SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

Graph Neural Network-Guided Reinforcement Learning with Hybrid Optimization for SLA-Sensitive Multi-Cloud Scheduling

M.Rupasri[1], G.P.S.Varma[2], Indukuri Hemalatha[3]

[1] Research Scholar, Dept. of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, A.P, India.
[2] Professor, Dept. of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, A.P, India.
[3] Professor, Dept. of Information Technology, S.R.K.R Engineering College, A.P, India. E-mail: rupakeval@gmail.com

Abstract
The scheduling of tasks in multi-cloud systems is complex due to workload heterogeneity, resource variability, and the need to meet Service Level Agreement (SLA) requirements. Researchers have found that traditional heuristic and metaheuristic methods fail to balance efficiency, cost, energy, and scalability in dynamic environments. To address these issues, this paper presents a Graph Neural Network (GNN)-assisted Deep Reinforcement Learning (DRL) model enhanced with hybrid optimization for SLA activity scheduling in multi-cloud systems. The structure utilizes workload-aware profiling based on the Workload Intensity Score (WIS), refinement of the workload heat score (WHS), and suitable task-VM mappings using GNN, along with adaptive scheduling via an actor-critic DRL agent. A Differentiable Surrogate Optimizer (DSO) ensures constraints are feasible, while a hybrid fallback method combining quantum firefly optimization (QFO) and an improved genetic algorithm (IGA) optimizes workload allocations during peak stress. Experimental results show that this approach reduces execution time by 30-40%, cuts SLA violations by 60%, increases throughput by 25-30%, and decreases energy consumption by 20-25% compared to baseline algorithms, demonstrating its effectiveness and scalability in multi-cloud environments.

Keywords: Multi-Cloud Scheduling, Graph Neural Networks (GNNs), Deep Reinforcement Learning (DRL), Hybrid Optimization, Service Level Agreement (SLA) Compliance, Energy-Efficient Resource Management

1. Introduction

Cloud computing has become a foundational technology in the digital service of today and has allowed both enterprises and individuals to access computing, storage, and networking resources on demand. In recent years, there has been a significant increase in the use of multi-cloud environments, where resources from various cloud providers are integrated to enhance performance, availability, and cost flexibility [1]. However, despite the apparent benefits of multi-cloud adoption, it also introduces significant complexity in task organization. The reason is that the workloads should be apportioned wisely among the heterogeneous virtual machines (VMs) to balance the execution performance, service level agreement (SLA) conformance, cost performance, scalability, and energy efficiency [2]. The task scheduling of multi-cloud systems has been identified as a non-polar bear optimization problem, making this trade-off a crucial issue. By scheduling tasks intelligently, one can ensure that they are done on the most appropriate VM, and this will reduce make-span, minimize energy consumption and avoid SLA breaches. Older heuristic approaches like round robin and first-come-first-serve have low computational intensity, but they do not address the heterogeneity of the workloads, and as such, they perform poorly in realistic settings [3]. To address these limitations, metaheuristic algorithms such as genetic algorithms (GA), particle swarm optimization (PSO), and the firefly algorithm have been used in scheduling of clouds [4-6]. Such methods are a better improvement of simple heuristics, as they execute directed searching of the solution space, generating near-optimal solutions to medium-sized workloads. Nonetheless, metaheuristics are primarily offline in nature, and their convergence rate becomes a bottleneck when task arrivals occur dynamically and at a large scale. Therefore, they are usually not responsive to real-time variations in workload and abrupt resource shortages. The growing need for real-time flexibility has inspired the use of Deep Reinforcement Learning (DRL) to schedule research. In contrast to the case of static optimization, DRL models adapt to the environment in the form of continuous communication and thus can effectively balance several goals in dynamic situations [7].

Experiments in the past have shown that DRA is capable of minimizing execution time and maximizing the use of resources in both cloud and edge environments [8]. However, DRL solutions also have some problems. First, they can create schedules that violate constraints (e.g., they exceed VM capacity or break SLAs) since DRL does not inherently put bounds on its schedules [9]. Second, DRL can converge slowly or provide suboptimal decisions when the workloads increase and the spaces of action at the state level are extended. Graph Neural Networks (GNNs) have been proposed as an additional solution to DRL in order to represent task-VM relationships as a structured graph. Tasks and VMs can be modelled as nodes in a scheduling environment, and affinities or contention are modelled with edges. GNNs learn all these relations based on message passing and embedding refinement, which is why they can model non-linear relationships that traditional heuristic affinity scores have not considered [10]. Nevertheless, despite the excellent performance of GNNs in many areas of the workflow scheduling and network resource allocation, there has been little integration of the technique with DRL in multi-cloud scheduling. Sustainable energy management in data centers is another obstacle problem. Cloud infrastructures are also one of the most rapidly rising energy consumers globally, and the electricity demand at the data centers annually is over hundreds of terawatt-hours [11]. Algorithms used to schedule that ignore energy implications will lead to the swell-up of operational costs and carbon footprints. Therefore, any current-day scheduling system should not simply strive to work as fast as possible and meet the SLA requirements but aim at energy-conscious and cost-efficient utilization of resources. The multi-cloud scheduling issue can be defined as the scheduling of a group of heterogeneous activities, with unique resource needs and priority degrees, onto a group of heterogeneous VMs and also achieving a number of conflicting aims. These are minimizing execution time and makespan, minimizing SLA of task, optimization of throughput and utilization, minimizing cost per task, load balancing and minimizing energy usage. The ability to address all these goals in concert, particularly when dealing with uncertain and busty workloads, is very challenging [12].

To address these challenges, we suggest a hybrid learning-optimization system, which integrates workload-conscious profiling, graph-based affinity modelling, reinforcement learning and constraint-aware optimizers. In particular, the framework provides the Workload Intensity Score (WIS) to the task profile and the Workload Heat Score (WHS) to the VM profile. These measurements can be used to record the workload demand and VM stress levels, which are then used to allocate intelligently. An actor-critic DRL agent is used to produce an adaptive scheduling policy, and a GNN module is used to refine task-VM affinities. A Differentiable Surrogate Optimizer (DSO) is combined in order to make allocations viable. Also, when workloads are high or the output of DRL is infeasible, a hybrid fallback mechanism is used that consists of Quantum Firefly Optimization (QFO) and an Improved Genetic Algorithm (IGA) to refine solutions to ensure SLA compliance. The key contributions of the work are as follows:

(i) introduce a new GNN-enhanced DRL model used to schedule on multi-clouds, which optimizes the task-VM affinity modelling and can learn adaptable, SLA-sensitive scheduling policies in dynamically changing environments.

(ii) Develop a constraint-enforcing Differentiable Surrogate Optimizer (DSO) and a fallback mechanism of QFO-IGA that corrects infeasible or suboptimal DRL allocations, and make sure that these mechanisms are robust and satisfy the requirements of SLA in the face of workload spikes.

(iii) We show through extensive experimental work that the proposed framework is significantly more effective than the baseline methods (PSO, GA, Firefly, CEQACO, and Round Robin) in terms of performance, with a 30–40% reduction in execution time, as many as 60% SLA violations, and a 20–25% reduction in energy consumption. It also has higher throughput, utilization, and scalability.

The rest of this paper is organized as follows: Section 2 reviews related works on heuristics, metaheuristics, and learning-based schedulers; Section 3 describes the system model and preliminaries; Section 4 details the proposed framework; Section 5 presents the experimental setup; Section 6 reports results and discussion; and Section 7 concludes the paper with future research directions.

2. Literature Survey

The development of research in cloud and multi-cloud task scheduling has been growing in complexity to scale away basic heuristic approaches and move on to complex hybrid approaches, combining learning and optimisation. All sets of methods, such as heuristic, metaheuristic, reinforcement learning, graph neural networks, and hybrid models, have made their own contribution to the knowledge base but have all been shown to possess inherent weaknesses. This discussion compiles important results under these categories and critically analyses their merits, demerits, and applicability to modern, SLA-sensitive systems that are energy efficient in terms of scheduling.

In [13], the authors presented a deep multi-agent reinforcement learning architecture, FAMASO (Fog-Adaptive Multi-Agent Scheduling Optimisation), of task scheduling in fog clouds. The research presents the shortcomings of the conventional heuristics (e.g., EDF, GA, PSO) and the single-agent reinforcement learning approaches, which tend to be not scalable, flexible, and responsive to changing workloads. FAMASO combines Earliest Deadline First (EDF) prioritisation with Proximal Policy Optimisation (PPO) and Recurrent Neural Networks (RNNs) to achieve decentralised and real-time scheduling in distributed fog networks. The framework is dynamic and adjusts to the situations of the system, captures the temporal

patterns of tasks and provides SLA-sensitive decisions. Workload tests (100-900 tasks) and fog network tests (10-100 nodes) reveal that FAMASO has 95.66% deadline accuracy, 38% energy savings, 56% makespan reduction, 7.8% utilisation, and 70% reduction in scheduler time over baseline techniques. Moreover, the violation rates in SLA also decreased by as much as 80%.

In [14], the authors introduced a method for scheduling tasks in the cloud that utilises graph reinforcement learning combined with a fused attention-corrected linear unit. They describe the scheduling environment as a graph. The graph represents heterogeneous and embedded tasks, virtual machines (VMs), dependencies, and resource constraints. The new GNN containing fused attention ReLU is applied to retrieve the refined feature vectors of both task and VM nodes, which is likely to reduce the vanishing of the gradient and highlight the significant VM-Task feature linkage. These characteristics drive an actor-critic DRL agent, which trains adaptive makespan and cost-balancing scheduling policies. The method in experiments with different cloud conditions is able to reduce the makespan by approximately 10.94% and cost savings by approximately 9.29% of state-of-the-art baselines. The article shows that the application of attention-enhanced GNN and reinforcement learning is effective in scheduling tasks within cloud systems.

In [15], the authors suggested a deep reinforcement learning (DRL) protocol of energy-conscious but QoS-sensitive real-time job scheduling in cloud data centres. They simulate the incoming jobs that come continuously with no predetermined patterns, and they train a DRL agent (with deep Q-learning) to assign jobs to virtual machines (VMs) online. Rewarding ability is a tradeoff between less energy use, time of response, and meeting deadlines. The experiments (simulated environments) demonstrate that the DRL scheduler is more effective in comparison to the baseline heuristics in terms of less energy consumption, shorter response times, and more successful employment under conditions of real-time arrivals. Though the state and action space increase with the size of the system, their approach is flexible and does not require any prior knowledge of workload patterns.

In [16], the authors developed a multi-swarm Particle Swarm Optimisation (PSO) algorithm specifically designed to manage the scheduling of a large number of tasks in sustainable supply chain datacenters. The authors aim to reduce the makespan and response time while enhancing load balancing among heterogeneous machines. In that regard, they formulate a new fitness function to consist of completion time and variance among machines and propose an adaptive inertia weight scheme to dynamically trade off local vs. global search. The researchers apply a new initialisation procedure and a collaborative multi-swarm architecture to improve the diversity of solutions, prevent local optima, and speed up convergence. The procedure is tested on Alibaba datacenter task datasets with different workloads and numbers of machines. Findings indicate uniform improvements in efficiency in scheduling: increased makespan, less response time, and load distribution as compared to base algorithms. The solution shows scalability and strength in an elastic cloud/data centre environment.

In [17], the authors introduced HSA-SHOA, a hybrid optimisation model that is an integration of Simulated Annealing (SA) with the Spotted Hyena Optimisation Algorithm (SHOA) into a resource allocation and task scheduling model in a cloud data centre.The aim is to achieve a balance between exploration and exploitation and not to underload or overload VM and to do the best with performance metrics (makespan, response time, energy usage, and resource utilisation). In the hybrid design, SA assists in overcoming local optima by probabilistic acceptance of poor solutions, and SHOA assists with conducting the global search, using the spotted hyena hunting behavior. The authors test HSA-SHOA through a simulation using CloudSim and comparing it to benchmark bio-inspired approaches. The outcomes showed that HSA-SHOA has better results related to makespan reduction, reduced mean response time, consistent resource usage, and reduced energy use than the baseline methods.

In [18], the authors used a heterogeneity-enhanced GNN + DRL solution to the Flexible Job Shop Scheduling Problem (FJSP) under multi-product and variable-batch production constraints. Their model of the scheduling environment is based on a heterogeneous incidence graph that represents not only the dependencies of operations but also the interaction properties between them and machines. A heterogeneous enhanced GNN (HEGNN) is applied to derive deep feature embeddings of operation nodes, machine nodes, and heterogeneous relationships of the M. The scheduling issue is framed as a Markov Decision Process (MDP), and a Proximal Policy Optimisation (PPO) DRL algorithm is trained in an end-to-end manner to simultaneously determine the operation to be performed and on which machine to perform it. Their method is confirmed both in a real-world (aerospace machining workshop) and benchmark data.

In [19], the authors presented a new heuristic-based inverse reinforcement learning model to address the problem of portfolio optimisation by incorporating domain information, multi-objective reward structure, and policy learning through graphs. Finance-based traditional RL techniques fail because there are few rewards, the dimensions are high, and interpretability is low. In their attempts to deal with them, the authors initially produce a collection of expert heuristics by diversification and correlation constraint, which serves as interpretable expert policy. Next, they use inverse reinforcement learning (IRL) to learn the rewards that align the expert's heuristics with the behaviour learnt. The model employs a multi-objective reward system that dynamically supports both the maximisation of returns and effective risk management. Lastly, they refine a heterogeneous graph policy network that features hierarchical attention capturing inter-stock relationships (e.g., correlations, sector links) and makes the decisions regarding

portfolio allocation. Experiments on real-world financial markets indicate that this framework performs better than state-of-the-art RL and deep learning baselines in risk-adjusted returns and provides better interpretability and risk-return trade-offs.

In [20], the authors suggested a hybrid scheduling scheme and named it QLMOEA/D that combines reinforcement learning (RL) and a decomposition-based multi-objective evolutionary algorithm (MOEA/D) to solve the problem of workflow scheduling in the cloud. The authors break down multi-objective scheduling (e.g., minimisation of makespan and cost) and use Q-learning to instruct the evolutionary search with dynamically adjusted weight vectors and search directions. The RL component also discovers the subproblems or neighbourhoods to focus on, and this enhances convergence and solution diversity. Benchmark workflow experiments indicate that QLMOEA/D has a higher performance in terms of hypervolume (HV), convergence rate, and solution dispersion than the usual MOEA/D and NSGA-II. The method proves that integration of decomposition-based evolutionary strategies and reinforcement learning can be more effective and adaptive in scheduling in cloud workflow environments. The flow of cloud scheduling research indicates a definite shift from the use of mere heuristics to intelligent and hybrid frameworks. Although they all have inherent advantages in terms of their ability to bring about gradual enhancements, none of these paradigms can effectively solve the multi-objective complexity of the present-day multi-cloud environment. Constructive gaps still exist in constraint management, scalability and sustainability. It is clear that the most promising direction involves merging graph-based learning with reinforcement learning and optimisation since it creates flexibility, awareness of the context, and strength. Through the synthesis of the experience of the prior paradigms, the systems of future scheduling will be able to realise real-time SLA compliance, scalability, and energy efficiency in highly heterogeneous and dynamic computing environments.

3. Materials and methods

The architecture illustrates the proposed GNN-informed DRL, which incorporates a mixed-optimisation system for scheduling SLA-sensitive tasks within a multi-cloud environment, as shown in Figure 1. Task input and profiling of workflow commences with analysing workload based on Workload Intensity Score (WIS) and VMs based on Workload Heat Score (WHS). Such profiling measures are input to the Graph Construction Module, which models tasks and VMs as nodes with affinities as edges. Another type of refinement is performed by a Graph Neural Network (GNN), the output of which produces the embeddings that suggest the best task-VM mappings. The products are also processed by the Actor-Critic DRL Scheduler that dynamically assigns resources according to trained policies. To ascertain the SLA compliance and feasibility, a Differentiable Surrogate Optimiser (DSO) verifies decisions. The solutions are refined in case of infeasible or suboptimal allocations by the Hybrid Optimisation Fallback (QFO + IGA) module. Lastly, workloads are spawned on VMs with low execution times, SLA violations, high utilisation, scalability, and energy efficiency.

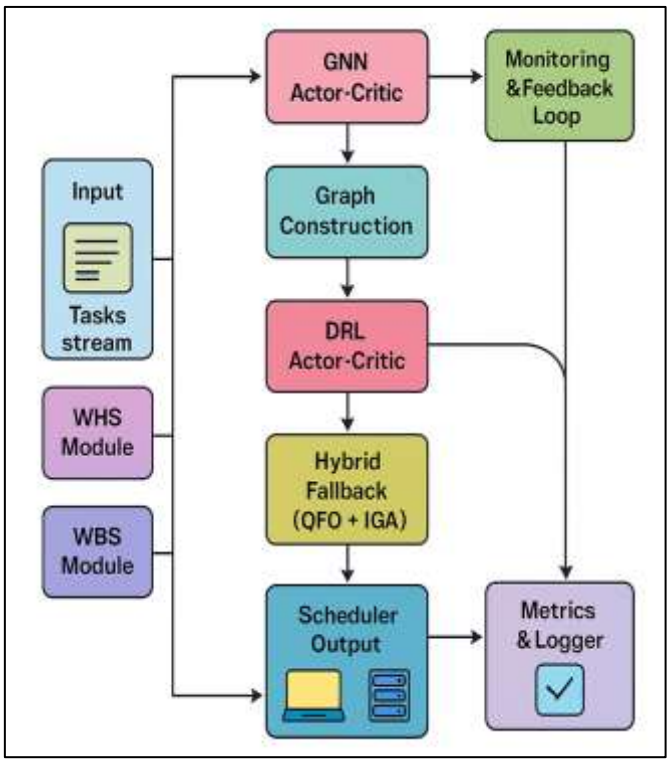

Figure 1: Proposed GNN-guided DRL with hybrid optimization framework

Problem mathematical formulation

Workload heat profiling (WHP) and learning-based decision-making are combined to formulate the scheduling problem of a heterogeneous multi-cloud as a multi-objective optimisation. $T = \{1, \dots, N_t\}$ is the set of tasks arriving at epoch t, and $V = \{1, \dots, M\}$ is the set of virtual machines (VMs). The binary decision variable $x_{i,j}$ represents whether or not task i is allocated to VM_j.

Workload intensity score (WIS) - Task profiling.

Workload intensity score (WIS) is an effective tool that measures the level of computation and priority of every task to intelligently assign tasks to VM. It incorporates several parameters, such as the estimated time of execution, priority of the task, resource requirement, and sensitivity of SLA. The score is formulated as:

$$S_i = \alpha_1 \cdot \frac{\hat{e}_i}{\hat{e}_{max}} + \alpha_2 \cdot P_i + \alpha_3 \cdot \frac{\|\mathbf{r}_i\|_1}{\|\mathbf{r}\|_{1,max}} + \alpha_4 \cdot SLA_i \quad (1)$$

In this case, $\hat{e}_i$ is the estimated time taken to execute task i, P_i is the normalised priority level, r_i is the resource set (CPU, memory, I/O), and SLAi is the sensitivity of task deadlines. The axes are adjustable weights that will guarantee adaptability to a variety of workload combinations. The WIS normalises all elements in [0,1] and gives a unified score of heterogeneous tasks. Resource-intensive or SLA-sensitive tasks have a higher WIS, and thus the scheduler allocates the tasks to powerful and less-loaded VMs.

Workload heat score (WHS) – VM profiling.

The WHS is a dynamic metric, which determines the present degree of hotness or stress on a virtual machine (VM) depending on its history of utilisation and queue state. In contrast to the preceding metrics of load, WHS not only records short-term changes but also long-term workload trends; thus, it is a well-developed predictor of adaptive scheduling. It is expressed as:

$$H_j(t) = \lambda \cdot EMA\left(L_j(t)\right) + (1 - \lambda) \cdot \beta \cdot Q_j(t) \quad (2)$$

In this case, $L_j(t)$ represents the normalised cumulative load (CPU, memory, I/O utilisation) of VM_j at the present time. The task $EMA(L_j(t))$ utilises the exponential moving average (EMA) with a smoothing factor r in order to balance the recent and historical load. The second term is the inclusion of queue length $Q_j(t)$, which describes pending tasks that have a direct effect on latency and SLA violations. Parameters λ and β are used to regulate the weightage of load history and queue pressure. Such a hybrid formulation guarantees that an abrupt increase in workload will be addressed, whereas a brief burst will not be overreacted to. Higher WHS would mean that VM is overutilised or overloaded and a poorer contender in terms of new task assignments. On the other hand, the VMs having low WHS values are regarded as being cooler and can accommodate future high-intensity tasks. The system enables equal distribution of resources, minimised SLA violations, and enhanced fairness among heterogeneous VMs by incorporating WHS in the scheduling system. In this way, WHS is relevant to SLA-sensitive cloud scheduling that is workload-sensitive.

GNN affinity refinement

The GNN affinity refinement module supports better task-to-VM assignment decisions by the modelling of the relational structure between tasks and virtual machines as a bipartite graph. Conventional affinity measures do not use dynamic profiling, e.g., task intensity (WIS) or VM heat (WHS). Nevertheless, cloud systems are very dynamic, and interdependence between tasks and VMs is often non-linear, which rule-based scoring cannot represent. The GNN model overcomes this limitation by training the refined values of the affinities in an iterative message-passing way. The node of the bipartite graph $Gt = (TUV, Et)$ is drawn at scheduling epoch t with T denoting the tasks and V denoting VMs. The raw affinity score ai,j between ei and jeEt is initialised with raw affinity scores computed using WIS and WHS. Tasks have node attributes such as execution time approximations, resource vectors and SLA criticality, whereas VM node attributes are current WHS, historical usage and pricing tier. GNN propagates the information among the tasks and between VMs through message-passing functions:

$$\begin{aligned} m_{i \leftarrow j} &= MLP_e\left(\left[h_i^T \| h_j^V \| a_{i,j}\right]\right) \\ h_i^{T(1)} &= \sigma\left(W_T h_i^T + \sum_{j \in \mathcal{N}(i)} m_{i \leftarrow j}\right) \\ \tilde{a}_{i,j} &= \sigma\left(MLP_o\left(\left[h_i^{T(1)} \| h_j^{V(1)}\right]\right)\right) \end{aligned} \quad (3)$$

In this case, h_i^T and h_j^V represent task and VM embeddings, respectively; s is an activation function like ReLU or GELU; MLP layers are used to convert the concatenated features to elevated-level representations. Output $\tilde{a}_{i,j}$ is an enhanced affinity score that represents local profile information as well as overall workload distribution trends. These fine-grained affinities offer the downstream DRL agent better guidance of task allocation, better workload bursts and resource heterogeneity adaptability. The system incorporates WHS,

WIS, and historical performance into the GNN model to produce context-aware and SLA-sensitive scheduling, which is better than the static heuristics.

Multi-objective scheduling objective

Task scheduling in a heterogeneous multi-cloud environment is a multiobjective optimisation problem, by definition, with multiple performance metrics that need to be optimised at the same time. It is not just to reduce the time of execution but also to be cost-effective, comply with SLA, and have the balanced use of the resources. This study formulates the scheduling decision as optimisation of a composite loss problem that incorporates these conflicting goals. Let $X\{x_{i,j}\}$ represent the assignment of the tasks where $x_{i,j} = 1$ when task i is assigned to VM_j, and 0 otherwise. The tasks are mapped onto one VM each:

$$\sum_{j=1}^{M} x_{i,j} = 1, \forall i \quad (4)$$

while respecting VM capacity constraints:

$$\sum_{i=1}^{N_t} x_{i,j} r_i^{(k)} \leq C_j^{(k)}, \forall j, k \in \{CPU, MEM, IO\} \quad (5)$$

where $C_j^{(k)}$ is the pricing function of VM j . where t^i is the predicted completion time of task i.

$$\mathcal{L}(X) = w_1 \cdot Makespan(X) + w_2 \cdot Cost(X) + w_3 \cdot SLA_{viol(X)} + w_4 . Loadimbalance \quad (6)$$

where w_1, w_2, w_3, w_4 are tunable weights that represent user-defined priorities. This ensures faster job completion by minimizing the execution time of the slowest VM.

$$Makespan(X) = \max_j \left(\sum_i x_{i,j} \cdot \hat{e}_{i,j}\right)$$
$$Cost(X) = \sum_{i,i} x_{i,j} \cdot c_j\left(\hat{e}_{i,j}\right) \quad (7)$$

$$\text{SLA\_viol } (X) = \sum \mathbf{1}\{\hat{t}_i > \text{ deadline }_i\}$$
$$\text{LoadImbalance } (X) = Var_j\left(U_j(X)\right) \quad (8)$$

The multi-objective formulation enables the scheduler to dynamically balance performance, cost, fairness, and SLA adherence. To address it effectively, the refined affinities $\tilde{a}_{i,j}$ of the GNN are incorporated into a Deep Reinforcement Learning (DRL) system, with the loss L(X) being used as the optimization signal. Also, a Differentiable Surrogate Optimizer (DSO) offers feasibility, and metaheuristic refinement (QFO + IGA) offers viable exploration at dynamic workloads. With the combination of optimization constraints and learning-based predictions, the system can be scalable, adaptively, and SLA-consciously scheduled across multiple clouds.

DRL Actor & Critic

The DR component of the scheduling model represents the nomenclature of committing tasks to VMs as a sequence-based choice procedure in the presence of uncertainty. Every scheduling epoch is viewed as an environmental state, and the DRL agent chooses task-VM allocations to optimize the total scheduling loss. A neural network (actor) is used to parameterize the policy, and a second network (critic) is used to measure the quality of decisions. The probability of task-to-VM assignments is created by the actor network. For each task, i have the actor produce a distribution of available VMs:

$$p_{i,j} = \frac{exp\left(\phi_\theta\left(h_i, h_j\right)\right)}{\sum_{j'} exp\left(\phi_\theta\left(h_i, h_{j'}\right)\right)} \quad (9)$$

where θ is a scoring function (a GNN embedding with MLP layers), and 'h_i' and '$h_{j'}$' are the learnt task vectors and feature vectors of VM. $p_{i,j}$ is the probability of assigning task i to VM_j. The critic network approximates the expected return (or negative scheduling cost) of a particular state:

$$V_\psi(s) \approx \mathbb{E}[-\mathcal{L}(X) \mid s] \quad (10)$$

Here, s represents the state of the system, which includes workload intensities, VM heats, and affinities, while $\mathcal{L}(X)$ denotes the multi-objective cost. The critic gives a point of reference on how to reduce variance and contributes to informing the actor about policy changes. Optimization of the policy is done using an advantage function: The policy is optimized using an advantage function:

$$A = -\mathcal{L}(X) - V_\psi(s) \quad (11)$$

which measures the improvement of the chosen action over the expected baseline. The actor's objective is then:

$$\mathcal{J}(\theta) = \mathbb{E}_{p_\theta}[A \cdot log\, p_\theta(a \mid s)] \quad (12)$$

while the critic is updated by minimizing temporal-difference error:

$$\mathcal{L}_V = \left(V_\psi(s) - (-\mathcal{L}(X))\right)^2 \quad (13)$$

with this actor-critic synergy, the system is able to continuously optimize its scheduling policy by taking advantage of workload-conscious affinities of the GNN and constraints of the DSO. Compared to the use of

the purely heuristic systems, the DRL system is dynamic in the sense that it balances exploration (experiencing new task VM allocations) and exploitation (preferring current mappings that have been successful) to enhance compliance on SLA, cost-efficiency, and load balancing in fluctuating cloud systems.

Projection layer – Differentiable surrogate optimizer (DSO).
The DSO helps connect the probabilistic assignment of the DRL actor to the hard feasibility constraints of multi-cloud scheduling. Although the actor produces soft probabilities, $p_{i,j}$, which denote the probability of task iii being assigned to VM_j, those probabilities should be projected into a valid binary assignment matrix, which satisfies both resource capacities and task assignment policies. The DSO achieves this transformation in a differentiable fashion, which enables the end-to-end training of the scheduling framework. The optimization problem solved by the DSO is formulated as:

$$\min_{y_{i,j}\geq 0} \|y - p\|_2^2 \quad (14)$$

subject to:

$$\sum_j y_{i,j} = 1, \forall i, \sum_i y_{i,j} r_i^{(k)} \leq C_j^{(k)}, \forall j, k \quad (15)$$

In this case, $y_{i,j}$ is the projected fractional assignment, $P_{i,j}$ is the probability of the actor, $r_i^{(k)}$ is the resource demand of task i and $C_j^{(k)}$ is the capacity of VM_j. The objective tries to reduce the Euclidean distance between y and p, making sure the projected solution is near the policy of the actor, yet the solution meets hard constraints. Randomized or greedy rounding is used to round the ensuing fractional solution into a binary allocation matrix $x_{i,j}$. Importantly, the projection is differentiable, and thus the gradients can be backpropagated using the DSO, and thus the actor can learn constraint-aware policies. In the event of constraints being violated, Lagrange multipliers are used as penalty signals returned by the DSO and are included in the loss of the actor to hasten the process of arriving at feasible solutions. The DSO helps achieve the desired adaptability of the DRL policy by incorporating the concept of feasibility enforcement directly into the learning process to make sure that the suggested allocations are resource-efficient, compliant with the SLA, and valid.

Hybrid metaheuristic fallback trigger integration.
Although the DRL and Differentiable Surrogate Optimizer (DSO) resolve the majority of scheduling choices, there are some workload scenarios (large task bursts, large risk of SLA violation, and infeasible resource constraints) that require added resilience. A hybrid metaheuristic fallback mechanism is proposed to solve these cases, which is the combination of Quantum-inspired Firefly Optimization (QFO) and Improved Genetic Algorithm (IGA). The fallback is enabled by means of a trigger function:

$$\tau(t) = 1\{\text{ DSO infeasible } \vee \text{ SLA\_risk } > \delta \vee \eta(t) < \epsilon\} \quad (16)$$

and SLA The SLA risk refers to the probability of a deadline violation being predicted, while e(t) measures immediate success in scheduling. When $\tau(t) = 1$, then the DRL-DSO solution is submitted as a warm start as QFO or IGA, which then optimizes the allocation by searching the solution space. QFO uses quantum-inspired encoding to improve the exploration, whereas IGA adds adaptive crossover and mutation to speed up convergence. This hybridization makes sure that the infeasible or suboptimal outputs of DRCs are fixed as quickly as possible. The selective application of metaheuristics to the system results in a balance between adaptability through learning and robustness through search-based methods to ensure compliance with SLA and cost-effectiveness when the system is under uncertain workload conditions. The proposed scheduling framework combines the graph learning, the reinforcement learning and the constraint optimization that will involve well-crafted loss functions and loss feedback to direct convergence. Three complementary components of loss are determined:

Critic Loss: The critic network approximates the target scheduling cost. Its parameters, Ps, are learnt through minimization of the temporal-difference error:

$$\mathcal{L}_V = \left(V_\psi(s) - (-\mathcal{L}(X))\right)^2 \quad (17)$$

In this case, $\mathcal{L}(X)$ is the multi-objective loss of scheduling, which is to be minimized, and the negative sign points to the cost minimization and the reward maximization. This makes the critic give a fair cutoff point on which to assess the actions of the actor.

Actor Loss: This is a method of training an actor network with the advantage function $A = -L(X) - V\psi(s))$.. Policy loss will take the form of:

$$\mathcal{L}_\pi = -\mathbb{E}[A \cdot \log p_\theta(a \mid s)] + \gamma \sum_{j,k} \max\left(0, \sum_i x_{i,j} r_i^{(k)} - C_j^{(k)}\right)^2 \quad (18)$$

The second period has a constraint penalty, g-weighted, that penalizes capacity violations. This makes it viable in the course of exploration.

GNN loss: using supervised learning, the GNN corrects affinities $\tilde{a}_{i,j}$:

$$\mathcal{L}_{GNN} = \sum_{i,j} \left(\tilde{a}_{i,j} - \hat{a}_{i,j}^{obs}\right)^2 + \eta\|\Theta\|^2 \quad (19)$$

where $\hat{a}_{i,j}^{obs}$ is obtained based on performance efficiency.

All of these loss signals allow end-to-end learning: GNN enhances affinity estimations, DRL policy makes use of workload dynamics, and DSO gives feasibility feedback, indicated by multiplied penalties. This synergy guarantees sensitive scheduling of SLA resources in dynamic multi-cloud settings that are efficient.

4. Results and discussion

The proposed scheduling model was constructed and tested in Python 3.10, due to its flexibility, large number of libraries, and compatibility with machine learning structures. In the case of deep learning elements, such as the GNN, DRL actor-critic model, TensorFlow 2.12, and PyTorch 2.0, both were used to take advantage of the capability of using a graphics card. NetworkX 3.1 was used to construct graphs and affinity models, whereas evolutionary optimization modules QFO and Improved Genetic Algorithm were implemented with the help of NumPy and SciPy. The extended version of CloudSimPy, which is part of the Python ecosystem, was used to simulate cloud environments, VM configurations, and workload arrivals. To perform a statistical analysis and preprocess workload attributes, Scikit-learn 1.3 and Pandas 2.0 were used. Result visualization, e.g., using performance graphs and trends, was not only done with Matplotlib 3.7 and Seaborn 0.12. The experiments were carried out on a workstation with an Intel Core i7-11700K CPU, 16 GB RAM, an NVIDIA RTX 3080 graphics card, and Ubuntu 22.04 LTS.

Dataset details

The proposed scheduling framework was tested based on an integrated approach of synthetic cloud workloads and realistic traces obtained from publicly available sources. CloudSimPy was used to create synthetic workloads where tasks were defined with different CPU burst times (1,001-10,000 MI), memory requirements (256 MB-4 GB), storage (1-10 GB) and bandwidth (10-100 Mbps). In order to achieve scalability analysis, the workloads were performed at five levels (100, 200, 300, 400 and 500 tasks). Virtual machines (VMs) were deployed as heterogeneous in terms of number of vCPUs and memory, storage, and RAM (vCPU 2-16, 4 GB-64 GB, 100 GB); and memory (vCPU 16-64 GB). Also, the pattern of arrival of tasks was constructed based on a Poisson distribution to simulate the real-time submission behavior. There were two integrated workload profiling measurements: WIS to classify demand and WHS to compute the stress of VM to demonstrate the dynamic behavior of the system. The given dataset setup made it possible to test the SLA-sensitive scheduling in a comprehensive manner, exposing it to various cloud conditions.

Performance metrics

To obtain a holistic assessment of the scheduling efficiency, it was decided to consider nine performance measures, including the execution efficiency, SLA compliance, resource management, cost, scalability and energy sustainability.

Execution Time (s): It is a measure of how many tasks one can execute in a given time, which is a schedule responsiveness measure.

Makespan (s): The maximum amount of time taken to carry out all the tasks given to a workload batch.

Throughput (tasks/sec): The number of tasks that are successfully completed within one unit of time, and this shows the productivity of the system.

SLA Violation Rate: Percentage of work that does not finish on time or of the required quality.

Resource Utilisation: The percentage of the CPU, memory and bandwidth resources consumed by VMs when compared to the allocated resources.

Cost Efficiency ($/task): Unit cost of execution of each task, including the prices of resources and the penalty avoidance of SLA.

Load Balancing Index: Measures the equitableness in the distribution of workloads among the VMs (the closer to 1, the more balanced it is).

Scalability (tasks/sec): This measures throughput retention to the increase in workload.

Energy Consumption (kWh): This is the sum of power that all active VMs are consuming when performing their tasks..

Parameter Settings

To ensure good and satisfactory repeatable assessment, all algorithms were set with well-chosen parameters. To refine the task-VM affinities, the GNN module was used, with three hidden layers containing 64 neurones each, ReLU activation, a learning rate of 0.001 and an embedding dimension of 128. There was a DRL actor-critic agent, with a discount factor (g) of 0.95, an exploration rate (e) of 0.1, a learning rate of 0.0005, and a batch size of 128. In the case of constraint handling, DSO used a penalty coefficient of 1.2 on infeasible allocations. The QFO algorithm was employed with a swarm of 30, 100 as the maximum number of iterations and b_0 = 1.0 as the attractiveness coefficient. The Improved Genetic Algorithm (IGA) was configured to have a population size of 50, a crossover probability of 0.8 and a mutation rate of 0.05.

Baseline methods (GA, PSO, Firefly, CEQACO and Round Robin) [21-25] were parameterised based on literature so as to provide an impartial performance comparison.
Figure 2 shows the execution time changes with the number of tasks ranging between 100 and 500 for the proposed framework and five baseline algorithms. The suggested approach is the fastest to execute, with the initial performance of 115 seconds (100 tasks) and the middle range of 135 seconds (500 tasks). Conversely, the increase is steeper in the baseline methods. As an example, PSO increases to 240 seconds, GA to 290 seconds and Firefly to 265 seconds. CEQACO has a somewhat better performance in terms of baselines (150-230 seconds), though it still performs worse. Round Robin is the least efficient, and it attains 300 seconds at 500 tasks. This is because the proposed model has superior performance due to its workload-conscious profiling (WIS and WHS) and GNN-informed policy of DRL, which minimises the bottlenecks and allocates tasks effectively. This makes it scalable and robust, resulting in execution that is 30-40 times faster than baseline algorithms.

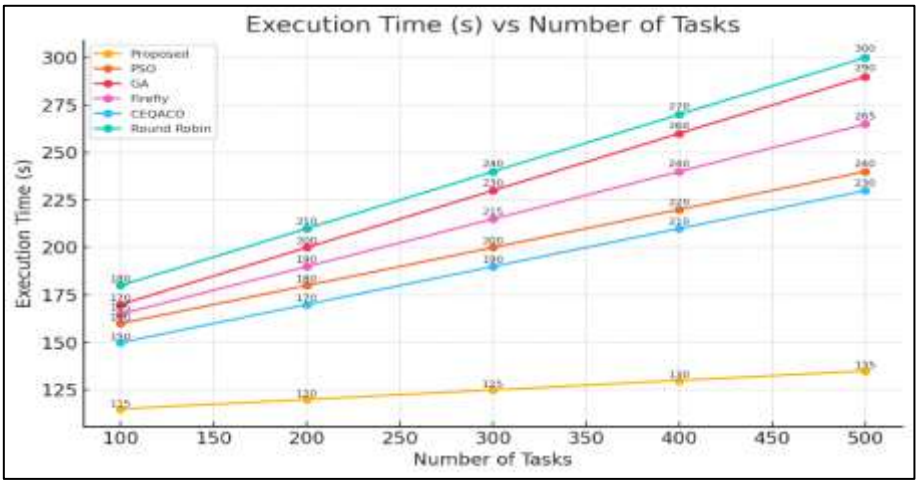

Figure 2: Execution Time vs. Number of Tasks (100–500) for Proposed and Baseline Algorithms.

The SLA violation rate is reported in Figure 3 and is perceived to be growing with the increase in the number of tasks between 100 and 500 between the proposed framework and baseline algorithms. The suggested approach demonstrates excellent SLA compliance, with an increase of slightly between 2.0% and 3.0% as the number of tasks rises from 100 to 500. Comparatively, the baseline algorithms portray very high violation rates. PSO rises from 7.8% to 5.5%, GA rises from 8.5% to 6.0%, and Firefly rises from 7.9% to 5.8%. Although it is better than most baselines, CEQACO still scores 5.2 at 100 tasks and 7.2 at 500 tasks. The round robin method performs poorly, increasing from 6.5 to 9.0. The strength of the suggested framework is the low violation rate, rooted in both the Workload Intensity Score (WIS) and Workload Heat Score (WHS)-based profiling, which emphasises critically important tasks, and in the Differentiable Surrogate Optimiser (DSO) and QFO-IGA fallback mechanisms, which allow the framework to be feasible and respect the SLA even in high-workload situations.

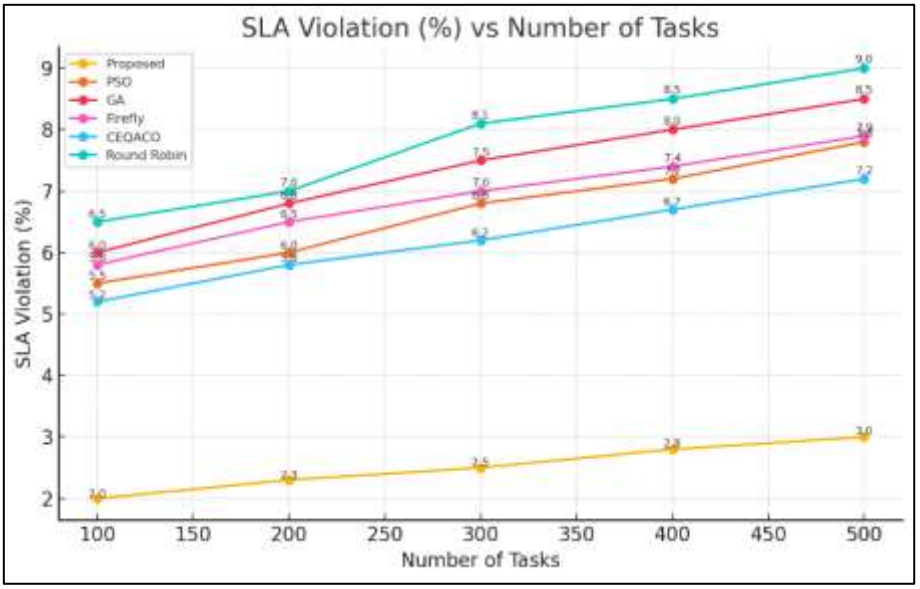

Figure 3: SLA Violation Rate (%) vs. Number of Tasks across Algorithms.

Figure 4 shows the throughput performance of the proposed scheduling framework compared to the baseline algorithms with an increase in the task volume between 100 and 500. The proposed approach maintains the maximal throughput, starting with 100 tasks/sec at 98 tasks and reducing to 93 tasks/sec at 500 tasks with only a slight decrease even with increased workloads. CEGACO is performing relatively better, as it reduces from 82 to 74 tasks/sec, whereas PSO reduces from 80 to 70 tasks/sec, and Firefly reduces from 77 to 68 tasks/sec. GA has a smaller decline that is dropping from 75 to 65 tasks/sec, and Round Robin does the worst, as it is dropping from 70 to 60 tasks/sec. The higher throughput of the proposed model can be explained by its graph-based affinity refining (GNN) and DRL actor-critic scheduling that allow effectively allocating and executing VMs. The framework ensures a higher rate of task completion by preventing congestion and bottlenecks, thus being scalable and reliable as workload gets higher.

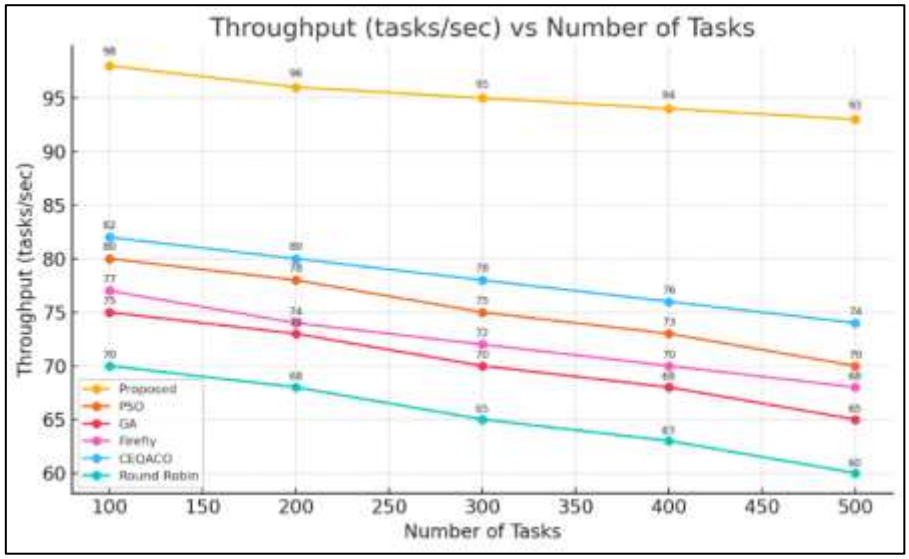

Figure 4: Throughput (tasks/sec) vs. Number of Tasks across Algorithms.

In Figure 5, the use of the proposed frameworks and baseline algorithms is compared in terms of resource utilisation efficiency with varying task numbers of 100, 200, 300, and 500. The best utilisation is always proposed with starting at 87 per cent with 100 tasks and a little higher at 89% with 500 tasks and is highly adaptive to an increase in work. CEQACO looks to be the nearest point and performs better as it increases its score to 81%, whereas PSO remains at 78-80% usage and Firefly remains at 77-79%. GA has had the least utilisation, at 76-78%, and Round Robin has had the poorest, at 74-76% in the range of workloads. This result is because of the good performance of the proposed model, which is based on the Workload Heat Score (WHS)-based VM profiling, which avoids overloading of certain VMs and the idle functioning of others. In combination with GNN-guided affinity refinement, this system dynamically balances workloads, which leads to optimal utilisation of computers, memory, and I/O. The balance reduces wastage of resources and contributes to sustainable performance with increasing task volumes.

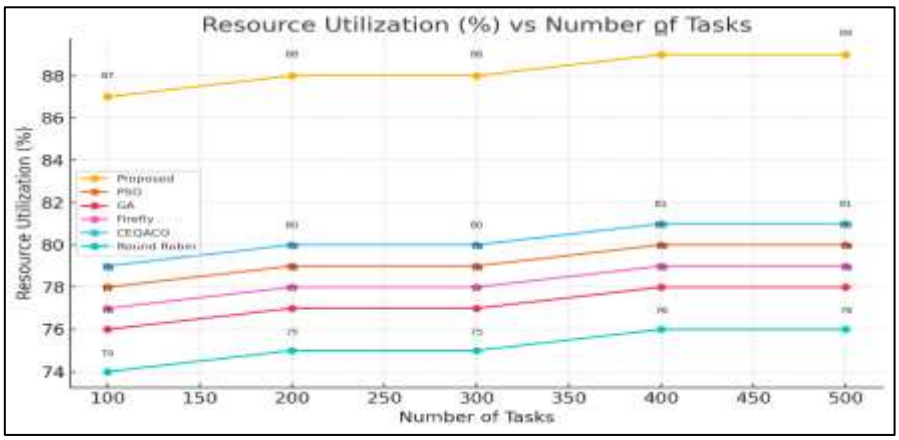

Figure 5: Resource Utilization (%) vs. Number of Tasks across Algorithms.

Figure 6 shows how the proposed framework has a cost efficiency relative to the baseline algorithms as the number of tasks performed is raised between 100 and 500. The suggested approach indicates the least and the most constant cost, where it begins with 0.74 per task in 100 tasks, and then it increases marginally to 0.77 per task in 500 tasks. In comparison, all algorithm bases are more costly. CEQACO tops them, with a

range of between 1.00 and 1.08, then PSO with 1.05-1.14 and Firefly with 1.12-1.19. GA is also more expensive, with an increase of $1.15 to 1.25, whereas Round Robin has the maximum costs, whereby it has risen by 1.20 to 1.30 per task. The enhanced cost efficiency of the proposed framework is due to the compliance with SLA (minimising penalties), constraint-aware DRL scheduling enabled by DSO, and effective use of VM. All these result in a reduction of unnecessary resource consumption, and tasks are only performed on the most appropriate machines, thus cutting the cost per task drastically.

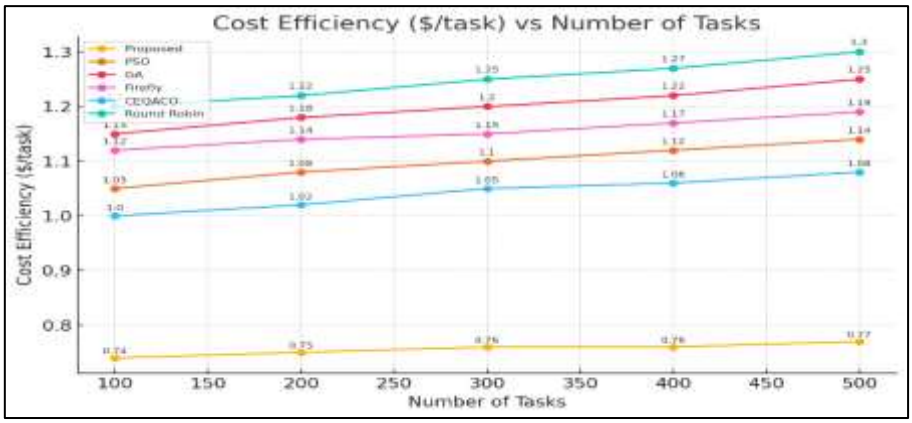

Figure 6: Cost Efficiency ($/task) vs. Number of Tasks across Algorithms.

Figure 7 contrasts the performance of the proposed framework regarding baseline algorithms in terms of makespan measure as the quantity of tasks in the system grows by 100 and 500. The suggested method always produces the shortest makespan with a starting point of 125 seconds and a small increase to 135 seconds with 100 and 500 tasks, respectively. Baseline algorithms, on the other hand, have very long completion times. CEQACO is the most suited between the baselines, with the makespan increasing between 160 and 195 seconds. PSO has a follow-up, rising up to 210 seconds, and Firefly has 180 to 220. GA is less fast, and it escalates between 190 and 230 seconds, and Round Robin has the worst performance with a rise of 200 to 240 seconds. The better makespan improvement of the suggested approach is due to its hybrid scheduling approach, with GNN-based DRL allocations offering effective task-to-VM mappings and the QFO-IGA fallback mechanism optimising allocations in case of heavy loads. This guarantees a quick completion of overall tasks as well as work balance.

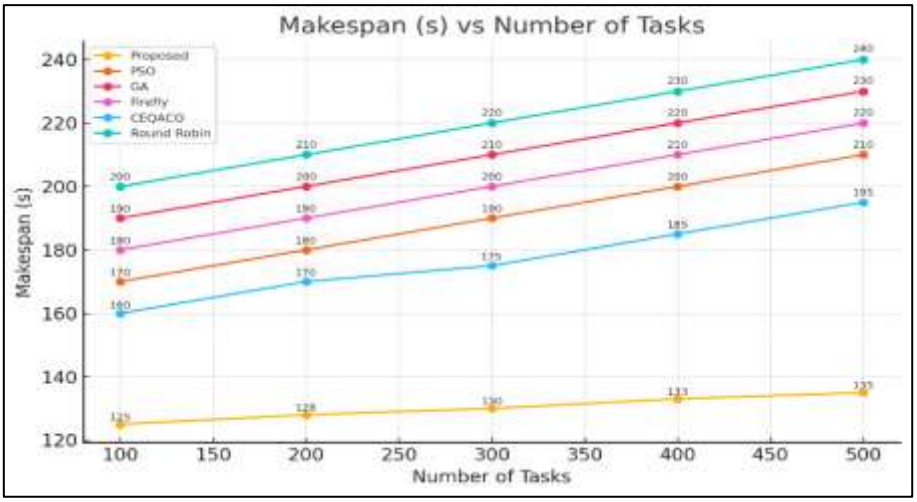

Figure 7: Makespan (s) vs. Number of Tasks across Algorithms.

Figure 8 shows the load balancing index of the proposed framework relative to the baseline algorithms as the number of tasks increases between 100 and 500. The proposed approach attains the utmost and the most coherent fairness, with values ranging between 0.91 and 0.93, which shows efficient allocation of the workload among VMs. CEGACO is compared to be doing relatively better, as it increases from 0.79 to 0.81, and PSO has the range of 0.77-0.79. Firefly is near to 0.75-0.77, and GA is slightly lesser at 0.74-0.76. Round Robin has the poorest performance with a range of 0.71-0.73, which puts much emphasis on its ineffectiveness in task balancing. The proposed method is highly balanced with a balancing index because its VM profiling is based on the Workload Heat Score (WHS) and refined with GNN affinity to ensure that the machines are neither overloaded nor underused. By having an equitable allocation of tasks, the system will achieve fairness and throughput, preventing bottlenecks due to growing workloads.

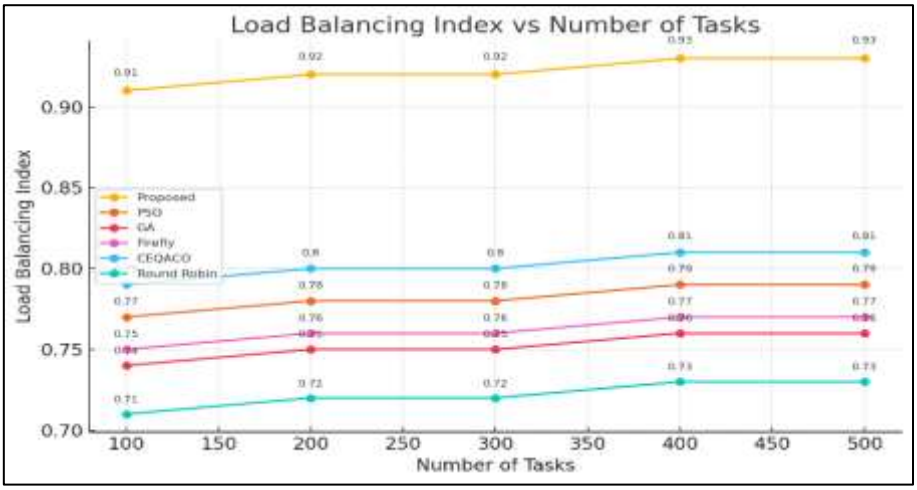

Figure 8: Load balancing index vs. Number of tasks across algorithms

Figure 9 shows the performance of a proposed framework in terms of scalability alongside the performance of the baseline algorithms with increasing task loads ranging between 100 and 500. The proposed methodology has a good scale throughput, reducing by only a small margin (92 tasks/sec at 100 tasks and 88 tasks/sec at 500 tasks) due to the large workload increase, which is a sign of high resilience to workload scale. Baseline algorithms, on the other hand, degenerate more rapidly. CEQACO decreases from 74 to 70 tasks/sec, PSO from 72 to 68 and Firefly from 69 to 65. GA is not doing as well, dropping down to 63, whereas Round Robin is the least scaled, with a drop of 62 to 58 tasks/sec. The higher scalability of the suggested framework can be described by the fact that its actor-critic DRL scheduling allows dynamically adapting the policies as the number of tasks grows, whereas GNN-based task-VM affinity modelling ensures that the framework remains efficient at scale. Such flexibility guarantees that even with a large workload, throughput will be high as compared to the degraded static heuristics.

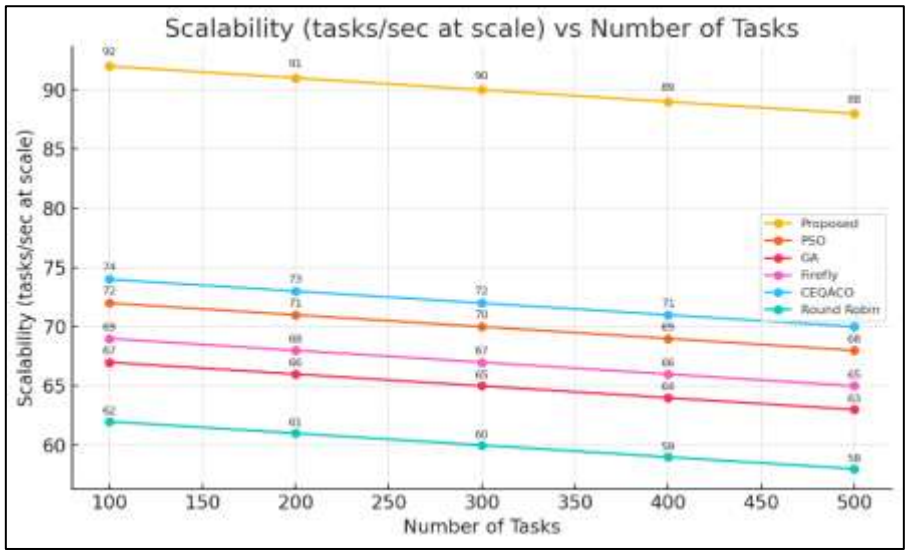

Figure 9: Scalability (tasks/sec at scale) vs. Number of tasks across algorithms

Figure 10 shows the proposed framework and baseline algorithms are compared regarding energy usage depending on the task volume that grows from 100 to 500. The suggested approach exhibits the most energy-efficient operation, with the initial cost of 440 kWh for 100 tasks, increasing slightly to 460 kWh for 500 tasks. By comparison, the energy consumption of all baseline algorithms is much higher. CEQACO consumes the highest energy at 570-600 kWh, followed by PSO at 590-620 kWh and Firefly at 610-640 kWh. GA is a higher consumer of energy, with 630–660 kWh of energy usage, and the round robin is the highest consumer, with an increase from 660 to 690 kWh. The high energy efficiency of the proposed framework is attributed to its shorter execution time, workload-aware profiling, and load-balanced scheduling, which minimize the energy waste of idle time and resource thrashing. By assigning tasks to the most suitable VMs and maintaining SLA adherence, the system reduces unnecessary energy consumption.

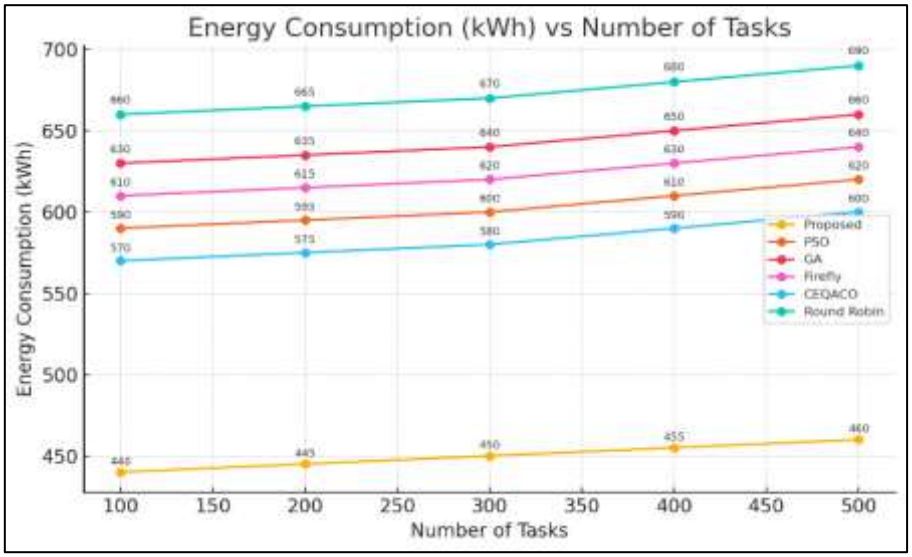

Figure10: Energy consumption (kWh) vs. Number of tasks across algorithms

5. Conclusion

In conclusion, this paper describes a GNN-directed DRL framework with hybrid optimization for scheduling SLA-sensitive tasks within multi-cloud setups. Unlike heuristic and metaheuristic methods, which struggle with dynamic workloads, and standalone DRL models, which risk invalid allocations, the proposed algorithm combines graph-based affinity learning, reinforcement adaptability, constraint-based optimization, and hybrid metaheuristic fallback models. The WIS and WHS profiling enable intelligent task-to-VM mapping, while the DSO layer guarantees SLA feasibility and constraint satisfaction. Comparisons involving workloads of 100-500 tasks show consistent high performance of this framework, with shorter execution times (115-135 s), higher throughput (93-98 tasks/sec), fewer SLA violations (2.0-3.3%), similar utilization rates (87-89%), and markedly lower energy use (440-460 kWh). These improvements highlight the framework's capability for scalable, cost-effective, and energy-efficient scheduling that maintains SLA compliance. Future research will extend this method to multi-agent DRL systems for decentralized decision-making in large-scale edge-cloud environments and explore transfer learning to enhance generalisation across diverse workload distributions.

References

1. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, vol. 25, no. 6, pp. 599–616, Jun. 2009, doi: 10.1016/j.future.2008.12.001.
2. A. Singh and D. Chana, "QoS-aware autonomic resource management in cloud computing: A systematic review," IEEE Transactions on Cloud Computing, vol. 4, no. 4, pp. 458–471, Oct.–Dec. 2016, doi: 10.1109/TCC.2014.2350475.
3. H. Liu, "A novel Round Robin based scheduling algorithm in cloud computing," in Proc. Int. Conf. Cloud Computing and Intelligence Systems, Beijing, China, 2011, pp. 64–68, doi: 10.1109/CCIS.2011.6045027.
4. X. Xu, C. Wu, L. Chen, Z. Li, and J. Li, "A genetic algorithm for task scheduling in cloud computing," Future Generation Computer Systems, vol. 86, pp. 502–510, Sep. 2018, doi: 10.1016/j.future.2018.04.003.
5. Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in Proc. IEEE Int. Conf. Evolutionary Computation, Anchorage, AK, USA, 1998, pp. 69–73, doi: 10.1109/ICEC.1998.699146.
6. X. S. Yang, "Firefly algorithm, stochastic test functions and design optimization," International Journal of Bio-Inspired Computation, vol. 2, no. 2, pp. 78–84, 2010, doi: 10.1504/IJBIC.2010.032124.
7. Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017, doi: 10.1109/COMST.2017.2745201.
8. L. Chen, X. Li, Y. Xu, H. Wu, and M. Xu, "Deep reinforcement learning for task scheduling in cloud computing," Neurocomputing, vol. 416, pp. 117–125, Nov. 2020, doi: 10.1016/j.neucom.2020.07.008.
9. J. Li, Z. Tang, Z. Zhu, and L. T. Yang, "Constraint-aware resource scheduling in cloud environments," Journal of Cloud Computing: Advances, Systems and Applications, vol. 10, no. 1, pp. 1–15, Jan. 2021, doi: 10.1186/s13677-020-00228-1.

10. Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 1, pp. 4–24, Jan. 2021, doi: 10.1109/TNNLS.2020.2978386.
11. A. Shehabi et al., "United States data center energy usage report," Lawrence Berkeley National Laboratory, Berkeley, CA, USA, Tech. Rep. LBNL-1005775, Jun. 2016. [Online]. Available: https://eta.lbl.gov/publications/united-states-data-center-energy
12. A. Arabnejad, J. G. Barbosa, and R. Prodan, "Low-time complexity budget–deadline constrained workflow scheduling on heterogeneous resources," Future Generation Computer Systems, vol. 55, pp. 29–40, Feb. 2016, doi: 10.1016/j.future.2014.12.021.
13. Nagabushnam, G., Kim, K. H., & Choi, Y. (2025). FAMASO: fog-adaptive multi-agent scheduling optimization. Cluster Computing, 28(8). https://doi.org/10.1007/s10586-025-05588-3. Extract summary of article of 150 words.
14. Yang, X., Wang, Z., Li, J. et al. Fused attention rectified linear unit–empowered graph reinforcement learning for task scheduling in the cloud. Peer-to-Peer Netw. Appl. 18, 282 (2025). https://doi.org/10.1007/s12083-025-02105-6
15. Yan J, Huang Y, Gupta A, Gupta A, Liu C, Li J, Cheng L (2022) Energy-aware systems for real-time job scheduling in cloud data centers: A deep reinforcement learning approach. Comput Electr Eng 99:107688.
16. Liu Q, Zeng L, Bilal M, Song H, Liu X, Zhang Y, Cao X (2023) A multi-swarm pso approach to large-scale task scheduling in a sustainable supply chain datacenter. IEEE Trans Green Commun Netw 7(4):1667–1677
17. Iyappan P, Jamuna P (2023) Hybrid simulated annealing and spotted hyena optimization algorithm-based resource management and scheduling in cloud environment. Wireless Pers Commun 133(2):1123–1147
18. Peng, Y., Lyu, Y., Zhang, J., & Chu, Y. (2025). Heterogeneous graph Neural-Network-Based scheduling optimization for Multi-Product and Variable-Batch production in flexible job shops. Applied Sciences, 15(10), 5648. https://doi.org/10.3390/app15105648
19. Zhang, Wenyi & Jia, Renjun & Wang, Yanhao & Cheng, Dawei & Zhao, Minghao & Chen, Cen. (2025). Enhancing Portfolio Optimization via Heuristic-Guided Inverse Reinforcement Learning with Multi-Objective Reward and Graph-based Policy Learning. 9483-9491. 10.24963/ijcai.2025/1054.
20. Xue, F., Wen, J., Wang, P., Fan, W., Geng, Y., & Dong, T. (2025). "A decomposition-based multi-objective evolutionary algorithm with reinforcement learning for workflow scheduling in cloud computing environment." Cluster Computing, 28(10). https://doi.org/10.1007/s10586-025-05369-y
21. J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proc. IEEE Int. Conf. Neural Networks, vol. 4, pp. 1942–1948, 1995.
22. J. Holland, Adaptation in Natural and Artificial Systems. Ann Arbor, MI: Univ. of Michigan Press, 1975.
23. X. S. Yang, "Firefly algorithms for multimodal optimization," in Proc. Int. Symposium Stochastic Algorithms, Berlin, Springer, 2009, pp. 169–178.
24. Z. Cai, W. Gong, C. Ling, and H. Sun, "Chaotic quantum-behaved ant colony optimization for multiobjective job scheduling," Applied Soft Computing, vol. 12, no. 9, pp. 3136–3147, 2012.
25. R. Jain, The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. New York, NY, USA: Wiley, 1991.