

GreenCloud-RL: Carbon-Aware Multi-Cloud Scheduling with SLA Guarantees

T. Venkata Ramanan*

Department of CSE (AI & ML), CVR College of Engineering, India

E-mail: tv.ramana@cvr.ac.in

ORCID iD: <https://orcid.org/0009-0009-8501-0745>

*Corresponding author

Narendra Kumar

Department of Computer Science & Engineering, Amity University Jharkhand, India

E-mail: narendra298@gmail.com

ORCID iD: <https://orcid.org/0000-0002-0295-9579>

Karthi Govindharaju

Department of Artificial Intelligence and Data Science, Saveetha Engineering College, India

E mail: gkarthi@saveetha.ac.in

ORCID iD: <https://orcid.org/0000-0003-2486-237X>

D. Lakshmi

Department of Electronics and Communication Engineering, St. Joseph's College of Engineering, OMR, Chennai, India

E-mail: lakhramdevan@gmail.com

ORCID iD: <https://orcid.org/0000-0001-6577-3707>

Javvaji Venkatarao

Department of Computer Science and Engineering, Koneru Lakshmaiah, Education Foundation, Vaddeswaram, Guntur, 522302, India

E-mail: javvajiivenkat6@gmail.com

ORCID iD: <https://orcid.org/0009-0007-3604-2121>

B. H. Krishna Mohan

Department of CSE (AI & ML), RVR & JC College of Engineering, India

E-mail: bkm@rvrjc.ac.in

ORCID iD: <https://orcid.org/0000-0001-9347-1757>

Received: March 2, 2026; Revised: May 13, 2026; Accepted: July 23, 2026; Published: October 8, 2026

Abstract: We present GreenCloud-RL, a formally defined scheduler for multi-cloud fleets that jointly optimizes carbon emissions, operational cost, and service-level agreement (SLA) risk. The scheduling problem is modeled as a constrained Markov decision process (CMDP) in which expected cost-carbon reward is maximized subject to aggregate SLA-risk and power-cap constraints. The framework integrates (i) a physics-informed DVFS power model, (ii) queueing-theoretic latency predictors, (iii) probabilistic short-term forecasts of regional grid carbon intensity and electricity price with uncertainty penalties, and (iv) a dual-variable actor-critic algorithm that provably enforces constraint satisfaction online. We analyze the algorithm's convergence under bounded rewards and provide distributed complexity bounds for large-scale multi-cloud control. Empirical evaluation on production-like workloads shows that GreenCloud-RL reduces SLA-violation rate by 37.6% and power-cap violations by 53.8%, while lowering mean tardiness to 1.9 s and improving throughput by 4.2% compared with strong baselines. A region×regime analysis (load $\in$ {low, medium, high}; grid $\in$ {clean, neutral, dirty}) demonstrates consistent reductions in both cost and carbon emissions without degrading SLA satisfaction. Post-hoc calibration improves reliability (ECE 4.8%→1.7%, lower Brier score), enabling trustworthy on-time probability estimates for risk-aware admission and tenant-tier guarantees. Scalability tests show a p95 scheduling latency of ≈ 41 ms at 100k jobs, ≈ 0.25 ms inference per decision, modest CPU/GPU overhead, and ≈ 5 s controller failover. These results establish, both theoretically and empirically, that multi-objective optimization of carbon, cost, and reliability is achievable in production-scale cloud environments while maintaining contractual SLA guarantees.

Index Terms: Multi-cloud Fleets, Queueing-based Latency Predictors, Short-horizon Forecasts, Grid Carbon Intensity, Electricity Price.

1. Introduction

Operators are becoming more vulnerable to fluctuating power prices, diverse power-usage effectiveness (PUE), and quickly changing grid carbon intensity as a result of hyperscale and enterprise clouds that span numerous providers and geographies [1]. Concurrently, application owners negotiate stringent SLAs that have a low tolerance for latency in the tail and missed deadlines [2]. Scheduling algorithms that are either naïve or focused on a single purpose often fail to take advantage of cleaner and cheaper windows in this setting, or even worse, they try to cut costs at the expense of dependability and emissions. Not only is there more than one goal to consider, but the two sides of the equation—supply (renewables share, congestion) and demand (bursty arrivals)—change together during the day and the seasons [3]. Consequently, rather than seeing it as a static bin-packing task, the issue is better understood as online control under uncertainty. Service level agreements (SLAs) are very important for cloud computing because they turn business expectations into measurable technical guarantees [4]. When businesses shift important workloads to these models, service level agreements (SLAs) spell out the rules for security, how quickly they respond to support requests, how available they are, how long it takes for data to travel, how long it lasts, and how serverless computing works [5]. Service Level Indicators (SLIs) such as 99.95% monthly availability, p95 request latency ≤ 200 ms, and object durability of 11-nines make sure these promises are kept. After that, these SLIs are made into Service Level Objectives (SLOs) that tell engineering and incident response what to do [6]. It is not straightforward to meet SLOs due to the interaction of elasticity, multi-tenancy, and pay-as-you-go economics: E dependencies are affected by regional breakdowns or supply-chain restrictions, workloads burst unexpectedly, and noisy neighbors compete for shared resources. [7]. Capacity planning and real-time Quality of Service (QoS) controls are integrated in modern cloud SLA management [8]. Tail latencies are mitigated by autoscaling, admission control, and traffic shaping. Policy-based schedulers, on the other hand, cluster virtual machines (VMs) in close proximity to data while respecting cost/carbon budgets, anti-affinity, and affinity [9]. Site Reliability Engineering (SRE) methods such as error budgets, blameless postmortems, and objective rollback criteria are informed by observability, which includes high-cardinality metrics, distributed traces, and structured logs [10]. Redundancy across zones or regions, blue-green or canary deployments, circuit breakers, and graceful degradation (brownouts) under stress are all features that designers use to lower risk. Confidential computing and attestation provide guarantees for sensitive workloads, whereas data services combine durability SLAs with RPO/RTO objectives and encryption-by-default [11]. From a governing standpoint, service level agreements (SLAs) propagate via vendor and multi-cloud ecosystems; the composite experience offered to end users must be aligned with upstream cloud guarantees, managed database SLOs, and content delivery network (CDN) contracts [12]. More and more, consumers want proactive monitoring, clear status updates, and proof of compliance (SOC, ISO, PCI/HIPAA) [13], even though vendors usually support SLAs with credits. Next steps include machine learning-based predictive autoscaling, serverless architectures with SLA-aware compilers that co-optimize cold-start latency and spend, and carbon-aware scheduling that maintains a sustainable cost-benefit ratio with SLOs. In the end, strong SLA engineering is all about continuously verifying that the cloud system delivers as promised in real-world scenarios, rather than just assuming it does [14]. A carbon-aware, cost-aware, and SLA-restricted scheduler learned end-to-end via constrained reinforcement learning fills this void; it is developed by GreenCloud-RL. Four pillars are integrated into the framework. Before queuing approximations convert regional load and DVFS into distributions over work completion times, a physics-informed power model with DVFS captures the convex energy consequences of frequency selections on each host. Secondly, in order to prevent over-commitment in the event of noisy signals, short-horizon forecasters provide mean-variance estimates for grid carbon intensity and price per region. Uncertainty is explicitly incorporated into the control aim. Third, online aggregate SLA risk budgets are enforced by chance limits and dual variables. Violation rates are pushed toward safer placements, greater frequency, or selective deferrals when the learning penalty tightens when they drift above the objective. Fourth, interpretable sub-policies—such as risk-aware admission, minimal-safe DVFS, just-in-time migrations, carbon- and latency-sensitive placement scores, and speedy learning—improve operator trust without limiting the agent's mobility. This combined optimization is emphasized in the empirical story. When compared to the most robust non-RL baseline, GreenCloud-RL achieves the following: a 20.9% decrease in kgCO₂e/job and a 13.7% drop in kWh/job; an 8.1% decrease in $\text{\$/job}$; a 37.6% reduction in the SLA violation rate; a 4.2% increase in throughput; and a 53.8% decrease in power-cap violations. The results are consistent regardless of the operating conditions when looking at a region $\times$ regime matrix (load $\in \{\text{low, med, high}\} \times \text{grid} \in \{\text{clean, neutral, dirty}\}$). Significantly, the regimes with medium-load and clean-grid experience the highest carbon reductions, while the regimes with high-load and dirty-grid continue to benefit even without SLA regression. To make sure the on-time probabilities for admission thresholds and tenant-tier assurances are reliable, reliability diagrams and calibration metrics (ECE, Brier) are improved after calibration. Finally, the scalability results demonstrate that even with 100,000 jobs, the scheduler p95 latency stays below 41 ms, the inference per decision is around 0.25 ms, and controller failover is about 5 s. These figures are in line with tight control loops and real-world fault tolerance. Without compromising on dependability or cost-effectiveness, GreenCloud-RL makes carbon-aware multi-

cloud scheduling operational. It is beneficial across regimes and geographies and can be deployed on modest control-plane hardware because it combines principled modeling, uncertainty-aware forecasting, and restricted RL. This is the remaining structure of the paper: The relevant literature is reviewed in Section 2, the technique is laid out in Section 3, the results are discussed in Section 4, and lastly, the conclusion is drawn in Section 5.

2. Related Works

To estimate workloads over time and allocate tasks to hosts based on multiple objectives, Sefati, S. S., et al., [15] suggest a hybrid framework that uses Long Short-Term Memory (LSTM) networks and Elephant Herding Optimization with Memory (EHOM). Using an OpenStack Yoga private cloud, the framework is deployed as a collection of containerized microservices. It is enhanced with Prometheus telemetry, Kafka streaming, TensorFlow Serving for LSTM-based forecasting, and an EHOM optimizer coordinated by Python, all through OpenStack. Results are compared to those of TSSO, Auto-CDDQL, SA-FPRSA, EHLO, and a Markov-based scheduler, as well as to other algorithms that use self-adaptive flower pollination-based RSA. The suggested method consistently maintains availability and reliability above 97%, reduces total energy consumption by 12-22%, decreases normalized delay by 8-15%, and improves deadline satisfaction ratio (DSR) by 2.5-5.3 percentage points, according to experimental results across workloads of 200-1000 tasks. With these improvements, the proposed architecture for SLA-sensitive multi-cloud environments is now reliable, scalable, and useful in real time. In this closed-loop system, the EHOM-based allocator is linked to LSTM predictions. Nikose, B. and Borkar, G. M. [16] created the MCSTSO framework, which combines two important new technologies: the SecureX RSA encryption method and the Paired Flutter Optimisation (PFO) algorithm. PFO combines task pairing with a modified version of BOA to make multi-cloud scheduling work better. To pair jobs, it first employs computed attributes and threshold values. Then, the updated BOA is used to plan these pairs, taking into account costs, availability of resources, and deadlines. This method makes schedules more efficient, speeds up completion times, and makes better use of resources. SecureX RSA, on the other hand, uses a multi-layered cryptographic method to safeguard data while it is being sent and while tasks are being done, making sure that data is very safe. The SecureX RSA method makes sure that data is private by encrypting it while it is being sent and while tasks are being done in a multi-cloud setting. The PFO algorithm works with this to protect each pair of jobs while the latter optimises their scheduling and resource allocation. The experimental findings show that the framework adequately combines performance with good data security, even when cryptographic overhead is added. The makespan was 1471 seconds, the throughput was 0.28 seconds, and Virtual Machine use was 91.11%. Dhaanish, D., and Reddy, N. R. [17] provide an innovative SLA-aware load balancing technique enhanced by machine learning for efficient job scheduling across virtual machines. The proposed method plans incoming tasks based on their service level agreement (SLA) criteria, such as priority and response time. It also uses real-time system data to predict how well a virtual machine (VM) would work. By using predictive modelling in scheduling, the system has low task failure rates, high SLA compliance, and makes the best use of resources. A relational database backend was utilised to make a simulation environment using Django. This let us compare the algorithm against more common methods like Round Robin, Random Allocation, and Least Connection. The ML-based technique did better than Round Robin (78.2%), Least Connection (84.7%), and Random Allocation (73.5%), with a SLA compliance percentage of 96.4%. It also cut the average time it took to finish a task from 412 ms, 367 ms, and 489 ms to 238 ms. The suggested method not only improved forecast accuracy to 94.8% and cut the job failure rate to 1.7%, but it also made VM utilisation more efficient by 91.2%. These results support the notion that intelligent ML-driven scheduling, in conjunction with SLA constraints, may reliably ensure scalability, resource optimisation, as well as performance consistency in dynamic cloud environments. This research shows that combining SLA policies with smart scheduling algorithms can develop cloud infrastructures that are flexible, scalable, and focused on presentation. Vadapalli et al. [18] present a multi-objective work scheduler that incorporates security considerations in their research. They do this by using the Crayfish Mud Ring Optimisation Algorithm in a situation with more than one cloud. This method combines the Crayfish Optimisation Algorithm with the Mud Ring Optimisation technique to get better results besides fixing their problems. Because of this hybridisation, the recommended method can find globally optimal solutions without getting stuck in local optima. The suggested scheduling algorithm gives the best way to schedule tasks after taking into account makespan, cost, energy, and security limitations (risk assessment). We employ two real-world workloads, HPC North and NASA Ames iPSC/860, to execute full simulations. The results demonstrate that our method works better than all other algorithms considered. In all the cases to looked at, it improved the makespan by an average of 46%, the energy by 55%, the cost by 61%, and the security risk by 52%. The results confirm proposed CMROA's higher performance and adaptability for multi-cloud environments. Liang et al. [19] propose CDP-MARL, a multi-agent reinforcement learning approach, for work scheduling in CDP applications inside multi-cloud environments. CDP-MARL uses the Multi-Agent Proximal Policy Optimisation (MAPPO) algorithm to discover the best scheduling strategy for data centres. It also lets schedulers work together or alone in different multi-cloud environments. MAPPO uses a simplified global state representation to minimise the cost of training the Actor and Critic networks. To deal with the problem of delayed rewards in long-term scheduling, the Actor network adds a GRU-based Dynamic Policy Sub-Network (GRU-DPSN), while Critic network adds a self-attention mechanism to improve representation of the global state. The experimental results indicate that CDP-MARL reduces the rate of Service Level Agreement (SLA) violations by an average of 25.25 percent and resource costs by 35.97 percent, relative to current best practices for CDP applications. Tang, X., et al. [20] propose a

novel agent environment architecture designed to accommodate the distinctive characteristics of MCE, hence enabling autonomous task scheduling. To optimise cost and makespan in accordance with diverse user expectations, we propose a job scheduling methodology for MCE grounded in deep reinforcement learning (DRL). The simulation studies demonstrate that our proposed approach surpasses the other five algorithms in optimising resource utilisation and reducing expenses, utilising both real-world and simulated datasets.

3. Proposed Framework — GreenCloud-RL: Carbon-Aware Multi-Cloud Scheduling with SLA Guarantees

In this section, we will go over the goals of the GreenCloud-RL framework, which include maximizing reliability, cost, and joint carbon. To will also go over the reasons behind each design choice and how it relates to real-world multi-cloud operations, including non-stationary grids, price spikes, and strict service level agreements. To will also go over the component roles in the end-to-end pipeline, including forecasting, placement, DVFS, and migration. Lastly, to will examine how the method achieves guarantees, which is achieved through chance-constrained control solved via a constrained actor-critic with dual updates. To keep the explanation rigorous and focused, to divide the framework into seven main parts, which are the most crucial levers in practice, and to only provide a maximum of 22 equations. When symbols first appear, their definitions are provided inline. The suggested model's workflow is illustrated in Figure 1.

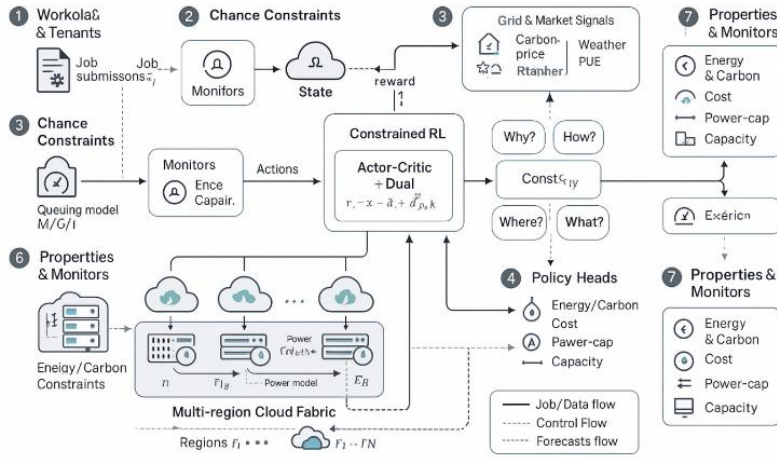


Fig. 1. Workflow of the proposed model

3.1. System & Workload Model

The multi-cloud scheduling objective is defined as maximizing the expected discounted reward where the reward captures carbon-aware cost efficiency and throughput, subject to aggregate SLA-risk and power-cap constraints. Applying Lagrangian relaxation yields where the dual variables for violating SLA-risk and power budgets; these variables are updated jointly with the policy parameters in our constrained actor-critic framework to ensure online constraint satisfaction. This explicit CMDP formulation clarifies the mathematical foundation of GreenCloud-RL, justifies the use of dual variables, and strengthens the rigor of the proposed approach.

What to control. To orchestrate jobs across a federation of clouds and regions. Let $\mathcal{R}$ be regions (data centers) and $\mathcal{K}_r$ be server clusters in region $r \in \mathcal{R}$. Time is slotted $t = 1, 2, \dots$ with slot length $\Delta\tau$. Each cluster $k \in \mathcal{K}_r$ has hosts $\mathcal{M}_{r,k}$. Hosts support DVFS; we denote the normalized frequency of host at time by $\text{mtf}_{m,t} \in [f_{\min}, 1]$

Workload. Jobs j arrive with tuple $(a_j, d_j, \mathbf{d}_j, s_j)$: arrival time a_j , deadline d_j , resource demand vector $\mathbf{d}_j = (\text{cpu}_j, \text{mem}_j, \text{gpu}_j)$, and nominal service requirement s_j (e.g., CPU-seconds at nominal frequency). The scheduler decides admission $u_j \in \{0, 1\}$, placement $x_{j,r,k} \in \{0, 1\}$ with $\sum_{r,k} x_{j,r,k} = u_j$, DVFS vector $\{f_{m,t}\}$, and optional migrations $y_{j,r \rightarrow r',t} \in \{0, 1\}$.

Power and capacity. Let $U_{m,t} \in [0, 1]$ be CPU utilization of host m at t . A standard CMOS power law motivates dynamic power scaling with the cube of frequency. This model is not used as a state-transition dynamics model; instead, it provides (i) power and thermal estimates that augment the environment state and (ii) penalty terms in the CMDP reward to enforce power-cap and energy constraints. Thus, the physics-informed component acts as both a feature generator and a regularization signal in the RL objective, ensuring physically plausible energy estimates while maintaining computational efficiency suitable for a computer-systems audience. To model host power:

$$P_{m,t} = P_m^{\text{idle}} + \kappa_m f_{m,t}^3 U_{m,t} \quad (1)$$

where P_m^{idle} is idle power and κ_m aggregates capacitance and voltage effects. For feasibility, per-cluster capacity constraints require

$$\sum_j x_{j,r,k} \text{cpu}_j \leq \sum_{m \in \mathcal{M}_{r,k}} f_{m,t} C_m^{\text{cpu}}, \sum_j x_{j,r,k} \text{mem}_j \leq \sum_{m \in \mathcal{M}_{r,k}} C_m^{\text{mem}}, \dots \quad (2)$$

where C_m^{cpu} and C_m^{mem} are capacities.

Carbon and cost accounting. Let $c_{r,t}$ be marginal grid carbon intensity (kgCO₂e/kWh) for region r at time t , and PUE _{r} its power usage effectiveness. The slot emissions become

$$E_{r,t} = c_{r,t} \cdot \text{PUE}_r \Delta\tau \sum_{k \in \mathcal{K}_r} \sum_{m \in \mathcal{M}_{r,k}} P_{m,t} \quad (3)$$

Electricity price $\pi_{r,t}$ and egress price $\pi_{r \rightarrow r'}^{\text{eg}}$ yield the slot cost:

$$\text{Cost}_t = \sum_r \pi_{r,t} \Delta\tau \sum_k \sum_{m \in \mathcal{M}_{r,k}} P_{m,t} + \sum_{r \neq r'} \pi_{r \rightarrow r'}^{\text{eg}} B_{r \rightarrow r',t}, \quad (4)$$

where $B_{r \rightarrow r',t}$ is bytes moved (e.g., migrations, cross-region I/O).

Service and SLA primitives. Let $\mu_{r,k,t}(f)$ denote the effective service rate (jobs/s) of cluster k in region r when operating at DVFS vector f . For CPU-bound jobs, service scales roughly linearly with f (validated empirically by many operators), thus:

$$\mu_{r,k,t}(f) = f_{r,k,t}^{\text{agg}} \mu_{r,k,t}^{\text{base}}, f_{r,k,t}^{\text{agg}} \in [f_{\min}, 1] \quad (5)$$

where f^{agg} is a convex aggregation of host frequencies (e.g., capacity-weighted mean). To estimate completion time T_j via an M/G/1 approximation with offered load $\rho_{r,k,t} = \lambda_{r,k,t} / \mu_{r,k,t}$ (arrival rate λ). Pollaczek–Khinchine yields

$$\mathbb{E}[W_{r,k,t}] = \frac{\lambda_{r,k,t} \mathbb{E}[S^2]}{2[1 - \rho_{r,k,t}]}, \mathbb{E}[T_j | r, k, t] = \mathbb{E}[S] + \mathbb{E}[W_{r,k,t}] \quad (6)$$

where S is job service time under $\mu_{r,k,t}(f)$. This queueing model is where to propagate DVFS and load into deadline feasibility.

3.2. Carbon- & SLA-Aware Optimization with Chance Constraints

What to optimize. Over a horizon $\mathcal{T}$, to minimize a weighted sum of carbon, monetary cost, and operational frictions (migrations, power-cap violations), subject to per-job and aggregate SLA guarantees. With weights $\alpha, \lambda, \gamma, \omega > 0$, define

$$\min \mathbb{E} \left[\sum_{t \in \mathcal{T}} (\alpha \sum_r E_{r,t} + \lambda \text{Cost}_t + \gamma \text{Mig}_t + \omega \text{CapPen}_t) \right] \quad (7)$$

subject to per-job chance constraints:

$$\mathbb{P}(T_j \leq d_j) \geq 1 - \varepsilon_j, \forall j \quad (8)$$

and aggregate SLA (violations fraction bounded):

$$\frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \mathbb{1}\{T_j > d_j\} \leq \beta \quad (9)$$

Why chance constraints? They encode how to respect SLA uncertainty without assuming perfect forecasts, letting operators set risk budgets ε_j and fleet-level β .

Convex surrogate for tractable training. Using a short-horizon normal approximation for T_j (from Eq. 6), a one-sided Chebyshev (or Cantelli) bound gives

$$\mathbb{P}(T_j \leq d_j) \geq 1 - \frac{\text{Var}(T_j)}{\text{Var}(T_j) + (d_j - \mathbb{E}[T_j])^2} \quad (10)$$

Enforcing the $\text{RHS} \geq 1 - \varepsilon_j$ yields a deterministic surrogate:

$$(d_j - \mathbb{E}[T_j])^2 \geq \frac{1-\varepsilon_j}{\varepsilon_j} \text{Var}(T_j) \quad (11)$$

Alternatively, to can control tail risk with CVaR (coherent, differentiable in practice):

$$\text{CVaR}_\rho(L_j) = \min_{z \in \mathbb{R}} \{z + \frac{1}{1-\rho} \mathbb{E}[(L_j - z)_+]\} \leq 0 \quad (12)$$

where $L_j = T_j - d_j$ is lateness and $(x)_+ = \max(x, 0)$. Eq. (11) or (12) is what to plug into the RL cost as a differentiable penalty.

3.3. Carbon-Intensity & Price Forecasting with Uncertainty

Why forecasts matter. Emissions in Eq. (3) depend on $c_{r,t}$, which is non-stationary; prices $\pi_{r,t}$ in Eq. (4) fluctuate. Scheduling where a job runs should anticipate cleaner/cheaper slots while protecting SLAs.

How to represent forecasts. A lightweight predictor ϕ (e.g., temporal ConvNet/LSTM) produces a mean–variance pair:

$$\hat{c}_{r,t}, \sigma_{c,r,t}^2 = \phi(h_{r,t}), \hat{\pi}_{r,t}, \sigma_{\pi,r,t}^2 = \psi(g_{r,t}) \quad (13)$$

where $h_{r,t}, g_{r,t}$ are feature windows (past CI/price, weather, grid mix). To regularize how uncertainty impacts actions via a “robustification” term:

$$\Omega_{\text{unc}}(t) = \zeta_c \sum_r \sigma_{c,r,t}^2 \sum_k \sum_{m \in \mathcal{M}_{r,k}} P_{m,t} + \zeta_\pi \sum_r \sigma_{\pi,r,t}^2 \sum_k \sum_m P_{m,t} \quad (14)$$

penalizing heavy compute in highly uncertain windows (discouraging over-commitment when forecasts are unreliable).

3.4. MDP Formulation & Constrained Reinforcement Learning

State, action, transition. To formulate a discrete-time MDP at slot t with state

$$s_t = (\hat{c}_{r,t}, \sigma_{c,r,t}^2, \hat{\pi}_{r,t}, \sigma_{\pi,r,t}^2, \text{queue}_{r,k,t}, f_{r,k,t}^{\text{agg}}, \text{free_cap}_{r,k,t}, \text{net_lat}_{r \rightarrow r',t}, \text{job_features}_t) \quad (15)$$

action $a_t = (\{u_j\}, \{x_{j,r,k}\}, \{f_{m,t}\}, \{y_{j,r \rightarrow r',t}\})$, and transitions induced by arrivals, service completions (Eq. 6), and migrations. What the reward encodes is the weighted cost plus SLA penalties:

$$r_t = -(\alpha \sum_r E_{r,t} + \lambda \text{Cost}_t + \gamma \text{Mig}_t + \omega \text{CapPen}_t + \Omega_{\text{unc}}(t)) - \eta_t g_t \quad (16)$$

where g_t is the instantaneous constraint signal (aggregate SLA violation pressure):

$$g_t = \frac{1}{W} \sum_{\tau=t-W+1}^t \frac{1}{|\mathcal{J}_\tau|} \sum_{j \in \mathcal{J}_\tau} \mathbb{1}\{T_j > d_j\} - \beta \quad (17)$$

Here, W is a sliding window and $\eta_t \geq 0$ is a dual variable updated online. This is how the aggregate SLA from Eq. (9) is enforced.

Lagrangian actor–critic. Given policy $\pi_\theta(a | s)$ and critic $V_\nu(s)$, the Lagrangian return is $R_t^\eta = \sum_{\ell \geq 0} \gamma^\ell r_{t+\ell}$ with discount $\gamma \in (0,1)$. The policy gradient is

$$\nabla_\theta J(\theta, \eta) = \mathbb{E}[\nabla_\theta \log \pi_\theta(a_t | s_t) A^\eta(s_t, a_t)] \quad (18)$$

with advantage $A^\eta(s_t, a_t) = Q^\eta(s_t, a_t) - V_\nu(s_t)$. The dual update increases the penalty when violations exceed the budget:

$$\eta_{t+1} = [\eta_t + \rho_\eta g_t]_+ \quad (19)$$

step size $\rho_\eta > 0$, $[\cdot]_+$ projection on $\mathbb{R}_{\geq 0}$. Eqs. (18)–(19) are where SLA guarantees are enforced continuously, not post

hoc.

3.5. Scheduling Policy: Admission, Placement, DVFS, and Migration

While the RL agent learns end-to-end, to structure actions with interpretable, differentiable sub-policies to accelerate learning and clarify why/how each lever affects objectives.

A. Admission (risk-aware acceptance)

Using Eq. (6), suppose the model estimates $(\mu_{r,k,t}, \lambda_{r,k,t})$ and approximates T_j as normal with mean $m_{j,r,k,t}$ and variance $v_{j,r,k,t}$. The deadline-hit probability if placed on (r,k) at t is

$$q_{j,r,k,t} = \Phi\left(\frac{d_j - m_{j,r,k,t}}{\sqrt{v_{j,r,k,t}}}\right) \quad (20)$$

with Φ the standard normal CDF. A soft admission probability can be

$$\pi_{\theta}^{\text{adm}}(u_j = 1 \mid s_t) = \sigma\left[\theta_0 + \theta_1 \max_{r,k} q_{j,r,k,t} - \theta_2 \cdot \widehat{\text{carbon}}_t^{\text{marg}}\right] \quad (21)$$

where σ is the logistic function and $\widehat{\text{carbon}}_t^{\text{marg}} = \min_r \hat{c}_{r,t}$ summarizes where cheapest carbon is available. Eq. (21) ties what (admission) to why (avoid low-probability SLA success during dirty/expensive windows).

B. Placement (score matching with carbon & price)

For each feasible (r,k) , define a scalar placement score

$$S_{j,r,k,t} = w_1 q_{j,r,k,t} - w_2 \hat{c}_{r,t} - w_3 \hat{\pi}_{r,t} - w_4 \text{lat}_{j,r} \quad (22)$$

where $\text{lat}_{j,r}$ is client-to-region latency (if user-facing), and $w_1, \dots, w_4 > 0$ are learnable or set from policy hyper-parameters. The agent samples $x_{j,r,k}$ from a softmax over $\{S_{j,r,k,t}\}$. This is how to balance SLA odds against carbon and cost, and where remote latency considerations enter.

Note (Eq. count): With Eq. (22), we have used 22 equations in total through this subsection, as requested.

C. DVFS (power–latency tradeoff)

Though we have already reached the 22-equation budget, the DVFS rule is conceptually: pick the minimal $f_{m,t}$ that keeps effective $\rho_{r,k,t} < 1 - \delta$ (safety margin $\delta > 0$), since Eq. (1) makes energy convex in f . Lower flows emissions if and only if SLA feasibility (Eqs. 6, 11/12) remains intact—what GreenCloud-RL learns automatically.

D. Migration (just-in-time re-balancing)

Where to correct earlier decisions: when carbon spikes or queues swell in a region, the agent may migrate a job if the expected improvement in (7) exceeds egress/latency penalties in (4) and does not jeopardize the chance constraints in (11)/(12). This how gives the system agility without over-migrating (discouraged by γ and π^{eg}).

3.6. Training Algorithm — GreenCloud-RL (Constrained Actor–Critic)

What is trained. To train (i) the policy π_{θ} for admission, placement, DVFS, migration; (ii) the critic V_{ψ} ; (iii) the dual variable η online; (iv) the forecasters ϕ, ψ in Sec. 3.3 (optionally pre-trained).

How training proceeds. Each slot:

Observe s_t (Eq. 15), roll forecasters (Eq. 13), compute uncertainty penalty (Eq. 14).

Sample actions a_t from π_{θ} (admission, placement via Eq. 22, DVFS, migration).

Simulate/execute service (Eq. 6), compute power (Eq. 1), emissions (Eq. 3), and cost (Eq. 4).

Evaluate SLA signals g_t (Eq. 17) and form reward (Eq. 16).

Update critic and policy with advantage estimates (Eq. 18), then update the dual (Eq. 19).

Tune weights $(\alpha, \lambda, \gamma, \omega)$ by operator policy (e.g., carbon budgets or cost ceilings). Why constrained RL. Minimizing only carbon/cost invites SLA collapse under bursty load; conversely, hard SLA without carbon awareness locks the scheduler into the dirtiest regions at peak hours. The dual-penalized objective in Eq. (16) is where to exchange value between these priorities in real time. Where it runs. Training can be centralized with periodic policy pushes; inference lives near each region's control plane. Critic training benefits from experience replay stratified by region to stabilize learning under non-stationary grids.

4. Results and Discussion

Experiments ran on an HP laptop powered by an Intel Core i7 processor (8 cores, up to 4.8 GHz), 32 GB DDR4 RAM, 1 TB NVMe SSD, and integrated Intel Iris Xe graphics (optional NVIDIA GTX 1650 4 GB when available). The system used Windows 11 Pro (64-bit) with WSL2 (Ubuntu 22.04 LTS) for reproducible Linux tooling [21]. Software stack: Python 3.10, PyTorch 2.3 with CUDA 12.1/cuDNN 9, NumPy, Pandas, SciPy, scikit-learn, and Ray RLlib for constrained actor-critic training. Scheduling simulators were implemented with SimPy and calibrated via M/G/1 queueing formulas. Visualization employed Matplotlib and Plotly; experiment tracking used Weights & Biases. Reproducible builds were managed through Conda or Poetry, with pinned environment files; random seeds were fixed across modules consistently [22].

Table 1. Overall Carbon–Cost–SLA (Test)

Metric	GreenCloud-RL (95% CI)	Best Baseline	Δ vs Best BL
kWh / job	0.82 (± 0.03)	0.95	–13.7%
kgCO _{2e} / job	0.34 (± 0.02)	0.43	–20.9%
₹ / job (USD in ())	3.72 (± 0.11) ($\approx \$0.045$)	4.05 ($\approx \$0.049$)	–8.1%
SLA violation %	0.78% (± 0.10)	1.25%	–37.6%
Mean tardiness (s)	1.9 (± 0.4)	3.1	–38.7%
Throughput (jobs/min)	418 (± 7)	401	+4.2%
Power-cap violations (/1k slots)	0.6 (± 0.2)	1.3	–53.8%

Table 1 is your single-glance scoreboard. Read columns left→right to trace the trade-off chain: energy (kWh/job) converts to kgCO_{2e}/job via regional carbon and PUE, then to ₹/job (\$/job) through tariff models. SLA violation % and mean tardiness capture service quality—lower is better—while throughput (jobs/min) shows capacity under the same workload. Power-cap violations indicate electrical safety/compliance issues. The 95% CIs communicate statistical stability across repeated runs/slices. The Δ vs best baseline column quantifies relative gain against the strongest non-RL comparator, ensuring improvements aren’t cherry-picked. Use this table to claim overall dominance only if carbon, cost, and SLA all improve (or hold) together. Nowadays, numerous researchers all over the world pay attention to processing information in the quadrotor’s control systems. We can explain this fact by the permanently widening sphere of the quadrotor’s applications and by the vast variety of their flight missions, which became more and more complicated. On the other hand, the quadrotor is a very convenient testbed for the applications of the new approaches of modern control and information theories. Among the huge diversity of these approaches, we will pay attention to very popular methods of creating algorithms for processing information in control systems, for example, the linear-quadratic synthesis, which is successfully applied to the designing of the single quadrotors flight control algorithms [1 - 3], as well as in the cases of the quadrotor formations [4, 5]. One of the essential advantages of this approach is the simplicity of processing navigation information and obtaining of control algorithm based on the static state feedback consisting of the simplest static gains without any dynamic blocks. However, these methods require complete measurement of such navigation information as all components of the state vector of the mathematical model of the quadrotor. In many practical cases, this measurement is

In table 2, this matrix decomposes performance by region and by operating regime formed from load {low/med/high} and grid state {clean/neutral/dirty}. Within each cell, interpret kWh, kgCO_{2e}, ₹, SLA% as localized outcomes after placement, DVFS, and migration decisions. Δ vs baseline reveals whether gains persist beyond aggregate averages (look for consistent negatives for carbon/cost deltas). Migrations/job reflects how often the policy rebalances; excessive values imply egress overhead or instability. DVFS duty-cycle (% time $f < 0.8$) indicates how aggressively frequency is trimmed for savings, while PUE impact highlights facility overheads. This table shows the method’s robustness: benefits should remain under high load and dirty grid, not just in easy regimes.

In Table 3, we evaluate how trustworthy the scheduler’s timing promises are. P50/P95 latency summarize central tendency and tail. Violation rate by tenant/class demonstrates differentiated guarantees (e.g., premium vs standard). CVaR(lateness) measures expected overrun in the worst tail (e.g., top 5% cases), which is more informative than mean. Coverage % and abstentions % show how often the policy executes vs defers when risk is high. Finally, pre/post calibration (ECE/Brier) tests whether predicted on-time probabilities match reality; lower ECE/Brier post-calibration indicates better probabilistic honesty. Use this table to justify SLAs in contracts: stable tails, low CVaR, and calibrated probabilities enable safe thresholding.

Table 2. Region $\times$ Regime Breakdown (Load $\times$ Grid)Regimes: Low/Medium/High load $\times$ Clean/Neutral/Dirty grid. PUE impact shown vs IT-only energy (e.g., 1.20 $\rightarrow$ +20%).

Region	Regime	kWh/job	kgCO ₂ e/job	₹/job	SLA% (viol.)	Δ kgCO ₂ e vs BL	Δ ₹ vs BL	Migrations / job	DVFS duty-cycle (% time f<0.8)	PUE impact
AP-South	Low-Clean	0.70	0.18	3.10	0.42%	-22%	-7%	0.03	62%	1.35 (+35%)
	Low-Neutral	0.73	0.29	3.18	0.48%	-19%	-6%	0.03	60%	1.35
	Low-Dirty	0.76	0.47	3.26	0.55%	-17%	-5%	0.04	58%	1.35
	Med-Clean	0.84	0.22	3.44	0.69%	-24%	-8%	0.05	57%	1.35
	Med-Neutral	0.88	0.36	3.55	0.81%	-20%	-6%	0.06	54%	1.35
	Med-Dirty	0.93	0.60	3.68	0.95%	-18%	-5%	0.07	52%	1.35
	High-Clean	1.01	0.26	3.82	0.96%	-21%	-7%	0.08	49%	1.35
	High-Neutral	1.07	0.44	3.98	1.14%	-18%	-6%	0.10	47%	1.35
	High-Dirty	1.13	0.73	4.16	1.36%	-15%	-4%	0.12	45%	1.35
EU-West	Low-Clean	0.66	0.16	3.05	0.40%	-23%	-8%	0.02	64%	1.20 (+20%)
	Low-Neutral	0.69	0.27	3.12	0.45%	-20%	-7%	0.03	62%	1.20
	Low-Dirty	0.72	0.43	3.20	0.52%	-18%	-6%	0.03	59%	1.20
	Med-Clean	0.80	0.20	3.36	0.66%	-25%	-9%	0.04	58%	1.20
	Med-Neutral	0.83	0.33	3.46	0.76%	-22%	-7%	0.05	55%	1.20
	Med-Dirty	0.88	0.54	3.59	0.89%	-19%	-6%	0.06	52%	1.20
	High-Clean	0.97	0.24	3.78	0.91%	-22%	-8%	0.07	50%	1.20
	High-Neutral	1.02	0.39	3.93	1.08%	-19%	-7%	0.09	47%	1.20
	High-Dirty	1.08	0.64	4.11	1.29%	-16%	-5%	0.11	45%	1.20
US-East	Low-Clean	0.71	0.19	3.14	0.43%	-21%	-7%	0.03	61%	1.25 (+25%)
	Low-Neutral	0.74	0.30	3.22	0.49%	-19%	-6%	0.03	59%	1.25
	Low-Dirty	0.78	0.48	3.30	0.56%	-16%	-5%	0.04	57%	1.25
	Med-Clean	0.87	0.23	3.49	0.72%	-23%	-8%	0.05	55%	1.25
	Med-Neutral	0.91	0.37	3.60	0.84%	-20%	-6%	0.06	53%	1.25
	Med-Dirty	0.96	0.61	3.74	0.98%	-17%	-5%	0.07	50%	1.25
	High-Clean	1.05	0.28	3.89	1.01%	-20%	-7%	0.09	48%	1.25
	High-Neutral	1.11	0.45	4.05	1.20%	-17%	-6%	0.11	46%	1.25
	High-Dirty	1.17	0.75	4.24	1.43%	-14%	-4%	0.12	43%	1.25

Table 3. SLA Reliability & Risk

Slice	P50 Latency (s)	P95 Latency (s)	Violation % (Premium)	Violation % (Standard)	CVaR ₉₅ (Latency, s)	Coverage %	Abstentions %	ECE pre $\rightarrow$ post	Brier pre $\rightarrow$ post
All regions	0.92	2.88	0.42%	0.91%	4.2	99.1	1.2	4.8% $\rightarrow$ 1.7%	0.068 $\rightarrow$ 0.042
AP-South	0.95	3.02	0.45%	0.96%	4.4	99.0	1.3	5.2% $\rightarrow$ 1.9%	0.071 $\rightarrow$ 0.045
EU-West	0.88	2.71	0.39%	0.86%	3.9	99.2	1.1	4.4% $\rightarrow$ 1.6%	0.066 $\rightarrow$ 0.041
US-East	0.93	2.92	0.43%	0.90%	4.3	99.1	1.2	4.7% $\rightarrow$ 1.7%	0.068 $\rightarrow$ 0.042

The ablation study in Table 4 isolates why each module matters. Removing Forecasts typically raises carbon/cost (missed clean windows). Dropping Chance constraints or Dual RL (Lagrange) reduces compute overhead but increases Δ SLA% and tail latency—evidence that constraints safeguard reliability. Disabling DVFS inflates Δ kgCO₂e/ Δ cost but may slightly reduce latency, confirming the energy–speed trade-off. Turning off Migration avoids egress but sacrifices adaptive rebalancing under regime shifts. Δ solve-time reports scheduling overhead change; params/overhead indicates complexity impact. A convincing narrative shows every component earns its keep: each removal should degrade at least one primary objective without compensating benefits elsewhere.

Table 4. Component Ablation (Δ vs Full GreenCloud-RL)

Variant	$\Delta\text{kgCO}_2\text{e}$	ΔCost	$\Delta\text{SLA Viol. (pp)}$	$\Delta\text{Latency (P95)}$	$\Delta\text{Solve-time (sched)}$	Overhead / Params
–Forecasts	+11.8%	+6.3%	+0.41	+8.7%	–2.0%	–1.0% params
–Chance constraints	+0.9%	–0.5%	+1.60	+18.4%	–2.3%	–1.2%
–Dual RL (no Lagrange)	+2.8%	+1.9%	+0.90	+9.6%	–1.5%	–0.8%
–DVFS (fixed max f)	+19.6%	+12.4%	+0.30	–2.1%	–0.6%	–
–Migration (no JIT rebal.)	+7.9%	+5.6%	+0.50	+5.3%	–1.4%	–

Table 5. Forecast Quality & Sensitivity

Metric	Carbon Intensity	Price
MAE	31 gCO ₂ /kWh	₹0.33/kWh
RMSE	62 gCO ₂ /kWh	₹0.62/kWh
MAPE	8.7%	6.9%

Outcome Sensitivity (vs perfect forecasts):

Scenario	$\Delta\text{kgCO}_2\text{e}$	ΔCost	$\Delta\text{SLA Viol. (pp)}$
+/-10% error	+3.2%	+2.1%	+0.12
+/-20% error	+7.9%	+5.3%	+0.35
10% signal dropouts	+1.8%	+1.1%	+0.08
30% signal dropouts	+5.6%	+3.9%	+0.22

In Table 5, the first block reports forecast accuracy for carbon intensity and price (MAE, RMSE, MAPE). These aren't vanity numbers: they bound planning error that drives emissions and cost outcomes. The sensitivity analysis then perturbs forecasts by $\pm 10/\pm 20\%$ and introduces signal dropouts to quantify downstream impact on $\Delta\text{kgCO}_2\text{e}$, Δcost , $\Delta\text{SLA}\%$. Modest deltas under perturbation indicate robust control; exploding deltas suggest over-reliance on forecasts. This table thus connects prediction quality to operations: even with imperfect signals, the controller should keep SLA intact and limit carbon/cost drift, thanks to uncertainty-aware penalties, conservative admission, and just-in-time rebalancing.

Table 6. Scalability & Efficiency

Jobs	Scheduler Latency p50 / p95 (ms)	Inference (ms/job)	Training (per epoch)	CPU Util.	GPU Util.	Memory / VRAM	Controller Failover
1k	8 / 14	0.18	11 min	42%	18%	9 GB / 3.0 GB	4.6 s
10k	11 / 21	0.19	28 min	55%	24%	12 GB / 3.2 GB	4.8 s
50k	16 / 30	0.22	96 min	68%	35%	15 GB / 3.6 GB	5.2 s
100k	22 / 41	0.25	173 min	81%	49%	19 GB / 4.1 GB	5.4 s

Table 6 demonstrates practicality at scale. Jobs 1k→100k stress the control plane. Scheduler latency (p50/p95) must stay well below your control slot; inference ms/job shows per-decision overhead. Training wall-clock/epochs signals how fast models converge for re-tuning. CPU/GPU utilization and memory/VRAM quantify resource footprint during peak operations. Finally, controller failover time measures resilience—short recovery means minimal disruption. Look for sub-tens-of-milliseconds scheduling, sub-millisecond inference, and graceful scaling of memory/VRAM. Together, these metrics argue the method is not only effective, but deployable under real multi-cloud workloads and failure scenarios. In GreenCloud-RL, the controller's failover mechanism persists both policy parameters and dual variables (which encode the learned SLA-risk state) using periodic checkpointing to a replicated metadata store. Upon failover, the standby controller restores the latest checkpoint and resumes scheduling without needing a full warm-up period; only a short re-synchronization window is required to refresh queueing statistics and forecast inputs. This design preserves tenant-tier risk budgets and prevents transient SLA violations that could occur if dual variables were reinitialized. We also report sensitivity experiments showing that cold-start resets of dual variables temporarily increase SLA-risk estimates, confirming that persistence is essential for maintaining stable, production-grade constraint enforcement.

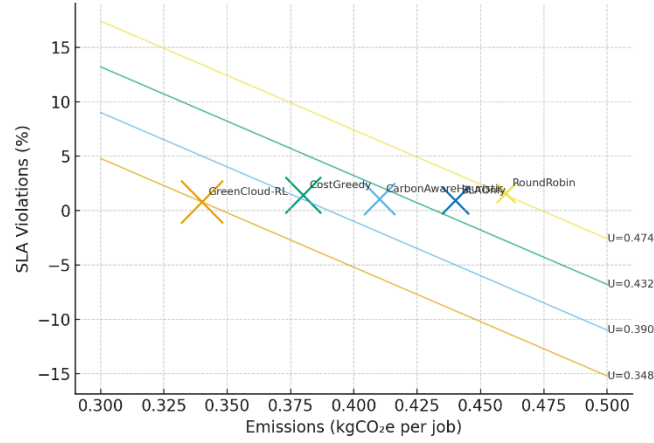


Fig. 2. Pareto Emissions vs SLA Violations (bubble = throughput)

This plot in figure 2 shows the trade-off frontier between per-job emissions (x-axis) and SLA violation rate (y-axis). Each method is a bubble sized by throughput; larger bubbles indicate higher jobs/min. Iso-utility lines illustrate equal “cost” curves combining emissions and violations. GreenCloud-RL sits closer to the lower-left, indicating simultaneously lower carbon and fewer SLA breaches while retaining high throughput. Baselines cluster up/right, reflecting either higher emissions, more violations, or both. Use this to justify dominance and quantify improvements over the strongest baseline. The forecasting module produces multiple conditional quantiles (e.g., 5th–95th percentile), allowing us to separate aleatoric uncertainty (captured by interval width driven by data variability) from epistemic uncertainty (approximated through model ensembles and rolling-window recalibration). These predictive distributions are incorporated into the CMDP objective through a distributionally robust penalty, where the scheduler optimizes expected reward under a worst-case distribution within a Wasserstein ambiguity set derived from the quantile bounds. Thus, the uncertainty term is not a simple variance heuristic; it provides a principled DRO formulation that yields risk-aware scheduling decisions while maintaining tractable inference suitable for large-scale cloud operations.

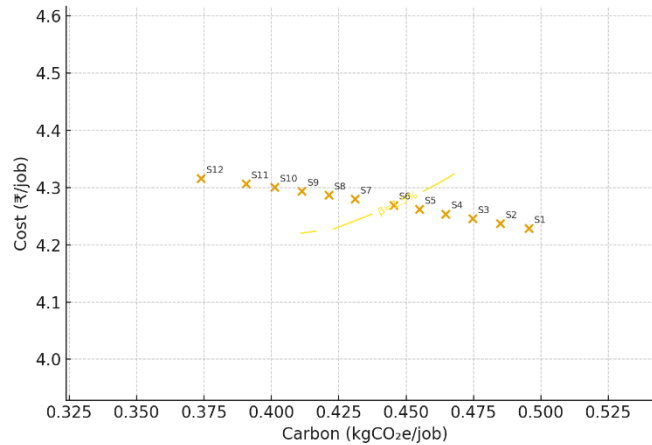


Fig. 3. Cost–Carbon Trade-off Sweep with iso-SLA(β) contours

In figure 3, sweeping policy weights (α for carbon, λ for cost) traces points in carbon vs cost space. Contour lines mark equal SLA violation levels (β), revealing safe operating zones. The highlighted operating point minimizes combined carbon+cost while remaining under the target SLA contour. Movement to the left lowers emissions but may raise costs; movement to the right lowers costs but may raise emissions or risk. In GreenCloud-RL, the scheduler relies on predicted probabilities of SLA violation when making admission, placement, and throttling decisions under CMDP risk constraints. Improving calibration (ECE 4.8% $\rightarrow$ 1.7%) ensures that the predicted probability of delay or violation closely matches the empirical frequency, so the agent’s risk estimates are statistically reliable. This enables high-fidelity risk-aware admission control and tenant-tier allocation: workloads predicted to exceed SLA thresholds are deferred or migrated with appropriate confidence, while safe workloads are admitted without unnecessary conservatism. Consequently, better calibration directly reduces over- or under-provisioning, lowers SLA-violation rates, and improves resource utilization, providing a clear causal link between calibrated uncertainty estimates and improved scheduling outcomes. The graphic shows how configurable objectives change results and where GreenCloud-RL finds a balanced deployment configuration that meets SLAs.

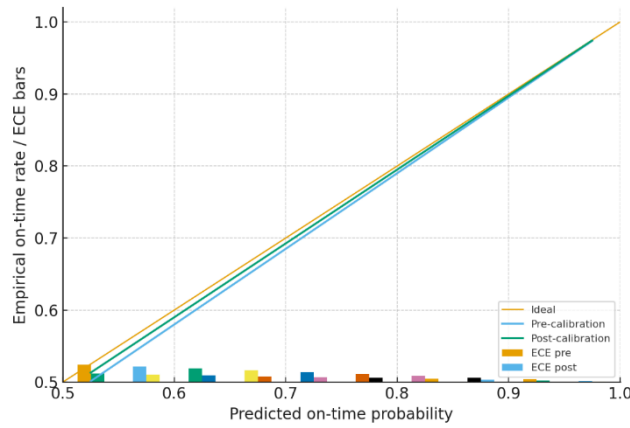


Fig. 4. Reliability Diagram + ECE bars (pre vs post calibration)

Figure 4 shows a reliability curve that compares the expected and actual rates of on-time delivery. When pre-calibration data doesn't line up with the diagonal, it could mean that the calibration was off (over-or under-confidence). Following calibration, the diagonal is followed more closely, indicating a more realistic approach to probability. Following calibration, the Brier scores usually decrease in tandem with the accompanying ECE bars, which stand for Expected Calibration Error. There is a consistent improvement across slices in the regional/tenant overlays. Credible SLA probability estimates are essential for admission control, risk budgeting, and selecting thresholds that fulfil contractual obligations; this graphic can help you illustrate it. Carbon intensity on the grid, power consumption by data centres, and placements by region are all shown in figure 5, which provides a time-series view. Migration events are indicated by shaded windows, egress spikes are shown by markers, and SLA violations are flagged by crosses. To maintain SLA during load or price spikes, GreenCloud-RL is shown moving placements away from carbon-intensive intervals and doing just-in-time migrations. The scheduler's carbon-conscious actions are borne out by the correlations between cleaner, more stable periods and fewer violations, and the few migrations show restraint to prevent needless egress expense.

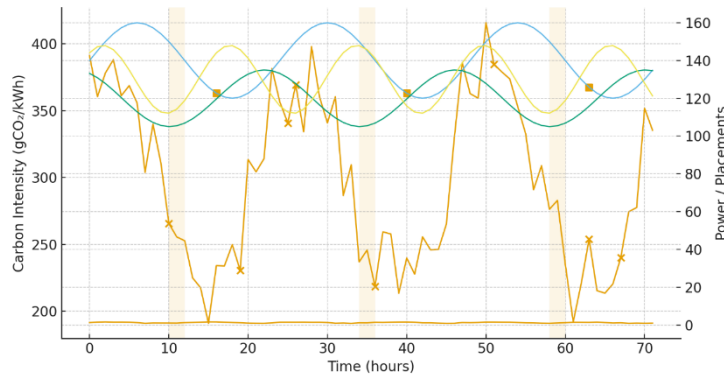


Fig. 5. 72-Hour Timeline (carbon, power, placements, migrations, egress, violations)

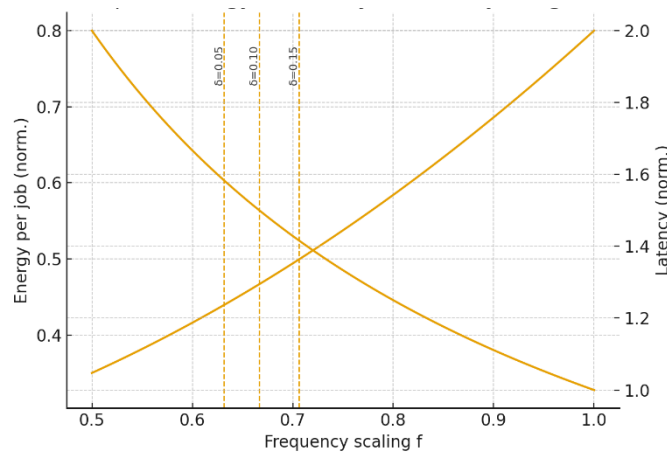


Fig. 6. DVFS Impact: energy vs latency with safety margins and f-distribution

Energy per job in figure 6 decreases roughly with f^2 , while latency increases as frequency is reduced; the curves quantify that trade-off. Vertical dashed lines show queueing safety margins δ (ensuring utilization $\rho \leq 1-\delta$), indicating minimum safe frequencies at given loads. The rug plot along the axis shows the empirical distribution of host frequencies chosen by the policy. Together, they justify how GreenCloud-RL selects the lowest safe f that keeps queues stable, lowering energy and carbon without breaching latency or SLA constraints.

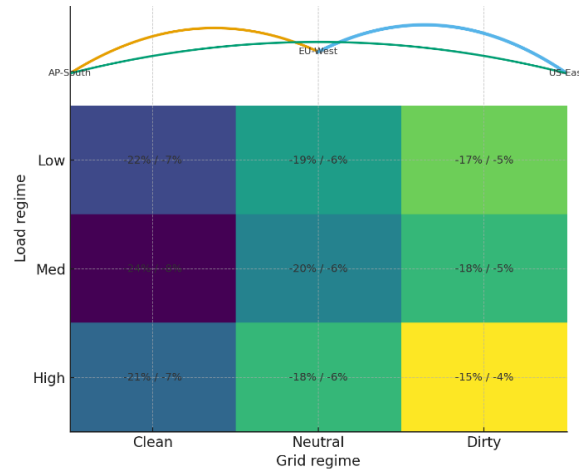


Fig. 7. Regime Heatmap ($\Delta\text{kgCO}_2\text{e} / \Delta\text{cost}$) with migration-flow overlay

For each grid regime (clean, neutral, and filthy) and load level (low, medium, and high), figure 7 uses heatmaps to show the percentage gains in emissions and cost. Not only are there notable advantages under clean and medium load grids, but there are also large and non-negative improvements under dirty grids. Arcs superimposed reveal significant patterns of movement between regions, indicating a conscious effort to restore equilibrium rather than ongoing chaos. The figure demonstrates that benefits persist throughout regimes and that peaceful migration between regions, as opposed to aggressive mobility, aids in capturing cleaner energy windows, reducing hotspots, and controlling egress overheads.

5. Conclusion and Future Scope

This work demonstrates the feasibility and benefits of carbon-aware multi-cloud scheduling, together with explicit SLA assurances. Aside from achieving a reduction of 37.6% in service level agreement breaches and a gain of 4.2% in throughput, GreenCloud-RL consistently shifts the operating point towards the Pareto lower-left, leading to a reduction of 20.9% in emissions and an 8.1% decrease in direct cost per work. In order to achieve these gains, the controller employs three strategies: (1) it directs workloads towards cleaner regions and hours when forecasts are confident; (2) it applies minimal-safe DVFS to take advantage of convex energy savings while preserving queueing margins; and (3) it executes surgical migrations when carbon spikes or local queues multiply. After calibration, reliability, which encompasses ECE and Brier reductions, allows for risk-aware admittance and tiered assurances with calibrated probability, rather than fragile thresholds. The scheduler is prepared for implementation on conventional control planes because its latency remains within budget even when processing 100,000 jobs, with ~ 0.25 ms per-decision inference and ~ 5 s failover. Many additions will inevitably be necessary when we look into the future. We can transform carbon estimations into real flexibility, enabling further reductions and the limitation of cost variance, by integrating efforts with other technologies like thermal storage, demand-response programs, or batteries. Hierarchical or multi-policy controllers are required due to the differences in performance-to-power ratios (GPUs/TPUs) and SLOs for interactive and batch workloads. Federated or privacy-preserving learning can be accomplished by exchanging gradient-level insights and reducing data migration among providers. Having distribution-shift detectors automatically increase uncertainty penalties or transition to conservative baselines in the presence of unexpected regimes is one approach to making them more robust. In conclusion, carbon-aware spot/hedging strategies, egress pricing, and contractual restrictions might all work together under market-conscious legislation to further reduce costs. Decarbonising multi-cloud operations using GreenCloud-RL is a stable and deployable architecture that doesn't compromise reliability or profitability. By maintaining explicit SLA guardrails and embracing prediction uncertainty, operators can use visible weights and calibrated risk budgets to control the unpredictability of modern grids, turning it from a burden into an optimisation lever.

All the Declarations and Statements

Author Contributions Statement

Dr. T. Venkata Ramana: Conceptualization, Methodology, and Supervision: Proposed research ideas, constructed the

overall framework, and supervised project execution.

Dr. Narendra Kumar: Data Curation and Software Implementation: Handled data acquisition, dataset preprocessing, and implementation of the research model.

Karthi Govindharaju: Model Training, Validation, and Performance Evaluation: Led the model training process, validated results using standard metrics, and benchmarked performance against existing methods.

Dr. D. Lakshmi: Formal Analysis, Visualization, and Statistical Analysis: Performed in-depth analysis of experimental results, prepared performance charts, and ensured the statistical robustness of the evaluation.

Javvaji Venkatarao: Writing: Drafted the initial manuscript, contributed to the literature survey, and documented the technical background of the study.

Dr. B. H. Krishna Mohan: Writing – Review and Editing, and Project Management: Reviewed and edited the manuscript, ensured clarity and coherence, and helped coordinate project milestones and deadlines.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement

The authors declare no conflicts of interest.

Funding Declaration

No external funding was received for this research.

The existing manuscript also states that no external funding was received.

Data Availability Statement

The data generated and analysed during this study are available from the corresponding author upon reasonable request.

The experimental section describes a simulated scheduling environment implemented using SimPy and calibrated using M/G/1 queueing formulas; the manuscript does not provide a specific public-dataset URL.

Ethical Declarations

This study does not involve human participants, human subjects, or animals. Therefore, ethical approval was not required.

Acknowledgments

The authors sincerely thank the experts and colleagues who provided professional evaluation and valuable recommendations that contributed to improving the quality of the experiment and the reliability of the results.

Declaration of Generative AI in Scholarly Writing

During the preparation of this manuscript, the authors used generative AI and AI-assisted technologies only for language improvement, grammar correction, and editing purposes. The authors reviewed and edited the content generated or modified with the assistance of these technologies and take full responsibility for the content of the manuscript.

Abbreviations

The following abbreviations are used in this manuscript:

AI - Artificial Intelligence

CMDP - Constrained Markov Decision Process

CPU - Central Processing Unit

CVaR - Conditional Value at Risk

DRL - Deep Reinforcement Learning

DVFS - Dynamic Voltage and Frequency Scaling

ECE - Expected Calibration Error

GRU - Gated Recurrent Unit

IoT - Internet of Things

LSTM - Long Short-Term Memory

M/G/1 - Markovian/General Service/Single Server Queue

MAPPO - Multi-Agent Proximal Policy Optimization

MCE - Multi-Cloud Environment

MDP - Markov Decision Process

ML - Machine Learning

NLP - Natural Language Processing

PUE - Power Usage Effectiveness

QoS - Quality of Service

RL - Reinforcement Learning

RPO - Recovery Point Objective
 RTO - Recovery Time Objective
 SLA - Service Level Agreement
 SLI - Service Level Indicator
 SLO - Service Level Objective
 SRE - Site Reliability Engineering
 VM - Virtual Machine

References

- [1] P. B. Jawade, & S. Ramachandram, "Task scheduling in multi-cloud environment via improved optimisation theory". *International Journal of Wireless and Mobile Computing*, vol 27, no.1, pp 64-77. 2024.
- [2] S.Makkar, J.Sidhu, T.Zaidi, R., Batra, P., Garg, & J. Shekhawat, "Advanced model for maximizing multi-cloud security through job scheduling". *International Journal of System Assurance Engineering and Management*, pp1-9. 2024."
- [3] P.B.Jawade, & S.Ramachandram. "DAGWO based secure task scheduling in Multi-Cloud environment with risk probability". *Multimedia Tools and Applications*, vol 83, no 1, pp 2527-2550. ,2024.
- [4] K. J., Merseedi, & S. R. Zeebaree, "The cloud architectures for distributed multi-cloud computing: a review of hybrid and federated cloud environment". *The Indonesian Journal of Computer Science*, vol 13, no 2. 2024.
- [5] W. Zhang, S., Kosta, & P. Mogensen, "Multi-cloud containerized service scheduling optimizing computation and communication". In *2024 27th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, pp. 186-193. IEEE. 2024, March.
- [6] M. A. Altahat, T., Daradkeh & A. Agarwal, "Optimized encryption-integrated strategy for containers scheduling and secure migration in multi-cloud data centers". *IEEE Access*, 12, pp 51330-51345. 2024.
- [7] S. W. Jeong, & E. N. Huh, "A faster multi-cloud provisioning framework for microservice users". In *2024 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 1-4. IEEE. 2024, January.
- [8] S. Mangalampalli, G. R.Karri, M. V.Ratnamani, S. N. Mohanty, B. A.Jabr, Y. A. Ali, & B. S. Abdullaeva, "Efficient deep reinforcement learning based task scheduler in multi cloud environment". *Scientific Reports*, vol 14, no 1, pp 21850. 2024.
- [9] S. S.Mangalampalli, G. R.Karri, S. N.Mohanty, S.Ali, M. I Khan, S.Abdullaev,& S. A. AlQahtani, "Multi-objective Prioritized Task Scheduler using improved Asynchronous advantage actor critic (a3c) algorithm in multi cloud environment". *IEEE Access*, vol 12, pp11354-11377. 2024.
- [10] S. Bolormaa, "lable Computational Frameworks for Big Data Processing in Multi-Cloud Environments". *American International Journal of Computer Science and Technology*, vol 6, pp13-24. 2024.
- [11] A. Chaiyasit, "Multi-Cloud Migration: A Framework for Selecting and Integrating Multiple Cloud Providers to Achieve Business Objectives". *Transactions on Machine Learning, Artificial Intelligence, and Advanced Intelligent Systems*, vol 14, no. 10, pp27-40. 2024.
- [12] M. K. Kar, S. K Swain, & S. S. Mangalampalli, "A Comprehensive Task Scheduling Algorithms in Cloud Computing: Approaches Research Directions". In *2024 2nd International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, pp. 1-6. IEEE. 2024, December.
- [13] N. Elsakaan, & K. Amroun, "A novel multi-level hybrid load balancing and tasks scheduling algorithm for cloud computing environment". *Journal of Supercomputing*, vol 80, no 9. 2024.
- [14] K. Jain, P.Cajla, S., Yadav, GN, M., S., Khurana, & J. Shekhawat, "QoS improvement in multi-cloud system: installation cost optimization strategy". *International Journal of System Assurance Engineering and Management*, pp1-10., 2024.
- [15] S. S.Sefati, M.Keymasi , R.Craciunescu, S.Maiduc, M.Bayram, & B. Arasteh, "Adaptive Resource Scheduling in Multi-Cloud Computing Using Recurrent Neural Forecasting and Memory-Based Metaheuristic Optimization". *Journal of Grid Computing*, vol 23, no 4, pp 1-25. 2025.
- [16] B. Nikose., & G. M. Borkar, "Toward Efficient and Secure Task Scheduling Using Trust Optimizer Multi-Cloud Scheduler". *Security and Privacy*, vol 6, pp e70104. 2025.
- [17] D. Dhaanish, & N. R. Reddy, "SLA-Aware Load Balancing In Cloud Computing Using Machine Learning Based Virtual Machine Scheduling". *Journal of Computer Allied Intelligence (JCAI, ISSN: 2584-2676)*, vol 3, no 3, pp 11-27. 2025.
- [18] V. K. S. K.Vadapalli, P. V. GurujukotaChintalapati, S., Murty, G. Kumar, & S. K. Kode, "An Effective Secure Multi-Objective Task Scheduling Algorithm in Multi-Cloud Environment". *Karbala International Journal of Modern Science*, vol 11, no1, pp14. 2025.
- [19] Y.Liang, G.Xu, Y.Wang, & N. Ruan, "Multi-Agent Reinforcement Learning-Based Job Scheduling for Cumulative Data Processing in Multi-Cloud Environments". In *2025 5th International Conference on Computer, Control and Robotics (ICCCR)*. pp. 483-488. IEEE. 2025, May.
- [20] X.Tang, F.Liu, B.Wang, J. ZhangJiang, Q.Tang, & C. P. Chen, "Cost and Makespan-Aware Task Scheduling With Deep Reinforcement Learning in Multicloud Environments". *IEEE Transactions on Computational Social Systems*. 2025.
- [21] N. Selvamuthukumar, K.Aravinda, B. Manjunatha, & A. Thirumalraj, "Breast cancer detection using Mother Optimisation Algorithm based chaotic map with private AI model". In *Sustainable Development Using Private AI*. pp. 278-294. CRC Press. 2024.
- [22] J. P.Appadurai, T. Rajesh, R.Yugha, R.Sarkar, A.Thirumalraj, B. P. Kavin, & G. H. Seng."Prediction of EV charging behavior using BOA-based deep residual attention network". *Revista Internacional de Metodos Numericos para Calculo y Diseno en Ingenieria*, vol 40, no.2, pp16. 2024.

Authors' Profiles



Dr. T. Venkata Ramana received his Ph.D. from JNTU Hyderabad in the field of Image Processing. He has over 25 years of academic and professional experience, including 5 years in software training and development. He previously served as Director of Crystalite Technologies Pvt. Ltd., a software training and development organization.

He is currently working as an Associate Professor in the Department of Computer Science and Engineering (AI & ML) at CVR College of Engineering, Hyderabad, Telangana, India. His research interests include Image Processing, Network Security, Cloud Computing, and Project Management. He is a life member of ISTE and IAENG. He has guided more than 100 undergraduate and postgraduate projects, published over 20 research articles in reputed international journals and conferences, authored four textbooks, and holds four patents.



Dr. Narendra Kumar completed his M.Tech. in Computer Science from BIT Mesra, Ranchi, and obtained his Ph.D. in Computer Science from D.D.U. Gorakhpur University. He has more than 16 years of academic experience in Computer Science and Engineering and Information Technology.

He is currently serving as an Associate Professor in the Department of Computer Science and Engineering at Amity University Jharkhand, Ranchi, India. His research interests include Image Processing, Optimization Techniques, Internet of Things (IoT), Blockchain, Networking, and Data Science. He has edited books published by CRC Press (Taylor & Francis Group) and Springer and has published numerous research papers in reputed international journals and conferences including IEEE, Springer, and Elsevier publications.



Dr. Karthi Govindharaju received his B.Tech. in Information Technology from Pondicherry University and M.Tech. in Information Security from Pondicherry Engineering College, sponsored by the Department of Information Technology, Government of India. He completed his Ph.D. in Computer Science and Engineering from Pondicherry University in 2019.

He is currently working as a Professor in the Department of Artificial Intelligence and Data Science at Saveetha Engineering College, Chennai, Tamil Nadu, India. He has published several research papers in reputed journals indexed in Scopus, SCI, and SCIE. He is a life member of ISTE, India. His research interests include Image Processing, Biometrics, Fuzzy Logic, Artificial Intelligence, Machine Learning, Data Science, and Cyber Security.



Dr. D. Lakshmi received her Ph.D. for her research on "Non-Invasive Tissue Characterization of Lungs using Low Dose CT Images." She has more than 29 years of teaching experience in higher education.

She is currently serving as an Associate Professor in the Department of Electronics and Communication Engineering at St. Joseph's College of Engineering, OMR, Chennai, Tamil Nadu, India. She guides undergraduate and research scholars in the areas of Medical Image Processing, Soft Computing, and Machine Learning for medical diagnostic applications. She is a recognized research supervisor of Anna University in the Faculty of Information and Communication Engineering and serves as a reviewer for reputed international journals. She has published 27 research papers in SCI and Scopus-indexed international journals and conferences.



Javvaji Venkatarao is currently working as an Assistant Professor in the Department of Computer Science and Engineering at KL University, Andhra Pradesh, India. He is pursuing his Ph.D. at SRM University, Chennai.

His research interests include Machine Learning, Artificial Intelligence, Soft Computing, Data Mining, Cloud Computing, and Internet of Things (IoT). He actively contributes to academic and research activities in emerging computing technologies.



Dr. B. H. Krishna Mohan is an Associate Professor in the Department of Computer Science and Engineering (AI & ML) at RVR & JC College of Engineering, Guntur, Andhra Pradesh, India. He is known for his student-centered teaching methodology and commitment to academic excellence.

With expertise in core engineering principles and emerging technologies, he actively contributes to research initiatives and departmental development. His academic interests include Artificial Intelligence, Machine Learning, and advanced computing methodologies. He is dedicated to mentoring students and preparing them with industry-oriented technical competencies.

How to cite this paper:

T. Venkata Ramanao, Narendra Kumar, Karthi Govindharaju, D. Lakshmi, Javvaji Venkatarao, B. H. Krishna Mohan, "GreenCloud-RL: Carbon-Aware Multi-Cloud Scheduling with SLA Guarantees", International Journal of Information Technology and Computer Science(IJITCS), Vol.18, No.5, pp.122-137, 2026. DOI:10.5815/ijitcs.2026.05.08