Energy-Efficient Task Scheduling for Data Centers via Dual-agent Constrained Semi-MDP Reinforcement Learning

1st Yiling Zhang
School of Electrical Engineering
Southeast University
Nanjing, China
230249171@seu.edu.cn

2nd Yujian Ye
Dept. of Electrical and Electronic Eng.
Imperial College London
London, United Kingdom
yujian.ye11@imperial.ac.uk

3rd Goran Strbac
Dept. of Electrical and Electronic Eng.
Imperial College London
London, United Kingdom
g.strbac@imperial.ac.uk

***Abstract*—The surge in global data generation and the rising energy demand of data centers (DCs) have made energy-efficient dynamic scheduling essential for sustainable operation. To handle uncertain workloads and fluctuating electricity prices, the scheduling problem is formulated as a Constrained Semi-Markov Decision Process (CSMDP). A Dual-Agent Safe Reinforcement Learning (DA-SRL) framework is proposed, consisting of a task selector and a server allocator that collaboratively minimize energy costs while meeting deadlines and resource constraints. A safety-enhanced RL strategy combining a constraint-value network and a shielding mechanism ensures compliance with capacity and time-coupled constraints, while an action-value network improves energy efficiency. Experiments based on Alibaba real-world production dataset demonstrate that the proposed approach achieves superior scheduling optimality and safety assurance with state-of-the-art methods.**

***Index Terms*—data center, constrained Semi-Markov Decision Process, dynamic task scheduling, safe reinforcement learning, energy efficiency**

I. INTRODUCTION

In the era of rapid digital transformation, the computing demands of modern data centers (DCs) have escalated dramatically, with global capacity estimated to climb to 171–219 GW by 2030, up from roughly 60 GW today [1]. Concurrently, electricity consumption by data centres already reached approximately 415 TWh in 2024, accounting for about 1.5% of global electricity use, and is projected to reach 945 TWh by 2030 [2]. This surge in both compute power and energy draw places significant pressure on infrastructure resources, operational budgets and sustainability targets, thereby heightening the imperative for energy-efficient and deadline-aware scheduling strategies within data center operations.

Task arrivals in data centers exhibit strong stochasticity and temporal dependence, intertwined with dynamic power, thermal, and electricity price variations. Moreover, heterogeneous servers and diverse application demands necessitate adaptive and safety-aware decision-making to balance energy consumption, quality of service (QoS), and resource utilization [3]. These complexities render static or rule-based scheduling insufficient to ensure reliable performance and energy efficiency, highlighting the need for intelligent and sustainable scheduling strategies under uncertainty.

Dynamic task scheduling approaches can be broadly categorized into three groups: (1) model predictive control (MPC)-based methods [4], (2) heuristic algorithms, and (3) machine learning approaches. MPC-based methods formulate scheduling as a rolling optimization problem over a prediction horizon, effectively capturing temporal coupling constraints in sequential decision-making. Heuristic and metaheuristic algorithms, such as First-in-First-Out (FIFO) [5], Genetic Algorithm (GA) [6] and Ant Colony Optimization (ACO) [7], offer fast and robust solutions for large-scale, real-time scheduling. Hybrid heuristic strategies further enhance performance by combining multiple optimization paradigms [8].

In recent years, Deep Reinforcement Learning (DRL) has gained significant attention for its strong capability in handling uncertain and dynamic operational problems. It has been increasingly applied to data center scheduling, effectively addressing the limitations of traditional methods [9]. The scheduling problem is typically modeled as a Markov Decision Process (MDP), where learning-based agents interact with the environment to derive adaptive scheduling policies [7].Despite these advances, existing DRL-based scheduling approaches still face two prominent challenges. First, the curse of dimensionality arises as task and server numbers grow, causing an exponential increase in state–action space complexity. Existing studies mitigate this issue by (1) simplifying allocation scenarios to a few tasks per step [10], (2) redesigning network structures, such as the Branching DQN (BDQN) [11], which uses multiple output branches to independently estimate Q-values for task–server pairs, achieving bilinear scalability—and (3) employing policy-based algorithms like Proximal Policy Optimization (PPO). Nevertheless, policy-based methods often fail to capture dependencies among discrete actions and show lower sample efficiency and slower convergence than value-based ones. Second, managing temporal constraints remains difficult, as most works rely on penalty function-based soft

This research was funded by Science and Technology Program of State Grid under Grant 1400-202455422A-3-5-YS.

979-8-3315-8138-1

constraints [12] that are sensitive to hyperparameters and lack proactive mechanisms to prevent violations. In response to the aforementioned challenges, this paper formulates the dynamic task scheduling problem in data centers as a Constrained Semi-Markov Decision Process (CSMDP) and develops a dual-agent safe reinforcement learning (DA-SRL) framework tailored for heterogeneous tasks and servers. The proposed framework enables adaptive and energy-aware scheduling under uncertainty while ensuring safety and QoS constraints. The main contributions of this work are summarized as follows:

- Propose a CSMDP-based model with temporally extended sub-actions and a dual-agent framework for scalable task–server coordination.
- Develop agent-specific safety mechanisms, where the task-selector employs a constraint-value function for deadline assurance, and the server-allocator applies a shielding strategy to ensure safe action selection.
- Design a CDDQN training framework with unified energy-efficiency rewards and constraint-value guidance to enhance QoS and safety.

II. PROPOSED CONSTRAINED SEMI-MDP FORMULATION

MDP provides a foundation for sequential decision-making in stochastic environments, while SMDP extends this framework with temporally extended actions (options) [13], consisting of multiple sub-actions executed over variable durations. Building upon this, CSMDP aims to maximize cumulative rewards while keeping cumulative safety costs within predefined limits [14]. To ensure tractable computation in large-scale data centers, a *waiting task queue* Q^w is constructed, containing up to N^w tasks (or placeholders) selected from Q^{tot} on a FIFO basis, along with a termination marker β_τ^{sl}. This reduces the number of tasks considered per step from N_τ^{tot} to N^w, maintaining fairness and computational efficiency.

To address the aforementioned problem, the data center task scheduling is formulated as a CSMDP, represented by the seven-tuple $< S, O, A, P, R, C, \gamma >$, where $\gamma \in [0, 1]$ is the discount factor that measures the importance of immediate and future costs. The rest definitions are as follows.

A. State space

Let f_τ^q denotes the *task queue feature vector* that collects the attribute set A^q for all tasks in the waiting task queue, formulated as $f_\tau^q = [A_i^q, \forall i \in Q_\tau^w, \beta_\tau] \in \mathbb{R}^{6N^w+1}$. Each task attribute vector is represented as $A_i^q = [t_i^{id}, t_i^{arr}, t_i^{ddl}, t_i^{pro}, R_i^{CPU}, R_i^{GPU}]$, where these elements correspond respectively to the task id, arrival time, deadline, processing time, required number of CPU/GPU. Similarly, let $f_{k,\tau}^s$ denotes the *server feature vector* that describes the operating status of all servers, expressed as $f_{k,\tau}^s = [z_{k,\tau}^{CPU}, z_{k,\tau}^{GPU}]$, where $z_{k,\tau}^{CPU} \in \mathbb{R}^{N_k^{CPU}}$ denotes the remaining processing time of each CPU core, and $z_{k,\tau}^{GPU} \in \mathbb{R}$ represents the number of unoccupied GPUs of server k at step τ. The overall system state can thus be defined as $s_\tau = [f_\tau^q, f_{k,\tau}^s, \forall k, \lambda_\tau]$, where λ_τ is the electricity price.

B. Option and action space

The option at step τ is denoted as $o_\tau = \langle S^o, \mu_\tau, \beta_\tau^o \rangle$, where S^o defines the set of states in which the option can be executed (in this work, it is available for all states, i.e., $S^o := S$), μ_τ denotes the control policy, and β_τ^o is the termination condition that may be triggered by three cases: completion of all tasks in the waiting queue, task selection being forgone (β^{sl}), or server assignment being forgone (β^{as}). Once option o_τ is initiated, a temporally extended decision-making process begins from step τ, during which actions at each sub-step m are determined by μ, i.e., $a_{\tau,m} = [a_{\tau,m}^{sl}, a_{\tau,m}^{as}] = \mu(s_\tau, o_\tau), \forall m \in M_\tau$. $a_{\tau,m}^{sl}$ and $a^{as}\tau, m$ respectively represent selecting a task from the waiting queue and assigning it to an appropriate server, with dimensions $N^w + 1$ and $K + 1$, where the additional "+1" accounts for the terminate action. The option terminates when all N^w tasks are successfully allocated or when task selection is forgone, resulting in a state transition from s_τ to $s_{\tau+1}$.

C. State and sub-process transitions

There are two types of transitions: *state transitions* and *sub-process transitions*. A state transition from step τ to $\tau + 1$ occurs when an option terminates, involving updates to the task queue feature f_τ^q and server status feature $f_{k,\tau}^s$. The total queue $Q_{\tau+1}^{tot}$ incorporates new arrivals, and the waiting queue is refreshed on a FIFO basis, while server features update the remaining processing times of CPUs and GPUs. Sub-process transitions from m to $m + 1$ update $f_{\tau,m}^q$ and $f_{k,\tau,m}^s$ within an option: $f_{\tau,m}^q$ marks the selected task (from $a_{\tau,m}^{sl}$) as empty, and $f_{k,\tau,m}^s$ adjusts the assigned server (from $a_{\tau,m}^{as}$) to reflect the updated CPU/GPU load.

D. Reward and cost function

As IT equipment and cooling systems account for nearly 90% of data center energy use [15], the total energy consumption is modeled as the sum of these two main components. The reward r_τ is defined as the immediate negative cost of the data center at time step τ, calculated as $r_\tau = -\lambda_\tau(P_\tau^{IT} + P_\tau^{CL})$.

$$P_t^{IT} = \sum\nolimits_{k=1}^{K} (P_{k,t}^{CPU} + P_{k,t}^{GPU}) \tag{1}$$

IT energy is modeled separately for CPUs and GPUs, using linear CPU utilization models [16] and GPU models incorporating utilization and dynamic voltage/frequency scaling [17].

$$P_{k,t}^{CPU} = (P_k^{full} - P_k^{idle}) \cdot u_{k,t}^{CPU} + P_k^{idle} \tag{2}$$

$$P_{k,t}^{GPU} = c_k \cdot f_k^{GPU} \cdot V_k^2 \cdot u_{k,t}^{GPU} \tag{3}$$

where $P_{k,t}^{CPU}, P_{k,t}^{GPU}$ are the power demand of CPU/GPU of server k; P_k^{full}, P_k^{idle} are IT power demand of server k in idle/fully loaded state; $u_{k,t}^{CPU}, u_{k,t}^{GPU}$ are CPU/GPU utilization rate of server k; c_k, f_k^{GPU}, V_k are GPU power calculation factor, rated GPU operating frequency and GPU power supply voltage of server k.

The cooling system is further modeled to capture thermal coupling effects on energy consumption [18].

$$P_t^{CL} = \left(\eta^h P_t^{IT} + k^{et} s^{dc}\right) / k^{per} \tag{4}$$

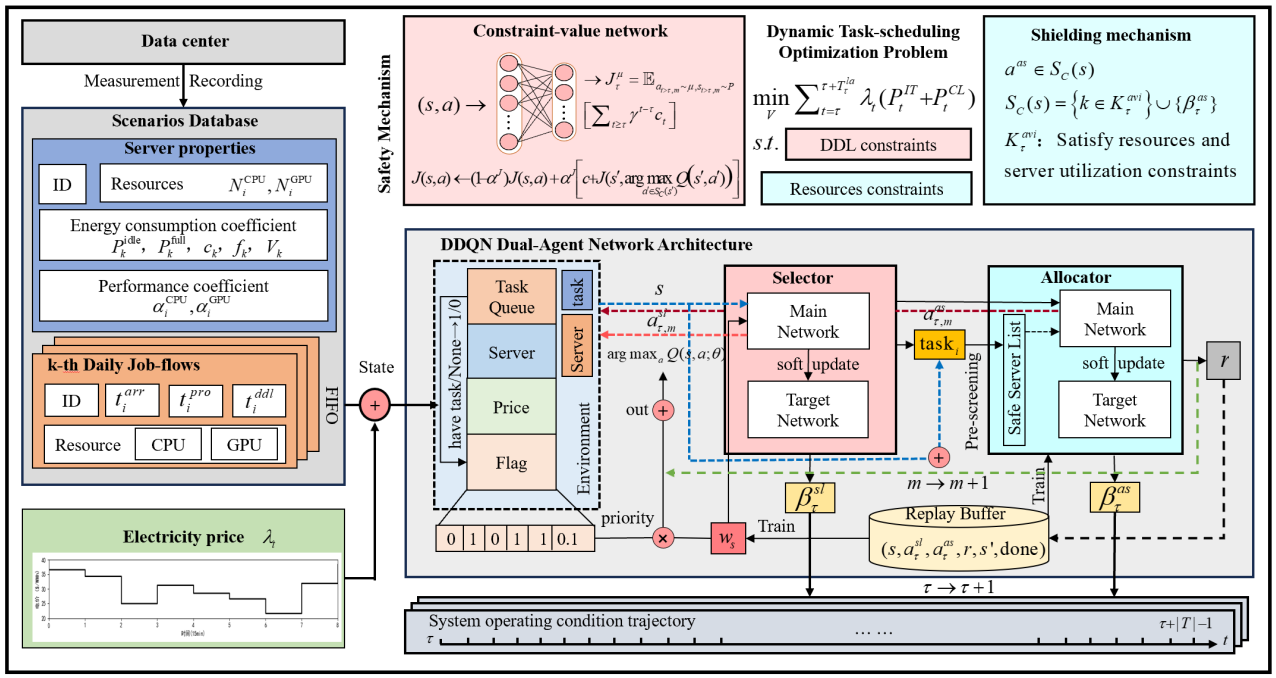

Fig. 1. Overall architecture of the proposed CSMDP-based dynamic task scheduling framework.

where η^h is the ratio of heat dissipation to power demand, $+k^{et}$ is the environmental temperature coefficient, s^{dc} is the projected area of all IT equipment of the DC, and k^{per} is the performance coefficient of the air conditioning equipment.

The cost C pertains to task deadline constraint violations, in which the task selector agent identifies the set of servers that are available to process task i at step τ as $K_\tau^{avi} = \{k : S_{k,\tau}^{CPU} + R_i^{CPU} - N_k^{GPU} > 0, S_{k,\tau}^{GPU} + R_i^{GPU} - N_k^{GPU} > 0\}$, where $S_{k,\tau}^{CPU}, S_{k,\tau}^{GPU}$ denote the number of currently occupied CPU or GPU resources, and N_k^{CPU}, N_k^{GPU} represent number of CPU cores and GPUs of server k.

$$C_{i,\tau}^{ddl} = \frac{1}{|K_\tau^{avi}|} \sum_{k \in K_\tau^{avi}} \frac{t_i^{pro}}{\alpha_k} + \tau - t_i^{ddl} \leq \epsilon_i, \forall i \in Q_\tau^w \tag{5}$$

where α_k are CPU performance factor of server k. ϵ_i is the tolerable threshold.

III. PROPOSED DA-SRL ALGORITHM

A. Dual-agent architecture

To address the action space dimensionality challenge, the task allocation process is decomposed into two cooperative stages, task selection and resource allocation, which managed by the selector and allocator agents. The selector chooses a task or a null action according to the current system state, while the allocator assigns the selected task to a feasible server that satisfies resource constraints. As illustrated in Fig. 1, server and task flow data are retrieved from the data center, together with real-time electricity price information. The system state vector comprises four components, where the flag indicates task priorities in Q^w based on urgency. At each time step τ, the selection and allocation actions are iteratively executed, updating sub-process states until the transition condition is met.

B. CDDQN-base dynamic task scheduling

This study leverages the DDQN algorithm [19], augmented with long-term safety evaluation models, to address multi-step constraints arising from task deadline requirements in dynamic task scheduling.

1) Constraint-value network for selector: We construct the action-value (Q-value) and constraint-value (J-value) functions for the task selection agent to handle the soft constraint. The core concept of the proposed CDDQN algorithm is to define a safe action set $S_C(s)$ that filters out unsafe actions during Q- and J-value evaluation and updating. Since a higher degree of constraint violation indicates greater task urgency, these tasks are assigned higher selection priorities. Accordingly, given the task queue feature f_τ^q, the priority of each unfinished task in Q^w is determined based on its corresponding J-value:

$$\phi(a) = \sigma\left(J(f_\tau^q, a)\right), \forall a \in S_\tau^{ne} \tag{6}$$

where $\sigma(x_i) = e^{x_i} / \sum_{j=1}^n x_j$ denotes the *softmax* operation and $S_\tau^{ne} = \{i \in Q_\tau^w : A_i^q \neq 0\} \subseteq Q_\tau^w$ denotes a subset of Q_τ^w excluding empty placeholders. The Q-value and J-value functions of the task selection agent can be updated as:

$$\begin{aligned} Q^{sl}(s,a) \leftarrow (1-\alpha^Q) Q^{sl}(s,a) + \\ \alpha^Q \left[r + \gamma Q^{sl}\left(s', \underset{a' \in S^{ne}}{\operatorname{argmax}} Q^{sl}(s',a')\phi(a')\right)\right] \end{aligned} \tag{7}$$

$$\begin{aligned} J^{sl}(s,a) \leftarrow (1-\alpha^J) J^{sl}(s,a) + \\ \alpha^J \left[c + \gamma J^{sl}\left(s', \underset{a' \in S^{ne}}{\operatorname{argmax}} Q^{sl}(s',a')\phi(a')\right)\right] \end{aligned} \tag{8}$$

where $c = c_\tau$ is the total violation of option o_τ:

$$c_\tau = \sum_{i \in S_\tau^{ne}} \left(\left[C_{i,\tau}^{ddl} \right]^+ \right) \tag{9}$$

2) Shielding mechanism for allocator: For the server allocation agent, resource requirements and utilization limits are treated as hard constraints that must be satisfied at each decision step. Accordingly, a shielding mechanism is employed through a safe action set $S_C(s)$ to filter out infeasible actions. A unified reward signal is used to update the Q-value networks of both the task selection and server allocation agents, fostering coordinated decision-making and enhancing overall energy cost efficiency. The safe action set $S_C(s)$ can be given as:

$$S_C(s) = \left\{k \in K_\tau^{avi}\right\} \cup \left\{\beta_\tau^{as}\right\} \tag{10}$$

IV. CASE STUDY AND RESULTS

A. Case configuration

To evaluate the proposed method, the Alibaba Trace_2020 open dataset [20] was utilized, which captures real-world production workloads containing mixed training and inference tasks from large-scale machine learning services. The dataset includes detailed attributes such as task arrival times, estimated execution durations, deadlines and CPU/GPU resource requests. Experiments are conducted over a daily scheduling horizon with 15-minute intervals, where 4–10 tasks are randomly sampled from the task pool per interval. Set up five types of heterogeneous servers, each characterized by distinct computing capacities, energy consumption parameters and performance parameters. In total, 500 training and 100 testing scenarios are constructed. The electricity price data are obtained from the EIA dataset at a 15-minute resolution [21].

The length of the waiting queue is set to $N^w = 10$. The selector and allocator employ Q-value network, all of which feature 2 fully-connected hidden layers, each layer has 256 neurons with a dropout rate of 0.5 and utilizes LeakyReLU as the activation function, and the respective target network adopts the same architecture with a soft update rate of 10^{-4}. All network parameters are updated using the Adam optimizer, with an initial learning rate of 10^{-6} and gradually decreasing. When selecting actions, the epsilon greeny strategy is adopted to balance the trade-off between exploration and utilization. The initial value of ϵ is set to 0.5 and gradually decreases with the increase of training rounds. The discount rate γ is set to 0.95. The examined methods are implemented using PyTorch v2.4.1, optimization problems are solved using Gurobi v12.0.0. Experiments are conducted on a computer with AMD Ryzen 99900X 12-core CPU, NVIDIA GeForce RTX 4090D GPU and 64GB of RAM.

B. Performance evaluation

As illustrated in Fig. 2, the proposed scheduling method demonstrates strong adaptability to fluctuating electricity prices and varying task arrivals. During low-price periods (around t=50-70), the total number of active tasks increases significantly, indicating that non-urgent workloads are effectively shifted to cost-efficient intervals. Conversely, when electricity prices rise, the system limits task execution to essential workloads, thereby reducing operational costs. Furthermore, tasks are preferentially assigned to servers with higher performance-to-cost efficiency, enabling improved resource utilization while maintaining an overall balanced load distribution. These results confirm that the proposed method achieves coordinated optimization between task scheduling and energy management, ensuring cost-effectiveness and system stability under dynamic conditions.

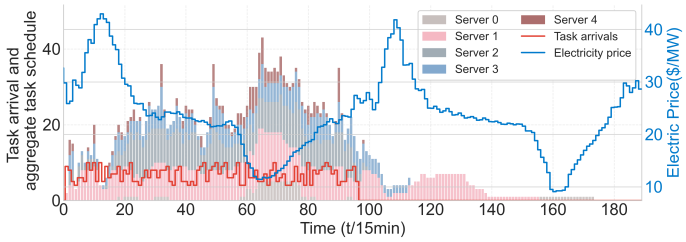

Fig. 2. Temporal variation of task arrivals, aggregated task allocation across servers, and electricity prices under the proposed method.

C. Comparison with benchmarks

We evaluate our approach against six baseline methods: (1) Theoretical Optimal: The task scheduling problem is formulated as a MILP and solved using Gurobi to obtain the global optimum, serving as a performance benchmark; (2) FIFO: Tasks are scheduled in the order of arrival. This method is simple, easy to implement, and ensures fairness, but it does not account for task priority, which may lead to inefficiency for long-running jobs; (3) Genetic Algorithm (GA): Task schedules are iteratively optimized through selection, crossover, and mutation of candidate solutions; (4) BDQN: Employs a multi-branch Q-network, with each branch evaluating the Q-value for a specific action; (5) Proximal Policy Optimization (PPO): which models discrete task allocation actions with categorical distribution; (6) DA-RL: An ablation variant of our method without safety mechanism to assess the contribution of reducing constraints violation.

Fig. 3(a) presents the learning curves obtained using sliding-window averaging. The theoretical optimal value is 107.22$, whereas FIFO exhibits the poorest performance, deviating by 28.65%. The heuristic GA achieves a closer result, with a deviation of 14.17% from the optimal solution. Among reinforcement learning approaches, the proposed DA-SRL method attains the best performance, differing by only 1.07% from the theoretical optimum. In contrast, its variant without the security mechanism, BDQN, and PPO exhibit deviations of 3.85%, 12.87%, and 6.16%, respectively. Regarding convergence behavior, the proposed method reaches stability after approximately 350 steps, while DA-RL and BDQN converge near 400 steps. PPO demonstrates a temporary decline around 900 steps before converging at about 1200 steps. Overall, the proposed DA-SRL achieves the closest approximation to the theoretical optimum, with faster convergence and a more stable learning trajectory than all benchmarks.

Fig. 3(b) illustrates the penalty associated with task deadline violations. The tolerance threshold is set to 0.25, corresponding to the minimum time allocation granularity. Except for the theoretical optimum and FIFO, only the proposed DA-SRL maintains a final violation value within the threshold

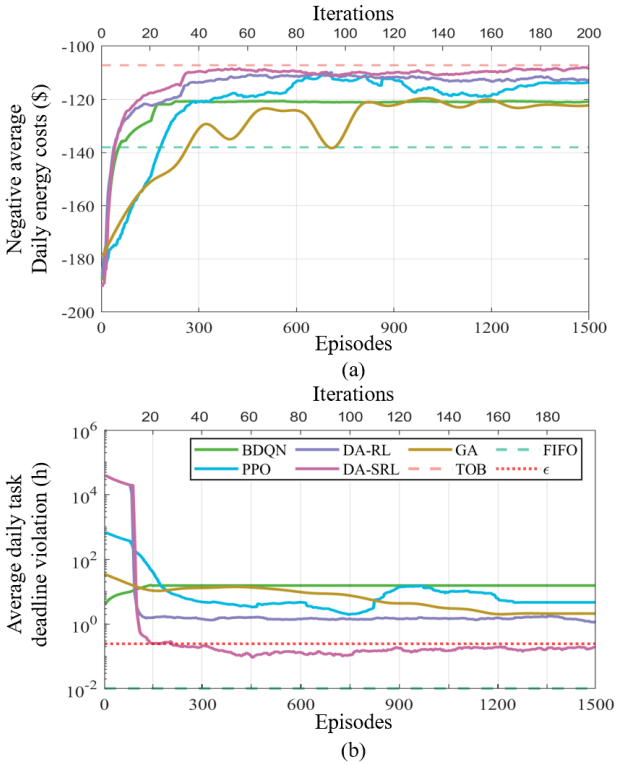

Fig. 3. Episodic (negative) average daily energy costs (a) and task deadline violations (b) under examined methods.

(0.18). By contrast, the variant without the security mechanism records a final violation amount of 1.11, exceeding the limit by 344%. The remaining algorithms show considerably larger deviations, underscoring the effectiveness of the proposed security mechanism in ensuring deadline compliance and system robustness.

V. CONCLUSIONS

This paper presents a Dual-Agent Safe Reinforcement Learning (DA-SRL) framework for energy-efficient dynamic task scheduling in data centers. Formulated as a Constrained Semi-Markov Decision Process, the framework jointly optimizes energy cost and task deadlines through coordinated task selection and server allocation. The safety mechanism, integrating a constraint-value network and shielding strategy, effectively enforces time and resource constraints. Experimental results on the Alibaba production dataset show that DA-SRL achieves energy costs only 1.07% above the theoretical optimum, with the fastest convergence and the lowest deadline violations among all methods. These results demonstrate that DA-SRL provides superior scheduling optimality, convergence stability, and safety assurance, offering a promising solution for sustainable data center operation.

REFERENCES

[1] M. . Company. (2024) Ai power: Expanding data center capacity to meet growing demand. Accessed: Oct. 25, 2025. [Online]. Available: https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/ai-power-expanding-data-center-capacity-to-meet-growing-demand

[2] I. E. Agency. (2024) Energy demand from artificial intelligence. Accessed: Oct. 25, 2025. [Online]. Available: https://www.iea.org/reports/energy-and-ai/energy-demand-from-ai

[3] D. Ma, Y. Ye, Y. Wu, D. Xu, Z. Ding, and G. Strbac, "Bi-level optimisation model for harvesting spatial-temporal load shifting flexibility of data centres using endogenously formed locational price signal," *IET Smart Grid*, vol. 8, no. 1, p. e70020, 2025.

[4] X. Guo, Y. Che, Z. Zheng, and J. Sun, "Multi-timescale optimization scheduling of interconnected data centers based on model predictive control," *Front. Energy*, vol. 18, no. 1, pp. 28–41, 2024.

[5] D. Olliaro, A. Anggraito, M. A. Marsan, S. Balsamo, and A. Marin, "The impact of service demand variability on data center performance," *IEEE Trans. Parallel Distrib. Syst.*, 2024.

[6] Z. Ding, Y.-C. Tian, Y.-G. Wang, W. Zhang, and Z.-G. Yu, "Progressive-fidelity computation of the genetic algorithm for energy-efficient virtual machine placement in cloud data centers," *Applied soft computing*, vol. 146, p. 110681, 2023.

[7] R. Chen, B. Liu, W. Lin, J. Lin, H. Cheng, and K. Li, "Power and thermal-aware virtual machine scheduling optimization in cloud data center," *Future Generation Computer Systems*, vol. 145, pp. 578–589, 2023.

[8] A. G. Jakwa, A. Y. Gital, S. Boukari, and F. U. Zambuk, "Performance evaluation of hybrid meta-heuristics-based task scheduling algorithm for energy efficiency in fog computing," *International Journal of Cloud Applications and Computing (IJCAC)*, vol. 13, no. 1, pp. 1–16, 2023.

[9] Y. Li, C. Yu, M. Shahidehpour, T. Yang, Z. Zeng, and T. Chai, "Deep reinforcement learning for smart grid operations: Algorithms, applications, and prospects," *Proceedings of the IEEE*, vol. 111, no. 9, pp. 1055–1096, 2023.

[10] D. Yi, X. Zhou, Y. Wen, and R. Tan, "Efficient compute-intensive job allocation in data centers via deep reinforcement learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 6, pp. 1474–1485, 2020.

[11] Y. Sun and Q. He, "Joint task offloading and resource allocation for multi-user and multi-server mec networks: A deep reinforcement learning approach with multi-branch architecture," *Eng. Appl. Artif. Intell.*, vol. 126, p. 106790, 2023.

[12] W. Liu, Y. Yan, Y. Sun, H. Mao, M. Cheng, P. Wang, and Z. Ding, "Online job scheduling scheme for low-carbon data center operation: An information and energy nexus perspective," *Appl. Energy*, vol. 338, p. 120918, 2023.

[13] R. S. SUTTON, D. PRECUP, and S. SINGH, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artif. Intell.*, vol. 112, no. 1-2, pp. 181–211, 1999.

[14] Y. Zhang, Y. Ye, J. Hu, H. Hu, X. Zhang, and D. Xu, "Constrained semi-MDP formulation and perception-enhanced safe policy learning for efficient dynamic task scheduling of data centers," *IEEE Trans. Smart Grid*, to be published.

[15] M. Chen, C. Gao, M. Song *et al.*, "Internet data centers participating in demand response: A comprehensive review," *Renew. Sustain. Energy Rev.*, vol. 117, p. 109466, 2020.

[16] H. Yuan, H. Liu, J. Bi, and M. Zhou, "Revenue and energy cost-optimized biobjective task scheduling for green cloud data centers," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, pp. 817–830, 2020.

[17] Y. Huang, B. Guo, and Y. Shen, "Gpu energy optimization based on task balance scheduling," *J. Syst. Archit.*, vol. 107, p. 101808, 2020.

[18] C. Nadjahi, H. Louahlia, and S. Lemasson, "A review of thermal management and innovative cooling strategies for data center," *Sustain. Comput.: Inform. Syst.*, vol. 19, pp. 14–28, 2018.

[19] H. VAN HASSELT, A. GUEZ, and D. SILVER, "Deep reinforcement learning with double q-learning," in *Pro. AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[20] Q. Weng, W. Xiao, Y. Yu, W. Wang, C. Wang, J. He, Y. Li, L. Zhang, W. Lin, and Y. Ding, "{MLaaS} in the wild: Workload analysis and scheduling in {Large-Scale} heterogeneous {GPU} clusters," in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, 2022, pp. 945–960.

[21] U.S. Energy Information Administration. (2025) Electricity. U.S. Department of Energy. [Online]. Available: https://www.eia.gov/electricity/wholesalemarkets/data.php?rto=caiso