Robust Dynamic Material Handling via Adaptive Constrained Evolutionary Reinforcement Learning

Chengpeng Hu[1], Ziming Wang[2], Bo Yuan[2], Jialin Liu[3], Chengqi Zhang[4], and Xin Yao[3]

[1] Eindhoven University of Technology, Eindhoven, the Netherlands
[2] Southern University of Science and Technology, Shenzhen, China
[3] Lingnan University, Hong Kong SAR, China
[4] Hong Kong Polytechnic University, Hong Kong SAR, China

Abstract. Dynamic material handling (DMH) involves the assignment of dynamically arriving material transporting tasks to suitable vehicles in real time for minimising makespan and tardiness. In real-world scenarios, historical task records are usually available, which enables the training of a decision policy on multiple instances. Recently, reinforcement learning has been applied to solve DMH. Due to the occurrence of dynamic events such as new tasks, adaptability is highly required. Solving DMH is challenging since constraints including task delay should be satisfied. A feedback is received only when all tasks are served, which leads to sparse reward. Besides, making the best use of limited computational resources and historical records for training a robust policy is crucial. The time allocated to different problem instances would highly impact the learning process. To tackle those challenges, this paper proposes a novel adaptive constrained evolutionary reinforcement learning (ACERL) approach, which maintains a population of actors for diverse exploration. ACERL accesses each actor for tackling sparse rewards and constraint violation to restrict the behaviour of the policy. Moreover, ACERL adaptively selects the most beneficial training instances for improving the policy. Extensive experiments on eight training and eight unseen test instances demonstrate the outstanding performance of ACERL compared with several state-of-the-art algorithms. Policies trained by ACERL can schedule the vehicles while fully satisfying the constraints. Additional experiments on 40 unseen noised instances show the robust performance of ACERL. Cross-validation further presents the overall effectiveness of ACREL. Besides, a rigorous ablation study highlights the coordination and benefits of each ingredient of ACERL.

Keywords: Dynamic material handling · constrained optimisation · evolutionary reinforcement learning· natural evolution strategy · experience-based optimisation.

This paper is accepted by IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS on 11 June 2025.

In modern smart logistics such as flexible manufacturing and warehouses, automated guided vehicles (AGVs) play a key role in dynamic material handling (DMH) [15, 14], where fleets of AGVs must serve transport tasks arriving in real time. These tasks typically involve moving materials between workstations, with the objective of minimising makespan and tardiness under dynamic events (e.g., breakdowns, new tasks) and strict operational constraints [12, 9, 6].

Historical task records (e.g., AGV states) are often available [7, 8]. Leveraging multiple instances for training improves generalisation but introduces trade-offs across instances and requires careful selection under limited computation [4]. Moreover, task contributions to system performance are hard to evaluate due to sparse feedback, as performance is only observable after task completion. Classical dispatching rules [2, 13] are simple and fast but lack adaptability. Search-based methods, such as evolutionary algorithms [3], restart from scratch after each event, leading to long search times unsuited for real-time DMH. Reinforcement learning (RL) offers faster online responses [10, 7, 8] by modelling DMH as a Markov decision process (MDP). However, RL requires carefully designed reward functions [5], which may not align with the true objective and often fail to generalise across instances. Handling constraints (e.g., AGV availability, task delays) remains challenging. Re-sampling for feasibility [7] or reward penalties [11, 1] are often ineffective. Constrained MDP approaches such as RCPOM [6] address feasibility but still suffer from sparse feedback and limited robustness across instances due to computational budget constraints.

In this paper, we consider DMH problem with uncertainties and sparse feedback. We proposed a robust adaptive constrained evolutionary reinforcement learning (ACERL) approach to achieve real-time decision-making in DMH with adaptability and effectiveness. ACERL leverages natural gradient ascent to update its parameters. Adaptive instance sampler keeps choosing the training instance from which the policy benefits the most for policy improvement. Reasonable computational resource allocation is allowed. To balance the rewards and penalties, we present intrinsic stochastic ranking with rank-based fitness. Instead of common reward-based, real-valued fitness metrics, we incorporate rank-based fitness to estimate the natural gradient, which implies the optimisation direction of both maximising rewards and constraint satisfaction at the same time. Intrinsic stochastic ranking provides an independent ranking for different instances based on rewards and penalties, which enables a bias for adaptive instance selection. Extensive experiments show that ACERL outperforms advanced reinforcement learning methods, constrained reinforcement learning methods and classic dispatching rules on eight training instances and eight test instances in terms of maximising rewards with constraint satisfaction. ACERL not only achieves the best makespan, but also fully satisfies tardiness constraints on nearly all instances. Besides, ACERL is evaluated on five datasets with a total of 40 noised instances to simulate real-world scenarios. ACERL presents a robust and outstanding performance on all datasets. Leave-one-out cross-validation is conducted on ACERL by splitting the training dataset into subsidiary training and test datasets, and presents the overall effectiveness of ACERL.

References

1. Achiam, J., Held, D., Tamar, A., Abbeel, P.: Constrained policy optimization. In: International Conference on Machine Learning. pp. 22–31. PMLR (2017)
2. Blackstone, J.H., Phillips, D.T., Hogg, G.L.: A state-of-the-art survey of dispatching rules for manufacturing job shop operations. The International Journal of Production Research **20**(1), 27–45 (1982)
3. Chryssolouris, G., Subramaniam, V.: Dynamic scheduling of manufacturing job shops using genetic algorithms. Journal of Intelligent Manufacturing **12**(3), 281–293 (2001)
4. Dennis, M., Jaques, N., Vinitsky, E., Bayen, A., Russell, S., Critch, A., Levine, S.: Emergent complexity and zero-shot transfer via unsupervised environment design. Advances in Neural Information Processing Systems **33**, 13049–13061 (2020)
5. Dewey, D.: Reinforcement learning and the reward engineering principle. In: 2014 AAAI Spring Symposium Series. pp. 1–4 (2014)
6. Hu, C., Wang, Z., Liu, J., Wen, J., Mao, B., Yao, X.: Constrained reinforcement learning for dynamic material handling. In: International Joint Conference on Neural Networks. pp. 1–9 (2023)
7. Hu, H., Jia, X., He, Q., Fu, S., Liu, K.: Deep reinforcement learning based AGVs real-time scheduling with mixed rule for flexible shop floor in industry 4.0. Computers & Industrial Engineering **149**, 106749 (2020)
8. Jeong, Y., Agrawal, T.K., Flores-García, E., Wiktorsson, M.: A reinforcement learning model for material handling task assignment and route planning in dynamic production logistics environment. Procedia CIRP **104**, 1807–1812 (2021)
9. Kaplanoğlu, V., Şahin, C., Baykasoğlu, A., Erol, R., Ekinci, A., Demirtaş, M.: A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles (AGV) in manufacturing systems by considering AGV breakdowns. International Journal of Engineering Research & Innovation **7**(2), 32–38 (2015)
10. Li, M.P., Sankaran, P., Kuhl, M.E., Ganguly, A., Kwasinski, A., Ptucha, R.: Simulation analysis of a deep reinforcement learning approach for task selection by autonomous material handling vehicles. In: Winter Simulation Conference. pp. 1073–1083. IEEE (2018)
11. Ng, A.Y., Harada, D., Russell, S.: Policy invariance under reward transformations: Theory and application to reward shaping. In: International Conference on Machine Learning. vol. 99, pp. 278–287 (1999)
12. Ouelhadj, D., Petrovic, S.: A survey of dynamic scheduling in manufacturing systems. Journal of scheduling **12**(4), 417–431 (2009)
13. Sabuncuoglu, I.: A study of scheduling rules of flexible manufacturing systems: A simulation approach. International Journal of Production Research **36**(2), 527–546 (1998)
14. Singh, N., Dang, Q.V., Akcay, A., Adan, I., Martagan, T.: A matheuristic for AGV scheduling with battery constraints. European Journal of Operational Research **298**(3), 855–873 (2022)
15. Zou, W.Q., Pan, Q.K., Wang, L.: An effective multi-objective evolutionary algorithm for solving the AGV scheduling problem with pickup and delivery. Knowledge-Based Systems **218**, 106881 (2021)

Robust Dynamic Material Handling via Adaptive Constrained Evolutionary Reinforcement Learning

Chengpeng Hu, *Student Member, IEEE*, Ziming Wang, *Student Member, IEEE*, Bo Yuan, *Member, IEEE*, Jialin Liu, *Senior Member, IEEE*, Chengqi Zhang, *Life Senior Member, IEEE*, and Xin Yao, *Fellow, IEEE*

***Abstract*—Dynamic material handling (DMH) involves the assignment of dynamically arriving material transporting tasks to suitable vehicles in real time for minimizing makespan and tardiness. In real-world scenarios, historical task records are usually available, which enables the training of a decision policy on multiple instances consisting of historical records. Recently, reinforcement learning (RL) has been applied to solve DMH. Due to the occurrence of dynamic events such as new tasks, adaptability is highly required. Solving DMH is challenging since constraints, including task delay, should be satisfied. A feedback is received only when all tasks are served, which leads to sparse reward. Besides, making the best use of limited computational resources and historical records for training a robust policy is crucial. The time allocated to different problem instances would highly impact the learning process. To tackle those challenges, this article proposes a novel adaptive constrained evolutionary RL (ACERL) approach, which maintains a population of actors for diverse exploration. ACERL accesses each actor for tackling sparse rewards and constraint violation to restrict the behavior of the policy. Moreover, ACERL adaptively selects the most beneficial training instances for improving the policy. Extensive experiments on eight training and eight unseen test instances demonstrate the outstanding performance of ACERL compared with several state-of-the-art algorithms. Policies trained by ACERL can schedule the vehicles while fully satisfying the constraints. Additional experiments on 40 unseen noised instances show the robust performance of ACERL. Cross validation further presents the overall effectiveness of ACREL. Besides, a rigorous ablation study highlights the coordination and benefits of each ingredient of ACERL.**

***Index Terms*—Constrained optimization, dynamic material handling (DMH), evolutionary reinforcement learning (ERL), experience-based optimization, natural evolution strategy (ES).**

Received 22 January 2024; revised 2 November 2024 and 30 March 2025; accepted 11 June 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFE0106300, in part by the National Natural Science Foundation of China under Grant 62250710682 and Grant 62476119, in part by Guangdong Major Project of Basic and Applied Basic Research under Grant 2023B0303000010, and in part by the Internal Grants of Lingnan University. *(Corresponding author: Jialin Liu.)*

Chengpeng Hu is with the Department of Industrial Engineering and Innovation Sciences, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands.

Ziming Wang and Bo Yuan are with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China.

Jialin Liu and Xin Yao are with the School of Data Science, Lingnan University, Hong Kong, SAR, China (e-mail: jialin.liu@ln.edu.hk).

Chengqi Zhang is with the Department of Data Science and Artificial Intelligence, The Hong Kong Polytechnic University, Hong Kong, SAR, China.

This article has supplementary downloadable material available at https://doi.org/10.1109/TNNLS.2025.3582299, provided by the authors.

Digital Object Identifier 10.1109/TNNLS.2025.3582299

I. Introduction

IN MODERN smart logistics such as flexible manufacturing systems and warehouse floors, the need of automated guided vehicles (AGVs) has grown fast. Dynamic material handling (DMH) [1], [2] involves scheduling a fleet of AGVs to serve dynamically arriving transporting tasks in real time. Transporting tasks typically include lifting and moving material between workstations using AGVs. The purpose of the DMH is to minimize makespan and tardiness, i.e., the maximal task finishing time and delay of tasks, in response to unexpected events and strict problem constraints. Due to the unexpected occurrence of dynamic events such as vehicle breakdowns and new tasks in DMH, the scheduling plans have to be determined frequently in real time [3], [4]. Problem constraints such as the delay of tasks should be also guaranteed to ensure the operation of manufacturing [5].

Usually, some historical completed task records, e.g., previous task contexts and AGV information, are available in real life [6], [7]. Training a decision policy using multiple instances consisting of historical records enhances the generalization, but the tradeoff across different instances is introduced. Selecting suitable instances at different training stages is crucial with consideration of limited computational budgets [8]. It is also hard to determine the unique contribution of each task assignment to the performance of the entire system since the overall performance can only be determined once all tasks are completed, which leads to sparse feedback.

Dispatching rule is a classic and common method for handling DMH [9], [10]. Simple yet practicable mechanism makes them easy to deploy to real-world operations quickly. However, this simplicity leads to limited performance and poor adaptability to real-world scenarios. Search-based methods, such as evolutionary algorithms (EAs), have been used for handling DMH when dynamic events occur, which restarts the search for a new solution [11]. However, a new search needs to be run from scratch for every single new scenario. The long search time merely meets the requirement of a fast response.

Recently, reinforcement learning (RL) has made some promising progress in DMH [6], [7], [12] by providing prompt online responses with trained policies. In the RL setting, DMH is formulated as a Markov decision process (MDP), where the reward function is manually constructed based on the makespan [6]. It takes massive effort and human knowledge to construct a "good" reward function since RL-based methods [7], [12] often suffer from sparse feedback [13]. However, those designed reward functions cannot guarantee consistency with the original objective function and fail to generalize on multiple instances.

How to handle constraints such as the availability of AGVs and task delays in DMH is crucial. The work of [6] repeatedly sampled the task assignment until the sampled assignment is feasible. However, ensuring adherence to feasible task assignments is hard by resampling. It is also not plausible to satisfy the long-term task delay via simply penalizing rewards [14], [15]. To handle constraints, the work of [5] formulated the DMH problem as a constrained MDP (CMDP) [16] and proposed reward-constrained policy optimization with masking (RCPOM) approach. However, RCPOM suffers from sparse feedback from both reward and constraint sides. The robust performance across multiple DMH problem instances is not fully addressed. A policy may perform well on the given instances while failing on others, as it might not have been sufficiently trained on certain instances with limited computational budget [5].

In this article, we consider the DMH problem with uncertainties and sparse feedback in the context of multiple instances. We proposed a robust adaptive constrained evolutionary RL (ACERL) approach to achieve real-time decision-making in DMH with adaptability and effectiveness. Unlike regular RL methods, ACREL inherits the gradient-free characteristic of natural evolution strategies (NES) [17], which tackles sparse feedback in both rewards and constraint violation penalties intuitively. No gradient calculation related to the backpropagation is required. ACREL maintains a population of actors for diverse exploration. Intrinsic stochastic ranking (ISR) using rank-based fitness is proposed to evaluate the actors. Those rank-based fitness values facilitate the estimation of natural gradients, which implies the optimization direction for maximizing rewards and satisfying constraints simultaneously. The limited computational budget leads to a tradeoff in the utilization of multiple instances consisting of historical records for training a robust policy. To tackle the tradeoff, we propose a novel training mode that adaptively selects a subset of training instances with which a population of actors interacts. The selected instances introduce a bias in estimating the natural gradients after being ranked by the ISR, which contributes to a remarkable performance among multiple unseen instances.

The main contributions of this article are summarized as follows.

1) We propose ACERL to address DMH with uncertainties and sparse feedback. ACERL balances the reward maximization and constraint satisfaction, even when the agent receives only limited feedback. Given multiple instances with different contexts, ACERL still provides robust scheduling solutions in real time.
2) We experimentally demonstrate the limitations of using a single instance or randomly selecting from multiple instances for training decision policies. Under this observation, we propose an adaptive training mode that deploys an adaptive instance sampler (AIS) to break the tradeoff of multiple instances. The limited computational resources are allocated by adaptively choosing the most beneficial instances for training a robust policy.
3) ACERL requires no domain knowledge and makes no assumption on reward or constraint-related functions. It is suitable to solve real-world problems with sparse feedback and constraints.
4) Extensive experiments show that ACERL outperforms five state-of-the-art algorithms on eight training and eight unseen test instances. Additional experiments on 40 instances with uncertainties and leave-one-out cross validation between heterogeneous training instances present its robust performance, in terms of maximizing the reward and satisfying constraints. The functionality and coordination of each ingredient in ACERL are further demonstrated through a rigorous ablation study.

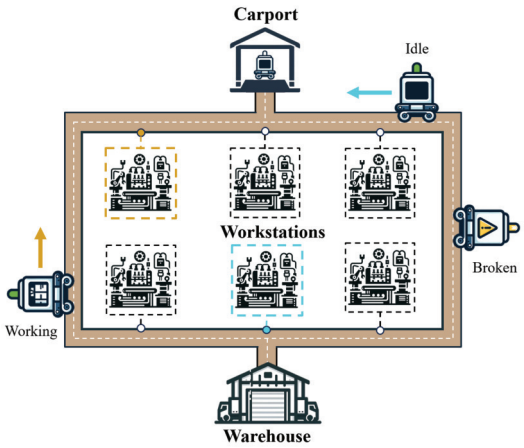


Fig. 1. Illustration of DMH with AGVs that are in one of the three possible states *Idle*, *Working*, and *Broken* at different time steps.

The remainder of this article is organized as follows. Section II introduces the DMH problem and related work. Section III details the proposed ACERL and its components. Section V presents the experimental studies. Section VI concludes and discusses some future directions.

II. Background

Section II-A describes and formulates DMH. Then, related works on scheduling DMH and evolutionary RL (ERL) are presented in Sections II-B and II-C, respectively.

A. Dynamic Material Handling

DMH is widely found in manufacturing, warehouse, and other systems with transporting scenarios [1], [2]. The problem is involved with transporting some goods from their storage sites to some delivery sites with regard to dynamic events, including new tasks and vehicle breakdowns. A policy is responsible for assigning dynamically arriving transporting tasks to AGVs of different types in real time. The objectives are minimizing the makespan and restricting the task delay within a tolerant threshold.

1) Problem Description: Fig. 1 presents an example of a material handling scenario on a manufacturing floor. The manufacturing floor is formed as a graph $G(\mathcal{L}, \mathcal{T})$, where $\mathcal{L}$ and $\mathcal{T}$ denote the sets of sites and paths, respectively. Sites are stop or working points like pickup points, delivery points, and parking points for AGVs. Each workstation serves as either a pickup or delivery point, where AGVs collect some material or transport carried material. The warehouse can only be the pickup point. Task $u_1, u_2, \ldots$ can be released at any moment. The total number of tasks m is unknown initially. A task u is determined by pickup point $u.s$, delivery point $u.e$, arrival time $u.o$, and expiry time $u.\tau$ when it is released. A fleet of AGVs $\mathcal{V}$ is arranged to complete tasks by a policy π. An AGV has one of the three possible states, namely *Idle*, *Working*, and *Broken*. Only available AGVs, i.e., in *Idle* state, can serve tasks, which

is an *instantaneous constraint*. An AGV can handle one and only one task at once while *Working*. If an AGV is broken, it should release the assigned task and be repaired in place for a certain amount of time $v.rp$ before being available again.

The longest total task finishing time among all AGVs after finishing serving m tasks, *makespan* $F_m(\pi)$, is formulated in the following equation:

$$F_m(\pi) = \max_{v \in \mathcal{V}^\pi} \mathrm{FT}\left(u^v_{|\mathrm{HL}(v)|}, v\right) \tag{1}$$

where $\mathrm{HL}(v)$ refers to the historical list of all tasks served by v and $\mathrm{FT}(u, v)$ is the time of finishing serving task u by v.

The average delay of all tasks, *tardiness* $F_t(\pi)$, is defined as follows:

$$F_t(\pi) = \frac{1}{m} \sum_{v \in \mathcal{V}^\pi} \sum_{i=1}^{|\mathrm{HL}(v)|} \max\left\{\mathrm{FT}\left(u^v_i, v\right) - u^v_i.o - u^v_i.\tau, 0\right\} \tag{2}$$

where $\mathcal{V}^\pi$ denotes the fleet scheduled by policy π and u^v_i represent the task u_i served by AGV v.

2) CMDP Formulation: DMH is formulated as a CMDP [5], [16], denoted by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{C}, \mathcal{P}, \gamma \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, and $\mathcal{R}{:}\mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is the reward function. $\mathcal{C}$ is the penalty functions related to the constraint with $\mathcal{C}{:}\mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$. $\mathcal{P}{:}\mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ is the transition probability function. $\gamma \in (0, 1)$ is the discount factor. In DMH, the state space, $\mathcal{S}$, is encoded by task and AGV information, such as the remaining time before timeout and waiting time of unassigned tasks in the task pool at decision time t. The action space, $\mathcal{A} = \mathcal{D} \times \mathcal{V}_t$, is a hybrid space combining AGV information $\mathcal{V}_t$ at t and dispatching rules $\mathcal{D}$; for instance, it involves deciding a specific dispatching rule to assign a task to a chosen AGV. To maintain consistency between the reward function and objective function, the policy receives only the negative number of the final makespan as its reward, i.e., $-F_m$, while receiving zero at other time. Similarly, the penalty function, $\mathcal{C}$, is constructed based on the tardiness, where the policy gets $-F_t$ once all tasks are served [18].

Policy $\pi(a_t|s_t)$ is the probability of taking action a_t in state s_t at time t. Usually, a cumulative constraint $\mathcal{C} = g(c(s_0, a_0, s_1), \ldots, c(s_t, a_t, s_{t+1}))$ is restricted by a threshold ξ where $c(s, a, s'){:}\mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is a per-step penalty. Let $J^\pi_\mathcal{C}$ denote the expectation of the cumulative constraint, formulated as $J^\pi_\mathcal{C} = \mathbb{E}_{\tau \sim \pi}[\mathcal{C}]$, where $\tau \sim \pi$ denotes a trajectory $(s_0, a_0, s_1, a_1, s_2, \ldots)$ sampled from π. In DMH, tardiness, formulated in (2), is considered as the *cumulative constraint*. The instantaneous constraint considers if an action a_t is legal at state s_t, i.e., only choose available vehicles. The policy π_θ, parameterized by θ, aims at maximizing the discounted cumulative reward while satisfying the constraints, formulated as [16]

$$\max_\theta \; J^{\pi_\theta}_\mathcal{R} = \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t, s_{t+1})\right] \tag{3}$$

$$\text{s.t. } J^\pi_\mathcal{C} \le \xi \tag{4}$$

$$a_t \text{is legal} \quad \forall t = 0, 1, 2, \ldots \tag{5}$$

where ξ represents the constraint threshold.

B. Scheduling DMH

Dispatching rules is a classic approach for DMH. Rules such as first come first serve (FCFS), earliest due date (EDD) first, and nearest vehicle first (NVF), usually have simple handcrafted mechanisms [10]. Due to their simplicity, they can be quickly implemented and deployed in simple manufacturing systems. However, the dispatching rules can hardly be improved and adapted to complex situations. Motivated by the poor generalization of using one single rule at once [9], some work combined multiple dispatching rules to make decisions [19], [20]. However, how to effectively coordinate multiple dispatching rules presents a new challenge.

Iterative methods can also be applied to solve DMH. According to the dynamic events, Iterative methods divide the scheduling problem into several subproblems. The process restarts to optimize each subproblem when a dynamic event occurs. Liu et al. [21] proposed a dynamic framework to schedule a fleet of robots for material transportation. When a new task arrives, the framework is triggered to search for a new scheduling solution with a mixed-integer programming (MIP) model. Yan et al. [22] integrated the MIP model to an iterated algorithm for job shop in material handling with multiple new tasks.

Instead of using an MIP model, population-based search methods have been applied to solve DMH by searching subproblem. Chryssolouris and Subramaniam [11] assumed that the number of tasks is known and the operations of each job may vary. Task assignments are encoded as permutation-based solutions, and then, a genetic algorithm searches for promising operation sequences whenever a dynamic event happens [11]. Umar et al. [23] proposed a hybrid genetic algorithm with a weighted sum fitness function of multiple objectives using random key representation for DMH. Wang et al. [24] optimized travel distance and energy consumption at the same time with the nondominated sorting genetic algorithm. If a trolley breaks or a task is canceled, the optimization restarts [24]. An adaptive parameter adjustment with discrete invasive weed optimization algorithm was investigated by Li et al. [25] to handle the dynamic scheduling, in which multiple dynamic events such as new tasks, emergent tasks, and task cancellation are considered. Although the aforementioned search-based methods may obtain promising solutions for the subproblems triggered by every occurrence of dynamic events, they are limited by the long optimization time [11]. Thus, they hardly perform a fast and adaptive response for some real-world scenarios. Moreover, search-based methods typically require a specific problem representation, which is challenging to directly transfer for optimizing other problems, particularly those involving additional dynamic events and constraints.

It has been witnessed that RL presents a competitive level beyond humans on some sequential decision-making problems, including video games [26], [27], Go [28], and robotic control [29]. RL-based methods have been applied to solve DMH problems for their advantages over sequential decision-making problems. Chen et al. [30] proposed a Q_λ algorithm with forecast information. The RL-based dispatching policy decides the tasks of the dolly train, considering multiple loads. Xue et al. [31] considered a flow shop scenario with multiple AGVs and applied Q-learning to find a suitable match between jobs and vehicles. Kardos et al. [32] optimized the choice of workstations for products, using an RL-based method. Instead of directly choosing tasks, Hu et al. [6] adapted deep Q-learning (DQN) to choose a pair of dispatching rule and AGV, inspired by the work of Chen et al. [20]. The chosen

rule then assigns a waiting task to the paired AGV, aiming at minimizing makespan and delay ratio. Li et al. [33] modeled the scheduling as a multiagent scenarios, where dispatching tasks and selecting vehicles are controlled by two agents.

However, RL-based methods have to put extra effort to design dense reward functions for specific problems due to the inherent sparsity of real-world problems. Although there are some techniques like intrinsic rewards [34], [35] to address the issue, they introduce a bias to the true reward function. Besides, RL-based methods are not originally designed to handle constraints such as vehicle breakdowns, which makes them hard to apply to real-world problems directly. To address those issues, Hu et al. [5] formulated the problem as a CMDP [16] and proposed a constrained RL (CRL) method, named RCPOM, by incorporating reward shaping and invalid action masking into reward constraint policy optimization (RCPO) [36]. Although RCPOM has shown superior performance on diverse DMH instances, it does not present enough constraint satisfaction [5], due to the temporal credit assignment problem with both sparse rewards and constraint violation. Moreover, as the RL agent is exposed to multiple instances during training, a critical question arises regarding the optimal allocation of training resources across these instances. The proportion of time or episodes dedicated to each instance can significantly impact the learning process. As highlighted in [37], different sampling proportions lead to variations in the distribution of experiences, which in turn introduces biases in gradient calculations. This bias can steer the policy optimization in suboptimal directions, potentially compromising the agent's generalization capabilities.

C. Evolutionary RL

Considering neural network optimization (weights or architecture) as a closed-box problem, ERL directly applies EAs [38] or integrates EA to RL to search for parameters of an actor [39], [40]. ERL typically uses a fitness-based metric for parent selection and survivor selection. It particularly works in the nondifferentiable case since no gradient calculation related to the backpropagation is necessarily required.

Evolution strategy (ES), a typical EA for numerical optimization, is often used to train neural network policies for RL tasks. Salimans et al. [17] leveraged NES for policy optimization. Noises are sampled from a factored Gaussian distribution and then added to the policy network to generate a population. The closed form of the gradient estimator is also given by Salimans et al. [17]. Such et al. [41] applied a genetic algorithm to evolve networks, in which the parameters of a neural network are treated as an individual. Conti et al. [42] validated the effectiveness of novelty search and quality diversity assisted with ES, when meeting sparse and deceptive reward functions. Yang et al. [43] proposed a cooperative coevolution algorithm based on a negatively correlated search to optimize parameters of policy network. Khadka and Tumer [44] proposed a hybrid framework that combines EA and MDP-based RL, addressing the sparse reward and exploration issues. Hu et al. [45] proposed an evolutionary CRL algorithm for robotic control. The constraint handling technique is incorporated into the EA to handle CRL problems. However, on the other hand, the inherent population and elite survival prevent it from training on multiple scenarios at the same time.

Algorithm 1 ACERL

Input: Generation number G, population size λ, number of Instances K, learning rate α, noise standard deviation σ
Output: π_θ

Initialize policy π_θ
Initialize reward buffers $\mathcal{B}_\mathcal{R} = \langle \mathcal{B}_i \rangle$, $i = \{1 \ldots, K\}$
Initialize number of selections $N = \langle N_i \rangle$, $i = \{1 \ldots, K\}$
for $n = 1$ to G **do**
  Initialize instance buffers $\mathcal{I} = \langle \mathcal{I}_i \rangle$, $i = \{1 \ldots, K\}$
  Sample noise $\epsilon_1 \cdots \epsilon_\lambda \in \mathcal{N}^{|\theta|}(0, I)$
  for $i = 1$ to λ **do**
    $\pi_{\theta_i} \leftarrow \pi_{\theta+\sigma\epsilon_i}$
    Sample instance $\eta \leftarrow \text{AIS}(\mathcal{B}_R, N)$ ▷*Algorithm 2*
    $J_\mathcal{R}^{\pi_{\theta_i}}, J_\mathcal{C}^{\pi_{\theta_i}} = \text{Evaluate}(\pi_{\theta_i}, \eta)$ ▷*Algorithm 3*
    $N_\eta \leftarrow N_\eta + 1$
    $\zeta \leftarrow i$
    Store $J_\mathcal{R}^{\pi_{\theta_i}}$ in $\mathcal{B}_\mathcal{R}^\eta$
    Store $J_\mathcal{R}^{\pi_{\theta_i}}, J_\mathcal{C}^{\pi_{\theta_i}}, \zeta$ in $\mathcal{I}_j$
  end for
  $f_1, \ldots, f_\lambda = \text{ISR}(\mathcal{I})$ ▷*Algorithm 4*
  $\theta \leftarrow \theta + \alpha \frac{1}{\lambda\sigma} \sum_i^\lambda f_i \epsilon_i$
end for

Algorithm 2 Adaptive Instance Sampler, AIS($\mathcal{B}_R, N$)

Input: Reward buffers $\mathcal{B}_\mathcal{R}$, counts N, exploration factor α_u
Output: η

for $\eta = 1$ to K **do**
  $u_\eta = \frac{1}{|\mathcal{B}_\mathcal{R}^\eta|} \sum_i^{|\mathcal{B}_\mathcal{R}^\eta|} \frac{\max_{1\le k\le|\mathcal{B}_\mathcal{R}^\eta|}(J_\mathcal{R}^k) - J_\mathcal{R}^i}{\max_{1\le k\le|\mathcal{B}_\mathcal{R}^\eta|}(J_\mathcal{R}^k) - \min_{1\le k\le|\mathcal{B}_\mathcal{R}^\eta|}(J_\mathcal{R}^k)}$.
end for
Sample $\eta = \text{softmax}_\eta \left(u_\eta + \alpha_u \sqrt{\frac{\log\left(\sum_\eta^M N_\eta\right)}{N_\eta}} \right)$

III. Adaptive Constrained ERL

We propose ACERL algorithm[1] to optimize the parameters θ of the policy π_θ for scheduling. The framework and pseudocode are presented in Fig. 2 and Algorithm 1, respectively.

ACERL models an actor (a neural network in our case) as an individual and maintains a population of those independent actors. Instead of evolving the population with genetic operators, at each generation, ACERL samples a population from a distribution based on the policy π_θ. The individuals and their corresponding fitnesses are used to update the distribution and discarded instantly later.

Note that all individuals of the population interact with the environment formed with a specific training instance. The corresponding training instances are chosen by an AIS (Algorithm 2), resulting in an adaptive training process. It estimates the advantage of each *candidate instance* according to its historical rewards based on the evaluation of the policy (Algorithm 3) and the number of selections. The advantage of each instance describes how good the instance is for policy improvement. The most beneficial instance for training the policy is selected according to the estimated soft probability formed with the advantages. If the policy shows inferior

[1]Code: https://github.com/HcPlu/ACERL

Algorithm 3 Evaluate(π) Evaluates a Policy π_θ by Interacting With a Given Environment

Input: Policy π_θ, Instance η
Output: $J_{\mathcal{R}}^{\pi_\theta}, J_{\mathcal{C}}^{\pi_\theta}$

$J_{\mathcal{R}}^{\pi_\theta} \leftarrow 0$
$J_{\mathcal{C}}^{\pi_\theta} \leftarrow 0$
Initialize environment *env* with the instance η
for $t = 0, 1, 2, \ldots, T-1$ **do**
  Sample action $a_t \in \pi_\theta(s_t)$
  Obtain r_t, c_t, s_{t+1} from *env* by acting a_t
  $J_{\mathcal{R}}^{\pi_\theta} \leftarrow J_{\mathcal{R}}^{\pi_\theta} + r_t$
  $J_{\mathcal{C}}^{\pi_\theta} \leftarrow J_{\mathcal{C}}^{\pi_\theta} + c_t$
end for

Algorithm 4 Intrinsic Stochastic Ranking, ISR($\mathcal{I}$). $p_f \in (0, 1)$ Is the Tolerate Probability. $\phi(\pi)$ Denotes the Penalty

Input: Instance buffers $\mathcal{I}_1, \ldots, \mathcal{I}_K$
Output: $f_1, \ldots, f_\mu$

for $k = 1$ to K **do**
  $\mu' = |\mathcal{I}_k|$
  for $i = 1$ to μ' **do**
    $l_i = i$
  end for
  Get $\langle J_{\mathcal{R}}^{\pi_{\theta_j}}, J_{\mathcal{C}}^{\pi_{\theta_j}}, \zeta_j^k \rangle$ from $\mathcal{I}_k, j = 1, \ldots, \mu'$
  for $i = 1$ to μ' **do**
    for $j = 1$ to $\mu' - 1$ **do**
      Sample δ uniformly at random in $(0, 1)$
      if $(\phi(\pi_{\theta_j}) = \phi(\pi_{\theta_{j+1}}) = 0)$ or $(\delta < P_f)$ **then**
        if $J_{\mathcal{R}}^{\pi_{\theta_j}} < J_{\mathcal{R}}^{\theta_{j+1}}$ **then**
          swap l_j and l_{j+1}
        end if
      else
        if $\phi(\pi_{\theta_j}) > \phi(\pi_{\theta_{j+1}})$ **then**
          swap l_j and l_{j+1}
        end if
      end if
    end for
  end for
  for $i = 1$ to μ' **do**
    $f_{\zeta_{l_i}^k} = \mu' - i + 1$
  end for
end for

performance on certain instances, the likelihood of choosing those instances increases accordingly.

After obtaining the episodic rewards and penalties, we design an ISR method (Algorithm 4) to group the sampled individuals that interact with the same training instance into the same buffer. The fitness of each individual is assigned with its own rank index in the descending intrinsic ranked buffer. The ranking method balances the rewards and penalties, seeking to maximize the long-term reward with constraint satisfaction. The impact of fitness weight is further involved in breaking the tradeoff among multiple training instances. Finally, ACERL updates the policy with NES according to sampled noises and the corresponding fitness values.

Sections III-A–III-C detail the core ingredients of ACERL, including adaptive training, ISR with rank-based fitness, and NES.

A. Efficient Instance Selection via Adaptive Training

To achieve a better computational resource allocation, we design an AIS to train ACERL by selecting suitable training instances adaptively. The pseudocode of AIS is shown in Algorithm 2. The idea behind AIS is that if the policies have performed well on a specific instance, then the probability of selecting this instance should be reduced and the likelihood of selecting the instances that were evaluated fewer times should increase.

Specifically, an inverted distance metric for evaluating the advantage of training instances is proposed

$$u_\eta = \frac{1}{|\mathcal{B}_{\mathcal{R}}^\eta|} \sum_{i=1}^{|\mathcal{B}_{\mathcal{R}}^\eta|} \frac{\max_{1\le k\le|\mathcal{B}_{\mathcal{R}}^\eta|} J_{\mathcal{R}}^k - J_{\mathcal{R}}^i}{\max_{1\le k\le|\mathcal{B}_{\mathcal{R}}^\eta|} J_{\mathcal{R}}^k - \min_{1\le k\le|\mathcal{B}_{\mathcal{R}}^\eta|} J_{\mathcal{R}}^k}. \quad (6)$$

The distance metric measures the performance of the policy on the training instance η at the time horizon. Specifically, it calculates how much each episode's reward deviates from the maximum reward, scaled by the batch's reward range. A larger u_η signifies poorer policy performance on η, which provides an insight that the instance should be chosen more.

The adaptive training takes the metric in (6) and the number of times of selecting each training instance into account by the widely used upper confidential bound (UCB) [46], [47]

$$\text{UCB} = u_i + \alpha_u \sqrt{\frac{\log\left(\sum_j^K N_j\right)}{N_i}} \quad (7)$$

where α_u is the exploration factor, K is the total number of training instances, and N_i is the number of instance i being selected as the training candidate. With the UCB-based AIS, instances are chosen to train the policy more efficiently. Thus, the computational resource can be dynamically allocated in a proper way.

B. ISR With Rank-Based Fitness

Assessing a policy for DMH that involves multiple instances and constraints can be challenging beyond the unconstrained optimization problems [17], [42]. Regular methods use a weighted sum of the objective value and penalties for constraint violation as the reshaped reward function, which, however, introduces the challenge of adjusting the weights [48].

Stochastic ranking (SR) [49] is an effective constraint handling technique, which has been successfully applied to constrained optimization [50] and combinatorial optimization [51]. SR makes no assumption on the problem, while only one parameter is introduced and is easy to tune [49]. Inspired by the work of [49], we propose ISR with rank-based fitness, shown in Algorithm 4, to handle the constraints in the case of sparse feedback. During the evolution process, the individuals that interact with the same training instance are collected and stored in the same buffer. Then, all the individuals in the same buffer are ranked according to their rewards and penalties. Notably, if solutions provided by two individuals are either

feasible or meet the probability threshold p_f, the solution with the higher reward is assigned a better rank; otherwise, the one with fewer constraint violations is given the higher rank.

An individual's fitness value is assigned based on the ranking in its own buffer by ISR. Following the setting of [49], the penalty function related to the constraint violation is:

$$\phi(\pi) = \left(\max\{0, J_C^\pi - \xi\}\right)^2. \tag{8}$$

ISR remains some infeasible solutions with high fitness values besides giving priority to feasible solutions. Although infeasible solutions cannot be executed to solve the problem, they may help escape from some infeasible areas. ISR not only balances the rewards and penalties but also allows the dynamic selection of suitable training instances with preference, which makes efficient use of the computational resource. Additionally, ISR only requires the final scalar value instead of temporal information, which aligns well with the sparse case.

C. Policy Improvement Through Searching Gradients

ACERL applies evolution strategies to update the policy π_θ, which is parameterized by a neural network θ. A population is sampled from a distribution over $p_\psi(\theta)$ with the given policy π_θ. The estimated gradient is given by [52]

$$\nabla_\psi \mathbb{E}_{\theta\sim p_\psi} F(\theta) = \mathbb{E}_{\theta\sim p_\psi} F(\theta)\nabla_\psi \log(p_\psi(\theta)) \tag{9}$$

where $F(\theta)$ is the fitness function related to the policy π_θ. The minimal requirement of temporary information makes natural gradient intuitive to tackle sparse reward and long-term horizon, compared with value-based methods and policy gradient, without calculating the gradients for the backpropagation. The perturbations in the parameter space also facilitate the exploration for collecting more diverse experiences [17].

In the DMH, the fitness of each individual is given by the ISR (Algorithm 4). More specifically, p_ψ is a factored Gaussian distribution $\mathcal{N}(\psi, \sigma^2)$, in which ψ is the mean value and σ is the covariance. The expectation of fitness under the distribution considering θ as the mean parameter [17] is written as follows:

$$\mathbb{E}_{\theta\sim p_\psi} F(\theta) = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)} F(\theta + \sigma\epsilon). \tag{10}$$

Finally, the policy is optimized by the gradient ascent with a vanilla estimator

$$\nabla_\theta^\epsilon F(\theta + \sigma\epsilon) = \frac{1}{\sigma}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\left[F(\theta + \sigma\epsilon)\epsilon\right] \tag{11}$$

or an antithetic estimator

$$\nabla_\theta^\epsilon F(\theta + \sigma\epsilon) = \frac{1}{2\sigma}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\left[(F(\theta + \sigma\epsilon) - F(\theta - \sigma\epsilon))\epsilon\right]. \tag{12}$$

All weights are perturbed in the case of factored Gaussian distribution, which is similar to the coupon collector problem in a continuous way. Considering ACERL as a randomized finite difference method, the optimization complexity scales linearly with the number of weights, i.e., $\mathcal{O}(|\theta|)$. The expected covering time to ensure that all weight dimensions are sufficiently perturbed is at least polynomial bound $\mathcal{O}(|\theta|\log|\theta|)$. Given the population size λ, the space complexity and the communication complexity are both $\mathcal{O}(\lambda|\theta|)$ for passing gradients since we only optimize the neural network. Notably, ACERL follows the optimization scheme of classic evolution strategies. As such, its optimization complexity, covering time, space, and communication complexities align with existing theoretical bounds [52], [53].

IV. Theoretical Analysis

Although Choromanski et al. [54] and Liu et al. [55] show that the gradient estimator is close to the true gradient, they only derive the theoretical results in the unconstrained setting. We extend the analysis in [54] and [55] and demonstrate the theoretical guarantee on the constrained optimization with SR.

First, we simplify the constrained problem formulation on the function landscape with one single instance for a better analysis as follows:

$$\max_\theta \; F(\theta) \quad \text{s.t. } g(\theta) \le \xi \tag{13}$$

where $F(\theta)$ is the objective function, $g(\theta)$ is the constraint function, and ξ is the constraint threshold.

Following the settings of Choromanski et al. [54], the following assumptions of the regularities of $F(\theta)$ and $g(\theta)$ are made.

Assumption 1: F and g are L-Lipschitz, i.e., $\forall\theta,\theta' \in \mathbb{R}^d, |F(\theta) - F(\theta')| \le L_f\|\theta - \theta'\|, \quad |g(\theta) - g(\theta')| \le L_g\|\theta - \theta'\|$.

Assumption 2: F has a τ-smooth third-order derivative tensor with respect to $\sigma > 0$ so that $F(\theta + \sigma\epsilon) = F(\theta) + \sigma\nabla F(\theta)^\top\epsilon + (\sigma^2/2)\epsilon^\top H(\theta)\epsilon + (1/6)\sigma^3 f'''(\theta)[v, v, v]$ with $v \in [0, \epsilon]$ satisfying $|F'''(v, v, v)| \le \tau\|v\|^3 \le \tau\|\epsilon\|^3$, where $H(\theta)$ and $F'''(\theta)$ denote the Hessian matrix and third derivative of F, respectively. Similarly, g has a τ-smooth third-order derivative tensor with the same regularities as F.

Assumption 3: σ is small enough, i.e., $0 < \sigma < (1/35)((\mathcal{E}/\tau d^3 \max\{L_f, L_g, 1\}))^{1/2}$, where $\mathcal{E} > 0$.

For a better analysis, we relax the penalty function $\phi(\theta)$ formulated in (8) to a smooth approximation since $\phi(\theta)$ is not differentiable at $g(\theta) = \xi$

$$\phi'(\theta) = \rho \ln\left(1 + e^{(g(\theta)-\xi)/\rho}\right) \tag{14}$$

where $\rho > 0$. This relaxation nearly preserves the ordering of the inequality as $\rho \to 0$.

Theorem 1: There exists a sufficiently small $\rho > 0$ such that

$$\forall\theta, \quad \theta' \in \mathbb{R}^d, \; \phi(\theta_i) \le \phi(\theta_j) \iff \phi'(\theta_i) \le \phi'(\theta_j)$$
$$i.e., \; \rho \ln\left(1 + e^{(g(\theta_i)-\xi)/\rho}\right) \le \rho \ln\left(1 + e^{(g(\theta_j)-\xi)/\rho}\right).$$

Intuitively, the relaxed penalty function $\phi'(\theta)$ is also L-Lipschitz. Besides, Assumption 2 applies to $\phi'(\theta)$.

Recalling SR, it assigns priority to individuals based on the objective and penalty function with probability p_f. Thus, we consider the objective function relaxed by SR as follows:

$$f_{\text{SR}}(\theta) = p_f F(\theta) - (1 - p_f)\phi'(\theta) \tag{15}$$

where p_f is the probability with values in $[0, 1]$. It is easy to see that Assumptions 1 and 2 also hold for f_{SR}.

Assuming two candidates π_{θ_i} and π_{θ_j} with $F(\theta_i) \le F(\theta_j)$, all three possible scenarios are described as follows.

Case 1: If $\phi'(\theta_i) = \phi'(\theta_j) = 0$, then $f_{\text{SR}}(\theta_i) \le f_{\text{SR}}(\theta_j)$.

Case 2: If $0 \le \phi'(\theta_j) \le \phi'(\theta_i)$ and $0 < \phi'(\theta_i)$, then $f_{\text{SR}}(\theta_i) \le f_{\text{SR}}(\theta_j)$.

Case 3: If $0 \le \phi'(\theta_i) \le \phi'(\theta_j)$ and $0 < \phi'(\theta_j)$, then the ranking $f_{\mathrm{SR}}(\theta_i) \le f_{\mathrm{SR}}(\theta_j)$ holds under the following inequality:

$$p_f\left(F\left(\theta_i\right)-F\left(\theta_j\right)\right) \le \left(1-p_f\right)\left(\phi'\left(\theta_i\right)-\phi'\left(\theta_j\right)\right)$$

which indicates that the final ranking relies on p_f, as well as both reward and penalty values.

With the above discussions, we show that f_{SR} captures the behaviors of SR.

Under the assumptions, we can derive an antithetic form that

$$\frac{f_{\mathrm{SR}}(\theta+\sigma\epsilon)-f_{\mathrm{SR}}(\theta-\sigma\epsilon)}{2\sigma} = \epsilon^{\top}\nabla f_{\mathrm{SR}}(\theta)+\zeta(\theta)$$

where $\zeta(\theta) \le (\tau/6)\sigma^2\|\epsilon\|^3$. Given f_{SR} is smooth with constant τ on the third-order derivative tensor, we have

$$\left|\frac{f_{\mathrm{SR}}(\theta+\sigma\epsilon)-f_{\mathrm{SR}}(\theta-\sigma\epsilon)}{2\sigma}-\epsilon^{\top}\nabla f_{\mathrm{SR}}(\theta)\right| \le \tau\sigma^2\|\epsilon\|^3.$$

Recall the gradient in the antithetic case [(12)]

$$\nabla_{\theta}^{\epsilon}F(\theta+\sigma\epsilon) = \frac{1}{2\sigma}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}[(F(\theta+\sigma\epsilon)-F(\theta-\sigma\epsilon))\epsilon].$$

We have $|(f_{\mathrm{SR}}(\theta+\sigma\epsilon)-f_{\mathrm{SR}}(\theta-\sigma\epsilon))/(2\sigma) - \epsilon^{\top}\nabla f_{\mathrm{SR}}(\theta)| \le \tau\sigma^2\|\epsilon\|^3$. Then, we can derive the following inequality:

$$\|\nabla_{\theta}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}f_{\mathrm{SR}}(\theta+\sigma\epsilon)-\nabla f_{\mathrm{SR}}(\theta)\| \le \mathbb{E}_{\epsilon}\tau\sigma^2\|\epsilon\|^4. \tag{16}$$

Theorem 2: The bias of the gradient estimator under SR is well bounded

$$\left\|\nabla_{\theta}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}f_{\mathrm{SR}}(\theta+\sigma\epsilon)-\nabla f_{\mathrm{SR}}(\theta)\right\| \le \mathcal{E}. \tag{17}$$

Proof: Considering that $\sigma \sim \mathcal{N}(0, I)$, we have $\mathbb{E}[\sigma(i)^4] = 3$, $\mathbb{E}[\sigma(i)^2] = 1$, $\forall i \in \{1, \ldots, d\}$. The following inequality holds:

$$\begin{aligned}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\left[\|\epsilon\|^4\right] &= \mathbb{E}_{\epsilon}\left[\sum_{i=1}^{d}\epsilon(i)^4+\sum_{i\neq j}\epsilon(i)^2\epsilon(j)^2\right]\\ &\le 3d+d^2 \le 3d^2.\end{aligned}$$

Thus, together with (16)

$$\begin{aligned}\left\|\nabla_{\theta}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}f_{\mathrm{SR}}(\theta+\sigma\epsilon)-\nabla f_{\mathrm{SR}}(\theta)\right\| &\le \mathbb{E}_{\epsilon}\tau\sigma^2\|\epsilon\|^4\\ &\le 3\tau\sigma^2d^2.\end{aligned}$$

Recall Assumption 3 that $0 < \sigma < (1/35)((\mathcal{E})/(\tau d^3\max\{L_f, L_g, 1\}))^{1/2}$. Finally, we derive the result

$$\|\nabla_{\theta}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}f_{\mathrm{SR}}(\theta+\sigma\epsilon)-\nabla f_{\mathrm{SR}}(\theta)\| \le \mathcal{E}.$$

■

V. Experimental Results and Analysis

We conduct several sets of experiments and an ablation study to comprehensively evaluate ACERL. The aims of experiments and compared methods are detailed as follows.

1) To validate the effectiveness of ACERL, we compare it on eight training instances and eight unseen test instances with several state-of-the-art methods and baselines categorized into groups: 1) "MAPPO" [33], "RCPOM" [5], and soft actor-critic (SAC) [29] with the fixed Lagrangian multiplier, "LSAC" [56]; 2) "SAC" and "PPO" [57]; 3) MAPPO, RCPOM, LSAC, SAC, and PPO equipped with the AIS denoted as "AMAPPO,""ARCPOM," "ALSAC," "ASAC," and "APPO," respectively; 4) classic dispatching rules, including "FCFS," "EDD," "NVF," and "STD" [10]; and 5) two random policies that randomly choose rules or tasks, denoted as "MIX" and "Random" on both training instances and test instances, respectively.
2) To present the robust performance of ACERL, it is tested on 40 instances with increasing extent of perturbations that simulate different degrees of dynamic events.
3) To further evaluate the performance and limitations of ACERL in out-of-distribution cases, a leave-one-out cross validation is conducted by dividing the training dataset into subsidiary training datasets and test datasets.
4) To examine each ingredient of ACERL including ISR, rank-based fitness, and adaptive training, an ablation study is performed to present their unique contributions.

A. Experiment Setting

Experiment settings, including hyperparameters, problem instances, and metrics for performance assessment, are described as follows.

1) Implementation Details: The implementations of RL and CRL policies, including SAC [29] and PPO [57], MAPPO [33], RCPOM [5], and LSAC [56], are adapted based on the Tianshou framework[2] [58]. Source codes of RCPOM are provided in [5]. The network structure is formed by two hidden fully connected layers 128×128. The discounted factor γ is 0.97. The initial multiplier λ and the learning rate of RCPOM are set as 0.001 and 0.0001, respectively. The population size λ is 256. The number of generations G is 128. The constraint threshold ξ is set as 50. Other common hyperparameters are set following the default setting of Tianshou.[3] All learning policies are trained for $1e^6$ steps with five different seeds and each is tested 30 times independently.

2) Problem Simulator and Instances: The experiments are conducted on publicly available DMH instances and simulator, DMH-GYM,[4] provided in [5]. The training instances (DMH-01–DMH-08) are drawn from different distributions using a searching-based method, while test instances (DMH-09–DMH-16) are generated by noising the training instances [5].

3) Evaluation Metrics: Three metrics are used to evaluate the policies, including the average normalized score of makespan M, the average normalized score of tardiness C, and the average constraint satisfaction percentage P, formulated as follows:

$$M = \frac{1}{K}\sum_{j=1}^{K}\frac{F_m^{\max}-F_m^j}{F_m^{\max}-F_m^{\min}} \tag{18}$$

$$C = \frac{1}{K}\sum_{j=1}^{K}\frac{F_t^{\max}-F_t^j}{F_t^{\max}-F_t^{\min}} \tag{19}$$

$$P = \frac{1}{K}\sum_{j=1}^{K}\mathbb{1}_{F_t^j<\xi} \tag{20}$$

[2]https://github.com/thu-ml/tianshou
[3]tianshou
[4]https://github.com/HcPlu/DMH-GYM

TABLE I

AVERAGE MAKESPAN AND TARDINESS OVER 30 INDEPENDENT TRIALS OF FIVE DIFFERENT SEEDS ON TRAINING INSTANCES. BOLD NUMBERS INDICATE THE BEST MAKESPAN AND TARDINESS. "+,""≈," AND "−" INDICATE THE POLICY PERFORMS STATISTICALLY BETTER/SIMILAR/WORSE THAN ACERL POLICY, RESPECTIVELY. THE NUMBER OF POLICIES THAT ARE "BETTER," "SIMILAR," AND "WORSE" THAN ACERL IN TERMS OF MAKESPAN AND TARDINESS ON EACH INSTANCE IS SUMMARIZED IN THE BOTTOM ROW. THE LAST COLUMN WITH HEADER "$M/C(P)$" INDICATES THE AVERAGE NORMALIZED MAKESPAN, TARDINESS, AND PERCENTAGE OF CONSTRAINT SATISFACTION. HORIZONTAL RULES IN THE TABLE SEPARATE DIFFERENT GROUPS OF ALGORITHMS

Algorithm	DMH-01 F_m/F_t	DMH-02 F_m/F_t	DMH-03 F_m/F_t	DMH-04 F_m/F_t	DMH-05 F_m/F_t	DMH-06 F_m/F_t	DMH-07 F_m/F_t	DMH-08 F_m/F_t	$M/C(P)$
ACERL	**1797.8/29.5**	**1859.2**/30.7	**1840.0**/28.6	**1896.6**/35.7	**1856.4/29.0**	**1864.4/35.5**	1929.6/**9.5**	**1856.8/25.4**	**0.90/0.90 (100%)**
MAPPO	1869.2-/68.3-	1926.6-/42.3-	1925.6-/53.0-	1908.2-/**32.2**+	1928.4-/42.1-	1955.8-/79.4-	1945.0≈/20.4-	1875.4-/30.2-	0.72/0.72 (62%)
RCPOM	1875.5-/62.8-	1932.6-/49.1-	1912.1-/51.8-	1947.7-/37.4≈	1937.6-/44.4-	1946.9-/58.5-	1937.9-/12.0-	1883.6-/27.9≈	0.71/0.75 (60%)
LSAC	1922.2-/65.7-	1968.1-/53.4-	1935.9-/49.2-	2012.0-/54.6-	1960.6-/48.1-	1997.7-/73.1-	1971.4-/15.6-	1934.3-/38.1-	0.55/0.68 (56%)
AMAPPO	1890.8-/73.2-	1936.4-/43.3-	1931.2-/42.9-	1928.6-/42.4-	1925.4-/47.1-	1952.0-/90.5-	**1927.0**+/14.3-	1897.6-/39.1-	0.70/0.69 (57%)
ARCPOM	1885.8-/76.0-	1948.3-/54.6-	1922.1-/56.3-	1979.2-/46.0-	1905.0-/37.6-	1948.5-/87.6-	1979.0-/18.0-	1928.1-/43.1-	0.63/0.65 (48%)
ALSAC	1903.6-/61.7-	1964.9-/51.6-	1899.1-/45.2-	1986.6-/47.8-	1967.0-/54.3-	2009.3-/73.7-	1966.5-/16.8-	1917.5-/38.3-	0.60/0.69 (53%)
SAC	1899.4-/59.6-	1973.1-/52.5-	1925.6-/43.9-	1991.6-/51.3-	1990.4-/52.9-	2013.4-/80.2-	1952.0-/14.7-	1933.3-/37.2-	0.57/0.69 (57%)
PPO	1877.4-/57.4-	1954.1-/50.6-	1920.2-/45.8-	1979.5-/51.8-	1955.4-/47.0-	1978.9-/68.9-	1973.3-/16.5-	1920.3-/34.1-	0.61/0.71 (58%)
ASAC	1913.2-/70.5-	1985.9-/55.3-	1944.8-/47.6-	1985.5-/49.1-	1987.7-/53.5-	2027.5-/92.3-	1950.2-/14.9-	1934.1-/39.8-	0.55/0.65 (52%)
APPO	1899.6-/72.9-	1951.4-/51.3-	1922.5-/54.3-	1972.5-/42.0-	1949.9-/54.3-	1992.8-/97.7-	1967.9-/19.3-	1905.6-/34.8-	0.62/0.65 (50%)
MIX	1939.9-/60.3-	2000.9-/60.4-	1932.2-/45.0-	2006.1-/52.8-	2028.2-/64.2-	2027.1-/69.5-	1969.2-/16.1-	1971.0-/46.9-	0.49/0.65 (54%)
FCFS	2081.1-/90.2-	2084.5-/82.3-	2014.7-/64.2-	2136.7-/105.0-	2194.6-/122.5-	2123.5-/85.2-	1927.9≈/11.2-	1933.6-/27.4≈	0.25/0.45 (37%)
EDD	1903.2-/33.9≈	1968.6-/**29.6**+	1977.8-/**26.5**≈	1988.1-/32.8+	1950.7-/33.7≈	2016.8-/46.8≈	1940.5-/12.1-	2020.8-/45.3-	0.52/0.85 (86%)
NVF	1876.5-/69.6-	1958.4-/56.4-	1946.7-/49.6-	2040.6-/66.7-	1933.9-/56.3-	1953.4-/78.5-	1996.5-/21.8-	1944.6-/35.5-	0.56/0.63 (51%)
STD	1868.8-/66.6-	1961.7-/49.2-	1921.3-/51.7-	1970.7-/35.6≈	1917.1-/41.4-	1955.4-/62.7-	1983.7-/30.4-	1883.5-/35.1-	0.65/0.70 (53%)
Random	2098.7-/124.5-	2113.8-/103.1-	2091.2-/143.7-	2135.1-/123.1-	2149.3-/119.0-	2159.1-/129.3-	2083.0-/70.2-	2067.8-/89.5-	0.02/0.01 (12%)
	(0/0/16) / (0/1/15)	(0/0/16) / (1/0/15)	(0/0/16) / (0/1/15)	(0/0/16) / (2/2/12)	(0/0/16) / (0/1/15)	(0/0/16) / (0/1/15)	(1/2/13) / (0/0/16)	(0/0/16) / (0/2/14)	

TABLE II

AVERAGE MAKESPAN AND TARDINESS OVER 30 INDEPENDENT TRIALS OF FIVE DIFFERENT SEEDS ON TEST INSTANCES. BOLD NUMBERS INDICATE THE BEST MAKESPAN AND TARDINESS. "+,""≈," AND "−" INDICATE THE POLICY PERFORMS STATISTICALLY BETTER/SIMILAR/WORSE THAN ACERL POLICY, RESPECTIVELY. THE NUMBER OF POLICIES THAT ARE "BETTER," "SIMILAR," AND "WORSE" THAN ACERL IN TERMS OF MAKESPAN AND TARDINESS ON EACH INSTANCE IS SUMMARIZED IN THE BOTTOM ROW. THE LAST COLUMN WITH HEADER "$M/C(P)$" INDICATES THE AVERAGE NORMALIZED MAKESPAN, TARDINESSM, AND PERCENTAGE OF CONSTRAINT SATISFACTION. HORIZONTAL RULES IN THE TABLE SEPARATE DIFFERENT GROUPS OF ALGORITHMS

Algorithm	DMH-09 F_m/F_t	DMH-10 F_m/F_t	DMH-11 F_m/F_t	DMH-12 F_m/F_t	DMH-13 F_m/F_t	DMH-14 F_m/F_t	DMH-15 F_m/F_t	DMH-16 F_m/F_t	$M/C(P)$
ACERL	**1801.6/29.9**	**1878.4/30.6**	**1892.7**/34.7	**1896.8**/35.6	**1894.9/24.2**	**1865.0/36.1**	**1932.8/9.6**	**1857.0/25.8**	**0.89/0.92 (97%)**
MAPPO	1880.8-/68.7-	1928.4-/43.4-	1931.8-/49.7-	1908.6-/**32.5**+	1923.2-/34.9-	2034.4-/92.2-	1972.8-/28.8-	1875.8-/30.9-	0.70/0.73 (65%)
RCPOM	1877.2-/63.2-	1935.0-/49.2-	1908.0-/51.7-	1946.8-/37.6≈	1928.2-/35.4-	1951.9-/59.9-	1970.4-/23.5-	1886.0-/30.4-	0.71/0.76 (63%)
LSAC	1918.0-/66.2-	1966.5-/53.4-	1934.4-/49.5-	2011.9-/54.6-	1961.6-/41.4-	1996.1-/74.1-	1972.1-/15.5-	1932.2-/39.0-	0.57/0.70 (56%)
AMAPPO	1879.8-/73.6-	1937.6-/43.3-	1937.6-/43.2-	1930.0-/41.2-	1924.8-/33.6-	2056.4-/105.8-	1954.8≈/24.4-	1894.0-/40.4-	0.67/0.69 (57%)
ARCPOM	1902.3-/78.3-	1949.0-/54.4-	1920.3-/55.4-	1974.0-/44.1-	1908.3-/28.0-	1963.4-/92.9-	1975.1-/16.2-	1931.5-/43.9-	0.65/0.68 (52%)
ALSAC	1893.7-/62.1-	1955.6-/49.8-	1929.9-/48.3-	2001.2-/47.5-	1955.9-/45.2-	2005.5-/73.1-	1974.5-/20.4-	1913.3-/38.7-	0.60/0.71 (55%)
SAC	1906.3-/62.0-	1972.8-/52.5-	1931.8-/44.9-	1993.4-/51.2-	1975.8-/49.3-	2012.3-/81.3-	1952.2-/14.7-	1924.9-/38.0-	0.59/0.70 (58%)
PPO	1875.4-/58.5-	1952.8-/50.4-	1916.3-/46.7-	1981.7-/52.0-	1956.3-/44.2-	1976.6-/70.6-	1975.2-/16.8-	1918.9-/34.9-	0.64/0.73 (59%)
APPO	1896.3-/73.2-	1950.9-/51.1-	1915.4-/55.1-	1975.2-/41.6-	1934.5-/38.5-	1996.4-/98.8-	1972.6-/20.2-	1890.4-/34.5-	0.65/0.68 (56%)
ASAC	1920.4-/72.3-	1987.4-/55.2-	1944.6-/48.4-	1988.1-/49.0-	1977.7-/47.8-	2031.9-/95.3-	1950.6-/14.7-	1934.6-/41.0-	0.57/0.66 (53%)
MIX	1941.7-/63.9-	2004.3-/60.2-	1932.3-/46.3-	2008.3-/52.7-	2030.6-/62.4-	2032.3-/72.5-	1971.5-/16.2-	1976.9-/49.4-	0.49/0.66 (54%)
FCFS	2089.3-/92.5-	2045.4-/75.7-	1996.9-/67.6-	2107.1-/95.2-	2191.9-/121.7-	2130.1-/89.9-	1934.1≈/11.1-	1946.8-/35.8-	0.27/0.45 (35%)
EDD	1996.9-/107.7-	1976.3-/33.4≈	1978.7-/**22.0**+	1997.6-/36.9≈	1962.1-/37.0-	1993.5-/44.6-	1934.8≈/11.0-	2052.3-/69.4-	0.48/0.74 (73%)
NVF	1847.5-/57.2-	1958.8-/54.3-	1926.6-/51.8-	2021.7-/65.9-	1939.2-/41.2-	1975.5-/89.8-	2003.5-/19.3-	1933.8-/40.9-	0.59/0.66 (58%)
STD	1894.1-/72.5-	1958.2-/49.7-	1923.1-/52.4-	1985.2-/38.1≈	1905.1≈/31.4-	1974.3-/71.9-	1990.3-/33.0-	1885.2-/38.6-	0.65/0.71 (55%)
Random	2132.7-/135.3-	2100.9-/99.5-	2076.2-/144.4-	2097.5-/106.3-	2144.1-/102.6-	2142.9-/123.6-	2101.3-/81.1-	2055.1-/94.1-	0.02/0.03 (11%)
	(0/0/16) / (0/0/16)	(0/0/16) / (0/1/15)	(0/0/16) / (1/0/15)	(0/0/16) / (1/3/12)	(0/1/15) / (0/0/16)	(0/0/16) / (0/0/16)	(0/3/13) / (0/0/16)	(0/0/16) / (0/0/16)	

where K is the number of instances and ξ is the constraint threshold. $F_m^{\max}$ and $F_m^{\min}$ represent the maximal and minimal makespan values F_m [(1)] among all the policies, respectively. Similarly, $F_t^{\max}$ and $F_t^{\min}$ represent the maximal and minimal tardiness values F_t [(2)] among all policies, respectively. All metrics follow the principle that a larger value indicates better performance of the policy.

B. Comparison With State of the Arts and Baselines

Tables I and II present the experiment results of ACERL on training instances (DMH-01–DMH-08) and test instances (DMH-09–DMH-16), respectively, compared with advanced RL methods, CRL methods, and classic dispatching rules. It is obvious that our proposed method, ACERL, statistically outperforms other algorithms on all the training and test instances except on DMH-07, DMH-13, and DMH-15. Compared to ACERL, the dispatching rule FCFS gets a competitive makespan but statistically worse tardiness on DMH-07, while other compared algorithms perform statistically worse than ACERL. Similar observations are also found on DMH-13 and DMH-15. Overall, ACERL has the best values in terms of normalized makespan M, normalized tardiness C, and constraint satisfaction percentage P and achieves the overall best performance on all instances.

Besides, ACERL achieves 100% and 97% constraint satisfaction on training and test instances, respectively, which is much better than other algorithms. For example, RCPOM [5],

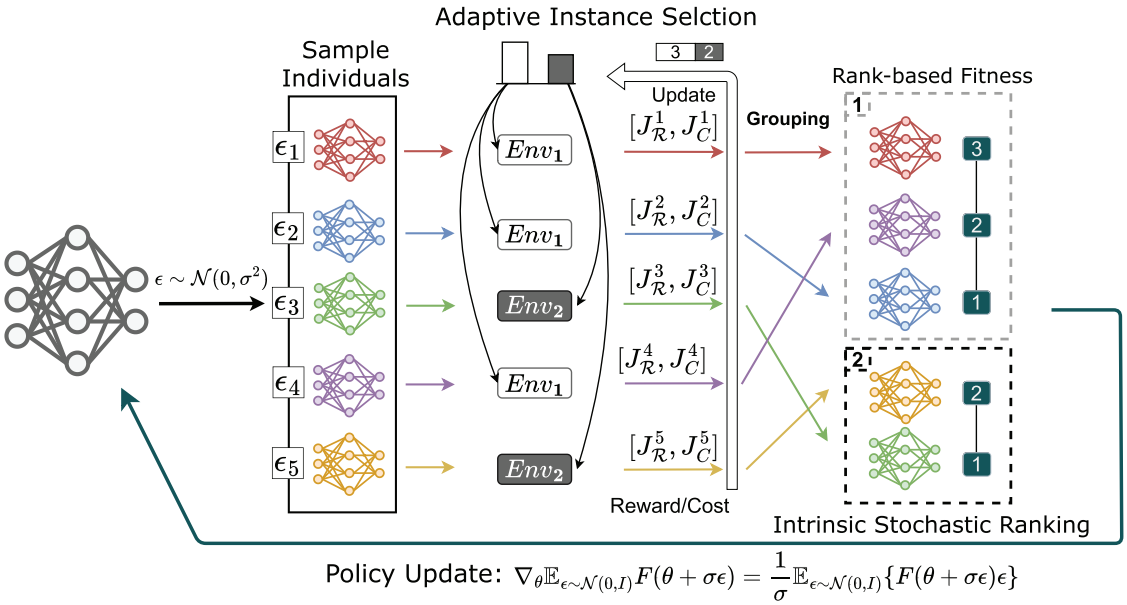


Fig. 2. Illustration of ACERL framework. A population of actors is sampled with Gaussian noise. All actors then interact with the specific subset of training instances determined by the AIS (see Section III-A). The collected rewards and selection number of training instances are used to update the instance sampler. The fitness of each actor is assigned according to its inner rank with the ISR (see Section III-B). A natural gradient ascent is applied to update the policy accordingly (see Section III-C).

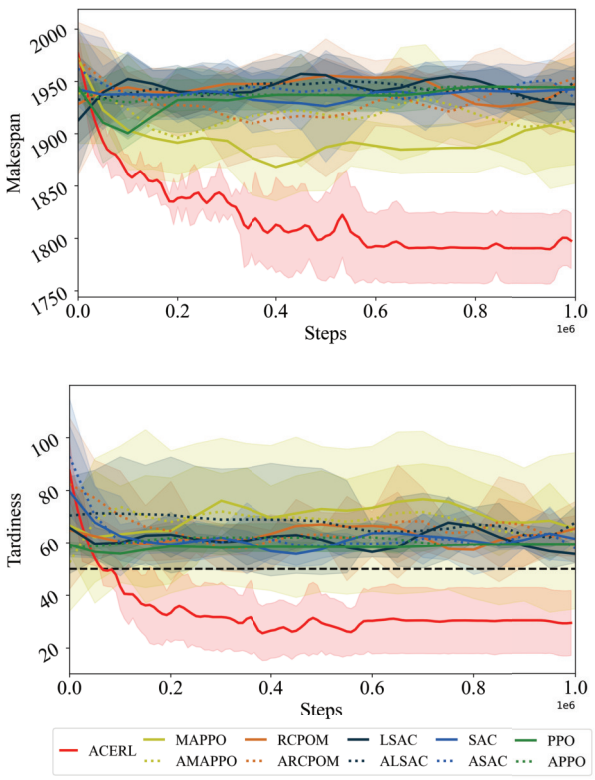


Fig. 3. Training curves on DMH-01. ACERL (red curve) obtains the best makespan while fully satisfying the tardiness constraint.

a state-of-the-art CRL method only gets 60% and 63% constraint satisfaction on training and test instances, respectively.

As an example, the training curves of DMH-01 are shown in Fig. 3. ACERL shows the best performance by fully satisfying the constraint in DMH-01. The training time of ACERL is about 30 min, while other RL and CRL-based methods take more than 1 h for training $1e^6$ steps. Moreover, the computation time of ACERL for one single decision is about 2 ms, which meets the real-time requirement [6]. More training curves on various instances are provided in Section A of Supplementary Material.

1) Tackling Sparse Feedback: Tables I and II show that none of the RL or CRL methods, including MAPPO, SAC, PPO, RCPOM, and LSAC, performs better than ACERL on all the 16 training and test instances. SAC and PPO are even worse than the dispatching rule STD, which achieves the best $M = 0.65$ among all dispatching rules. We attribute the phenomenon to the lack of temporary information. Regular environments and benchmarks considered in RL studies usually are well defined with suitable reward functions [59]. It is hard to guarantee that the same conditions are provided in real-world scenarios. Objectives, including makespan and tardiness, are only obtained after all tasks are completed in DMH. Given an RL agent, it only receives one meaningful reward at the end of an episode while getting zero for other steps. This also applies to the constraint satisfaction part, as the cost function related to tardiness is also sparse. MAPPO [33] and RCPOM [5] achieve similar results. RCPOM applies invariant reward shaping to address sparse feedback but still fails to achieve superior results. On the other hand, MAPPO suffers from the high requirement of training policies with a globally centralized critic.

ACERL is validated to handle sparse feedback by the experiment results. Instead of stochastic gradient descent with value function or policy gradient, ACERL applies the natural gradient ascent to update a policy. The usage of rank-based fitness makes it possible to get rid of the temporary information that regular RL methods require since only the last episodic value is needed. Thus, ACERL can tackle sparse feedback without reward shaping relying on domain knowledge.

2) Promising Constraint Satisfaction: Delay of task finishing time, i.e., tardiness, should be restricted within the

given threshold in DMH. It also needs to guarantee safety while assigning tasks since only AGVs in the idle state are available to serve tasks. They are considered as cumulative and instantaneous constraints and are handled as long-term behaviors and short-term behaviors, respectively. However, it is an intractable problem for RL methods to handle constraints [36]. The complexity arises from the need of maximizing reward, i.e., minimizing makespan while ensuring strict adherence to constraints. The balance is crucial that agents achieving high reward can still be outperformed by seemingly "poor" agents that prioritize constraint satisfaction. This phenomenon shows the critical importance of constraint management in real-world applications, where violating operational limits (such as excessive tardiness) can have severe consequences like human safety. RCPOM [5] combines RCPO [36] and invalid action masking to deal with the tardiness and available vehicle constraints, respectively. Although the masking technique guarantees the satisfaction of the available vehicle constraint, it actually does not perform well with an inferior constraint satisfaction percentage in terms of tardiness than EDD, a time-prefer dispatching rule.

The inherent nature of DMH systems introduces an additional layer of complexity through sparse feedback mechanisms. In DMH, meaningful rewards or penalty signals are only available once all tasks are served. This sparsity in the feedback loop exacerbates the difficulty of policy optimization, as the RL agent must learn to make decisions with limited immediate guidance on the long-term consequences of actions. Tables I and II show that regular CRL methods like LSAC fail to achieve high constraint satisfaction. CRL methods, including RCPOM and LSAC, require temporary penalty values to restrict the behavior of the policy, which, however, are usually missing in DMH. This is why, although RCPOM shows promising results compared to RL and CRL methods, it has a worse constraint satisfaction in terms of tardiness than EDD, even with a higher makespan.

ACERL benefits from the balance of maximizing rewards and satisfying constraints by the ISR with rank-based fitness. Individuals are grouped into distinct buffers based on the selected problem instances. This grouping allows for more meaningful comparisons between solutions within similar problem contexts as performances among different problems is incomparable. Instead of using the weighted sum of rewards and penalties, ACERL estimates the fitness of each individual based on its rank. The fitness of each individual is determined by its rank within its instance-specific group, serving as a unified metric, which enables fair comparisons across potentially diverse problem instances. The natural gradient induced by these fitness metrics efficiently guides the policy update process with the most promising direction for improving both reward and constraint satisfaction simultaneously. The population evolves toward feasible regions of the policy space that offer superior performance across multiple problem instances, promoting the development of policies that can generalize well to unseen DMH scenarios.

3) Stable Performance Across Multiple Instances: DMH considers training an agent on multiple instances and generalizing to multiple unseen instances. Each instance has its own context, like different objective value ranges and representations. This means that the performance of the same agent on different instances is incomparable. The challenge lies not only in achieving high performance on individual instances but in developing a robust policy that can adapt to the underlying patterns and principles of the dynamic system.

As shown in Tables I and II, ACERL shows the best performance on all instances. The diverse problem instances present a challenge in achieving an overall good performance for a single policy. There is no single dispatching rule that performs the best on all instances. EDD, a time-prefer rule, has the minimal tardiness F_t in DMH-02–DMH-04 with 29.6, 26.5, and 32.8, respectively. However, it rarely gets good results on makespan. A similar case happens on NVF, a distance-prefer rule, which performs fairly on makespan $F_m = 1876.5$ but violates the tardiness constraint on DMH-01 with $69.6 < 50$.

ACERL tackles the tradeoff of multiple training instances. Historical metrics such as obtained episodic reward and the number of selecting an instance are collected. An inverted distance metric is used to estimate how good the individual is in the view of the time horizon. We apply a UCB-based sampler to implement the adaptive training with the inverted distance metric and the number of selections. If an instance has been optimized well, i.e., the distance metric is small enough, then the proportion of the training instance decreases accordingly. The sampled population then interacts with environments instantiated by the adaptively selected training instances, for which the policy benefits the most. By adaptive training, the corresponding computational resource is allocated suitably. ACERL can explore and exploit more instances on which it performs badly to achieve an overall good performance. Besides, bias estimated by the ISR provokes some optimization pressure, which guides the gradient directions and helps with the next round's instance selection.

C. Robust Performance on Noised Instances

To further validate the effectiveness of ACERL, we validate the algorithm on some noised datasets. We extend the problem dataset by introducing random perturbations to the original data. The arrival time of each task is perturbed by this random noise, which is defined using the noise operator denoted as $\pm$. The number following $\pm$ specifies the magnitude of the perturbations. The initial training dataset and test dataset are labeled as DMH$\pm$0 and DMH$\pm$5, respectively. Subsequently, we created extended instance datasets with increasing levels of perturbation strength, labeled as DMH$\pm$10, DMH$\pm$15, DMH$\pm$20, DMH$\pm$25, and DMH$\pm$30. Table III summarizes the experiment results on noised datasets. More results are detailed in Section B of Supplementary Material.

It is observed that the performance rank of dispatching rules changes across different problem datasets. In DMH$\pm$0, the original training dataset, EDD has the highest constraint satisfaction. However, its constraint satisfaction descends and is even worse than NVF in DMH$\pm$20, DMH$\pm$25, and DMH$\pm$30. A similar case happens in makespan, NVF gradually achieves a better M over STD. ACERL outperforms other algorithms on several noised datasets, on which it has the best performance on metrics M, C, and P. The constraint satisfaction percentage of ACERL is much higher than the other algorithms, e.g., 100% on DMH$\pm$0, 97% on DMH$\pm$5, 95% on DMH$\pm$15, and 93% on DMH$\pm$30. ACERL still holds the first place in makespan with respect to metric M. Although RCPOM

TABLE III

PERFORMANCE ON NOISED INSTANCES. "$M/C(P)$" INDICATES THE AVERAGE NORMALIZED MAKESPAN, TARDINESS, AND PERCENTAGE OF CONSTRAINT SATISFACTION OVER 30 INDEPENDENT TRIALS OF FIVE DIFFERENT SEEDS. BOLD NUMBER INDICATES THE BEST VALUE IN THE CORRESPONDING COLUMN. HORIZONTAL RULES IN THE TABLE SEPARATE DIFFERENT GROUPS OF ALGORITHMS

Algorithm	DMH±0 $M/C(P)$	DMH±5 $M/C(P)$	DMH±10 $M/C(P)$	DMH±15 $M/C(P)$	DMH±20 $M/C(P)$	DMH±25 $M/C(P)$	DMH±30 $M/C(P)$
ACERL	**0.90/0.90 (100%)**	**0.89/0.92 (97%)**	**0.90/0.94 (80%)**	**0.94/0.94 (95%)**	**0.87/0.92** (75%)	**0.89/0.96 (84%)**	**0.93/0.96 (93%)**
MAPPO	0.72/0.72 (62%)	0.70/0.73 (65%)	0.79/0.83 (65%)	0.87/0.86 (85%)	0.83/0.89 (72%)	0.70/0.77 (55%)	0.78/0.82 (70%)
RCPOM	0.71/0.75 (60%)	0.71/0.76 (63%)	0.85/0.88 (64%)	0.85/0.88 (82%)	**0.87**/0.91 (66%)	0.81/0.84 (60%)	0.87/0.87 (70%)
LSAC	0.55/0.68 (56%)	0.57/0.70 (56%)	0.71/0.81 (57%)	0.67/0.74 (56%)	0.71/0.81 (56%)	0.70/0.81 (60%)	0.74/0.79 (60%)
AMAPPO	0.70/0.69 (57%)	0.67/0.69 (57%)	0.78/0.82 (67%)	0.79/0.79 (70%)	0.86/0.85 (70%)	0.77/0.77 (60%)	0.83/0.84 (70%)
ARCPOM	0.63/0.65 (48%)	0.65/0.68 (52%)	0.83/0.81 (57%)	0.76/0.76 (61%)	0.82/0.83 (57%)	0.85/0.84 (63%)	0.84/0.82 (69%)
ALSAC	0.60/0.69 (53%)	0.60/0.71 (55%)	0.77/0.84 (63%)	0.76/0.78 (66%)	0.78/0.85 (60%)	0.72/0.80 (60%)	0.79/0.79 (68%)
SAC	0.57/0.69 (57%)	0.59/0.70 (58%)	0.73/0.82 (59%)	0.70/0.76 (66%)	0.72/0.81 (57%)	0.71/0.80 (57%)	0.75/0.80 (62%)
PPO	0.61/0.71 (58%)	0.64/0.73 (59%)	0.77/0.83 (60%)	0.76/0.79 (63%)	0.80/0.85 (60%)	0.81/0.84 (60%)	0.83/0.83 (66%)
ASAC	0.55/0.65 (52%)	0.65/0.68 (56%)	0.70/0.78 (55%)	0.68/0.74 (63%)	0.68/0.77 (53%)	0.67/0.76 (55%)	0.69/0.76 (60%)
APPO	0.62/0.65 (50%)	0.57/0.66 (53%)	0.82/0.79 (58%)	0.78/0.75 (63%)	0.83/0.83 (61%)	0.84/0.82 (61%)	0.83/0.80 (64%)
MIX	0.49/0.65 (54%)	0.49/0.66 (54%)	0.60/0.76 (54%)	0.60/0.73 (59%)	0.60/0.75 (47%)	0.63/0.77 (57%)	0.62/0.75 (53%)
FCFS	0.25/0.45 (37%)	0.27/0.45 (35%)	0.24/0.54 (37%)	0.18/0.37 (35%)	0.26/0.42 (30%)	0.26/0.45 (34%)	0.12/0.39 (32%)
EDD	0.52/0.85 (86%)	0.48/0.74 (73%)	0.55/0.84 (65%)	0.52/0.87 (80%)	0.43/0.73 (57%)	0.46/0.75 (62%)	0.46/0.76 (53%)
NVF	0.56/0.63 (51%)	0.59/0.66 (58%)	0.77/0.82 (61%)	0.76/0.77 (65%)	0.86/0.89 (**77%**)	0.84/0.86 (62%)	0.92/0.89 (81%)
STD	0.65/0.70 (53%)	0.65/0.71 (55%)	0.86/0.83 (62%)	0.82/0.82 (76%)	0.82/0.85 (59%)	0.72/0.80 (58%)	0.82/0.84 (72%)
Random	0.02/0.00 (12%)	0.02/0.03 (11%)	0.00/0.00 (10%)	0.04/0.04 (13%)	0.04/0.01 (9%)	0.03/0.02 (7%)	0.01/0.00 (9%)

achieves a slightly higher M with 0.88 over ACERL with 0.87, its constraint satisfaction percentage P is rarely higher than ACERL, i.e., 66% <75% in DMH±20.

MIX selects dispatching rules uniformly at random and guarantees a lower performance bound of assigning tasks. From Table III, MIX has a better performance than FCFS in both makespan and tardiness and is better than EDD in makespan. Learning policies, including ACERL, MAPPO, RCPOM, as well as other RL and CRL policies, aim at learning a suitable policy to choose a proper rule within a given state, i.e., improving the lower bound. However, they can hardly make it due to the tradeoff among multiple instances, as well as sparse feedback.

ACERL learns from the diverse experiences sampled by the population. The adaptive selection of training instances prevents ACERL from focusing solely on specific instances during training, enabling a strong overall performance across multiple instances. ISR balances the conflict in terms of maximizing rewards and satisfying constraints according to the episodic values including makespan and tardiness. The obtained rank-based fitness helps with the estimation of the natural gradient for updating the policy. ACERL is particularly well-suited to tackling sparse reward and penalty in DMH, as it requires only episode-level values to guide the learning process.

A notable strength of ACERL lies in maintaining robust performance even when the noised dataset gradually differs from the original training dataset. This robustness suggests that the learned policy captures fundamental principles of DMH rather than merely memorizing specific instance characteristics. It indicates potential applications of ACERL in complex real-world scenarios with dynamics and constraints.

D. Leave-One-Out Cross Validation

To further explore ACERL's effectiveness and limitations, we divide the training set into subsidiary training and test sets by the leave-one-out method. For example, DMH-01 is stripped out as the test instance, and then, the remaining instances (DMH-02–DMH-08) are used to train the policy. Table IV shows the experiment results by leaving DMH-01 out. More results are attached in Section C of Supplementary Material.

Still, our proposed method, ACERL, achieves the highest M value of 0.98% and 87% constrained satisfaction. Across DMH-02–DMH-08, ACERL achieves the best makespans (F_m). However, we also find that ACERL does not perform well on the left instance DMH-01 as expected, with F_m = 1914.1 and F_t = 54.0. At the same time, STD, a dispatching rule achieves the best makespan value of 1868.8 and EDD achieves the lowest tardiness F_t = 33.9. The phenomenon is attributed to the out of distribution. The diverse training instances are intentionally designed to facilitate a comprehensive evaluation, and thus, instances may differ from each other significantly. It is not surprising to observe the limited performance of learning policies on DMH-01, which is separated from the training set as they have never encountered the instance before. At the same time, policies can truly learn something from the remaining training instances. Learning policies, including ACERL, RCPOM, and SAC, show superior performance than the MIX policy, which selects dispatching rules randomly. Some special knowledge is able to transfer from some "similar" instances to the split instance. Although ACERL does not demonstrate superior performance in terms of makespan on the split instance, it still achieves the overall best performance.

E. Ablation Study

To fully evaluate ACERL and analyze each ingredient, a rigorous ablation study is conducted. Compared strategies are organized into the following two groups.

1) Four ranking strategies are compared. ISR with rank-based fitness is denoted as "ISR." The weighted sum of raw rewards and penalties is denoted as "CF." Rank-based fitness is denoted as "R." Raw reward is denoted as "F."

TABLE IV

CROSS VALIDATION USING LEAVE-ONE-OUT ON DMH-01 (HIGHLIGHTED WITH GRAY BLOCKS): AVERAGE MAKESPAN AND TARDINESS OVER 30 INDEPENDENT TRIALS OF FIVE DIFFERENT SEEDS ON EACH INSTANCE. BOLD NUMBERS INDICATE THE BEST MAKESPAN AND TARDINESS. "+,""≈," AND "−" INDICATE THE POLICY PERFORMS STATISTICALLY BETTER/SIMILAR/WORSE THAN ACERL POLICY, RESPECTIVELY. THE NUMBER OF POLICIES THAT ARE "BETTER," "SIMILAR," AND "WORSE" THAN ACERL IN TERMS OF MAKESPAN AND TARDINESS ON EACH INSTANCE IS SUMMARIZED IN THE BOTTOM ROW. THE LAST COLUMN WITH HEADER "$M/C(P)$" INDICATES THE AVERAGE NORMALIZED MAKESPAN, TARDINESS, AND PERCENTAGE OF CONSTRAINT SATISFACTION. HORIZONTAL RULES IN THE TABLE SEPARATE DIFFERENT GROUPS OF ALGORITHMS

Algorithm	DMH-01 F_m/F_t	DMH-02 F_m/F_t	DMH-03 F_m/F_t	DMH-04 F_m/F_t	DMH-05 F_m/F_t	DMH-06 F_m/F_t	DMH-07 F_m/F_t	DMH-08 F_m/F_t	$M/C(P)$
ACERL	1914.1/54.0	**1861.8/28.6**	**1834.8**/31.0	**1919.8**/37.8	**1870.0**/35.3	**1878.8/38.6**	**1927.0/9.1**	**1864.4**/34.1	0.97/0.94 (87%)
MAPPO	**1867.4**+/62.9-	1908.8-/36.7-	1907.0-/47.0-	1945.2-/41.1-	1962.2-/55.4-	2023.8-/102.3-	1927.0≈/10.8-	1881.2-/39.2≈	0.82/0.76 (55%)
RCPOM	1900.7+/72.5-	1957.5-/54.1-	1925.1-/57.5-	1970.1-/46.0-	1919.0-/39.5≈	1976.1-/85.9-	1985.0-/27.2-	1929.3-/42.1-	0.71/0.71 (49%)
LSAC	1926.8≈/69.7-	1970.5-/57.4-	1924.5-/49.9-	1963.3-/44.4-	1940.4-/42.6-	1996.9-/78.2-	1952.9-/15.6-	1901.6-/30.9+	0.72/0.77 (52%)
AMAPPO	1904.2+/66.1≈	1949.2-/46.3-	1889.8-/42.6-	1950.6-/39.9-	1892.4-/**33.0**+	1989.8-/79.2-	1932.8-/13.4-	1889.0-/**23.4**+	0.81/0.83 (62%)
ARCPOM	1901.3≈/59.1≈	1988.2-/53.1-	1927.1-/46.4-	1986.8-/51.6-	1996.5-/50.6-	1991.6-/67.8-	1955.7-/17.1-	1932.7-/34.2-	0.67/0.78 (54%)
ALSAC	1901.3≈/56.4-	1968.8-/53.1-	1936.9-/42.7-	1985.3-/49.1-	1968.0-/51.2-	1990.3-/72.1-	1959.6-/13.3-	1933.5-/34.5≈	0.68/0.79 (58%)
SAC	1893.1+/64.5-	1972.1-/58.3-	1925.5-/43.6-	1971.6-/44.3-	1936.2-/41.8-	1978.5-/72.2-	1955.2-/16.4-	1925.9-/33.1+	0.73/0.78 (53%)
PPO	1901.9+/60.2≈	1994.2-/56.6-	1931.4-/43.9-	1989.7-/50.9-	1979.5-/54.4-	2008.2-/84.8-	1957.8-/17.3-	1922.9-/37.1≈	0.67/0.74 (53%)
ASAC	1901.6≈/55.4≈	1985.6-/54.6-	1924.2-/44.2-	2010.8-/53.0-	1974.1-/47.7-	1997.7-/70.1-	1964.6-/14.9-	1925.1-/36.8≈	0.66/0.78 (58%)
APPO	1921.4≈/62.0≈	2006.3-/59.7-	1942.5-/45.7-	2001.9-/53.4-	2005.3-/57.2-	2057.6-/95.2-	1948.0-/15.7-	1940.7-/37.4≈	0.60/0.71 (52%)
MIX	1939.9-/60.3≈	2000.9-/60.4-	1932.2-/45.0-	2006.1-/52.8-	2028.2-/64.2-	2027.1-/69.5-	1969.2-/16.1-	1971.0-/46.9-	0.57/0.72 (54%)
FCFS	2081.1-/90.2-	2084.5-/82.3-	2014.7-/64.2-	2136.7-/105.0-	2194.6-/122.5-	2123.5-/85.2-	1927.9≈/11.2-	1933.6-/27.4+	0.28/0.49 (37%)
EDD	1903.2≈/**33.9**+	1968.6-/29.6≈	1977.8-/**26.5**≈	1988.1-/**32.8**+	1950.7-/33.7≈	2016.8-/46.8≈	1940.5-/12.1-	2020.8-/45.3≈	0.62/0.94 (86%)
NVF	1876.5+/69.6-	1958.4-/56.4-	1946.7-/49.6-	2040.6-/66.7-	1933.9-/56.3-	1953.4-/78.5-	1996.5-/21.8-	1944.6-/35.5≈	0.66/0.70 (51%)
STD	1868.8+/66.6≈	1961.7-/49.2-	1921.3-/51.7-	1970.7-/35.6+	1917.1-/41.4-	1955.4-/62.7-	1983.7-/30.4-	1883.5-/35.1≈	0.77/0.78 (53%)
Random	2098.7-/124.5-	2113.8-/103.1-	2091.2-/143.7-	2135.1-/123.1-	2149.3-/119.0-	2159.1-/129.3-	2083.0-/70.2-	2067.8-/89.5-	0.02/0.00 (12%)
	(7/6/3) / (1/7/8)	(0/0/16) / (0/1/15)	(0/0/16) / (0/1/15)	(0/0/16) / (2/0/14)	(0/0/16) / (1/2/13)	(0/0/16) / (0/1/15)	(0/2/14) / (0/0/16)	(0/0/16) / (4/8/4)	

TABLE V

ABLATION STUDY ON TRAINING INSTANCES (DMH-01–DMH-08). BOLD NUMBER INDICATES THE BEST MAKESPAN AND TARDINESS. "$M/C(P)$" INDICATES THE AVERAGE NORMALIZED MAKESPAN, TARDINESS, AND PERCENTAGE OF CONSTRAINT SATISFACTION. STRATEGY DENOTES THE RANKING STRATEGIES, INCLUDING ISR, CF, R, AND F. MODE DENOTES TRAINING MODES, INCLUDING AIS, UNIFORM, AND RANDOM. DIFFERENT GROUPS OF ALGORITHMS ARE DIVIDED BY THE ROW LINES ACCORDING TO THE RANKING STRATEGIES AND TRAINING MODES

Strategy	Mode	DMH-01 F_m/F_t	DMH-02 F_m/F_t	DMH-03 F_m/F_t	DMH-04 F_m/F_t	DMH-05 F_m/F_t	DMH-06 F_m/F_t	DMH-07 F_m/F_t	DMH-08 F_m/F_t	$M/C(P)$
ISR	AIS	1797.8/**29.5**	1859.2/30.7	1840.0/28.6	**1896.6**/35.7	1856.4/29.0	1864.4/35.5	1929.6/9.5	1856.8/25.4	0.90/0.90 (100%)
ISR	Uniform	1817.1/47.5	1869.8/33.3	1839.4/25.8	1906.6/37.1	**1842.4**/23.7	1873.8/**31.9**	1932.8/10.0	1863.8/24.5	0.88/0.89 (91%)
ISR	Random	1800.0/42.0	1881.9/36.9	1860.4/30.2	1902.2/34.3	1881.2/35.8	1865.6/37.7	1929.6/9.1	**1852.2**/32.0	0.87/0.86 (94%)
CF	AIS	1802.0/35.2	1858.2/30.5	1848.4/20.2	1901.9/32.7	1855.8/21.8	1872.2/38.6	1932.8/6.0	1874.6/20.6	0.88/0.93 (95%)
CF	Uniform	1802.5/41.8	1874.8/33.7	1840.0/24.6	1930.0/**32.0**	1845.8/23.4	1875.0/38.2	**1927.0/4.8**	1864.0/25.7	0.88/0.90 (92%)
CF	Random	1815.4/42.6	1883.2/39.6	**1837.0/19.9**	1930.2/35.3	1844.6/**20.2**	1891.6/39.4	1932.8/7.0	1862.6/24.4	0.86/0.89 (92%)
R	AIS	1794.8/34.9	1858.2/**28.6**	1840.0/22.5	1928.7/37.6	1863.0/33.7	**1854.8**/33.2	1927.0/7.9	1860.8/**19.7**	0.89/0.91 (93%)
R	Uniform	1824.2/43.9	1860.9/31.5	1849.0/27.2	1926.3/33.8	1861.8/30.7	1865.2/37.1	1927.0/7.2	1859.0/27.7	0.87/0.88 (91%)
R	Random	1815.6/42.0	**1849.4**/28.8	1839.6/36.2	1937.2/34.7	1857.0/26.2	1883.1/41.7	1927.0/7.6	1882.6/25.6	0.86/0.88 (87%)
F	AIS	**1794.0**/46.9	1866.8/32.4	1843.0/33.2	1928.2/39.0	1865.4/41.4	1856.0/35.0	1932.8/10.0	1854.8/24.6	0.88/0.86 (87%)
F	Uniform	1807.4/52.7	1863.4/29.4	1844.5/29.5	1933.8/36.6	1874.2/36.9	1876.8/38.1	1927.0/10.3	1859.2/28.4	0.87/0.85 (88%)
F	Random	1800.2/46.4	1881.8/38.4	1837.0/24.7	1924.6/38.2	1850.4/24.9	1872.0/38.1	1935.4/8.5	1857.4/25.7	0.88/0.87 (87%)

"ISR": Intrinsic stochastic ranking with rank-based fitness "CF": Weighted sum of raw rewards and penalties
"R": Raw reward with rank-based fitness "F": Raw reward
"AIS": Adaptive instance sampler "Uniform": Fixed instances "Random": Randomly selecting instances

2) "AIS," "Uniform," and "Random" are three different training modes. "AIS" denotes the policy trained with the AIS. "Uniform" denotes that each instance is equally selected and fixed during training. "Random" means that instances are selected randomly.

1) Effect of ISR With Rank-Based Fitness: The experimental results are presented in Tables V and VI. Ranking strategy "F" can obtain good results on makespan, but it hardly satisfies the constraints with a high rate. Similarly, although "R," a rank-based fitness strategy, has a slightly better makspan than "F," it still does not perform well in satisfying the problem constraints, i.e., making the tardiness of tasks descend to a promising threshold. "CF," which applies the regular weighted sum of makespan and tardiness, tries to improve the likelihood of satisfying the constraints while only obtaining comparable results with "F" and "R." The results from the ablation study verify the effectiveness of the "ISR" strategy, in which we obtain excellent performance on makespan in various training modes with better constraint satisfaction, compared with "F," "R," and "CF." ACERL that uses the "ISR" strategy can achieve 100% constraint satisfaction on the training set in the AIS mode. Our proposed "ISR" strategy not only can eliminate

TABLE VI

ABLATION STUDY ON TEST INSTANCES (DMH-09–DMH-16). BOLD NUMBER INDICATES THE BEST MAKESPAN AND TARDINESS. "$M/C(P)$" INDICATES THE AVERAGE NORMALIZED MAKESPAN, TARDINESS, AND PERCENTAGE OF CONSTRAINT SATISFACTION. STRATEGY DENOTES THE RANKING STRATEGIES, INCLUDING ISR, CF, R, AND F. MODE DENOTES TRAINING MODES, INCLUDING AIS, UNIFORM, AND RANDOM. DIFFERENT GROUPS OF ALGORITHMS ARE DIVIDED BY THE ROW LINES ACCORDING TO THE RANKING STRATEGIES AND TRAINING MODES

Strategy	Mode	DMH-09 F_m/F_t	DMH-10 F_m/F_t	DMH-11 F_m/F_t	DMH-12 F_m/F_t	DMH-13 F_m/F_t	DMH-14 F_m/F_t	DMH-15 F_m/F_t	DMH-16 F_m/F_t	$M/C(P)$
ISR	AIS	1801.6/**29.9**	1878.4/30.6	1892.7/34.7	**1896.8**/35.6	1894.9/24.2	1865.0/36.1	1932.8/9.6	1857.0/25.8	0.89/0.92 (97%)
ISR	Uniform	1823.0/47.3	1891.2/34.2	1868.6/28.4	1913.7/37.2	1878.7/26.4	1867.6/**32.6**	1934.8/10.0	1888.2/26.8	0.87/0.90 (92%)
ISR	Random	1804.6/38.7	1879.6/36.5	1877.0/29.3	1919.6/34.2	1923.4/35.0	1871.8/37.2	1932.9/9.3	**1852.0**/32.3	0.88/0.89 (93%)
CF	AIS	1802.0/35.5	1877.6/30.1	1876.0/23.8	1925.0/34.2	1887.1/**23.0**	1888.2/44.1	1934.8/7.7	1874.4/**20.8**	0.87/0.93 (92%)
CF	Uniform	1801.7/42.2	1884.5/33.7	1870.8/24.3	1931.4/**31.8**	**1853.8**/23.6	1874.0/38.6	1941.2/**6.5**	1870.0/26.7	0.88/0.92 (92%)
CF	Random	1816.0/42.9	1882.4/39.3	1870.8/**23.6**	1940.8/35.2	1886.8/24.2	1906.0/43.0	1934.8/8.6	1863.6/24.7	0.85/0.90 (92%)
R	AIS	1807.2/34.6	1869.6/**29.6**	1886.8/26.8	1932.2/37.3	1884.3/31.0	**1854.2**/33.6	**1930.0**/8.8	1874.6/21.4	0.88/0.93 (93%)
R	Uniform	1824.8/44.4	**1863.5**/31.6	1874.1/29.5	1955.6/34.6	1886.7/31.3	1886.0/35.3	1930.0/8.2	1858.8/28.0	0.86/0.90 (91%)
R	Random	1813.5/42.4	1873.5/32.1	1916.9/46.3	1936.8/34.7	1885.0/29.2	1883.7/42.9	1930.0/7.7	1880.4/25.6	0.84/0.88 (82%)
F	AIS	1801.7/47.0	1876.2/34.7	1864.8/35.4	1927.8/38.7	1924.6/39.7	1856.4/35.6	1965.6/21.2	1862.6/25.9	0.86/0.85 (82%)
F	Uniform	1802.8/45.4	1906.4/32.9	1876.1/37.4	1946.8/37.6	1915.5/37.3	1876.9/38.8	1995.4/31.2	1860.0/28.9	0.80/0.84 (80%)
F	Random	**1800.2**/46.8	1881.0/38.0	**1864.2**/27.5	1932.8/38.5	1935.0/36.4	1888.0/43.9	1937.6/8.7	1866.2/27.1	0.86/0.87 (82%)

"ISR": Intrinsic stochastic ranking with rank-based fitness "CF": Weighted sum of raw rewards and penalties
"R": Raw reward with rank-based fitness "F": Raw reward
"AIS": Adaptive instance sampler "Uniform": Fixed instances "Random": Randomly selecting instances

the effects of scale between rewards and constraint values but also can well consider reward maximization and constraint satisfaction simultaneously.

2) Necessity of Adaptive Training: When a policy is trained on one single instance, i.e., without assuming multiple instances available as historical records, the policy performs well on the training instance but fails to generalize to new instances. Specifically, it results in a high makespan and struggles to meet constraints on different instances, which underscores the limitations of a model trained on just one instance. Detailed experiment results are presented in Section A of Supplementary Material. Subsequently, we trained the policies on three training modes, including "AIS," "Uniform," and "Random" to verify the effectiveness of our adaptive approach. From Tables V and VI, we can observe that using our proposed "AIS" as the training model results in a better makespan and a higher constraint satisfaction percentage.

Compared with the training modes of "Uniform" and "Random," our proposed "AIS" training mode is able to dynamically and adaptively adjust its proposition of training instances during the training process. The suitable choice of training instances enables a more reasonable and efficient allocation of computational resources, hence helping ACERL to achieve an overall good performance across all instances.

3) Coordination of ACERL: We demonstrated the effectiveness of our proposed ACERL and the unique contributions of its ingredients through rigorous and comprehensive ablation experiments, respectively. Overall, as observed in Tables V and VI, our ACERL achieves the best makespan and the highest constraint satisfaction percentage compared to all other combinations of ranking strategies and training modes. The AIS training mode considers multiple instances and adaptively allocates resources based on performance. The ISR strategy effectively mitigates the effect of value scales and achieves a well-balanced tradeoff between reward and constraint, resulting in excellent performance across all instances. All ingredients of ACERL, including ISR with rank-based fitness and adaptive training, cooperate well and are indispensable for solving the problem.

VI. CONCLUSION

In this article, we consider the DMH problem with uncertainties and sparse feedback, where unexpected dynamic events and constraints are present. Due to the dynamics, it is hard to determine the unique contribution of each task assignment, which leads to an issue of sparse feedback in both rewards and penalties. Although the existence of historical task records enables training a policy with multiple instances, the tradeoff across training instances poses a challenge with limited computational resources. To address the challenges, we proposed a robust adaptive constrained ERL approach, called ACERL. ACERL leverages natural gradient ascent to update its parameters. The population-based exploration provides diverse experiences by sampling noised actors at each generation. The adaptive instance sampler keeps choosing the training instance from which the policy benefits the most for policy improvement. Reasonable computational resource allocation is allowed. To balance the rewards and penalties, we present ISR with rank-based fitness. Instead of common reward-based, real-valued fitness metrics, we incorporate rank-based fitness to estimate the natural gradient, which implies the optimization direction of both maximizing rewards and constraint satisfaction at the same time. ISR provides an independent ranking for different instances based on rewards and penalties, which enables a bias for adaptive instance selection. Extensive experiments show that ACERL outperforms advanced RL methods, CRL methods, and classic dispatching rules on eight training instances and eight test instances in terms of maximizing rewards with constraint satisfaction. ACERL not only achieves the best makespan but also fully satisfies tardiness constraints on nearly all instances. Besides, ACERL is evaluated on five datasets with a total of 40

noised instances to simulate real-world scenarios. ACERL presents a robust and outstanding performance on all datasets. Leave-one-out cross validation is conducted on ACERL by splitting the training dataset into subsidiary training and test datasets and presents the overall effectiveness of ACERL. Furthermore, a comprehensive ablation study demonstrates the unique contribution of ACERL's core ingredients.

In future work, it is worth applying ACERL to more real-world problems such as vehicle routing and grid optimization. It is also interesting to figure out delicate mechanisms for a better generalization. Besides, the theoretical aspect presents an intriguing avenue for future research, potentially bridging the gap between empirical success and theoretical understanding in constrained ERL. Theoretical analysis will be a valuable yet challenging future work.

REFERENCES

[1] W.-Q. Zou, Q.-K. Pan, and L. Wang, "An effective multi-objective evolutionary algorithm for solving the AGV scheduling problem with pickup and delivery," *Knowl.-Based Syst.*, vol. 218, Apr. 2021, Art. no. 106881.

[2] N. Singh, Q.-V. Dang, A. Akcay, I. Adan, and T. Martagan, "A matheuristic for AGV scheduling with battery constraints," *Eur. J. Oper. Res.*, vol. 298, no. 3, pp. 855–873, May 2022.

[3] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *J. Scheduling*, vol. 12, no. 4, pp. 417–431, Aug. 2009.

[4] V. Kaplanoğlu, C. Şahin, A. Baykasoğlu, R. Erol, A. Ekinci, and M. Demirtaş, "A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles (AGV) in manufacturing systems by considering AGV breakdowns," *Int. J. Eng. Res. Innov.*, vol. 7, no. 2, pp. 32–38, 2015.

[5] C. Hu, Z. Wang, J. Liu, J. Wen, B. Mao, and X. Yao, "Constrained reinforcement learning for dynamic material handling," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2023, pp. 1–9.

[6] H. Hu, X. Jia, Q. He, S. Fu, and K. Liu, "Deep reinforcement learning based AGVs real-time scheduling with mixed rule for flexible shop floor in industry 4.0," *Comput. Ind. Eng.*, vol. 149, Nov. 2020, Art. no. 106749.

[7] Y. Jeong, T. K. Agrawal, E. Flores-García, and M. Wiktorsson, "A reinforcement learning model for material handling task assignment and route planning in dynamic production logistics environment," in *Proc. CIRP*, vol. 104, Jan. 2021, pp. 1807–1812.

[8] M. D. Dennis et al., "Emergent complexity and zero-shot transfer via unsupervised environment design," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, Jan. 2020, pp. 13049–13061.

[9] J. H. Blackstone, D. T. Phillips, and G. L. Hogg, "A state-of-the-art survey of dispatching rules for manufacturing job shop operations," *Int. J. Prod. Res.*, vol. 20, no. 1, pp. 27–45, 1982.

[10] I. Sabuncuoglu, "A study of scheduling rules of flexible manufacturing systems: A simulation approach," *Int. J. Prod. Res.*, vol. 36, no. 2, pp. 527–546, Feb. 1998.

[11] G. Chryssolouris and V. Subramaniam, "Dynamic scheduling of manufacturing job shops using genetic algorithms," *J. Intell. Manuf.*, vol. 12, no. 3, pp. 281–293, Jun. 2001.

[12] M. P. Li, P. Sankaran, M. E. Kuhl, A. Ganguly, A. Kwasinski, and R. Ptucha, "Simulation analysis of a deep reinforcement learning approach for task selection by autonomous material handling vehicles," in *Proc. Winter Simul. Conf. (WSC)*, Dec. 2018, pp. 1073–1083.

[13] D. Dewey, "Reinforcement learning and the reward engineering principle," in *Proc. AAAI Spring Symp. Ser.*, Mar. 2014, pp. 1–4.

[14] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proc. Int. Conf. Mach. Learn.*, vol. 99, San Francisco, CA, USA, Sep. 1999, pp. 278–287.

[15] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 22–31.

[16] E. Altman, *Constrained Markov Decision Processes: Stochastic Modeling*. Evanston, IL, USA: Routledge, 1999.

[17] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," 2017, *arXiv:1703.03864*.

[18] Z. Hu and W. Gong, "Constrained evolutionary optimization based on reinforcement learning using the objective function and constraints," *Knowl.-Based Syst.*, vol. 237, Feb. 2022, Art. no. 107731.

[19] J. C. Tay and N. B. Ho, "Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems," *Comput. Ind. Eng.*, vol. 54, no. 3, pp. 453–473, Apr. 2008.

[20] C. Chen, L.-F. Xi, B.-H. Zhou, and S.-S. Zhou, "A multiple-criteria real-time scheduling approach for multiple-load carriers subject to LIFO loading constraints," *Int. J. Prod. Res.*, vol. 49, no. 16, pp. 4787–4806, Aug. 2011.

[21] S. Liu, P. H. Tan, E. Kurniawan, P. Zhang, and S. Sun, "Dynamic scheduling for pickup and delivery with time windows," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, Feb. 2018, pp. 767–770.

[22] P. Yan, S. Q. Liu, T. Sun, and K. Ma, "A dynamic scheduling approach for optimizing the material handling operations in a robotic cell," *Comput. Oper. Res.*, vol. 99, pp. 166–177, Nov. 2018.

[23] U. A. Umar, M. K. A. Ariffin, N. Ismail, and S. H. Tang, "Hybrid multiobjective genetic algorithms for integrated dynamic scheduling and routing of jobs and automated-guided vehicle (AGV) in flexible manufacturing systems (FMS) environment," *Int. J. Adv. Manuf. Technol.*, vol. 81, nos. 9–12, pp. 2123–2141, Dec. 2015.

[24] W. Wang, Y. Zhang, and R. Y. Zhong, "A proactive material handling method for CPS enabled shop-floor," *Robot. Comput.-Integr. Manuf.*, vol. 61, Feb. 2020, Art. no. 101849.

[25] Z. Li, H. Sang, Q. Pan, K. Gao, Y. Han, and J. Li, "Dynamic AGV scheduling model with special cases in matrix production workshop," *IEEE Trans. Ind. Informat.*, vol. 19, no. 6, pp. 7762–7770, Jun. 2023.

[26] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[27] C. Hu et al., "Reinforcement learning with dual-observation for general video game playing," *IEEE Trans. Games*, vol. 15, no. 2, pp. 202–216, Feb. 2023.

[28] D. Silver et al., "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[29] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

[30] C. Chen, B. Xia, B.-H. Zhou, and L. Xi, "A reinforcement learning based approach for a multiple-load carrier scheduling problem," *J. Intell. Manuf.*, vol. 26, no. 6, pp. 1233–1245, Dec. 2015.

[31] T. Xue, P. Zeng, and H. Yu, "A reinforcement learning method for multi-AGV scheduling in manufacturing," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Feb. 2018, pp. 1557–1561.

[32] C. Kardos, C. Laflamme, V. Gallina, and W. Sihn, "Dynamic scheduling in a job-shop production system with reinforcement learning," in *Proc. CIRP*, vol. 97, May 2021, pp. 104–109.

[33] Y. Li et al., "Real-time scheduling for flexible job shop with AGVs using multiagent reinforcement learning and efficient action decoding," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 55, no. 3, pp. 2120–2132, Mar. 2025.

[34] M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, Dec. 2016, pp. 1479–1487.

[35] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. D. Turck, and P. Abbeel, "VIME: Variational information maximizing exploration," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2016, pp. 1117–1125.

[36] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2018, pp. 1–15. [Online]. Available: https://openreview.net/forum?id=SkfrvsA9FX

[37] M. Jiang, E. Grefenstette, and T. Rocktäschel, "Prioritized level replay," in *Proc. Int. Conf. Mach. Learn.*, Jan. 2020, pp. 4940–4950.

[38] X. Yao, "Evolving artificial neural networks," *Proc. IEEE*, vol. 87, no. 9, pp. 1423–1447, Sep. 1999.

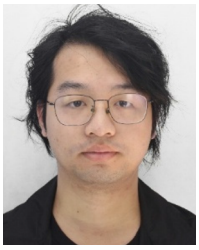
[39] O. Sigaud, "Combining evolution and deep reinforcement learning for policy search: A survey," *ACM Trans. Evol. Learn. Optim.*, vol. 3, no. 3, pp. 1–20, Sep. 2023.

[40] H. Bai, R. Cheng, and Y. Jin, "Evolutionary reinforcement learning: A survey," *Intell. Comput.*, vol. 2, p. 25, Jan. 2023.

[41] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," 2017, *arXiv:1712.06567*.

[42] E. Conti, V. Madhavan, F. P. Such, J. Lehman, K. O. Stanley, and J. Clune, "Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, Jan. 2018, pp. 5027–5038.

[43] P. Yang, H. Zhang, Y. Yu, M. Li, and K. Tang, "Evolutionary reinforcement learning via cooperative coevolutionary negatively correlated search," *Swarm Evol. Comput.*, vol. 68, Feb. 2022, Art. no. 100974.
[44] S. Khadka and K. Tumer, "Evolution-guided policy gradient in reinforcement learning," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1196–1208.
[45] C. Hu, J. Pei, J. Liu, and X. Yao, "Evolving constrained reinforcement learning policy," in *Proc. Int. Joint Conf. Neural Netw.*, Jun. 2023, pp. 1–8.
[46] R. Agrawal, "Sample mean based index policies by O(logn) regret for the multi-armed bandit problem," *Adv. Appl. Probab.*, vol. 27, no. 4, pp. 1054–1078, Dec. 1995.
[47] B. Hao, Y. Abbasi-Yadkori, Z. Wen, and G. Cheng, "Bootstrapping upper confidence bound," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, Jan. 2019, pp. 1–11.
[48] K. Watanabe, M. Hashem, K. Watanabe, and M. Hashem, "Evolutionary optimization of constrained problems," in *Evolutionary Computations: New Algorithms and Their Applications to Evolutionary Robots*. Berlin, Germany: Springer, 2004, pp. 53–64.
[49] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 284–294, Sep. 2000.
[50] T. Runarsson and X. Yao, "Constrained evolutionary optimization," in *Evolutionary Optimization*. Cham, Switzerland: Springer, 2003, pp. 87–113.
[51] K. Tang, Y. Mei, and X. Yao, "Memetic algorithm with extended neighborhood search for capacitated arc routing problems," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 1151–1166, Oct. 2009.
[52] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, "Natural evolution strategies," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 949–980, Mar. 2014.
[53] Y. Nesterov and V. Spokoiny, "Random gradient-free minimization of convex functions," *Found. Comput. Math.*, vol. 17, pp. 527–566, Apr. 2017.
[54] K. Choroma´nski, A. Pacchiano, J. Parker-Holder, Y. Tang, and V. Sindhwani, "From complexity to simplicity: Adaptive ES-active subspaces for blackbox optimization," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, vol. 32, Mar. 2019, pp. 10299–10309.
[55] F.-Y. Liu, Z.-N. Li, and C. Qian, "Self-guided evolution strategies with historical estimated gradients," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 1474–1480.
[56] A. Tamar, D. Di Castro, and S. Mannor, "Policy gradients with variance related risk criteria," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2012, pp. 387–396.
[57] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[58] J. Weng et al., "Tianshou: A highly modularized deep reinforcement learning library," *J. Mach. Learn. Res.*, vol. 23, no. 267, pp. 1–6, 2022.
[59] G. Brockman et al., "OpenAI gym," 2016, *arXiv:1606.01540*.

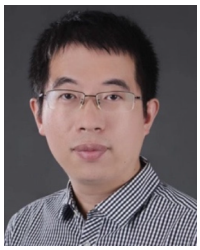


Chengpeng Hu (Student Member, IEEE) received the B.E. and M.E. degrees from the Southern University of Science and Technology, Shenzhen, China, in 2017 and 2021, respectively. He is currently pursuing the Ph.D. degree in industrial engineering and innovation sciences with Department of Industrial Engineering and Innovation Sciences of Eindhoven University of Technology, Eindhoven, The Netherlands.

His research interests include explainable AI, reinforcement learning, evolutionary computation, and their applications in combinatorial optimization and games.

Ziming Wang (Student Member, IEEE) received the B.S. degree in software engineering from South China Agricultural University, Guangzhou, China, in 2021, and the M.S. degree in electronics science and technology from the Southern University of Science and Technology (SUSTech), Shenzhen, China, in 2024, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering.

His research interests include explainable artificial intelligence and multiobjective optimization.



Bo Yuan (Member, IEEE) received the B.Sc. and Ph.D. degrees in electronic science and technology from the University of Science and Technology of China, Hefei, China, in 2009 and 2014, respectively.

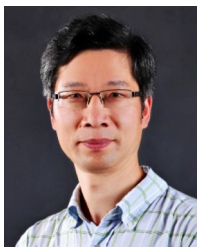
He is currently an Assistant Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His current research interests include computational intelligence, reinforcement learning, and electronic design automation.

Jialin Liu (Senior Member, IEEE) received the B.E. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2010, the Diplôme d'Ingénieur degree from Polytech'Paris-Sud, Orsay, France, in 2012, the dual M.Sc. degree from École Polytechnique, Palaiseau, France, and Université Paris-Sud in 2013, and the Ph.D. degree from Universit é Paris-Saclay, Orsay, in 2016.

She is currently an Associate Professor at Lingnan University, Hong Kong, SAR, China. Her research interests include artificial intelligence in games, evolutionary computation, smart logistics, and fair machine learning.

Dr. Liu is an Associate Editor of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE, IEEE TRANSACTIONS ON GAMES, and *Knowledge-Based Systems*.

Chengqi Zhang (Life Senior Member, IEEE) joined the Department of Data Science and Artificial Intelligence, The Hong Kong Polytechnic University (PolyU), as a Chair Professor of Artificial Intelligence in September 2024. He was the Director of Shenzhen Research Institute, PolyU, in January 2025. From December 2017 to November 2025, he was a Pro Vice-Chancellor (China Enterprise) at the University of Technology Sydney (UTS), Ultimo, NSW, Australia; from February 2017 to 2027, he was a Distinguished Professor at UTS; and a Research Professor of information technology at UTS in December 2001. He was an Associate Professor at Deakin University, Burwood, VIC, Australia, from January 1999 to December 2001; an Associate Professor at the University of New England (UNE), Biddeford, ME, USA, from January 1998 to January 1999; a Senior Lecturer at UNE from January 1994 to January 1998; and a Lecturer at UNE from January 1990 to January 1994. In addition, he was appointed as an Adjunct Professor at the University of New South Wales (UNSW), Sydney, NSW, Australia, from March 2017 to March 2020; and an Honorary Professor at the University of Queensland (UQ) from March 2015 to December 2017.



Xin Yao (Fellow, IEEE) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1982, the M.Sc. degree from the North China Institute of Computing Technology, Beijing, China, in 1985, and the Ph.D. degree from USTC, in 1990.

He worked at the University of Birmingham, Birmingham, U.K.; SUSTech, Shenzhen, China; ADFA@UNSW, Campbell, ACT, Australia; CSIRO, NSW, Australia; ANU, Canberra, Australia; and USTC, Hefei, China, previously. He is the Vice President (Research and Innovation) and the Tong Tin Sun Chair Professor of machine learning at Lingnan University, Hong Kong, China.