

Hybrid Offline-Online Multi-Agent Decision Transformers for Wireless Resource Management

Yiming Zhang, Kun Yang, Cong Shen, Dongning Guo

Abstract—This paper develops a hybrid offline-online multi-agent reinforcement learning framework based on decision transformers. The policy is first pretrained offline via supervised sequence modeling of trajectories generated by existing policies, providing a safe and sample-efficient initialization. It is then fine-tuned online using a hybrid objective that incorporates critic-guided gradients, enabling performance improvements beyond the offline policy. To facilitate stable offline-to-online transfer and effective multi-agent coordination, the framework incorporates return-weighted sampling, a critic conditioned on neighbors’ actions, and neighborhood-correlated exploration. The approach is fully distributed: both training and execution rely only on local observations and limited information exchange among neighboring agents. Evaluations with dynamic traffic arrivals in two settings: (i) joint scheduling and power allocation and (ii) coordinated beamforming, show that the proposed method achieves quality-of-service (QoS) performance comparable to centralized methods. Moreover, when pretrained on lower-quality datasets, online fine-tuning is also observed to surpass the initial offline policy. These results demonstrate a promising learning-based alternative for wireless resource management.

Index Terms—Decision transformer, hybrid offline-online learning, multi-agent reinforcement learning, Poisson arrivals.

I. Introduction

Interference management is a key determinant of quality of service (QoS) in dense wireless networks. Model-based optimization methods, such as weighted minimum mean-squared error (WMMSE) [1] and fractional programming (FP) [2], achieve near-optimal performance, but their dependence on global channel state information (CSI) and centralized computation becomes prohibitive in practical deployments. In contrast, widely deployed heuristics such as greedy max-SINR selection [3] rely only on local measurements and simple decision rules, often at the cost of substantial performance degradation. This motivates the central question of this work: can a distributed solution operate efficiently using only local information while achieving QoS performance comparable to that of centralized optimization methods?

To address this challenge, we introduce a hybrid offline-online multi-agent reinforcement learning (MARL) framework and evaluate it on wireless resource allocation

problems. As a model-free framework, MARL naturally handles distributed systems where agents make sequential decisions under partial observability — a setting common to wireless networks, traffic control, robotics, and smart grids. In the context of our wireless resource allocation application, MARL’s architecture aligns perfectly with the problem structure: interference coupling creates distributed decision-making interactions among transmitters, network dynamics require continual adaptation, and devices typically have access only to local measurements. Consequently, online MARL has been studied for power control [4], joint resource allocation [5]–[8], and user scheduling [9]–[12]. However, online learning presents a critical deployment barrier: the random exploration required in the early learning stages can lead to poor QoS, while learning an effective policy from scratch can be highly sample-inefficient.

Offline reinforcement learning provides a natural complement to online learning [13]–[16]. Instead of relying on potentially risky and inefficient online exploration, offline RL trains on pre-collected datasets. For example, the offline method in our prior work [15] succeeds in both sum-rate maximization and delay minimization in the joint scheduling and power allocation task and exhibits these advantages. In particular, high-performing but impractical centralized algorithms can generate synthetic trajectories from which their behavior is distilled into deployable distributed policies. The resulting agents can therefore benefit from mature optimization techniques without requiring centralized computation during deployment. Offline pretraining also offers improved safety, stability, and convergence over learning online from scratch.

A purely offline approach, however, has important limitations. A policy may degrade under distributional shift when deployment conditions differ from those represented in the training data. More fundamentally, its performance is constrained by dataset quality: policies trained on low-reward trajectories generally struggle to achieve expert-level performance [17]. We therefore propose a hybrid offline-online framework that combines safe and efficient offline pretraining with online fine-tuning. The pretrained policy provides a strong initialization, while subsequent interaction enables adaptation to deployment conditions and improvement beyond the data-generating policy, even when only low-quality offline data are available.

A key design choice in this hybrid framework is the policy architecture. Our prior work [12] showed that sequential information and decision is crucial for dynamic

This work was supported in part by the National Science Foundation under Grant Nos. 2132700, 2216970, and 2434044.

Y. Zhang and D. Guo are with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: yimingzhang2026@u.northwestern.edu; dguo@northwestern.edu).

K. Yang and C. Shen are with the Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904 USA (e-mail: ky9tc@virginia.edu; cong@virginia.edu).

wireless resource allocation, as CSI and queues evolve over time. Transformers [18] have emerged as a leading architecture for sequence modeling, and the decision transformer [19] brings this capability to RL by casting policy learning as return-conditioned sequence modeling. This formulation supports supervised pretraining on offline data and is therefore well suited to our hybrid framework. Our prior conference work [15] introduced the first multi-agent decision transformer (MADT) implementation for wireless resource management but considered only offline pretraining. Concurrent work [20] employs hybrid offline-online training of a single-agent decision transformer and targets sum-rate maximization in an unmanned aerial vehicle scenario, whereas we develop a fully distributed multi-agent framework for QoS optimization.

The main contributions of this paper are as follows:

- We develop a hybrid offline-online MARL framework based on MADTs. Offline pretraining uses trajectories from existing policies to provide a safe and sample-efficient initialization. Online fine-tuning then combines supervised loss with critic-guided gradients, enabling improvement beyond the original offline policy.
- We address three challenges of fine-tuning under partial observability: nonstationarity from co-evolving neighbors, credit assignment with unobserved concurrent actions, and coordinated exploration. Our solutions consist of return-weighted anchored replay with critic warm-up, a critic conditioned on neighbors' actions, and neighborhood-correlated exploration, all using only local observations and neighborhood information exchange.
- The framework is fully distributed and agnostic to the controlled resource: new resource-management problems require redefining only the per-agent observation, action, and reward. We demonstrate this generality on two instantiations with distinct actions and network deployment: joint scheduling and power allocation among interfering links and coordinated beamforming in a sectorized multi-cell cluster. Both adopt packet delay as the QoS metric because it better reflects user experience than sum-rate in realistic traffic demand lighter than the network capacity. In both settings, simulation results show that the learned policies achieve QoS performance comparable to genie-aided centralized methods, and that online fine-tuning is able to improve upon the very policies that generated the offline data.

The remainder of this paper is organized as follows. We describe the general online-offline framework in Sec. II. Sec. III introduces the wireless network scenarios, QoS, and MARL design. Sec. IV presents the simulation setup and numerical results. Concluding remarks are given in Sec. V.

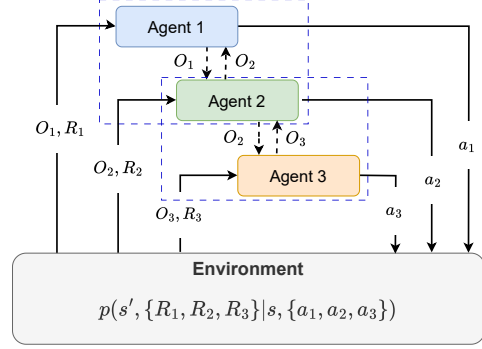


Fig. 1. An example of Dec-POMDP-IR.

II. MARL Framework

A. MARL Framework

We consider a multi-agent system in which K agents, indexed by $k \in \mathcal{K} = \{1, \dots, K\}$, interacting with the environment over episodes of length- $\mathcal{T}$ time slots. The following is defined for each agent $k \in \mathcal{K}$ and time slot $t \in \{1, \dots, \mathcal{T}\}$:

- The environment is described by a global state $\mathbf{s}^{(t)} \in \mathcal{S}$, where $\mathcal{S}$ denotes the state space.
- Agent k observes only a partial view $O_k^{(t)} \in \Omega_k$ of the global state, where Ω_k is agent k 's observation space.
- Agent k forms a belief, or history, $h_k^{(t)}$ based on its local observation $O_k^{(t)}$, the observations shared by the agent's neighbors, and the history of these observations. The agent then takes an action $a_k^{(t)} \in \mathcal{A}_k$ based on the belief, where $\mathcal{A}_k$ denotes agent k 's action space.
- The agents' actions collectively form the joint action $\mathbf{a}^{(t)} = (a_1^{(t)}, \dots, a_K^{(t)}) \in \mathcal{A}$, where $\mathcal{A} = \mathcal{A}_1 \times \dots \times \mathcal{A}_K$.
- The environment transitions to the next state according to the Markovian transition kernel $\mathcal{P}(\mathbf{s}^{(t+1)} | \mathbf{s}^{(t)}, \mathbf{a}^{(t)})$.
- Agent k receives an individual reward denoted as $r_k^{(t)} = R_k(\mathbf{s}^{(t)}, \mathbf{a}^{(t)}, \mathbf{s}^{(t+1)})$.

We model the system as a decentralized partially observable Markov decision process with individual rewards (Dec-POMDP-IR) [8], which replaces the single team reward in a Dec-POMDP [21] by the K individual reward functions $R_1, \dots, R_K$ introduced above. In addition, each agent's observation is a deterministic function of the global state rather than a stochastic one.

We define a neighbor relation between agents: two agents are said to be neighbors if they share their local observations with each other. Let l_k denote the number of neighbors of agent k , and $\nu_{k,1}, \dots, \nu_{k,l_k} \in \mathcal{K}$ denote their indices. Agent k 's neighborhood $\mathcal{D}(k) = (k, \nu_{k,1}, \dots, \nu_{k,l_k})$ comprises the agent itself and its neighbors. Fig. 1 illustrates this framework with three agents: agents 1 and 2 are neighbors, as are agents 2 and 3, while agents 1 and 3 are not neighbors and do not directly share information.

Agent k 's policy, π_k , represents a conditional probability distribution from which it samples its action a_k . Agent k 's

learning goal is to compute a good policy to maximize its own expected return accumulated over an episode of $\mathcal{T}$ slots:

$$\mathbb{E}_\pi \left[\sum_{t=1}^{\mathcal{T}} \gamma^{t-1} R_k \left(\mathbf{s}^{(t)}, \mathbf{a}^{(t)}, \mathbf{s}^{(t+1)} \right) \right], \quad (1)$$

where π denotes the joint policy and $\gamma \in [0, 1]$ is the discount factor.

B. Offline Pretraining with Decision Transformers

We adopt a decision transformer as the policy network in the multi-agent setting. Let the superscript $(s : t)$ denote a sequence spanning slots from s to t . For agent k , the local aggregate information $X_k^{(t)} = (O_k^{(t)}, O_{\nu_{k,1}}^{(t)}, \dots, O_{\nu_{k,l_k}}^{(t)})$ collects its own and its neighbors' observations. The return-to-go $\hat{R}_k^{(t)} = \sum_{l=t}^{\mathcal{T}} r_k^{(l)}$ is the tail of the finite-horizon objective (1) with $\gamma = 1$. The belief $h_k^{(t)} = (X_k^{(t-L+1:t)}, a_k^{(t-L+1:t-1)}, \hat{R}_k^{(t-L+1:t)})$ collects the length- L history of aggregate information, actions, and returns-to-go. The transformer maps the belief to an action distribution, $a_k^{(t)} \sim \pi_k(\cdot | h_k^{(t)})$, forming an auto-regressive model of order L . In addition to embeddings of the return-to-go, aggregate information, and action, the decision transformer uses a learned embedding of the within-episode slot index t as positional encoding.

Offline pretraining offers two key advantages: i) safe initial deployment by imitating proven policies, and ii) improved training efficiency through curated data rather than online exploration. In [15], [22], decision transformers trained offline converge to the performance level of the behavior policies that generated the data.

Offline pretraining consists of two phases. First, during data collection, a baseline method is executed over multiple simulated episodes. At each slot, we record the local observations, actions, and rewards available to each agent and compute the corresponding returns-to-go to construct the offline dataset. Second, all agents' trajectories are used to train a shared policy network π_{DT} via supervised learning. Let a length- L trajectory window beginning at slot t be $\tau^{(t)} = (\hat{R}_k^{(t:t+L-1)}, X_k^{(t:t+L-1)}, a_k^{(t:t+L-1)})_{k \in \mathcal{K}}$. For each training step, a mini-batch $(\tau^{(l)})_{l \in \mathcal{B}}$ of such windows is sampled, and the policy is updated by minimizing the cross-entropy loss over all agents and slots:

$$\mathcal{L}_{\text{CE}} = - \frac{1}{|\mathcal{B}|KL} \sum_{k=1}^K \sum_{l \in \mathcal{B}} \sum_{t=l}^{l+L-1} \log \pi_{\text{DT}} \left(a_k^{(t)} \middle| X_k^{(l:t)}, \hat{R}_k^{(l:t)}, a_k^{(l:t-1)} \right). \quad (2)$$

A causal attention mask ensures that the prediction at slot t conditions only on tokens up to t . At execution, the target return is initialized to the mean of the top-quartile episode returns of the offline dataset and decremented by the realized reward at each slot, prompting the transformer to reproduce the dataset's best behavior. The trained policy is executed independently by each agent.

C. Online Fine-Tuning

Purely offline learning has two key limitations. First, high-quality policies may be unavailable or impractical for real-world data collection. For instance, FP and WMMSE require centralized coordination and real-time global CSI, so they can be emulated to synthesize data but cannot be deployed to collect real-world data. Second, deployment conditions may differ substantially from the training distribution, leading to out-of-distribution actions and compounding errors that hinder adaptation.

To mitigate distribution shift, the online decision transformer [23] fine-tunes an offline-pretrained model through online interaction while retaining the auto-regressive supervised-learning paradigm. However, supervised learning alone cannot overcome suboptimal low-reward data because it provides no mechanism for improving beyond the collected trajectories.

We therefore augment the supervised objective with a policy-improvement term, allowing the policy to surpass rather than merely imitate the behavior in its training data. During online fine-tuning, the actor minimizes

$$\mathcal{L}_{\text{online}} = \mathcal{L}_{\text{CE}} + \lambda_{\text{RL}} \mathcal{L}_{\text{RL}} - \lambda_{\text{ent}} \bar{\mathcal{H}}, \quad (3)$$

where $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss in (2), evaluated on the fine-tuning batch, which regularizes the policy toward proven behavior, $\mathcal{H}(\cdot)$ denotes the policy entropy, whose mini-batch average $\bar{\mathcal{H}} = \frac{1}{|\mathcal{B}|K} \sum_{k=1}^K \sum_{l \in \mathcal{B}} \mathcal{H}(\pi_{\text{DT}}(\cdot | h_k^{(l)}))$ encourages continued exploration, while the reinforcement-learning term $\mathcal{L}_{\text{RL}}$, defined shortly, drives improvement beyond the dataset. The hyperparameters λ_{RL} and λ_{ent} balance the three objectives.

The RL term requires estimating how much each action improves upon the current policy. A Monte-Carlo policy gradient could use the empirical return-to-go $\hat{R}_k^{(t)}$ for this purpose, but such estimates have high variance and do not bootstrap. Following the RL-as-fine-tuning approach of [24], we instead learn twin critics, Q_{ϕ_1} and Q_{ϕ_2} , shared by all agents, which maps an agent's aggregate information $X_k^{(t)}$ to one value per discrete action. This constitutes a discrete-action counterpart of TD3 [25]. Unlike the sequence-model actor, each critic is a lightweight, memoryless network that consumes only the current aggregate information, a choice observed in [24] to be substantially more stable than a transformer critic. As in TD3, the second critic is only used to reduce overestimation in the bootstrap target introduced below, whereas the actor is updated using the first critic. The resulting advantage of action a is

$$A_{\phi_1} \left(X_k^{(t)}, a \right) = Q_{\phi_1} \left(X_k^{(t)}, a \right) - \sum_{a' \in \mathcal{A}_k} \pi_{\text{DT}} \left(a' \middle| h_k^{(t)} \right) Q_{\phi_1} \left(X_k^{(t)}, a' \right). \quad (4)$$

The RL loss increases the likelihood of above-average actions:

$$\mathcal{L}_{\text{RL}} = -\frac{1}{|\mathcal{B}|K} \sum_{k=1}^K \sum_{l \in \mathcal{B}} \sum_{a \in \mathcal{A}_k} \pi_{\text{DT}}(a | h_k^{(l)}) \frac{A_{\phi_1}(X_k^{(l)}, a)}{\sigma_A}$$

where σ_A is the standard deviation of the advantage, making the gradient scale insensitive to reward magnitude. Unlike hard filtering of positively-advantaged samples, this expectation-based objective yields a smoother update that reweights the entire action distribution.

The critics are trained by temporal-difference learning, i.e., by regressing $Q_{\phi_i}(X_k^{(t)}, a_k^{(t)})$ onto a bootstrap target formed from the reward and the estimated value of the next slot. Because a target computed directly from the networks being trained would change after every gradient step, regression onto it can be unstable [25]. We therefore compute the target from quasi-static copies of the learned networks: target critics $\{Q_{\bar{\phi}_1}, Q_{\bar{\phi}_2}\}$ and a target actor $\bar{\pi}_{\text{DT}}$, which are held fixed within each update and slowly track their online counterparts via Polyak averaging, $\bar{\phi}_i \leftarrow \kappa \phi_i + (1 - \kappa) \bar{\phi}_i$ with $\kappa \ll 1$ (likewise for $\bar{\pi}_{\text{DT}}$). The target evaluates the next slot by averaging, over the target actor's next-step action distribution, the element-wise minimum of the two target critics' predictions, which counteracts the overestimation bias of learned value functions [25]:

$$y_k^{(t)} = r_k^{(t)} + \gamma_c \sum_{a' \in \mathcal{A}_k} \bar{\pi}_{\text{DT}}(a' | h_k^{(t+1)}) \min_{i \in \{1, 2\}} Q_{\bar{\phi}_i}(X_k^{(t+1)}, a')$$

where $\gamma_c \in (0, 1)$ is the bootstrapping discount of the critic. Both critics regress onto this target:

$$\mathcal{L}_{\text{critic}} = \frac{1}{|\mathcal{B}|K} \sum_{k=1}^K \sum_{l \in \mathcal{B}} \sum_{i \in \{1, 2\}} \left(Q_{\phi_i}(X_k^{(l)}, a_k^{(l)}) - y_k^{(l)} \right)^2. \quad (5)$$

Online fine-tuning proceeds iteratively. In each iteration, agents collect trajectories and append them to a replay buffer that retains the best offline trajectories. Mini-batches drawn from this buffer are used for several critic gradient steps minimizing (5), followed by actor updates minimizing (3), with target networks updated through Polyak averaging after each step. More frequent critic updates help maintain accurate advantage estimates [25].

D. Stability and Coordination in Online Fine-Tuning

Online fine-tuning in a partially observable multi-agent environment introduces challenges absent from the single-agent setting. Each agent's reward depends on its neighbors' concurrent actions. Meanwhile, co-evolving neighbor policies create non-stationary learning targets, and improvements often require coordinated behavior across a neighborhood, which independent exploration is unlikely to discover. The offline-to-online transition is also delicate because an untrained critic can destabilize a capable pretrained policy. We employ several mechanisms to address these challenges.

Stabilizing the offline-to-online transition. We first warm up the critics with the actor frozen, and ramp λ_{RL} from zero over the first few iterations, so the RL gradient is trusted only once the critic is informative. The replay buffer keeps a fixed anchor set of the highest-return offline trajectories; each mini-batch mixes anchor and online data in fixed proportion, sampling trajectories with probability increasing in their standardized returns, so the supervised term imitates the best behavior seen so far rather than the latest exploratory episodes. Finally, the return target $\hat{R}_{\text{tar}}$ used for data collection is the running maximum of observed episode returns, preventing poor exploratory episodes from lowering the target.

Conditioning the critic on neighbors' actions. Agent k 's reward depends on its neighborhood's joint action, yet $X_k^{(t)}$ carries no information about the neighbors' concurrent actions. We condition the critic on the neighbors' intended actions at training time, $Q_{\phi}(X_k^{(t)}, a_k^{(t)}; \{a_j^{(t)}\}_{j \in \mathcal{D}(k) \setminus \{k\}})$. This allows the critic to evaluate an action in the context of the neighbors' behavior.

Temporally-extended, neighborhood-correlated exploration. Coordinated actions may require neighboring agents select compatible actions over several consecutive slots. Independent ϵ -greedy exploration lacks coordination while isolated deviation yields noisy learning signals for the critic because rewards depend on neighborhood joint actions. We therefore correlate exploration across both agents and time. In each slot, an agent k is selected as the exploration center with probability ϵ and sends a one-bit trigger to its neighbors. Each agent in $\mathcal{D}(k)$ then independently samples an exploratory action and holds it for a fixed number of slots, while agents outside the neighborhood continue to follow the policy. An agent still committed to an earlier exploratory action retains that action. Each trigger thus perturbs the joint action over one complete neighborhood — precisely the configuration on which the center agent's reward and critic depend — without requiring common randomness or centralized coordination. This mechanism extends ϵ_z -greedy exploration [26] to the multi-agent setting.

The latter two mechanisms trade implementation complexity for coordination capability. They are particularly useful in large or tightly coupled systems, where rewards strongly depend on neighbors' concurrent actions. In smaller or weakly coupled systems, a per-agent critic and standard action sampling may suffice.

III. Wireless Model and Problem Formulation

We instantiate the MARL framework in two wireless resource-allocation scenarios. Fig. 2 summarizes the three-phase training pipeline of Sec. II: (i) an offline dataset is either synthesized by simulating some existing policies or collected in a deployed system; (ii) offline pretraining employs supervised learning on this dataset to establish a safe initial decision-transformer policy; and (iii) online fine-tuning refines the policy through environmental interaction using the hybrid loss, enabling improvement

beyond the quality of the original dataset while preserving distributed execution.

A. Joint Scheduling and Power Allocation

1) System Model: We consider downlink in a network of K transmitter-receiver pairs, representing a single channel or subchannel in a mobile ad hoc network or a cellular network in which each access point (AP) serves one active device. An example is shown in Fig. 3. Each link is treated as an agent. Neighboring agents communicate through low-latency interfaces, such as the Xn interface between gNBs [27]. The associated fiber backhaul latency is typically 1 ms or less [28], well below the slot duration $T = 20$ ms considered here.

Each transmitter and receiver has a single antenna. Under flat fading, the channel power gain from transmitter i to receiver j in slot t is

$$g_{i \rightarrow j}^{(t)} = \alpha_{i \rightarrow j} |\beta_{i \rightarrow j}^{(t)}|^2, \quad t = 1, 2, \dots \quad (6)$$

where $\alpha_{i \rightarrow j} \geq 0$ is the large-scale path loss component, which remains constant over many slots, and $\beta_{i \rightarrow j}^{(t)}$ is the small-scale Rayleigh fading coefficient. To capture temporal channel correlation, we model the latter as a first-order complex Gauss-Markov process:

$$\beta_{i \rightarrow j}^{(t)} = \rho \beta_{i \rightarrow j}^{(t-1)} + \sqrt{1 - \rho^2} e_{i \rightarrow j}^{(t)} \quad (7)$$

where $\rho \in [0, 1]$ is the temporal correlation coefficient, and $(\beta_{i \rightarrow j}^{(0)}, e_{i \rightarrow j}^{(1)}, e_{i \rightarrow j}^{(2)}, \dots)$ are independent and identically distributed circularly symmetric complex Gaussian random variables with unit variance.

Agent neighborhoods are defined by interference coupling. Specifically, agent j belongs to agent i 's neighborhood if the relative path loss $\alpha_{j \rightarrow i} / \alpha_{i \rightarrow i}$ exceeds a predefined threshold, indicating nonnegligible interference from transmitter j to receiver i . In a hexagonal deployment, we assume each agent typically has at most six neighbors.

Let $p_k^{(t)}$ denote the transmit power of transmitter k in slot t , and let $\mathbf{p}^{(t)} = (p_1^{(t)}, \dots, p_K^{(t)})$ denote the global power-allocation vector. Assuming additive white Gaussian noise with variance σ^2 at each receiver, the spectral efficiency of link k is

$$\mathcal{C}_k^{(t)}(\mathbf{p}^{(t)}) = \log_2 \left(1 + \frac{g_{k \rightarrow k}^{(t)} p_k^{(t)}}{\sum_{j \in \mathcal{K}, j \neq k} g_{j \rightarrow k}^{(t)} p_j^{(t)} + \sigma^2} \right) \quad (8)$$

We consider a traffic-driven system in which each link maintains a first-in, first-out (FIFO) queue. Let $\zeta_k^{(t)}$ denote the number of packets arriving at link k at the beginning of slot t , and let M denote the packet size in bits. Given bandwidth W , the queue length $q_k^{(t)}$, measured in bits at the end of the slot, evolves as

$$q_k^{(t)} = \max \left(0, q_k^{(t-1)} + \zeta_k^{(t)} M - \mathcal{C}_k^{(t)} W T \right). \quad (9)$$

2) Agent Design: At the beginning of slot t , agent k 's local observation $O_k^{(t)}$ comprises:

- $g_{k \rightarrow k}^{(t)}$: the current direct channel gain,
- $p_k^{(t-1)}$: agent k 's previous action,
- $q_k^{(t-1)} + \zeta_k^{(t)} M$: the queue backlog after new arrivals,
- $\sum_{j \in \mathcal{K}, j \neq k} g_{j \rightarrow k}^{(t-1)} p_j^{(t-1)} + \sigma^2$: the total interference-plus-noise power,
- $\mathcal{C}_k^{(t-1)}$: spectral efficiency of link k computed from (8).

The last two quantities result from the previous slot's transmissions and are therefore available with a one-slot delay. Through neighborhood information exchange, agent k obtains the aggregate information $X_k^{(t)} = \{O_k^{(t)}, O_{\nu_{k,1}}^{(t)}, \dots, O_{\nu_{k,l_k}}^{(t)}\}$ before decision making in slot t .

Based on $X_k^{(t)}$, agent k decides whether to transmit and, if so, at what power. It selects $p_k^{(t)}$ from a discrete action set comprising zero, representing no transmission, and power levels equally spaced in dB between the minimum nonzero power $\underline{P}$ and the maximum power $\bar{P}$:

$$p_k^{(t)} \in \mathcal{A}_k = \left\{ 0, \underline{P}, \underline{P}(\bar{P}/\underline{P})^{\frac{1}{|\mathcal{A}_k|-2}}, \dots, \bar{P} \right\}. \quad (10)$$

We use queue length as a surrogate QoS objective because, by Little's law, shorter time-averaged queues corresponds to shorter delays. The utility of agent k is therefore $u_k^{(t)} = -q_k^{(t)}$. To promote coordination, agent k 's reward also includes the utilities of its neighbors. This discourages aggressive scheduling and transmission that reduces one's own backlog at the cost of greater interference and larger neighboring queues. Specifically,

$$r_k^{(t)} = \sum_{i \in \mathcal{D}(k)} u_i^{(t)} = - \sum_{i \in \mathcal{D}(k)} q_i^{(t)}. \quad (11)$$

B. Beamforming

1) Network Model: Our second instantiation considers downlink beamforming in a sectorized multi-cell network. Each site is partitioned into three 120° sectors, with inter-cell interference dominated by sectors directly facing one another. We therefore consider a cluster of K APs arranged on a ring with inward-facing sectors. Each AP serves one active device over a shared frequency band and is regarded as one agent. Its two adjacent APs are regarded as its neighbors, with which it exchanges observations as described in Sec. II. In each slot, an AP either makes no transmission or selects one beam from a codebook and transmits at a fixed power. Each device moves randomly across slots, producing temporally correlated channels. Fig. 4 depicts an example.

Because the sectors face one another, their beam selections are coupled: a beam aimed at one's own device also leaks power into other cells. Each AP must therefore balance its own rate against the interference it inflicts on others. As in the joint scheduling and power allocation setting, we consider traffic-driven delay-minimization. Poisson packet arrivals accumulate in a FIFO queue at each AP, which is served at the realized rate according to the queueing dynamics in (9).

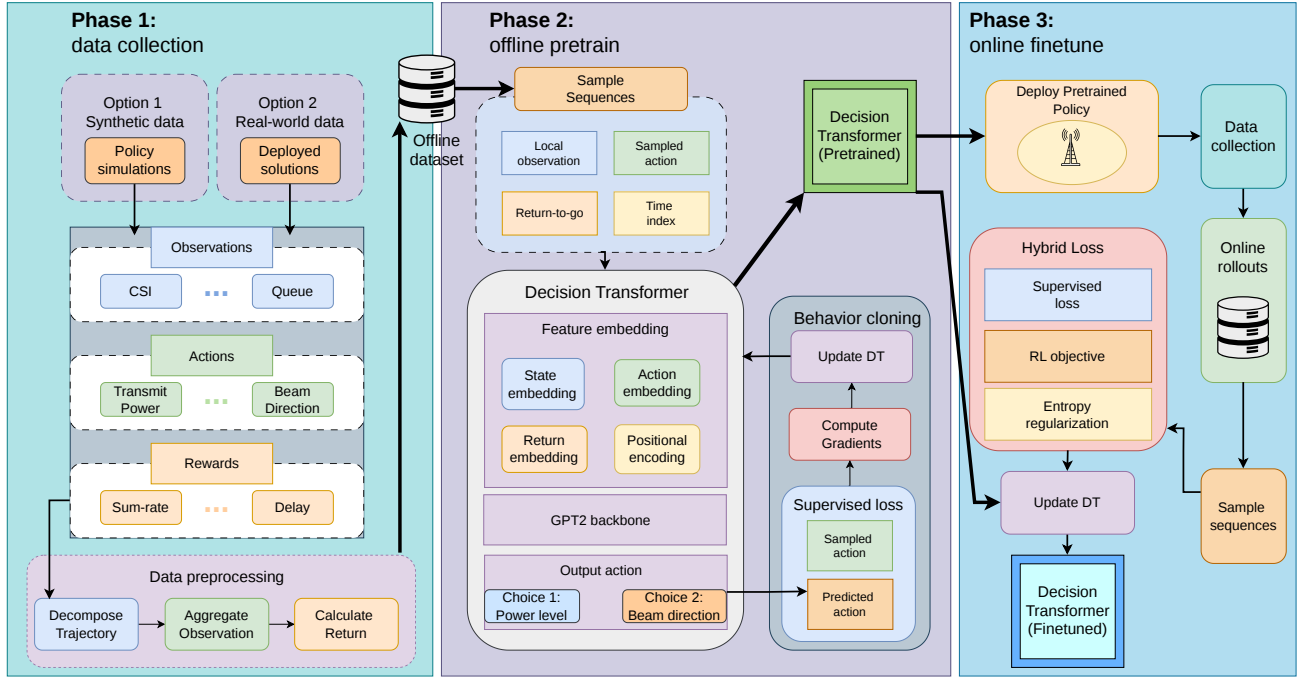


Fig. 2. Three-phase hybrid offline-online MADT framework for wireless resource allocation.

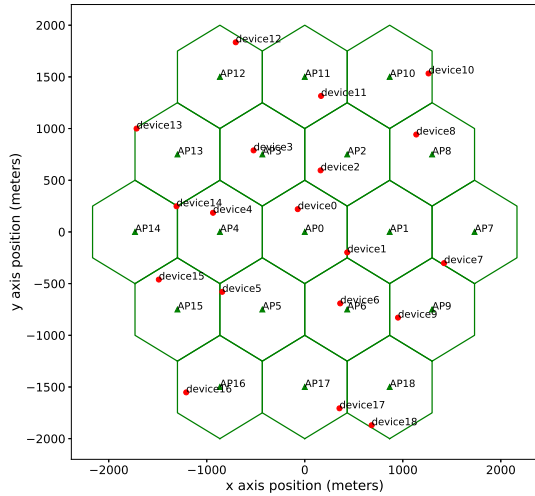


Fig. 3. Nineteen hexagonal cells, each serving one active device.

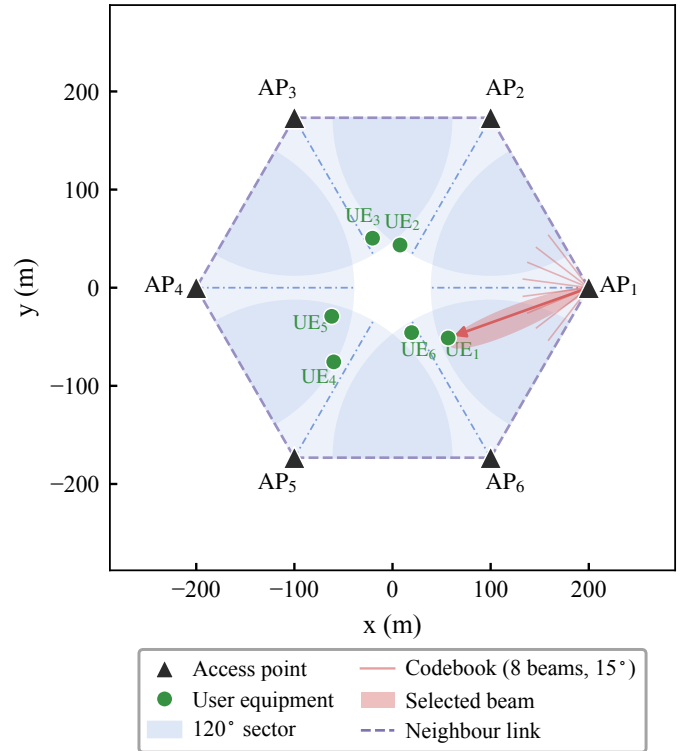


Fig. 4. Beamforming deployment: six APs with inward-facing 120° sectors form one cluster of mutually-interfering agents.

2) Agent Design: Agent k stays silent or selects one of B beams spanning its sector, i.e., $a_k^{(t)} \in \mathcal{A}_k = \{0, 1, \dots, B\}$ where 0 corresponds to silence. Let $G(a, \theta)$ denote the gain of beam a toward angle θ , with $G(0, \theta) = 0$. Let $\theta_{j \rightarrow k}^{(t)}$ denote the angle from AP j to user k , $d_{j \rightarrow k}^{(t)}$ their distance, $\ell(\cdot)$ the path loss as a function of distance, and

P the transmit power. The spectral efficiency of user k is

$$\mathcal{C}_k^{(t)} = \log_2 \left(1 + \frac{P G(a_k^{(t)}, \theta_{k \rightarrow k}^{(t)}) / \ell(d_{k \rightarrow k}^{(t)})}{I_k^{(t)} + \sigma^2} \right) \quad (12)$$

where $I_{j \rightarrow k}^{(t)} = P G(a_j^{(t)}, \theta_{j \rightarrow k}^{(t)}) / \ell(d_{j \rightarrow k}^{(t)})$ denotes the interference that AP j 's transmission places on user k , and $I_k^{(t)} = \sum_{j \neq k} I_{j \rightarrow k}^{(t)}$ denotes the aggregate interference.

The beam gains toward the served device are measured at the beginning of each slot. Quantities resulting from transmission, including the realized spectral efficiency and received interference, become available after the slot and therefore carry a one-slot delay. Agent k 's local observation $O_k^{(t)}$ comprises the direct beam gains $G(b, \theta_{k \rightarrow k}^{(t)})$, $b = 1, \dots, B$, distance $d_{k \rightarrow k}^{(t)}$, queue backlog $q_k^{(t-1)} + \zeta_k^{(t)} M$, the previous action $a_k^{(t-1)}$, the previous realized spectral efficiency $\mathcal{C}_k^{(t-1)}$, the previous aggregate interference $I_k^{(t-1)}$, and the previous per-source interference $I_{\nu \rightarrow k}^{(t-1)}$ of each neighbor, estimated at user k 's receiver and fed back to AP k .

In this beamforming setting, agent ν shares only

$$m_\nu^{(t)} = (\mathcal{C}_\nu^{(t-1)}, a_\nu^{(t-1)}, q_\nu^{(t-1)} + \zeta_\nu^{(t)} M) \subset O_\nu^{(t)} \quad (13)$$

with its neighbors. Assuming agent k has two neighbors, i.e., $\mathcal{D}(k) = \{k, \nu_{k,1}, \nu_{k,2}\}$, its aggregate information is $X_k^{(t)} = (O_k^{(t)}, m_{\nu_{k,1}}^{(t)}, m_{\nu_{k,2}}^{(t)})$. The utility and the reward follow Sec. III-A, i.e., $u_k^{(t)} = -q_k^{(t)}$, and $r_k^{(t)} = -\sum_{i \in \mathcal{D}(k)} q_i^{(t)}$.

IV. Performance Evaluation

We evaluate the hybrid-RL framework on both settings introduced in Sec. III. Throughout this section, the joint scheduling and power allocation task of Sec. III-A is referred to simply as power allocation for brevity.

A. Power Allocation

1) Simulation setup: We consider a network of 19 devices in 19 hexagonal cells, as illustrated in Fig. 3. The channel, traffic, and action-space models follow Sec. III, with the parameters listed in Table I; an agent's neighborhood is determined by the interference-coupling rule defined therein. Also, each agent digests its neighbors' shared observations by local index (its first neighbor, second neighbor, and so on), so the shared policy is agnostic to any global indexing of agents or links.

We compare the learned policies against the following baselines:

- Full Power: all APs transmit at maximum power;
- Random Power: Each AP selects its power uniformly;
- ITLinQ [29]: a distributed scheduling method inspired by information theory, realizable with minor adaptation;
- ideal WMMSE [1]: centralized WMMSE with instantaneous global CSI, included as a genie-aided benchmark;

TABLE I
Simulation Parameters of the Two Instantiations

Parameter	Power allocation	Beamforming
Cell radius	500 m	160 m
Path loss (d in km)	$128.1 + 37.6 \log_{10} d$ dB	exponent = 2
Transmit power	$(P_{\min}, \bar{P}) = (0, 23)$ dBm	fixed P
Noise power	-114 dBm	10 dB cell-edge SNR
Action space $\mathcal{A}_k$	7 power levels	8 beams + silent
Channel variation	$f_d = 10$ Hz, $\rho = 0.64$	user random walk
Arrival rate λ	25 packets/s	10 packets/s
Common: $W = 10$ MHz, $T = 20$ ms, $M = 500$ kbits, $\mathcal{T} = 1000$ slots.		

- delayed WMMSE: centralized WMMSE with one-slot-delayed global CSI. This is realizable but unscalable.

Offline datasets are generated by two behavior policies of different quality: an expert dataset produced by ideal WMMSE, and a medium dataset produced by delayed WMMSE. Each dataset contains 40 episodes (40,000 transitions) collected under independent channel and traffic realizations.

Online fine-tuning then performs 200 iterations, each of which collects one exploratory episode under a fresh channel/traffic realization and applies the critic and actor updates of Sec. II-C, i.e., an online interaction budget of only 200 episodes. All reported evaluations are rollouts on a common held-out channel/traffic realization, whose seed is disjoint from those of the offline dataset and the online exploration episodes, so that no evaluation data is seen during training, and every stage and baseline is scored on the same episode for a controlled comparison.

2) Training dynamics and convergence: Figs. 5 and 6 show the training dynamics for the expert and medium datasets, respectively. Since the per-step reward is the negative queueing cost, the episode return and the mean end-to-end packet delay are almost perfectly affinely related in our experiments. Each panel therefore carries a second vertical axis on the right that reads the same curve as mean delay.

During offline pretraining (panels (a)), the greedy evaluation return approaches the average return of the offline dataset (dashed line) within roughly 20 epochs in both cases: supervised pretraining on only 40 episodes suffices to imitate the behavior policy, without any environment interaction.

During online fine-tuning (panels (b)), both runs exhibit a transient performance drop over the first ~ 25 iterations. Two factors contribute to this decline. (i) the ϵ -greedy, temporally-committed exploration deliberately injects sustained off-policy power choices for exploration, which depresses the return of the collected episodes (thin curve); and (ii) the newly initialized twin critics are still uninformative while the RL weight λ_{RL} ramps up from zero, so early updates cannot yet exploit the RL gradient. As the critics warm up and the return-weighted replay — anchored by the best offline trajectories — concentrates the supervised term on the best behavior observed so far, the evaluation return recovers. Starting from the expert

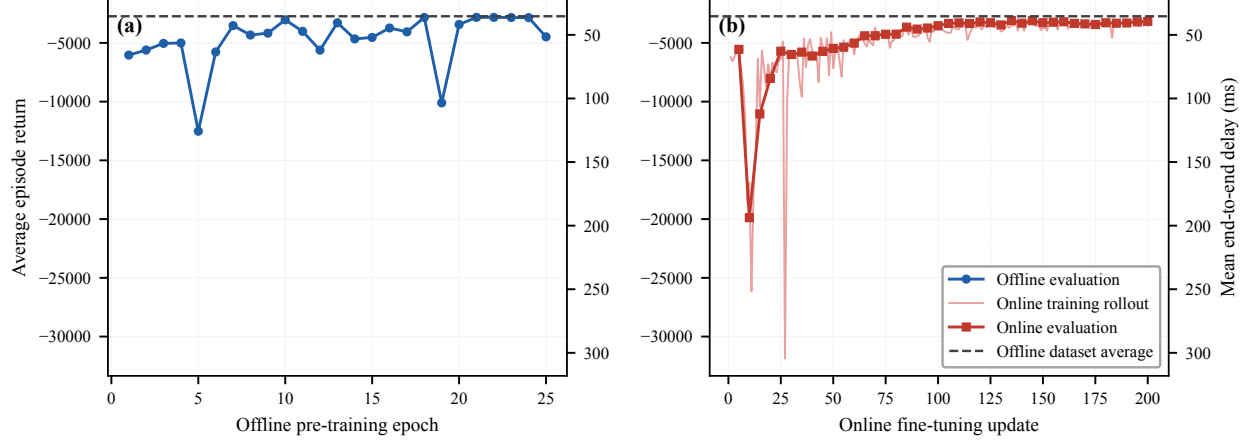


Fig. 5. Learning curves when pretraining on the expert (ideal WMMSE) dataset: (a) offline pretraining and (b) online fine-tuning. The dashed line marks the average episode return of the offline dataset. Each curve is read against both vertical axes: average episode return (left) and the corresponding mean end-to-end delay (right, inverted so that smaller delay is at the top).

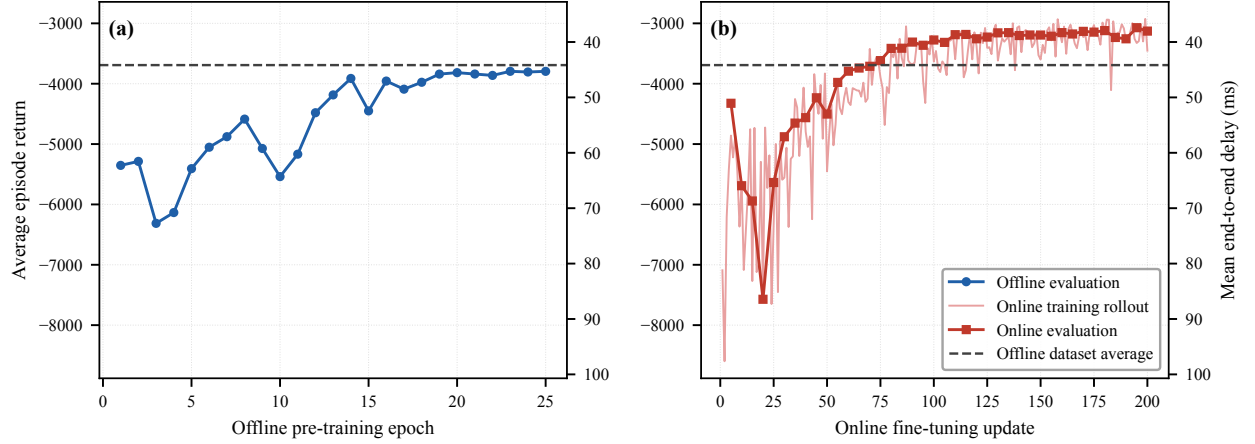


Fig. 6. Learning curves when pretraining on the medium (delayed WMMSE) dataset: (a) offline pretraining and (b) online fine-tuning. Axes as in Fig. 5. After the initial exploration phase, the fine-tuned policy surpasses the average return of the offline dataset

pretrained policy (Fig. 5(b)), there is little headroom above the behavior policy, and fine-tuning converges back to the dataset-average level. Starting from the medium pretrained policy (Fig. 6(b)), the evaluation return crosses the offline dataset average at around iteration 70 and settles clearly above it: the RL gradient enables the policy to improve beyond the quality of its own training data, which pure supervised (offline or online) decision-transformer training cannot do. As a reference point for sample efficiency, pure online MARL in a similar environment requires roughly 500 episodes of interaction for stable convergence to its optimal performance [8], several times the online interaction budget used here. We also note that the stability and coordination mechanisms of Sec. II-D are critical to this success: without them, online fine-tuning in this experiment simply fails to converge.

3) QoS performance: Fig. 7 presents the CCDF of the per-packet delays of all schemes on the common evaluation episode ($\sim 9,500$ delivered packets), and Table II reports

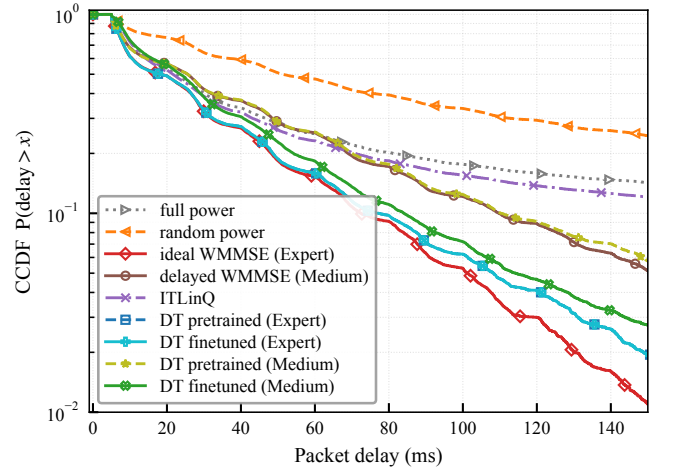


Fig. 7. CCDF of packet delays for the 19-device network (log scale), highlighting the delay tail.

TABLE II
Average Packet Delay (ms)

Method	Mean	90th pct.	99th pct.
full power	96.4	273.2	1029.2
random power	468.2	1984.5	5362.8
ITLinQ	104.1	210.2	1501.6
ideal WMMSE (Expert)	30.5	72.4	151.7
delayed WMMSE (Medium)	44.1	109.7	252.3
DT pretrained (Expert)	31.8	77.3	182.6
DT fine-tuned (Expert)	31.8	77.3	182.6
DT pretrained (Medium)	44.8	110.3	252.0
DT fine-tuned (Medium)	37.0	84.1	221.7

the corresponding statistics. Three observations follow.

First, offline pretraining faithfully reproduces its behavior policy. The DT pretrained on the expert dataset achieves a mean delay of 31.8 ms, similar to the genie-aided ideal WMMSE (30.5 ms), while operating fully distributedly — in contrast to the global CSI and centralized computation the expert requires. Likewise, the DT pretrained on the medium dataset matches delayed WMMSE almost exactly.

Second, online fine-tuning is most valuable precisely when the offline data is suboptimal. Fine-tuning the expert-pretrained policy leaves its performance unchanged (31.8 ms), consistent with Fig. 5(b): there is essentially no headroom, and the hybrid objective preserves the near-optimal policy. Fine-tuning the medium-pretrained policy, however, reduces the mean delay from 44.8 to 37.0 ms — a 17.5% improvement that also outperforms its own centralized teacher, delayed WMMSE. The gains are even more pronounced in the delay tail that dominates user experience: the 90th-percentile delay drops, clearly visible in the CCDF, where the fine-tuned (Medium) curve separates from the overlapping delayed-WMMSE/pretrained pair and moves toward the expert cluster.

Third, all learned policies decisively outperform the practical non-learning references — full power, ITLinQ, and random power — whose delays are markedly larger, with heavy tails exceeding one second at the 99th percentile (Table II).

4) Discussion: Beyond the raw delay numbers, two broader implications of this experiment are worth noting. First, the learned DT policy is fully distributed: each agent makes its decision using only its own local measurements and limited neighborhood information exchange, with no global CSI and no central coordinator. The per-agent computation and signaling therefore do not grow with the network size, making the solution scalable and feasible for real-world deployment — in sharp contrast to unscalable centralized WMMSE-type methods. This observation is consistent with the networked-MARL result that policies restricted to κ -hop neighborhood information can approach globally optimal performance when inter-agent influence decays with graph distance [30]. In our setting the inter-agent influence is interference, which decays with distance through the path loss, and policies acting on one-hop neighborhood information indeed attain QoS close to that of the centralized, global-information benchmark.

Second, the experiment shows that synthesized data from a deployment-infeasible algorithm is sufficient to train a deployable policy: ideal WMMSE is unrealizable in a real network, yet the data it generates in simulation distills into a distributed policy that attains nearly the same QoS performance using only local information.

B. Beamforming

1) Simulation setup: We instantiate the beamforming environment of Sec. III on the six-sector cluster of Fig. 4: six APs on a hexagonal ring with 200 m inter-site distance and inward-facing 120° sectors, each serving a single user, with a codebook of 8 beams of 15° width plus the silent action (9 discrete actions per agent). We deliberately confine every user to a narrow angular band at the cell edge for mobility. This stress test places each user in the region of strongest inter-cell interference, keeping the beam decisions of adjacent cells strongly coupled at all times.

Traffic is Poisson and we report per-packet delay; the remaining parameters are listed in Table I. We compare against three references: random beam selection; a greedy max-gain policy that points each AP directly at its own user (staying silent when its queue is empty) — a fully distributed heuristic that ignores the interference it inflicts on neighboring cells; and a queue-weighted coordinate-descent expert that performs Gauss-Seidel descent over the discrete joint beam space to maximize the queue-weighted sum-rate using global information — the analogue of ideal WMMSE over the discrete beam set, and likewise a genie-aided benchmark that is infeasible to deploy. Mirroring the power-allocation experiment, the expert offline dataset is generated by this coordinate-descent expert and the medium dataset by the max-gain policy, each containing 10 episodes (10,000 transitions). Since the problem is much smaller than the 19-link power-allocation task, we use a lighter decision transformer; all other components of the training pipeline of Sec. II are unchanged, and all reported evaluations again use the identical channel/traffic/mobility realizations across every stage and baseline.

2) Training dynamics and convergence: Fig. 8 shows the training dynamics for the medium dataset. The delay axis of Figs. 5 and 6 is omitted here for brevity, as delay remains affinely related to the episode return and the return curve therefore conveys the same information. In this experiment the per-slot utility is offset by a constant, $u_k^{(t)} = u_0 - q_k^{(t)}$, which better separates trajectories of different quality in the return-to-go signal without affecting the ranking of policies; it is also why the plotted returns can be positive. We also omit the expert-dataset curve: as in the power-allocation experiment (Fig. 5), pretraining imitates the near-optimal expert within a few epochs and fine-tuning leaves it unchanged.

The medium curve is the informative one. In the offline phase (Fig. 8(a)), the greedy evaluation return climbs from a large negative value to just below the dataset

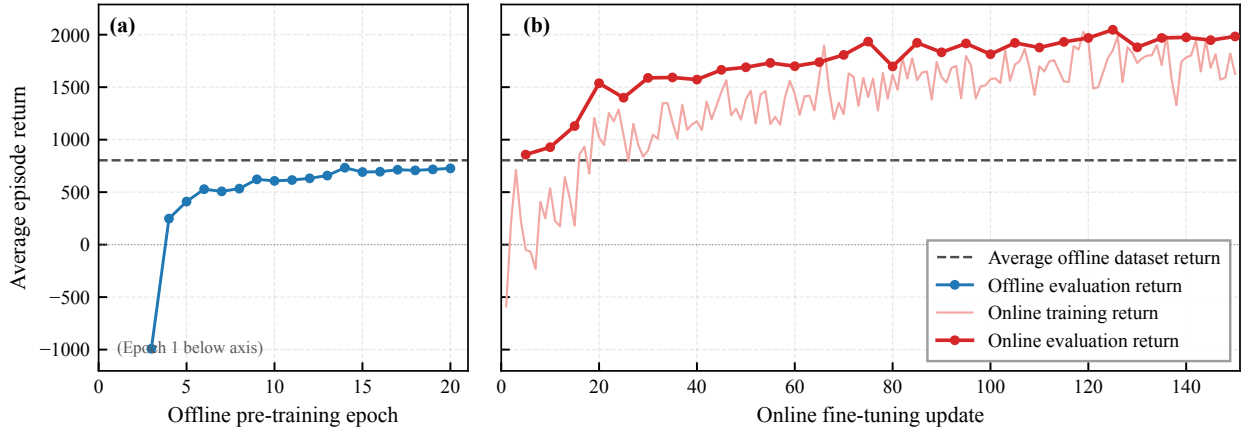


Fig. 8. Beamforming learning curves when pretraining on the medium (max-gain) dataset: (a) offline pretraining and (b) online fine-tuning. The dashed line marks the average episode return of the offline dataset.

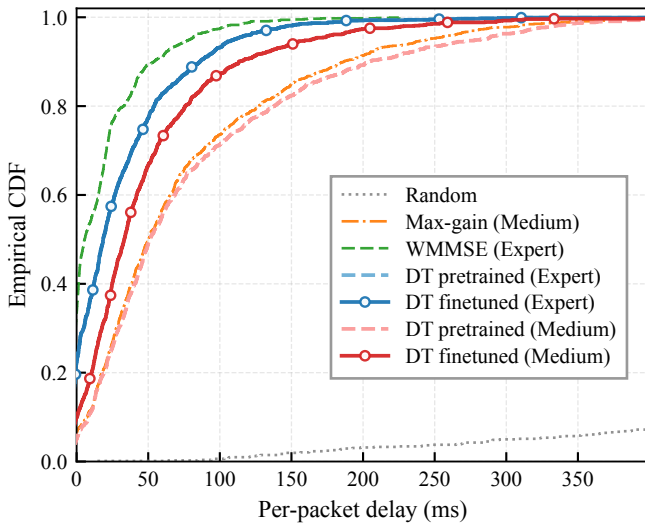


Fig. 9. CDF of per-packet delays for the beamforming network. The DT pre-trained (Expert) curve is hidden beneath the DT fine-tuned (Expert) curve, with which it exactly coincides.

average within roughly 15 epochs — imitation of the max-gain behavior from only 10 episodes. In the online phase (Fig. 8(b)), the exploratory episodes (thin curve) are initially depressed by the committed exploration, but the evaluation return crosses the dataset average within about 20 iterations and settles well above it. The improvement beyond the behavior policy is considerably larger than in the power-allocation experiment (Fig. 6(b)), for a simple reason: in this interference-limited deployment the greedy max-gain policy is much farther from the coordinated optimum than delayed WMMSE is from ideal WMMSE, leaving the RL gradient far more room to exploit.

3) QoS performance: Fig. 9 shows the per-packet delay CDF of all schemes on the common evaluation episode ($\sim 1,200$ delivered packets), and Table III reports the corresponding statistics. The overall picture matches the power-allocation experiment, with a few aspects worth

TABLE III
Beamforming Packet Delay (ms)

Method	Mean	90th pct.	99th pct.
random	3097.8	6241.9	7324.9
max-gain (Medium)	76.2	186.4	344.3
ideal WMMSE (Expert)	17.6	53.8	121.5
DT pre-trained (Expert)	31.6	86.1	184.4
DT fine-tuned (Expert)	31.6	86.1	184.4
DT pre-trained (Medium)	82.7	204.2	363.3
DT fine-tuned (Medium)	49.5	113.4	291.4

noting:

First, the offline imitation gap on the expert dataset is larger here than in power allocation. We attribute this to the leaner information structure of the beamforming agents: each AP exchanges observations with only its two adjacent neighbors, whereas the interference at a cell-edge user is shaped by up to five other sectors. With only partial visibility of the interferers, a distributed policy cannot fully reproduce the decisions of a centralized coordinate-descent expert that sees everything.

Second, online fine-tuning again delivers its largest gains on the medium dataset. The DT pre-trained on the medium dataset imitates its max-gain teacher (82.7 vs. 76.2 ms; their CDFs in Fig. 9 nearly coincide), and fine-tuning then reduces the mean delay by 40% to 49.5 ms, outperforming its teacher. We examined the policies' behavior to find the source of improvement: comparing their actions on the identical evaluation episode, in almost every slot the fine-tuned policy schedules fewer simultaneously transmitting APs than max-gain. Guided by its critic, the fine-tuned policy learns when yielding (going silent or steering away) benefits the neighborhood, and by conceding transmission opportunities to one another the agents jointly reduce interference and raise the realized spectral efficiency, so queued packets drain faster — exactly the coordination this deployment demands. The random policy, for completeness, is catastrophic: with six sectors interfering over the same region, uncoordinated beams render the network

unusable.

Finally, we emphasize the scope of these gains: the advantage of learned coordination is concentrated in the interference-limited regime deliberately constructed here. In additional experiments where users are spread over the full cell area so that inter-cell interference is light, the greedy max-gain policy already attains delay statistics close to the expert's, and there is no meaningful room — or need — for a learning-based method. Learned beam coordination is therefore best viewed as a tool for the high-interference operating points (e.g., dense deployments and cell-edge-heavy traffic) rather than as a blanket replacement for simple beam selection.

4) Discussion: Beyond the delay statistics, two broader implications of this experiment are worth noting. First, the beamforming instantiation demonstrates the transferability of the framework: relative to the power-allocation experiment, the action semantics (discrete beam selection vs. multi-level power control), the deployment and agent granularity, and the model and data budgets all differ, yet accommodating the new problem required redefining only the per-agent observation, action, and reward, with no algorithmic change to the training pipeline of Sec. II.

Second, the experiment characterizes the framework's behavior under restricted information. The offline imitation gap to the centralized expert, attributed above to the agents' partial visibility of their interferers, is an honest one that fine-tuning correctly does not overfit away, consistent with the framework's premise that achievable performance is bounded by the local information available to the agents.

V. Conclusion

This paper presented a hybrid offline-online MARL framework that distills the capabilities of model-based optimization methods into practical distributed policies for wireless resource management. In the joint scheduling and power allocation setting as well as the coordinated beamforming setting, the learned policies achieved QoS performance comparable to genie-aided centralized methods using only local measurements and limited neighborhood information exchange. When trained on suboptimal offline data, online fine-tuning further improved performance beyond the data-generating policies.

The framework combines the safety, stability, and sample efficiency of offline pretraining with the adaptability and policy-improvement capability of online RL. Its effectiveness across two settings with different actions, network deployments, agent structures, and training budgets demonstrates its generality. More broadly, the framework enables deployment-infeasible algorithms to provide synthetic training data whose knowledge is distilled into distributed policies. These results position hybrid offline-online MADT as a promising approach to intelligent wireless network management and other distributed multi-agent systems.

References

- [1] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted mmse approach to distributed sum-utility maximization for a mimo interfering broadcast channel," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4331–4340, 2011.
- [2] K. Shen and W. Yu, "Fractional programming for communication systems—part i: Power control and beamforming," *IEEE Transactions on Signal Processing*, vol. 66, no. 10, pp. 2616–2630, 2018.
- [3] 3GPP, "NR; Physical layer procedures for control," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.213, 2024.
- [4] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2239–2250, 2019.
- [5] J. Tan, Y.-C. Liang, L. Zhang, and G. Feng, "Deep reinforcement learning for joint channel selection and power control in D2D networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1363–1378, 2021.
- [6] Y. S. Nasir and D. Guo, "Deep actor-critic learning for distributed power control in wireless mobile networks," in *2020 54th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2020, pp. 398–402.
- [7] A. A. Khan and R. S. Adve, "Centralized and distributed deep reinforcement learning methods for downlink sum-rate optimization," *IEEE Transactions on Wireless Communications*, vol. 19, no. 12, pp. 8410–8426, 2020.
- [8] Y. Zhang and D. Guo, "Multi-agent reinforcement learning for multi-cell spectrum and power allocation," *IEEE Transactions on Communications*, vol. 73, no. 8, pp. 5980–5992, 2025.
- [9] K. Yang, D. Li, C. Shen, J. Yang, S.-p. Yeh, and J. Sydir, "Multi-agent reinforcement learning for wireless user scheduling: Performance, scalability, and generalization," in *The 56th Asilomar Conference on Signals, Systems, and Computers*, 2022, pp. 1169–1174.
- [10] M. K. Tefera, S. Zhang, and Z. Jin, "Deep reinforcement learning-assisted optimization for resource allocation in downlink OFDMA cooperative systems," *Entropy*, vol. 25, no. 3, p. 413, 2023.
- [11] J. Ge, Y.-C. Liang, L. Zhang, R. Long, and S. Sun, "Deep reinforcement learning for distributed dynamic coordinated beamforming in massive MIMO cellular networks," *IEEE Transactions on Wireless Communications*, pp. 4155–4169, 2023.
- [12] Y. Zhang and D. Guo, "Traffic-driven spectrum and power allocation via scalable multi-agent reinforcement learning," in *2024 60th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2024, pp. 1–8.
- [13] K. Yang, C. Shi, C. Shen, J. Yang, S.-p. Yeh, and J. J. Sydir, "Offline reinforcement learning for wireless network optimization with mixture datasets," *IEEE Transactions on Wireless Communications*, vol. 23, no. 10, pp. 12 703–12 716, Oct. 2024.
- [14] K. Yang, S.-P. Yeh, M. Zhang, J. Sydir, J. Yang, and C. Shen, "Advancing RAN slicing with offline reinforcement learning," in *2024 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2024, pp. 331–338.
- [15] Y. Zhang, K. Yang, C. Shen, and D. Guo, "Multi-agent decision transformer for power control in wireless networks," in *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025, pp. 1–5.
- [16] E. Eldeeb and H. Alves, "An offline multi-agent reinforcement learning framework for radio resource management," *IEEE Transactions on Mobile Computing*, vol. 25, no. 1, Jan. 2026.
- [17] M. Nakamoto, Y. Zhai, A. Singh, M. Sobol Mark, Y. Ma, C. Finn, A. Kumar, and S. Levine, "Cal-QL: Calibrated offline RL pre-training for efficient online fine-tuning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 62 244–62 269, 2023.
- [18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.
- [19] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," *Advances in*

neural information processing systems, vol. 34, pp. 15 084–15 097, 2021.

- [20] J. Zhang, J. Li, Z. Wang, L. Shi, S. Jin, W. Chen, and H. V. Poor, “Decision transformers for wireless communications: A new paradigm of resource management,” *IEEE Wireless Communications*, vol. 32, no. 2, pp. 180–186, Apr. 2025.
- [21] F. A. Oliehoek, C. Amato et al., *A concise introduction to decentralized POMDPs*. Springer, 2016, vol. 1.
- [22] S. Hu, Z. Fan, C. Huang, L. Shen, Y. Zhang, Y. Wang, and D. Tao, “Q-value regularized transformer for offline reinforcement learning,” in *Forty-first International Conference on Machine Learning*, 2024.
- [23] Q. Zheng, A. Zhang, and A. Grover, “Online decision transformer,” in *international conference on machine learning*. PMLR, 2022, pp. 27 042–27 059.
- [24] K. Yan, A. G. Schwing, and Y.-X. Wang, “Reinforcement learning gradients as vitamin for online finetuning decision transformers,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 38 590–38 628, 2024.
- [25] S. Fujimoto, H. Hoof, and D. Meger, “Addressing function approximation error in actor-critic methods,” in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.
- [26] W. Dabney, G. Ostrovski, and A. Barreto, “Temporally-extended ϵ -greedy exploration,” in *Proc. International Conference on Learning Representations (ICLR)*, 2021.
- [27] 3GPP, “NG-RAN; Xn general aspects and principles,” 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.420, 2018.
- [28] V. Jungnickel, K. Manolakis, S. Jaeckel, M. Lossow, and V. Braun, “Backhaul requirements for inter-site cooperation in heterogeneous LTE-Advanced networks,” in *Proc. IEEE Int. Conf. Commun. Workshops (ICC)*, 2013, pp. 905–910.
- [29] N. Naderializadeh and A. S. Avestimehr, “Itlinq: A new approach for spectrum sharing in device-to-device communication systems,” *IEEE journal on selected areas in communications*, vol. 32, no. 6, pp. 1139–1151, 2014.
- [30] G. Qu, A. Wierman, and N. Li, “Scalable reinforcement learning for multiagent networked systems,” *Operations Research*, vol. 70, no. 6, pp. 3601–3628, 2022.