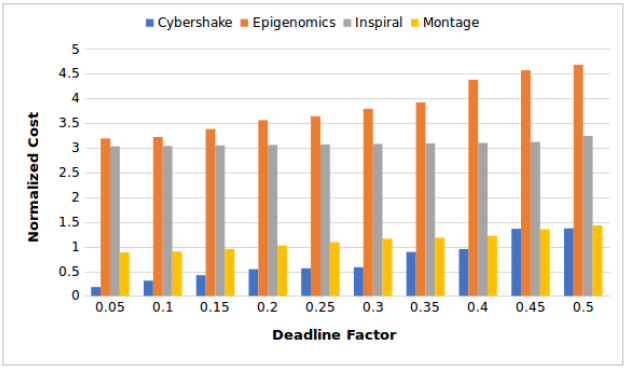


FIGURE 13. Normalized Cost Performance of Proposed Algorithm on Varying Deadline Factors.

7) COMPARATIVE PERFORMANCE ON HEURISTIC ALGORITHMS

With the LLHs, such as PSO, ACO, and SA, the proposed hyperheuristic generation algorithm provides an optimal scheduling solution for scientific workflows that are aware of deadlines and budgets in a variety of experimental configurations. Under different host and VM configurations in the cloud, the proposed algorithm consistently achieves better tradeoffs between deadlines and budget constraints when compared to baseline algorithms like GA, separate PSO, ACO, and SA algorithms. The makespan results provided in this section are obtained from the execution of the Cybershake-100 workflow size and repeated 10 times.

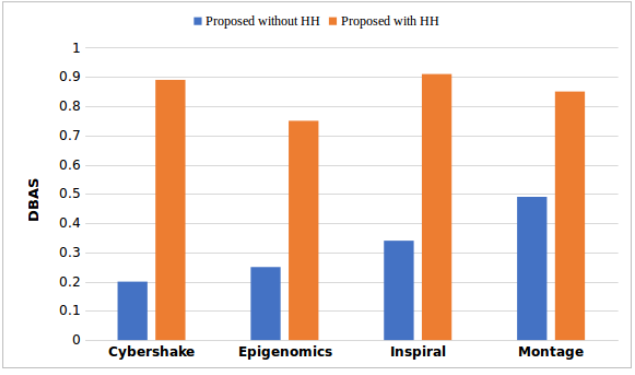


FIGURE 14. Balanced Deadline-Budget Performance of Proposed Algorithm.

Figure 13 compares the proposed workflow scheduling algorithm with hyperheuristics and without hyperheuristics, in which without hyperheuristics is evaluated by a single heuristic algorithm, such as ACO. When the proposed algorithm incorporates hyper-heuristics, the results for DBAS illustrate a prominent increase in adherence when compared to the workflow scheduling without hyper-heuristics. In particular, the proposed algorithm with HH performs remarkably better in terms of compliance across heterogeneous workflows, such as CyberShake, Epigenomics, Inspiral, and Montage, demonstrating its ability to manage time and cost limitations. In the CyberShake workflow, for example, the proposed method with HH shows a significant improvement, highlighting its resilience to changing scheduling requirements. Similarly, the proposed method with HH outperforms other complex workflows, such as Inpiral and Epigenomics, proving its flexibility in handling varying workload complexity. The potential of hyper-heuristics to improve resource utilization and guarantee this improvement demonstrates better adherence to scheduling constraints.

TABLE 11. Experimental Scenario for Heuristic Performance.

Experiment	Number of Hosts	Number of VMs	Number of Cloudlets
Scenario 1	5	10	200
Scenario 2	15	60	800
Scenario 3	25	75	1200
Scenario 4	30	200	1500

In scientific workflow scheduling, deadlines are difficult to meet in settings with limited resources, including small host and virtual machine configurations, as mentioned in Scenario 1. For example, comprehensive search by GA or local refinement methods, such as SA, are computationally demanding and frequently confront deadlines; hence, baseline algorithms, GA, and SA perform poorly in this situation. In contrast, ACO and PSO are highly appropriate to these environments. However, when deadlines are short, the convergence speed of these heuristic models becomes insufficient. LLHs are dynamically selected by the proposed hyperheuristic algorithm according to the workflow environment and objective factors. PSO adjusts the scheduling to mitigate time and avoid higher cost consumption. The results of the experiments show that the hyperheuristic algorithm is effective in situations with limited resources, achieving faster job completion than PSO and ACO, respectively.

Furthermore, it is challenging for baseline algorithms to balance deadlines and budgets for medium-scale host and virtual machine configuration, as modeled in Scenario 2. GA prioritizes one constraint over the other, whereas SA, ACO, and PSO exhibit limitations in terms of efficiently exploring a large solution space under balanced conditions. The proposed hyperheuristic generation algorithm balances the tradeoff by utilizing its predictive state-aware approach and LLHs. ACO finds an effective scheduling path that adheres to deadline restrictions, whereas SA or PSO optimization aims to reduce resource cost without compromising the deadline. According to results from experiments using medium configurations, the hyperheuristic algorithm increases the performance by achieving a minimal makespan compared to PSO and lowers overall workflow execution costs.

TABLE 12. Comparative Heuristic Performance.

	Makespan (s)			
Models	Scenario 1	Scenario 2	Scenario 3	Scenario 4
GA	122.0	220.5	250.3	310.2
SA	119.8	234.7	272.5	328.1
ACO	114.8	217.8	254.6	312.7
PSO	111.9	221.3	241.4	286.8
Proposed Hyperheuristic	107.7	187.3	214.5	248.1

Baseline algorithms, such as ACO and PSO, provide better performance under the settings of higher resources without constraints; however, separate heuristic models frequently confront cost savings and optimal resource utilization. LLHs are optimally integrated by the hyperheuristic algorithm to maximize available resources while maintaining cost-effectiveness. Tasks are assigned to VMs using ACO's capabilities, and allocations are further refined using PSO and SA's budget-sensitive heuristic with priority consideration to reduce the utilization of redundant resources, with optimal tradeoff between the deadline and budget during workflow scheduling. Scheduling workflows becomes highly difficult in heterogeneous configurations as modeled, which are defined by dynamic virtual machine and host performance capabilities. Baseline algorithms are ineffective in changing resource characteristics, which is resolved by the proposed algorithm associated with adaptive reward modeling and hyperheuristic generation in DRL. For example, ACO has trouble handling a variety of task requirements, and GA and PSO have high task execution times due to these algorithms being unable to adapt dynamically to the availability of heterogeneous resources. In this context, the combination of ACO, PSO, and SA in the proposed hyperheuristic algorithm outperforms the scheduling performance by objective-aware fitness computation over state-aware methodology to dynamically modify LLH selection according to workload parameters and virtual machine settings.

VI. CONCLUSION

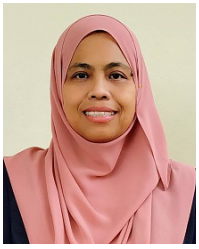
This work addressed the challenges in scheduling scientific workflows in a dynamically changing cloud environment with the optimal balancing of deadlines and budget. The proposed system presented a predictive state-aware deep reinforcement learning framework that incorporates hyper-heuristic generation for optimal workflow scheduling in order to address these challenges. In order to accurately predict the states and represent the environment, the proposed system applied an MGAN to capture the complex interactions between workflow tasks and cloud resources. Additionally, using DQN-based hyper-heuristic generation improves

decision-making in scheduling that is aware of deadlines and budgets, guaranteeing effective and flexible performance even in unpredictable and changing conditions in the cloud. Thus, the proposed method ensured an effective solution that enhances workflow task execution and scalable and efficient workflow scheduling in cloud environments. Furthermore, the experiments are conducted for heterogeneous scientific workflows, such as Cybershake, Epigenomics, Inspiral, and Montage, under various experimental settings and deadline-budget factors, proving the performance improvement and adaptability of the proposed workflow scheduling algorithm in the cloud.

REFERENCES

[1] M. S. Jawed and M. Sajid, "A comprehensive survey on cloud computing: Architecture, tools, technologies, and open issues," *Int. J. Cloud Appl. Comput.*, vol. 12, no. 1, pp. 1–33, Sep. 2022.
[2] N. Garg, Neeraj, M. Raj, I. Gupta, V. Kumar, and G. R. Sinha, "Energy-efficient scientific workflow scheduling algorithm in cloud environment," *Wireless Commun. Mobile Comput.*, vol. 2022, no. 1, pp. 1–12, Jan. 2022, doi: 10.1155/2022/1637614.
[3] M. Farid, H. S. Lim, C. P. Lee, and R. Latip, "Scheduling scientific workflow in multi-cloud: A multi-objective minimum weight optimization decision-making approach," *Symmetry*, vol. 15, no. 11, p. 2047, Nov. 2023, doi: 10.3390/sym15112047.
[4] M. Menaka and K. S. S. Kumar, "Workflow scheduling in cloud environment–challenges, tools, limitations & methodologies: A review," *Measurement, Sensors*, vol. 24, Dec. 2022, Art. no. 100436, doi: 10.1016/j.measen.2022.100436.
[5] A. Choudhary, M. C. Govil, G. Singh, L. K. Awasthi, and E. S. Pilli, "Energy-aware scientific workflow scheduling in cloud environment," *Cluster Comput.*, vol. 25, no. 6, pp. 3845–3874, Dec. 2022, doi: 10.1007/s10586-022-03613-3.
[6] X. Zhang, "Optimizing scientific workflow scheduling in cloud computing: A multi-level approach using whale optimization algorithm," *J. Eng. Appl. Sci.*, vol. 71, no. 1, Dec. 2024, doi: 10.1186/s44147-024-00512-9.
[7] T. Dokeroglu, T. Kucukyilmaz, and E.-G. Talbi, "Hyper-heuristics: A survey and taxonomy," *Comput. Ind. Eng.*, vol. 187, Jan. 2024, Art. no. 109815, doi: 10.1016/j.cie.2023.109815.
[8] E. N. Alkhanak and S. P. Lee, "A hyper-heuristic cost optimisation approach for scientific workflow scheduling in cloud computing," *Future Gener. Comput. Syst.*, vol. 86, pp. 480–506, Sep. 2018, doi: 10.1016/j.future.2018.03.055.
[9] C. Li, X. Wei, J. Wang, S. Wang, and S. Zhang, "A review of reinforcement learning based hyper-heuristics," *PeerJ Comput. Sci.*, vol. 10, p. e2141, Jun. 2024, doi: 10.7717/peerj-cs.2141.
[10] T. Zheng, J. Wan, J. Zhang, and C. Jiang, "Deep reinforcement learning-based workload scheduling for edge computing," *J. Cloud Comput.*, vol. 11, no. 1, Dec. 2022, doi: 10.1186/s13677-021-00276-0.
[11] H. Xie, D. Ding, L. Zhao, K. Kang, and Q. Liu, "A two-stage preference driven multi-objective evolutionary algorithm for workflow scheduling in the cloud," *Expert Syst. Appl.*, vol. 238, Mar. 2024, Art. no. 122009.
[12] S. Tao, Y. Xia, L. Ye, C. Yan, and R. Gao, "DB-ACO: A deadline-budget constrained ant colony optimization for workflow scheduling in clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 2, pp. 1564–1579, Apr. 2023, doi: 10.1109/TASE.2023.3247973.
[13] Y. Yang, G. Chen, H. Ma, S. Hartmann, and M. Zhang, "Enhancing generalization in genetic programming hyper-heuristics through mini-batch sampling strategies for dynamic workflow scheduling," *Inf. Sci.*, vol. 678, Sep. 2024, Art. no. 120975.
[14] Q. Wu, F. Ishikawa, Q. Zhu, Y. Xia, and J. Wen, "Deadline-constrained cost optimization approaches for workflow scheduling in clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 12, pp. 3401–3412, Dec. 2017.
[15] N. Shafinezhad, H. Abrishami, and S. Abrishami, "An adaptive budget and deadline-aware algorithm for scheduling workflows ensemble in IaaS clouds," in *Proc. 13th Int. Conf. Comput. Knowl. Eng. (ICCKE)*, Nov. 2023, pp. 432–438, doi: 10.1109/iccke60553.2023.10326305.
[16] N. Zhou, W. Lin, W. Feng, F. Shi, and X. Pang, "Budget-deadline constrained approach for scientific workflows scheduling in a cloud environment," *Cluster Comput.*, vol. 26, no. 3, pp. 1737–1751, Jun. 2023, doi: 10.1007/s10586-020-03176-1.
[17] L. Tian, H. Li, J. Huang, H. Zhang, S. Chai, and Y. Xia, "Self-detection and comprehensive learning-based BRO for cloud workflow scheduling under budget constraints," in *Proc. 42nd Chin. Control Conf. (CCC)*, Jul. 2023, pp. 1737–1742, doi: 10.23919/ccc58697.2023.10240068.
[18] P. V. Reddy and K. G. Reddy, "A multi-objective based scheduling framework for effective resource utilization in cloud computing," *IEEE Access*, vol. 11, pp. 37178–37193, 2023, doi: 10.1109/ACCESS.2023.3266294.
[19] A. Mohammadzadeh and M. Masdari, "Scientific workflow scheduling in multi-cloud computing using a hybrid multi-objective optimization algorithm," *J. Ambient Intell. Humanized Comput.*, vol. 14, no. 4, pp. 3509–3529, Apr. 2023.
[20] L. Ye, L. Yang, Y. Xia, and X. Zhao, "A cost-driven intelligence scheduling approach for deadline-constrained IoT workflow applications in cloud computing," *IEEE Internet Things J.*, vol. 11, no. 9, pp. 16033–16047, May 2024.
[21] Y. Gu, F. Cheng, L. Yang, J. Xu, X. Chen, and L. Cheng, "Cost-aware cloud workflow scheduling using DRL and simulated annealing," *Digit. Commun. Netw.*, vol. 10, no. 6, pp. 1590–1599, Dec. 2024.
[22] L. Ye, L. Yang, Y. Xia, Y. Zhan, and X. Zhao, "Deadline-constrained and cost-effective multi-workflow scheduling with uncertainty in cloud control systems," *J. Syst. Sci. Complex.*, vol. 37, no. 5, pp. 1–26, Oct. 2024.
[23] H. Mikram, S. El Kafhali, and Y. Saadi, "HEPGA: A new effective hybrid algorithm for scientific workflow scheduling in cloud computing environment," *Simul. Model. Pract. Theory*, vol. 130, Jan. 2024, Art. no. 102864.
[24] Z. Sun, Y. Mei, F. Zhang, H. Huang, C. Gu, and M. Zhang, "Multi-tree genetic programming hyper-heuristic for dynamic flexible workflow scheduling in multi-clouds," *IEEE Trans. Services Comput.*, vol. 17, no. 5, pp. 2687–2703, Sep. 2024.
[25] T. Zaki, Y. Zeiträg, R. Neves, and J. R. Figueira, "A cooperative coevolutionary genetic programming hyper-heuristic for multi-objective makespan and cost optimization in cloud workflow scheduling," *Comput. Operations Res.*, vol. 172, Dec. 2024, Art. no. 106805.
[26] Z. Sun, F. Zhang, Y. Mei, H. Huang, C. Gu, B. Qian, and M. Zhang, "Evolving scheduling heuristics for energy-efficient dynamic workflow scheduling in cloud via genetic programming hyper-heuristics," in *Proc. Int. Conf. Intell. Comput.* Singapore: Springer Nature, Jan. 2024, pp. 169–182.
[27] Q.-z. Xiao, J. Zhong, L. Feng, L. Luo, and J. Lv, "A cooperative coevolution hyper-heuristic framework for workflow scheduling problem," *IEEE Trans. Services Comput.*, vol. 15, no. 1, pp. 150–163, Jan. 2022, doi: 10.1109/TSC.2019.2923912.
[28] P. Zhang, Y. Huang, B. Hu, S. Wang, H. Duan, N. Al Moubayed, Y. Zheng, and Y. Long, "Knowing the past to predict the future: Reinforcement virtual learning," 2022, *arXiv:2211.01266*.
[29] Y. Shao, R. Li, B. Hu, Y. Wu, Z. Zhao, and H. Zhang, "Graph attention network-based multi-agent reinforcement learning for slicing resource management in dense cellular network," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10792–10803, Oct. 2021, doi: 10.1109/TVT.2021.3103416.
[30] R. V. Sudhakar, C. Dastagiraiah, S. Pattem, and S. Bhukya, "Multi-objective reinforcement learning based algorithm for dynamic workflow scheduling in cloud computing," *Indonesian J. Electr. Eng. Informat. (IJEEI)*, vol. 12, no. 3, pp. 640–649, Sep. 2024, doi: 10.52549/ijeei.v12i3.5728.
[31] N. Manome, S. Shinohara, and U.-I. Chung, "Simple modification of the upper confidence bound algorithm by generalized weighted averages," 2023, *arXiv:2308.14350*.

AWADH SALEM BAJAHER received the bachelor's degree in computer science (systems and networking) from Universiti Tenaga Nasional, in 2016, and the master's degree in computer science from Universiti Putra Malaysia, in 2018, where he is currently pursuing the Ph.D. degree with the Faculty of Computer Science and Information Technology. His research interests include the Internet of Things, cloud computing, computer networks, workflow and task scheduling, and parallel and distributed computing.



NOR ASILAH WATI ABDUL HAMID (Senior Member, IEEE) received the Ph.D. degree in computer science from The University of Adelaide, in 2008. From 2013 to 2015, she was a Visiting Scholar with the High-Performance Computing Laboratory, George Washington University, Washington, DC, USA. She is currently the Head of the Laboratory of Computational Sciences and Mathematical Physics, Institute for Mathematical Research, University Putra Malaysia (UPM), Selangor, Malaysia. She is also an Associate Professor with the Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, UPM. She has authored or co-authored more than 80 journals and conference proceedings in her research area. Her work was funded by the government and some by industry. Her research interests include parallel and distributed high-performance, cloud, and data-intensive computing. She received the CUDA Teaching Center Award from NVIDIA, in 2015, and established the CUDA Laboratory. She serves as a reviewer for various renowned journals and conference proceedings.

IDAWATY AHMAD received the bachelor's and master's degrees in information science from Saga University, Japan, and the Ph.D. degree in computer networks from University Putra Malaysia. She has been a Lecturer with the Faculty of Computer Science and Information Technology, University Putra Malaysia, since 2000. Her research interests include real-time systems, network protocols, and simulation/modeling, which are core areas in computer science.

ZURINA MOHD HANAPI (Senior Member, IEEE) received the B.Sc. degree in computer and electronic system from the University of Strathclyde, Glasgow, U.K., in 1999, the M.Sc. degree in computer and communication system engineering from Universiti Putra Malaysia (UPM), Malaysia, in 2004, and the Ph.D. degree from Universiti Kebangsaan Malaysia, in 2011. She is currently an Associate Professor with the Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, UPM, where she has been a Lecturer, since 2004. She has authored more than 70 papers in cited journals and conferences in the area of security and wireless sensor networks. Her current research interests include security, routing, wireless sensor networks, wireless networks, distributed computing, and cyber-physical-systems. She is a member of Malaysian Security Committee Research.