Toward Sustainable and Cost-Efficient HPC systems: A Deep Reinforcement Learning Job Scheduling approach

Thanh Hoang Le Hai[✉]1[0000−0001−7821−9158], Nhan Nguyen Phuc1[0009−0009−7081−3940], Minh Bui Ngoc2[0009−0008−5898−2015], Mai Nguyen Tran Phuong2[0009−0001−3135−8529], and Nam Thoai[✉]1[0000−0003−0499−8640]

High Performance Computing Laboratory, Faculty of Computer Science and Engineering,
Advanced Institute of Interdisciplinary Science and Technology,
Ho Chi Minh City University of Technology (HCMUT),
268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Vietnam
Vietnam National University Ho Chi Minh City,
Linh Trung Ward, Thu Duc City, Ho Chi Minh City, Vietnam
{thanhhoang,nhan.nguyen2005phuyen,minh.buingocbkhoa,mai.nguyen0505,namthoai}@hcmut.edu.vn

Abstract. Hybrid-powered High Performance Computing (HPC) centers can reduce both operational costs and carbon footprints by aligning job execution with renewable supply and time-varying electricity prices. However, existing Deep Reinforcement Learning (DRL) schedulers often neglect price signals, rely on synthetic energy traces, or conflate job selection and delay, limiting their real-world applicability. This work introduces a cost-aware, multi-action DRL framework that embeds a hybrid energy cost model into an HPC scheduling environment, enabling agents to incorporate dynamic price signals and renewable forecasts while decoupling job selection from execution delay. Evaluated on production-scale workload traces and measured renewable profiles, the proposed scheduler consistently reduces energy costs, improves renewable utilization, and maintains competitive job responsiveness compared with heuristic and learning-based baselines. These results demonstrate a practical path toward economically and environmentally sustainable HPC operations.

Keywords: High Performance Computing · Job Scheduling · Green-Aware Scheduling · Reinforcement Learning · Multi-objective Reinforcement Learning

1 Introduction

High Performance Computing (HPC) systems are increasingly becoming the backbone of modern scientific research, industrial design, and data-driven technologies[1]. As their usage expands to support massive-scale tasks like training large language models and simulating large-scale systems, the environmental impact of these systems has come under scrutiny. The world's top supercomputers

now draw over 20 megawatts, emitting on the order of 100,000 tonnes CO_2 per year when powered by conventional grids [2]. As global carbon-reduction policies tighten, the imperative to decarbonize HPC operations has never been greater.

To address this challenge, integrating renewable energy into HPC operations has been recognized as a critical pathway for reducing carbon emissions [3]. However, renewable sources are inherently intermittent, driven by weather variability and diurnal cycles. Moreover, their on-peak tariffs can exceed conventional electricity rates by up to two to three times [4], and the infrastructure and storage needed to stabilize supply often impose higher operational costs than grid power [5]. As a result, both the variability and higher associated costs make renewable energy integration an ongoing challenge for HPC operations. HPC administrators must balance electricity costs, renewable utilization, and system performance. Among the available strategies, job scheduling, which involves assigning jobs to resources and determining their execution order and timing, provides a flexible and low-cost lever that avoids major infrastructure upgrades. Prior works have advanced Deep Reinforcement Learning (DRL)-based job scheduling for core performance metrics [6, 7], and recent green-aware methods show that delaying work to match renewable supply can increase the renewable share [8]. However, these methods overlooked electricity costs, thereby reducing the feasibility of integrating renewable power into the system due to cost barriers.

In this paper, we propose a novel green-aware HPC scheduling approach that integrates explicit cost-awareness into the decision-making process and grounds its evaluation in realistic operational data. Our main contributions are as follows:

1. We embed a novel model of hybrid energy supply (grid, solar, wind) into the HPC scheduling agent, converting measured generation and consumption into optimization-ready metrics for renewable share and time-varying, source-specific electricity costs.
2. We design a scalar reward function that jointly optimizes renewable utilization, normalized cost per kWh, and bounded slowdown. Tunable penalty weights enable explicit trade-off control between environmental, economic, and performance objectives.
3. We evaluate the proposed RL scheduler using production-scale HPC workload traces with realistic per-job power measurements, measured renewable generation profiles, and real-world time-of-use tariffs.

The remainder of this paper is organized as follows. Section 2 reviews related work on HPC scheduling and renewable integration. Section 3 presents the proposed framework and methodology. Section 4 describes the experimental setup and analyzes the results. Finally, Section 5 concludes the study and outlines directions for extending the framework toward multi-objective reinforcement learning and more realistic electricity models.

2 Related Work

Job scheduling in HPC systems refers to the process of mapping submitted jobs to available computing resources and determining their execution order and start

times. Its fundamental role is to coordinate shared access to limited resources in order to optimize system-level objectives such as throughput, utilization, fairness, and turnaround time. Scheduling is therefore a cornerstone of HPC resource management, directly influencing both operational efficiency and user-perceived performance.

Formally, the scheduling problem is NP-hard, since the number of feasible allocations grows combinatorially with the size of the job queue and the cluster's resources [9]. Consequently, production systems employ approximate strategies. Classical heuristics such as First Come First Served (FCFS), Shortest Job First (SJF), and backfilling remain popular because they achieve low overhead and predictable behavior. More advanced approaches employ metaheuristics such as Genetic Algorithms (GA) or Ant Colony Optimization (ACO), which search larger solution spaces and can yield higher quality schedules at the expense of longer runtime [10]. Despite their effectiveness for traditional objectives, these methods were primarily designed to optimize performance metrics and seldom incorporate external factors such as time-varying renewable supply or dynamic electricity pricing.

To overcome these limitations, DRL has emerged as a promising paradigm for HPC scheduling. DRL learns adaptive dispatch policies directly from system state and workload dynamics, often outperforming fixed heuristics on throughput and slowdown. Representative examples include hierarchical models such as DRAS [11] and PPO-based agents for direct queue-to-dispatch mapping [12]. Practical DRL methods address the combinatorial action space with techniques like action masking. Yet despite these advances, most DRL schedulers remain performance-centric, leaving external factors such as renewable variability and electricity price signals largely unaddressed.

In response to these shortcomings, subsequent research has begun to extend classical heuristics and DRL approaches toward environmental objectives. Heuristic methods such as LPTPN prioritize jobs by processing-time and power demand to better fit renewable envelopes [13], while recent DRL efforts (e.g., GAS-MARL, RARE) incorporate renewable forecasts to jointly optimize utilization and performance [8, 14]. These methods improve renewable share or throughput, but they generally omit explicit electricity pricing in the objective. Consequently, a scheduler that jointly optimizes performance, renewable usage, and cost remains an open problem - providing the motivation for our proposed cost-aware, multi-action DRL framework.

3 Proposed Method

3.1 Problem Formulation

Modeling of HPC Clusters with a Hybrid Energy Supply We consider a homogeneous HPC cluster of N nodes, each with C cores and an idle power consumption of P_{idle} per powered-on node. Following the modeling assumptions in GAS-MARL [8], the instantaneous cluster power consumption at time t is

expressed as

$$P_{\text{total}}(t) = N_{\text{on}}(t) \cdot P_{\text{idle}} + \sum_{j \in \mathcal{R}(t)} P_j, \tag{1}$$

where $N_{\text{on}}(t)$ is the number of active nodes, $\mathcal{R}(t)$ denotes the set of running jobs, and P_j is the average power of job j. To reduce overhead, nodes remain powered on as long as the total core demand does not exceed $N \times C$. Jobs arrive from traces or stochastic models, wait in the queue, and are either dispatched or deferred by the scheduler.

The site is supplied by a hybrid power system comprising solar $G_{\text{sol}}(t)$, wind $G_{\text{wind}}(t)$, and conventional grid energy. Renewable supply is always prioritized, and any residual demand is covered by the grid. Time is discretized into one-hour slots, with perfect 24-hour forecasts for renewable generation and electricity prices. The total energy cost is computed as the sum of consumed energy from each source multiplied by its respective time-varying unit price.

Optimization Problem Definition We formulated the scheduling task as a constrained multi-objective optimization over a horizon of T discrete time slots and M completed jobs. Denote:

$$\begin{aligned} E_{\text{total}} = \sum_{t=0}^{T-1} P_{\text{total}}(t), \quad R_{\text{total}} = \sum_{t=0}^{T-1} \min\{P_{\text{total}}(t),\, G(t)\}, \\ \text{and} \quad \text{BSLD}_{\text{avg}} = \frac{1}{M} \sum_{j=1}^{M} \max\Big(\frac{w_j + e_j}{\max(\tau, e_j)},\, 1\Big). \end{aligned} \tag{2}$$

We then seek a policy π that simultaneously:

$$\begin{cases} \max\limits_{\pi} \;\; f_1(\pi) \;=\; \dfrac{R_{\text{total}}}{E_{\text{total}}} \quad \text{(maximize renewable utilization)}, \\ \min\limits_{\pi} \;\; f_2(\pi) \;=\; \dfrac{\sum_{t=0}^{T-1} \pi_{\text{grid}}(t)\,[P_{\text{total}}(t) - G(t)]_+}{E_{\text{total}}} \quad \text{(minimize average cost per kWh)}, \\ \min\limits_{\pi} \;\; f_3(\pi) \;=\; \text{BSL}_{\text{avg}} \quad \text{(minimize bounded slowdown)}. \end{cases} \tag{3}$$

subject to the resource constraint at each slot t:

$$\sum_{j \in \mathcal{R}(t)} c_j \;\le\; N \times C, \tag{4}$$

and to the feasibility of scheduling and delay actions. This formulation is designed to capture the inherent trade-off among three critical objectives—sustainability through green energy use, economic efficiency via low cost per unit energy, and performance via bounded slowdown.

3.2 Markov Decision Process modeling

State Space At each decision epoch t the scheduler observes a compact state $\mathbf{s}_t$ formed by three fixed-size components: a job-queue matrix of shape $Q \times F_j$, a running-job matrix of shape $R \times F_r$, and an energy-context matrix of shape $T \times F_e$. Queue rows encode normalized submission wait, requested cores, estimated runtime, total and per-core power, renewable-suitability flags and an allocation-feasibility flag; rows are ordered by submission time. Running rows encode allocated cores, current and per-core power, and remaining execution time; rows are ordered by predicted completion time. The energy matrix embeds slot duration, available renewable power, grid costs and renewable prices to enable short-term price and supply anticipation. All features are normalized by their respective maxima before being supplied to the learning agent.

Action Space At time t the agent issues an action $\mathbf{a}_t = \{j_t, d_t\}$, where j_t selects one of the earliest queued jobs and $d_t \in D$ selects a discrete delay. The delay set is partitioned as $D = \{D_0\} \cup D_{\text{th}} \cup D_{\text{to}}$, where D_0 is the single no-delay option, D_{th} contains threshold delays that defer dispatch until a given number of queued jobs complete, and D_{to} contains concrete time offsets (e.g. 300, 600, 900 s). Invalid job–delay pairs (those exceeding capacity or referencing jobs outside the observable queue) are masked, so the agent evaluates only feasible pairs $\mathcal{A}_t$. The number of delay alternatives is $|D| = 1 + |D_{\text{th}}| + |D_{\text{to}}|$. This multi-action design decouples job selection from timing, allowing the policy to trade immediate throughput for postponement to exploit favorable energy conditions.

Reward Function We evaluate each transition by a scalar reward $r_t = r(\mathbf{s}_t, \mathbf{a}_t)$ that balances three operational objectives: system responsiveness, renewable usage, and cost-effectiveness. Let BSLD_{avg} denote the average bounded slowdown across all completed jobs, $\text{GR} = \frac{\text{renewable energy used}}{\text{total energy used}}$ be the green-energy ratio, and $\text{CE} = \frac{\text{total cost}}{\text{total energy used}}$ the normalized cost-per-unit-energy. We then define

$$r_t = \text{GR} \; - \; \eta \, \text{BSLD}_{\text{avg}} \; - \; \beta \, \text{CE}, \tag{5}$$

where $\eta, \beta > 0$ are tunable penalty weights for slowdown and cost. By rewarding high green ratios while penalizing delay and energy expense, this single objective drives the RL agent toward sustainable, low-cost, high-throughput scheduling.

3.3 Training Algorithm

Figure 1 gives an overview of the interaction between the HPC cluster and the agent. The agent receives queue, running-job, and energy matrices, selects a job and delay, and applies the action to the cluster. Rewards based on performance and energy are returned, completing the feedback loop.

Building on this framework, we train an actor–critic policy with Proximal Policy Optimization (PPO) augmented by action masking [15]. The policy $\pi_\theta(a \mid$

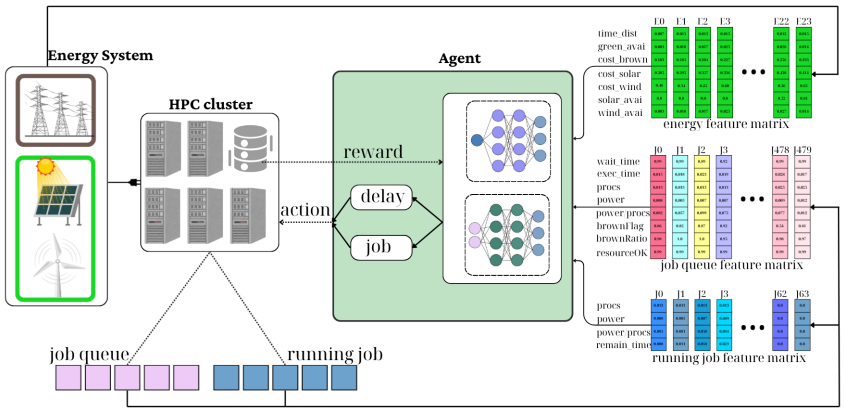


Fig. 1: Overview of the proposed scheduling framework

s) and value $V_\phi(s)$ are learned from trajectories collected by multiple parallel actors. PPO stabilizes updates via the clipped ratio

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \tag{6}$$

and the surrogate loss

$$L^{\text{PPO}}(\theta) = \mathbb{E}_t\Big[\min\{r_t(\theta)\,\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\,\hat{A}_t\}\Big], \tag{7}$$

where $\hat{A}_t$ is the generalized advantage estimate. Feasibility is enforced by a binary mask $M(s_t) \in \{0,1\}^{|\mathcal{A}|}$; masked logits are set to a large negative constant before the final softmax, so invalid actions have (effectively) zero probability, improving sample efficiency without changing PPO's clipping mechanism.

We decomposed action selection into two jointly trained actors (job selector and delay controller) under the same PPO–Mask framework. The joint policy factorizes as

$$\pi_\theta(a \mid s) = \pi_{\theta_1}(j \mid s)\,\pi_{\theta_2}(d \mid s, j), \qquad \theta = \{\theta_1, \theta_2\}, \tag{8}$$

with j sampled (masked) first and d conditioned on (s, j). Both actors share the value network and receive identical advantage signals derived from a delayed reward that combines green-energy ratio, cost efficiency and bounded slowdown.

Figure 2 illustrates the overall network architecture. The input consists of three observation matrices—job queue, running jobs, and energy context—which are first processed by dedicated encoders and then fused through a shared encoder. The resulting embedding is fed into both an actor and a critic. The actor comprises two output heads: (i) a job-selection head that generates an intermediate embedding, and (ii) a delay-selection head that conditions on both the

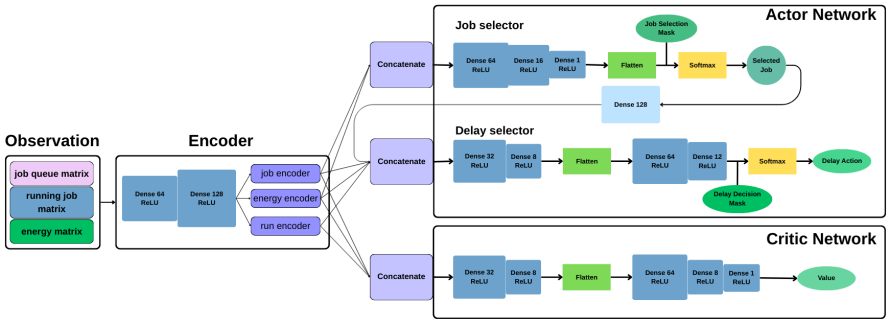


Fig. 2: Network structure for the proposed scheduling approach

global state representation and the selected job embedding. Action masks are applied to both heads prior to the softmax layer to ensure only feasible actions are considered. The critic, in parallel, outputs a scalar value based on the shared state encoding. Training is conducted for 300 epochs; in each epoch, the workload trace is initialized at a random offset and 100 independent trajectories are collected, each with up to 1,024 scheduling events. A trajectory terminates once the job queue becomes empty and no further arrivals are expected, following the protocol of [16].

4 Experiments and Results

4.1 Simulation Environment

We evaluated our scheduling policies in a unified simulation environment that builds on SchedGym[17]—which faithfully replays HPC workloads stored as Standard Workload File (SWF) traces[18] and manages job submission, resource allocation, and power profiling—with MESSPy[19]—which generates detailed renewable and grid supply data and computes per-hour techno-economic metrics.

4.2 Datasets

For workload logs, we used two traces: F-DATA, collected from the Fugaku supercomputer [20], and PM100, collected from the Marconi100 supercomputer between May and October 2020 [21] for training and testing. These datasets were chosen because they include per-job power measurements, which avoid the need for synthetic power assignment and thereby increase experimental realism and external validity. Regarding the energy system, we modeled hourly solar and wind generation using the NREL Typical Meteorological Year (TMY) dataset [22], aligned with the geographical locations of the HPC centers (Nagoya, Japan for F-DATA; Perugia, Italy for PM100). Electricity prices follow local time-of-use industrial tariffs.

4.3 Training Procedure

At the start of each trajectory, the environment was initialized with $\eta = 0.002$, $\beta = 1.5$, MAX_QUEUE_SIZE $= 480$, run_win $= 64$, green_win $= 24$, delayMaxJobNum $= 5$, and delayTimeList $= [120, 300, 600, 900, 1200, 1800, 2400, 3600]$. At each step we build a binary action mask from queue occupancy and delay constraints, sample one or two masked actions, and accumulate a scalar reward that weights performance, fairness and energy. Trajectories are stored; we compute GAE with $\gamma = 1$, $\lambda = 0.97$, standardize advantages, then run 8 epochs of mini-batch PPO (clip $\epsilon = 0.2$). Updates use Adam (actor learning rate $1{\times}10^{-4}$, critic learning rate $5{\times}10^{-4}$), gradient-norm clipping $= 0.5$, and an entropy term (coef $= 1{\times}10^{-3}$). Hyperparameters were tuned across traces; a full sensitivity analysis is beyond this work.

4.4 Scheduling Baselines

We compared the proposed cost-augmented Multi-Action RL scheduler against targeted baselines chosen to isolate distinct objectives. For *performance* we include FCFS and the F2 heuristic [23]; for *renewable integration* we include LPTPN [13]; and for *multi-objective* comparison we include a Genetic Algorithm and Ant Colony Optimization tuned for energy-aware scheduling [24, 25]. In addition, we evaluated two controlled RL variants: a *selection-only* variant that constrains the agent to job-selection decisions while disabling auxiliary control actions, and the Multi-Action RL agent trained without the explicit cost signal. These ablations respectively isolate the contributions of the selection policy and of explicit cost feedback. All methods are assessed on three complementary axes: scheduling performance (average bounded slowdown, average waiting time, and makespan, reported in hours with $t_{\mathrm{h}} = t_{\mathrm{s}}/3600$), renewable integration (green energy ratio, %), and energy cost (total cost [€] and cost efficiency [€/kWh]). This evaluation protocol permits direct, interpretable comparisons and the identification of regimes in which the cost-augmented policy attains Pareto improvements.

4.5 Results and Analysis

Table 1: Comparison results of scheduling strategies on F-DATA.

Algorithm	Total Cost (€)	Green Ratio (%)	Cost Eff. (€/kWh)	Avg B. Slowdown	Avg Waiting (h)	Makespan (h)
ACO	995,156.48	9.68	0.213	20.50	36.56	415.63
F2	995,922.77	9.42	0.214	21.40	37.47	413.77
FCFS	1,001,097.06	9.17	0.213	37.85	57.95	409.81
GA	995,978.68	9.85	0.214	26.85	45.21	416.56
LPTPN	962,720.73	59.03	0.200	82.40	214.81	1062.39
Multi-Action RL	947,928.48	32.77	0.185	21.70	38.14	414.39
No Cost Signal	955,086.46	19.46	0.199	21.96	38.25	425.82
One-Action	997,764.96	9.56	0.214	21.07	37.75	414.71

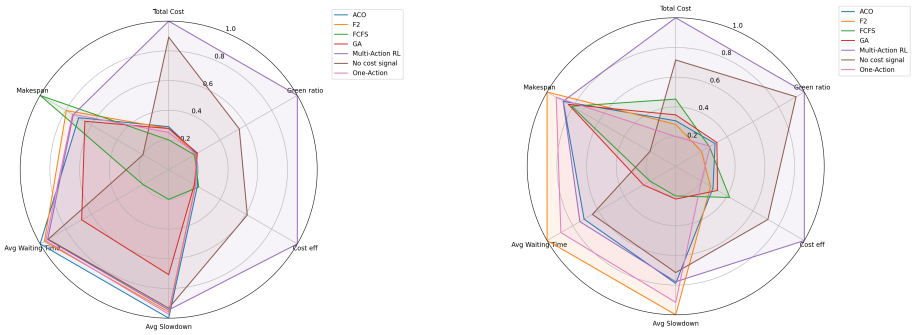
We ran ten independent trials per trace and scheduler with different random start offsets and 1,240 jobs per trial. This repeated-start protocol yields independent samples that capture intra-trace variability and workload heterogeneity.

Summary statistics are reported in Table 1 and Table 2, and normalized mean trade-offs are visualized in the radar plots of Figure 3. To preserve plot scale and improve visual discrimination, LPTPN is omitted from the radar plots because its high green ratio coincides with substantially worse cost and latency, which would compress the chart and obscure differences among the remaining schedulers.

Table 2: Comparison results of scheduling algorithms on PM100.

Algorithm	Total Cost (€)	Green Ratio (%)	Cost Eff. (€/kWh)	Avg B. Slowdown	Avg Waiting (h)	Makespan (h)
ACO	1,127,325.75	28.96	0.359	132.16	322.76	2214.60
F2	1,128,252.73	26.96	0.359	125.70	300.53	2210.81
FCFS	1,121,708.15	28.14	0.357	150.19	362.34	2216.58
GA	1,125,787.72	29.22	0.359	149.55	358.41	2215.80
LPTPN	1,410,847.06	64.68	0.425	478.61	1425.91	5787.23
Multi-Action RL	1,100,652.95	42.03	0.349	132.51	320.77	2214.00
No Cost Signal	1,111,576.30	40.83	0.353	134.41	327.72	2234.92
One-Action	1,131,408.03	28.24	0.360	128.26	308.82	2212.44

The experimental results highlight the effectiveness of the proposed cost-aware Multi-Action RL scheduler in balancing economic, environmental, and performance objectives in hybrid-powered HPC systems. As shown in Table 1 and 2 and illustrated by the radar plots in Figure 3, our method consistently outperforms both heuristic baselines (e.g., FCFS, F2, LPTPN) and ablated RL variants.



(a) Dataset F-DATA (b) Dataset PM100

Fig. 3: Multi-criteria radar plot comparison.

Compared with heuristic approaches, Multi-Action RL achieves significant cost reductions and higher renewable utilization without incurring the severe performance penalties seen in strategies like LPTPN, which increases green share but at the expense of much longer waiting times and higher energy costs. The cost feedback mechanism enables the scheduler to make economically informed delay decisions, shifting non-urgent jobs to periods with abundant renewable supply while preserving responsiveness for time-sensitive workloads.

Against single-action RL and the cost-unaware variant, the multi-action decomposition and explicit cost signal yield more adaptive and stable policies. This design allows finer control over job dispatching and delay, producing better Pareto trade-offs among cost, green ratio, and bounded slowdown. Moreover, trial-to-trial variability remains comparable to heuristic baselines, suggesting that the learned policies are robust even under heterogeneous and heavy-tailed workloads.

Overall, the results demonstrate that incorporating price-awareness and multi-action control can lower the economic barriers to renewable integration while maintaining competitive scheduling performance, providing a practical path toward sustainable and cost-efficient HPC operations.

5 Conclusion and Future Works

This work presented a cost-aware, green-oriented scheduling framework for hybrid-powered HPC systems. By integrating dynamic price signals, renewable forecasts, and a multi-action decision model, our approach achieved a balanced optimization of cost-efficiency, renewable energy utilization, and job responsiveness. Comparative experiments against heuristic and learning-based baselines demonstrated that the proposed scheduler consistently reduced energy costs, increased the share of green energy, and maintained competitive scheduling performance, highlighting its practical applicability for sustainable HPC operations.

Future work will focus on extending the framework toward a full multi-objective reinforcement learning (MORL) formulation and integrating more realistic electricity system models. This includes dynamic grid constraints, storage-aware actions, and richer cost structures to better capture real-world operational conditions.

Acknowledgments. We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Deelman, E., Dongarra, J., Hendrickson, B., Randles, A., Reed, D., Seidel, E., Yelick, K.: High-performance computing at a crossroads. Science 387(6736), 829–831 (2025), https://www.science.org/doi/abs/10.1126/science.adu0801
2. https://top500.org/lists/top500/2024/11/
3. Silva, C., Vilaça, R., Pereira, A., Bessa, R.: A review on the decarbonization of high-performance computing centers. Renewable and Sustainable Energy Reviews 189, 114019 (2024), https://doi.org/10.1016/j.rser.2023.114019
4. Bharany, S., Sharma, S., Khalaf, O.I., Abdulsahib, G.M., Al Humaimeedy, A.S., Aldhyani, T.H.H., Maashi, M., Alkahtani, H.: A systematic survey on energy-efficient techniques in sustainable cloud computing. Sustainability 14(10) (2022), https://doi.org/10.3390/su14106256

5. Oliveira, C.d., Pablo: On the environmental impact of high-performance computing. Online; accessed via Sifflez.org (2022), https://sifflez.org/publications/environment-hpc/
6. Mao, H., Alizadeh, M., Menache, I., Kandula, S.: Resource management with deep reinforcement learning. In: Proceedings of the 15th ACM Workshop on Hot Topics in Networks. p. 50–56. HotNets '16, Association for Computing Machinery, New York, NY, USA (2016), https://doi.org/10.1145/3005745.3005750
7. Wang, Q., Zhang, H., Qu, C., Shen, Y., Liu, X., Li, J.: Rlschert: An hpc job scheduler using deep reinforcement learning and remaining time prediction. Applied Sciences 11(20), 9448 (2021)
8. Chen, R., Lin, W., Huang, H., Ye, X., Peng, Z.: Gas-marl: Green-aware job scheduling algorithm for hpc clusters based on multi-action deep reinforcement learning. Future Generation Computer Systems 167, 107760 (2025), https://www.sciencedirect.com/science/article/pii/S0167739X2500055X
9. Khemka, B., Machovec, D., Blandin, C., Siegel, H.J., Hariri, S., Louri, A., Tunc, C., Fargo, F., Maciejewski, A.A.: Resource management in heterogeneous parallel computing environments with soft and hard deadlines. In: Proceedings of the XI Metaheuristics International Conference (MIC 2015). pp. 1–10. Metaheuristics International Conference, Fort Collins, CO, USA (June 2015)
10. Razzaq, S., Wahid, A., Khan, F., Amin, N., Shah, M., Akhunzada, A., Ihsan, A.: Scheduling Algorithms for High-Performance Computing: An Application Perspective of Fog Computing, pp. 107–117 (01 2019)
11. Fan, Y., Lan, Z., Childers, T., Rich, P., Allcock, W., Papka, M.E.: Deep reinforcement agent for scheduling in hpc. In: 2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS) (2021)
12. Wang, L., Rodriguez, M.A., Lipovetzky, N.: Optimizing hpc scheduling: A hierarchical reinforcement learning approach for intelligent job selection and allocation. The Journal of Supercomputing 81(8), 918 (2025), https://doi.org/10.1007/s11227-025-07396-3
13. Kassab, A., Nicod, J.M., Philippe, L., Rehn-Sonigo, V.: Green power constrained scheduling for sequential independent tasks on identical parallel machines. In: 2019 IEEE Intl Conf on Parallel Distributed Processing with Applications, Big Data Cloud Computing, Sustainable Computing Communications, Social Computing Networking (ISPA/BDCloud/SocialCom/SustainCom). pp. 132–139 (2019)
14. Venkataswamy, V., Grigsby, J., Grimshaw, A., Qi, Y.: Rare: Renewable energy aware resource management innbsp;datacenters. In: Job Scheduling Strategies for Parallel Processing: 25th International Workshop, JSSPP 2022, Virtual Event, June 3, 2022, Revised Selected Papers. p. 108–130. Springer-Verlag, Berlin, Heidelberg (2022), https://doi.org/10.1007/978-3-031-22698-4_6
15. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms (2017), https://arxiv.org/abs/1707.06347
16. Mao, H., Schwarzkopf, M., Venkatakrishnan, S.B., Meng, Z., Alizadeh, M.: Learning scheduling algorithms for data processing clusters. In: Proceedings of the ACM Special Interest Group on Data Communication. p. 270–288. SIGCOMM '19, Association for Computing Machinery, New York, NY, USA (2019), https://doi.org/10.1145/3341302.3342080
17. Zhang, D., Dai, D., He, Y., Bao, F.S., Xie, B.: Rlscheduler: An automated hpc batch job scheduler using reinforcement learning. In: SC20: International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 1–15 (2020), https://doi.org/10.1109/SC41405.2020.00035

18. Chapin, S.J., Cirne, W., Feitelson, D.G., Jones, J.P., Leutenegger, S.T., Schwiegelshohn, U., Smith, W., Talby, D.: Benchmarks and standards for the evaluation of parallel job schedulers. In: Feitelson, D.G., Rudolph, L. (eds.) Job Scheduling Strategies for Parallel Processing, Lecture Notes in Computer Science, vol. 1659, pp. 66–89. Springer-Verlag (1999)
19. Pasqui, M., Mati, A., Ademollo, A., Calabrese, M., Lubello, P., Carcasci, C.: Messpy: Multi-energy system simulator (2022), https://github.com/pielube/MESSpy, open-source Python 3.9 package for techno-economic assessment of Renewable Energy Communities and multi-energy systems
20. Antici, F., Bartolini, A., Domke, J., Kiziltan, Z., Yamamoto, K.: F-data: A fugaku workload dataset for job-centric predictive modelling in hpc systems. Scientific Data 12(1), 1321 (2025), https://doi.org/10.1038/s41597-025-05633-1
21. Antici, F., Seyedkazemi Ardebili, M., Bartolini, A., Kiziltan, Z.: Pm100: A job power consumption dataset of a large-scale production hpc system. In: Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis. p. 1812–1819. SC-W '23, Association for Computing Machinery, New York, NY, USA (2023), https://doi.org/10.1145/3624062.3624263
22. National Renewable Energy Laboratory: Typical meteorological year (tmy) — national solar radiation database (nsrdb) (2024), https://nsrdb.nrel.gov/data-sets/tmy
23. Carastan-Santos, D., de Camargo, R.: Obtaining dynamic scheduling policies with simulation and machine learning. In: SC17: International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 1–13 (2017)
24. Kolodziej, J., Khan, S.U., Xhafa, F.: Genetic algorithms for energy-aware scheduling in computational grids. In: 2011 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing. pp. 17–24 (2011)
25. Baydoun, A.M., Zekri, A.S.: Network-, cost-, and renewable-aware ant colony optimization for energy-efficient virtual machine placement in cloud datacenters. Future Internet 17(6) (2025), https://www.mdpi.com/1999-5903/17/6/261