Conditional Diffusion Model With OOD Mitigation as High-Dimensional Offline Resource Allocation Planner in Clustered Ad Hoc Networks

Kechen Meng, *Student Member, IEEE*, Sinuo Zhang, *Graduate Student Member, IEEE*, Rongpeng Li, *Senior Member, IEEE*, Chan Wang, *Member, IEEE*, Ming Lei, and Zhifeng Zhao, *Senior Member, IEEE*

***Abstract*—In modern clustered ad hoc networks, efficient and dynamic resource allocation is crucial for ensuring Quality of Service (QoS) under dynamic and uncertain environments. However, the challenges posed by limited sample efficiency, high interaction cost, and high-dimensional action space limit the effectiveness of the widely adopted Model-Free Reinforcement Learning (MFRL) solutions. In contrast, Model-Based RL (MBRL) offers an alternative approach to boost sample efficiency and stabilize the training by explicitly leveraging a learned environment model. Nevertheless, designing accurate and stable dynamics models in noisy, real-world communication scenarios remains a key bottleneck. To address these issues, we propose a Conditional Diffusion Model Planner (CDMP) for high-dimensional offline resource allocation in clustered ad hoc networks. By leveraging the powerful generative capability of Diffusion Models (DMs), our approach enables the accurate modeling of complex environmental dynamics and utilizes an inverse dynamics model for effective policy planning. Beyond simply adopting DMs in offline RL, we further incorporate the CDMP algorithm with a theoretically guaranteed, uncertainty-aware penalty metric, which theoretically and empirically manifests itself in mitigating the Out-of-Distribution (OOD)-induced distributional shift, a common issue for offline settings with scarce training data. Extensive experiments also show that our model outperforms MFRL in average reward and QoS, while demonstrating superior performance over other MBRL algorithms. These results highlight the practicality and scalability of our model in real-world network resource allocation tasks.**

***Index Terms*—Ad hoc network, communication resource allocation, model-based reinforcement learning, conditional diffusion model, model-uncertainty quantification.**

Received 11 April 2025; revised 24 July 2025; accepted 27 August 2025. Date of publication 4 September 2025; date of current version 23 December 2025. This work was primarily supported in part by the National Key Research and Development Program of China under Grant 2024YFE0200600, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LR23F010005, in part by Huawei Cooperation Project under Grant TC20240829036, in part by the National Key Laboratory of Wireless Communications Foundation under Grant 2023KP01601, and in part by the Big Data and Intelligent Computing Key Lab of CQUPT under Grant BDIC-2023-B-001. The associate editor coordinating the review of this article and approving it for publication was P. Sarigiannidis. *(Corresponding authors: Rongpeng Li; Chan Wang; Ming Lei.)*

Kechen Meng, Sinuo Zhang, Rongpeng Li, Chan Wang, and Ming Lei are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: mengkechen@zju.edu.cn; 22431100@zju.edu.cn; lirongpeng@zju.edu.cn; 0617464@zju.edu.cn; lm1029@zju.edu.cn).

Zhifeng Zhao is with Zhejiang Laboratory and the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: zhaozf@zhejianglab.com).

Digital Object Identifier 10.1109/TCOMM.2025.3606417

I. Introduction

WIRELESS ad hoc networks have been widely applied in military and civilian fields, due to its merits in flexibility, low cost, and robustness [1]. In this context, Time Division Multiple Access (TDMA) and its multi-frequency variant MF-TDMA have laid the very foundation for resource management therein [2]. Moreover, the complexity of the topology and the locality of node observations in ad hoc networks significantly constrain the system's stability and reliability. To overcome these challenges, clustering methods [3], [4] can be naturally engaged to enhance coordination and improve efficiency. Nonetheless, the dynamic nature of service arrivals and departures often leads to underutilization in static resource allocation solutions, highlighting the need for a dynamic solution to adapt to underlying variations. Several algorithms have been proposed to address this challenge [5], [6]. Prominently, the success of Reinforcement Learning (RL) [7], [8] motivates the emergence of smart resource allocation methods [2]. Typically, Model-Free RL (MFRL), such as DQN [9], DDPG [10], PPO [11], are chosen to guide the learning of an optimal policy through direct interaction with the physical environment, which is costly and potentially unsafe in real-world applications. Moreover, MFRL suffers from low sample efficiency and unstable training in scenarios that require complex planning [12]. Specifically, in the context of resource allocation in ad hoc networks with N nodes, M time slots, and L channels, a typical setting of the action space could be $N^{M\times L}$, awfully scaling along with the increase of L, M and N. To overcome these limitations, Model-Based RL (MBRL) has recently emerged as a promising approach to significantly improve the overall efficiency and effectiveness of RL algorithms [13].

Compared to MFRL, MBRL constructs an environment model from historical logs to simulate potential outcomes and generate synthetic trajectories, enabling the agent to fully leverage the experienced data while reducing trial-and-error costs [12]. Consequently, decision-making can be framed as a planning problem, where the agent simulates future trajectories to select actions that achieve desired outcomes [14]. This approach improves the agent's ability to handle long-term dependencies and make informed decisions. However, since MBRL performance depends on accurate environment

modeling, prediction errors, which naturally arise from the complexity and uncertainty of the dynamic environment, pose significant challenges. Additionally, traditional MBRL methods for planning problems rely on dynamic programming to stitch together sub-optimal reward-labeled trajectories in the training dataset to reconstruct optimal trajectories [15], [16]. To enable dynamic programming, these approaches learn a value function that estimates the discounted sum of rewards from a given state. However, value function estimation faces significant instabilities due to function approximation, off-policy learning, and bootstrapping-collectively referred to as the "deadly triad" [17]. These issues pose substantial barriers to scaling existing MBRL algorithms effectively.

On the other hand, since online policy training in ad hoc networks often faces high costs to interact with the environment, offline RL [18] is favored to train agents only with pre-collected offline datasets while saving the cost of online interaction. However, relying solely on previously collected data often leads to poor policies that prefer Out-of-Distribution (OOD) actions with overestimated values, resulting in unsatisfactory performance [19]. This phenomenon is primarily attributed to the distributional shift [20] between the learning and behavior policies, induced by the limited size and diversity of the training data. As a result, developing effective techniques to address this challenge has become a critical focus in the offline MBRL research community. Recent methods like MOPO [21] and MOREC [22], which incorporate a reward penalty based on an estimate of the uncertainty using model ensembles and learn reward-consistent dynamics through adversarial games respectively, partially mitigate this issue but still struggle to balance between model accuracy and computational complexity & stability.

In this paper, to accommodate allocated resources to imbalanced traffic loads in clustered ad hoc networks, we propose a Conditional Diffusion Model Planner (CDMP) with offline training. Unlike earlier deep generative models such as Energy-Based Models (EBMs) [23], Variational Autoencoders (VAEs) [24] and Generative Adversarial Networks (GANs) [25], which often suffer from issues like mode collapse, limited diversity, or training instability, Diffusion Models (DMs) offer a principled and robust alternative. They iteratively refine samples through a noise-to-data denoising process, enabling the modeling of complex, multi-modal distributions with greater sample quality and improved convergence behavior. These characteristics make DMs particularly appealing for high-dimensional offline RL, where capturing the diverse, uncertain nature of environment transitions is essential. Recent successes in vision and language domains [26], [27] further demonstrate their modeling strength, motivating our use of DM-based planning in wireless ad hoc networks. Notably, different from the DM-assisted MFRL [10], [28], [29], [30], we first use a conditional DM to characterize environmental dynamics and guide planning towards trajectories with high cumulative future rewards. Afterward, we employ an inverse dynamics model to generate actions based on state transitions between adjacent time steps, thus benefiting the obtaining of a dynamic resource allocation policy. Therefore, our methodology works completely in an offline MBRL manner. Furthermore, to mitigate potential OOD issues in offline training, we develop CDMP-pen, a variant of CDMP, with a theoretically guaranteed penalty metric that can more effectively quantify the uncertainty of the estimated dynamics model. Compared to the existing literature, the contribution of the paper can be summarized as follows:

- To implement load-aware resource allocation in clustered ad hoc networks, we propose a sample-efficient MBRL solution (i.e., CDMP) that leverages dynamic programming to combine sub-optimal trajectories into an optimal one without relying on inaccurate value estimation. By modeling the environmental dynamics as a conditional DM, CDMP can optimize the resource scheduling strategy in MF-TDMA through offline training. Due to the complexity of modeling high-dimensional action sequences, we model the diffusion process between states only and then utilize an inverse dynamics model to predict actions.
- To mitigate OOD issues in the MBRL offline setting, in CDMP-pen, we quantify the risk arising from imperfect dynamics and appropriately balance that risk with return. By calibrating a "smoothed distance to data" metric to measure model uncertainty, we can guide the generation of high-confidence trajectories, ensuring that the decision variables remain consistently close to the offline dataset. Furthermore, we theoretically demonstrate that the smoothed distance to data serves as an upper bound for the true uncertainty based on the Lipschitz constant, thereby validating the soundness and effectiveness of the proposed metric.
- We further illustrate our approach experimentally on high-fidelity OPNET simulations and validate the superiority of our proposed framework over other offline RL methods [31], [32], [33], [34], [35]. In addition, we conduct extensive ablation studies to corroborate the robustness and reliability of our approach under varying conditions and parameter settings, further highlighting its practicality and adaptability.

The remainder of the paper is organized as follows. Section II briefly introduces the related works. In Section III, we introduce the preliminaries and formulate the system model. The details of our proposed framework are presented in Section III. In Section IV, we provide the results of extensive simulations and numerical analysis. Finally, the conclusion is summarized in Section V.

For convenience, we list the major notations of this paper in Table I.

II. Related Works

A. Resource Schedule in Ad Hoc Networks

In the TDMA Multiple Access Channel (MAC) protocol, efficient time slot scheduling algorithms are essential for managing channel access and optimizing resource utilization [36]. Classical methods, such as DTSS [5], a bandwidth-efficient dynamic TDMA slot scheduling scheme to accommodate unpredictable network traffic, and DDMC-TDMA [37], a

TABLE I
MAIN PARAMETERS AND NOTATIONS USED IN THIS PAPER

Notations	Definition
N	Number of nodes
M	Number of time slots in a frame
L	Number of channels
K	Diffusion steps
H	Planning horizon
N_{epoch}	Number of training epochs
N_{step}	Number of training steps per epoch
ϵ_θ	Diffusion noise model
f_ϕ	Inverse dynamics model
$\boldsymbol{x}_k$	The intermediate latent variable at diffusion step k
$\boldsymbol{y}$	The conditioning information in conditional diffusion model
s_t, a_t, $\mathcal{R}_t$	State, action and reward of the agent at t
$\text{Return}(\tau)$	The return of the trajectory τ
$\widetilde{s}_t$	The perturbed state
$\hat{s}_t$	The reconstructed state
s_c	The closest data-point in the state dataset
$d_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}})^2$	The smoothed distance to data
$d(s;\mathcal{D}_{\text{stat}})^2$	The standard squared distance
$e(s)$	The true uncertainty between the data distribution and the surrogate distribution
$\text{gen}_t^{(i)}$	Number of packets generated within t
$\text{sen}_t^{(i)}$	Number of packets sent within t
$T_t^{(i)}$	Length of packet transmission queue at t
$T_{\max,t}^{(i)}$	Maximum length of packet transmission queue within t
n_{lm}	Node n allocated to time slot m of channel l
$d_t^{(i)}$	Time delay of node i at t
$u_t^{(i)}$	Throughput of node i at t
l_t	System packet loss rate at t
λ, η	Factors balancing the metrics in weighted reward function
ω	Conditional guidance scale
ξ	Low-temperature sampling factor
β	Parameter controlling the probability of removing conditional information
ζ	Parameter controlling the weight of OOD penalty term
$\mathcal{T}_{\text{total}}, \mathcal{S}_{\text{total}}$	Time complexity and space complexity of our algorithm
$\mathcal{T}_\theta, \mathcal{S}_\theta$	Time complexity and space complexity of the diffusion noise model
$\mathcal{T}_\phi, \mathcal{S}_\phi$	Time complexity and space complexity of the inverse dynamics model
d_{act}	Dimension of action embedding in the inverse dynamics model
d_{trans}	Dimension of state-transition pair embedding in the inverse dynamics model
D_{state}	Dimension of the agent's state representation
C_{out}	Base channel count of the temporal U-Net backbone
k_{conv}	Kernel size of the temporal U-Net backbone

topology-agnostic protocol designed for large-scale, high-density ad hoc networks, represent significant developments in this field. However, these approaches rely on predefined rules and static strategies, limiting adaptability in complex, real-time scenarios with high loads or bursty traffic. To overcome these limitations, MFRL-based methods have been introduced to enable more flexible and responsive scheduling. For instance, [38] proposes an actor-critic algorithm to optimize TDMA scheduling and minimize weighted end-to-end delay, a metric reflecting tactical traffic criticality. Similarly, [39] utilizes node features for state representation and applies diverse heuristics across time slots, enabling adaptive decisions that account for long-term deadline adherence. While these RL-based methods leverage real-time feedback to dynamically adjust scheduling strategies and enhance adaptability over traditional rule-based approaches, they still encounter challenges in long-term decision-making, convergence to local optima, and training instability. Moreover, their online RL nature incurs substantial interaction costs. To address these issues, we incorporate DMs with offline RL for MF-TDMA resource allocation, which better captures long-term dependencies and enables global optimization, thereby providing a more robust solution for high-dimensional resource management.

B. Diffusion Models for Resource Allocation in Communication Systems

Although prior studies have exemplified the expanding role of DMs in tackling complex problems beyond their traditional generative contexts [40], [41], their application to wireless networks for optimizing resource allocation strategies is still evolving [42]. These networks exhibit complex dynamics and highly mobility of wireless devices [43]. Additionally, the impact of noise and interference in wireless channels, which affects the generalization of RL policies, is often overlooked [44]. Recently, several studies have attempted to incorporate DMs into wireless communication tasks such as data augmentation, latent space representation, and policy optimization. For example, [28] proposes a DDPM-based resource allocation framework that generates optimal block lengths conditioned on channel state information. D3PG [29] introduces an access mechanism that integrates DMs with DDPG [10] framework to jointly adjust the contention window and the aggregation frame length in Wi-Fi networks. In addition, [45] develops a hierarchical architecture for resource coordination, where a DM-driven soft actor-critic [46] is employed at the high level to generate spectrum allocation strategies, while QMIX [47] operates at the low level for fine-grained joint power control and resource assignment. Moreover, [30] explores a multi-agent setting using DMs for optimal DNN partitioning and task offloading. However, these approaches are predominantly built upon the MFRL paradigm, where DMs are employed to directly model the agent's policy. Despite demonstrating certain effectiveness, they typically suffer from low sample efficiency, unstable convergence, and suboptimal decision quality, particularly under dynamic and uncertain wireless environments. To tackle these limitations, we adopt a MBRL framework, wherein DMs are used to model the environment dynamics by approximating the state transition distribution. This modeling approach enables the agent to plan actions without relying on frequent interaction with the real environment, thereby significantly improving sample efficiency, enhancing convergence stability, and allowing for long-horizon planning in complex wireless network scenarios.

C. Planning With Model-Based Reinforcement Learning

To tackle the sampling inefficiency and communication overhead in MFRL, MBRL naturally emerges as an alternative solution. By leveraging a learned dynamics model, agents

can predict future trajectories and achieve higher returns through planning, which is a fundamental research focus in MBRL. However, MBRL algorithms often struggle to scale effectively as the complexity of RL tasks increases. Moreover, modeling errors in dynamics tend to accumulate over time, significantly limiting the practical applications of MBRL approaches. Focusing on these challenges, PETS [16] combines uncertainty-aware dynamics models with sampling-based uncertainty propagation using probabilistic ensembles. POPLIN [48] formulates action planning at each time-step as an optimization problem, and argues that optimization in the policy network's parameter space will be more efficient. On the other hand, PILCO [49] explicitly marginalizes aleatoric and epistemic uncertainty in the dynamics model to directly optimize the surrogate expected reward for long-term planning. Although these works mitigate some of the aforementioned problems, they either rely on model ensembles for uncertainty estimation, which becomes computationally prohibitive in high-dimensional environments, or require value function fitting for dynamic programming, resulting in severe cumulative errors and training instability. Additionally, conventional MBRL approaches typically rely on fully connected networks parameterizing diagonal-covariance Gaussian distributions as the environment model, which struggle to accurately represent complex multi-modal distributions. Inspired by recent advancements [34], [50], we reinterpret the planning problem in MBRL as a sequence generation task, thereby alleviating the computational burden while fully leveraging the expressiveness and robustness of DMs.

D. Addressing Out-of-Distribution in Offline RL

Although offline RL avoids the risks and time-consuming nature of real-world interactions, it faces large extrapolation errors when the Q-function is evaluated on OOD actions, which can lead to unstable learning and potential divergence. To address these challenges, recent offline RL algorithms have introduced conservative value function estimation to enhance the reliability of model-based rollouts. For instance, COMBO [51] learns a conservative critic by penalizing the value function for state-action pairs outside the support of the offline dataset, while OSR [52] further incorporates Conservative Q-Learning (CQL) [32] with an estimated inverse dynamics model to implement state recovery. Another class of approaches focuses on optimizing a lower bound of policy performance, which is constructed using uncertainty estimates of the learned model. MOPO [21] penalizes rewards through the aleatoric uncertainty in next-state predictions, and MOBILE [53] introduces an uncertainty quantifier to estimate the Bellman errors induced by the learned dynamics through the inconsistency of Bellman estimations under different learned models. Drawing from these methodologies, we integrate uncertainty-based penalization to avoid unreliable model predictions, aiming to more effectively balance between model accuracy and computational complexity & stability.

III. Preliminaries & System Model

In this section, we briefly introduce the framework of MBRL and some fundamentals of DMs. On this basis, we highlight how to apply DMs to tackle the sequential decision-making issue for resource allocation in clustered ad hoc networks.

A. Preliminaries

1) Model-Based Reinforcement Learning: We formulate the sequential decision-making problem as a discounted Markov Decision Process (MDP), defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}(s_{t+1}|s_t, a_t), \mathcal{R}(s_t, a_t), \gamma \rangle$ [35]. In this formulation, $\mathcal{S}$ represents the set of possible states and $\mathcal{A}$ is the set of actions the agent can take. The transition function $\mathcal{T}$ describes the probability of transitioning from state s_t to s_{t+1} after executing an action a_t. The reward function $\mathcal{R}$ gives the immediate reward the agent receives after taking action a_t in state s_t, and the discount factor $\gamma \in [0, 1)$ balances the importance of future rewards, with lower values favoring immediate rewards. The agent follows a stochastic policy π, mapping states to actions. In RL, the standard goal of the agent is to learn a policy that maximizes the expected cumulative discounted sum of rewards in trajectory $\tau := \{s_0, a_0, s_1, a_1, \cdots\}$ as

$$\pi^* := \arg\max_{\pi} \mathbb{E}_{\tau \sim p_\pi} \left[\sum\nolimits_{t \geq 0} \gamma^t \mathcal{R}(s_t, a_t) \right]. \tag{1}$$

In MBRL, the agent first estimates the dynamics model from collected environmental transition samples. Afterward, based on the learned dynamics model, the agent predicts subsequent actions and executes a planning phase to optimize its policy [54]. This characteristic significantly reduces the physical interactions between the agent and the environment, which results in higher data efficiency and lower sampling costs [12].

2) Diffusion Probabilistic Models: We try to use $p_\theta(\boldsymbol{x})$ to approximate the data distribution $q(\boldsymbol{x})$ from the dataset $\mathcal{D} := \{\boldsymbol{x}^i\}_{0 \leq i \leq M}$ based on DMs, which can be achieved by minimizing the KL divergence between both distributions [55]. By the definition of divergence, this objective can be reformulated as maximizing the log-likelihood as

$$\begin{aligned} \theta^* &:= \arg\min_{\theta} D_{\mathrm{KL}}(q(\boldsymbol{x}) \parallel p_\theta(\boldsymbol{x})) \\ &= \arg\min_{\theta} \mathbb{E}_{\boldsymbol{x} \sim q(\boldsymbol{x})} [\log q(\boldsymbol{x}) - \log p_\theta(\boldsymbol{x})] \\ &= \arg\max_{\theta} \mathbb{E}_{\boldsymbol{x} \sim q(\boldsymbol{x})} [\log p_\theta(\boldsymbol{x})] \\ &\approx \arg\max_{\theta} \sum\nolimits_{i=1}^{M} \log p_\theta(\boldsymbol{x}^i). \end{aligned} \tag{2}$$

As a powerful class of generative models, DMs transform the objective above into maximizing the Evidence Lower BOund (ELBO) of a Hierarchical Variational Auto-Encoder (HVAE) [56]. By modeling the data generation process through reversing a forward noising process, DMs effectively perform a denoising process. For each data point $\boldsymbol{x}_0 \sim q(\boldsymbol{x})$ from the dataset $\mathcal{D}$, the predefined forward noising process can be modeled as a discrete Markov chain $\boldsymbol{x}_{0:K}$ such that $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \mathcal{N}(\boldsymbol{x}_k|\sqrt{\alpha_k}\boldsymbol{x}_{k-1}, (1-\alpha_k)\boldsymbol{I})$. Here, $\mathcal{N}(\mu, \Sigma)$ denotes a Gaussian distribution with mean μ and variance Σ, and $0 < \alpha_K < \cdots < \alpha_0 < 1 \in \mathbb{R}$ are hyperparameters controlling the variance schedule. For appropriately chosen α_k and a sufficiently large K, we assume that $\boldsymbol{x}_K \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. The trainable reverse process modeled as the variational

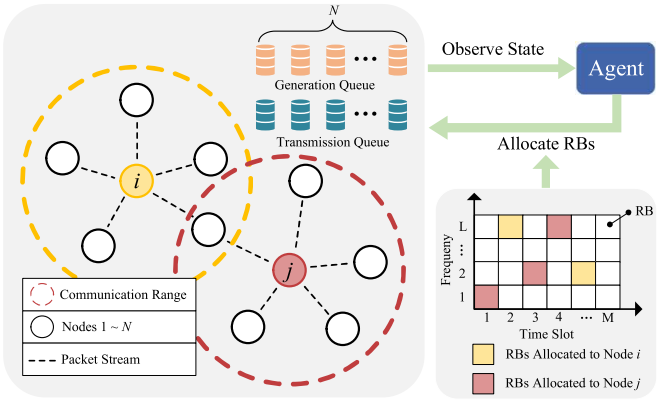


Fig. 1. Ad hoc network topology under MF-TDMA MAC protocol.

reverse Markov chain is parameterized with $p_\theta(\boldsymbol{x}_{k-1}|\boldsymbol{x}_k) = \mathcal{N}(\boldsymbol{x}_{k-1}|\mu_\theta(\boldsymbol{x}_k,k),\Sigma(\boldsymbol{x}_k,k))$. This allows us to sample $\boldsymbol{x}_K$ from standard Gaussian noise and progressively denoise it until obtaining $\boldsymbol{x}_0$. The reverse process can be learned by optimizing a simplified surrogate loss as

$$\mathcal{L}(\theta) := \mathbb{E}_{\substack{k\sim[1,K]\\ \boldsymbol{x}_0\sim q\\ \epsilon\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}}\left[\| \epsilon - \epsilon_\theta(\boldsymbol{x}_k,k)\|^2\right]. \tag{3}$$

Here, the deep neural network ϵ_θ parameterized by θ is trained to predict the noise $\epsilon \sim \mathcal{N}(\mathbf{0},\boldsymbol{I})$, which is added to the dataset sample $\boldsymbol{x}_0$ to produce $\boldsymbol{x}_k$ as

$$\boldsymbol{x}_k := \sqrt{\overline{\alpha}_k}\boldsymbol{x}_0 + \sqrt{1-\overline{\alpha}_k}\epsilon, \tag{4}$$

where $\overline{\alpha}_k := \prod_{s=1}^{k}\alpha_s$. This is equivalent to predicting the mean of $p_\theta(\boldsymbol{x}_{k-1}|\boldsymbol{x}_k)$, since $\mu_\theta(\boldsymbol{x}_k,k)$ can be calculated as a function of $\epsilon_\theta(\boldsymbol{x}_k,k)$ [57] as

$$\mu_\theta(\boldsymbol{x}_k,k) = \frac{1}{\sqrt{\alpha_k}}\boldsymbol{x}_k - \frac{1-\alpha_k}{\sqrt{1-\overline{\alpha}_k}\sqrt{\alpha_k}}\epsilon_\theta(\boldsymbol{x}_k,k). \tag{5}$$

B. System Model

We consider an adaptive, resource-constrained clustered ad hoc network scenario with N nodes, as illustrated in Fig. 1. In this setup, each node communicates directly with its one-hop neighbors using wireless transceivers, while communication with remote nodes requires multi-hop relaying. Every node is equipped with packet queues to cache data packets. If a sent packet remains unacknowledged, it will be appended to the end of the cache queue of forwarding nodes in case. We assume the traffic loads therein are prioritized, where a portion of nodes generate packets at high rates while others at relatively slower rates. However, the extent of imbalance and variations in traffic loads are unknown in advance. Specifically, we employ MF-TDMA [2] as the MAC layer protocol. In this setting, the practical implementation in one frame encompasses two stages (i.e., network management and traffic transmission). In the first stage, nodes perform time synchronization and interact with the agent to determine the allocation scheme. The second stage can be divided into ML Resource Blocks (RBs), where the nodes start to transmit packets within allocated RBs efficiently.

To optimize resource utilization and minimize collision probability, it is essential to leverage a dynamic resource allocation mechanism to allocate an appropriate number of RBs to each node according to individual characteristics. Therefore, we formulate the resource allocation problem as an MDP, and define the decision interval as one frame. Specifically, at each time-step (i.e., one frame) t, the agent observes the packet queue length of all nodes in the system. The state of node i is defined as $s_t^{(i)} := \left[\text{gen}_t^{(i)}, \text{sen}_t^{(i)}, T_t^{(i)}, T_{\max,t}^{(i)}\right]$, where $\text{gen}_t^{(i)}$ and $\text{sen}_t^{(i)}$ denote the numbers of packets generated and sent within the frame t, while $T_t^{(i)}$ and $T_{\max,t}^{(i)}$ represent the real-time transmission queue length at t and maximum length within frame t, respectively. The overall state is the concatenation of the states of all nodes, i.e., $s_t := \left[s_t^{(1)}, s_t^{(2)}, \cdots, s_t^{(N)}\right]$. Based on the observed state s_t, the agent takes action $a_t := [n_{lm}] \in \mathbb{Z}^{L\times M}$ to allocate RBs, where n_{lm} denotes the node n assigned to time slot m of channel l. To improve the overall system Quality of Service (QoS), we define the reward function as

$$\mathcal{R}_t := -\sum_{i=1}^{N} d_t^{(i)}, \tag{6}$$

where $d_t^{(i)}$ indicates the time delay for packets received by node i. Our experiment experience shows that by minimizing the average delay, the communication system can achieve improved throughput and reduced packet loss rate, enhancing the overall QoS. Meanwhile, it avoids the cumbersome tuning of multi-objective optimization weights. In this paper, we resort to a conditional DM-based high-dimensional resource allocation planner for maximizing the cumulative reward function.

IV. Resource Allocation With CDMP & CDMP-Pen

In this section, we discuss how to apply conditional DMs to guide the generation of trajectories with high confidence toward superior cumulative returns. We elaborate on the CDMP framework for policy optimization through offline training. Furthermore, we develop CDMP-pen with uncertainty estimation to effectively combat the OOD problem. The entire procedure for training and implementation is illustrated in Fig. 2.

A. Planning Over Decision Sequences

In the planning phase of MBRL, it usually needs to simulate or explore different sequences of actions and states, thus evaluating outcomes and evolving decisions from the perspective of a longer horizon. Correspondingly, we formulate the state and/or action trajectory modeling as a DM-based conditional generative problem [58], that is

$$\max_\theta \mathbb{E}_{\tau\sim\mathcal{D}}\left[\log p_\theta(\boldsymbol{x}_0(\tau)|\boldsymbol{y}(\tau))\right]. \tag{7}$$

In other words, after characterizing the distribution p_θ of the trajectories sampled from an MDP, we can conveniently generate portions of a trajectory $\boldsymbol{x}_0(\tau)$ from information $\boldsymbol{y}(\tau)$ about it. Here, in the context of RB allocation in clustered ad hoc networks, the discreteness of actions, with higher frequency and less smooth characteristics, adds to the difficulty of predicting and modeling. Therefore, we define $\boldsymbol{x}_k(\tau)$ as a noisy sequence of states with a length H, namely

$$\boldsymbol{x}_k(\tau) := (s_t, s_{t+1}, \cdots, s_{t+H-1})_k. \tag{8}$$

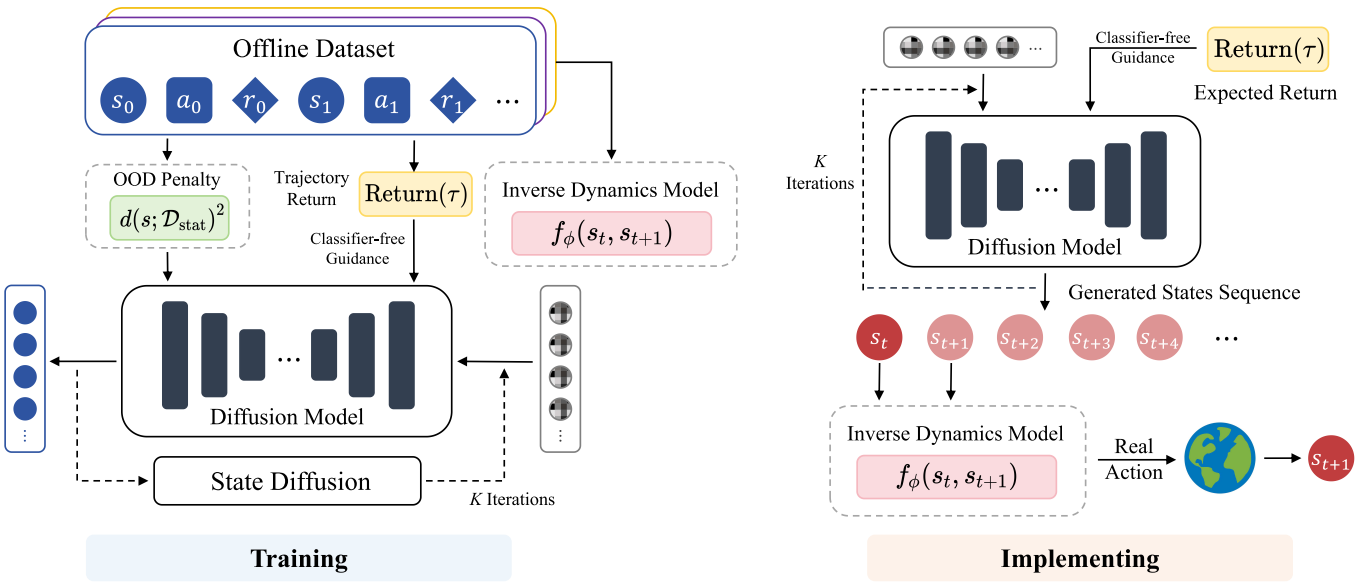


Fig. 2. Training and implementation of CDMP & CDMP-pen.

Meanwhile, to guide the denoising process toward generating trajectories with high cumulative rewards, $\boldsymbol{y}(\tau)$ could refer to the return of the trajectory $\mathrm{Return}(\tau)$, that is

$$\boldsymbol{y}(\tau) := \mathrm{Return}(\tau) := \sum_{t'=t}^{t+H-1} \gamma^{t'-t} \mathcal{R}_{t'}. \tag{9}$$

Next, contingent on a learned inverse dynamics model $f_\phi(s_t, s_{t+1})$ [59], the training of which will be given in Section IV-B, for any timestep t in $\boldsymbol{x}_0(\tau)$, the policy is inferred by estimating the action that leads to the transition from state s_t to s_{t+1}, namely

$$a_t := f_\phi(s_t, s_{t+1}). \tag{10}$$

Therefore, we can accomplish trajectory-based planning to implement policy optimization.

B. Classifier-Free Guidance-Based Training

In order to guide trajectory generation for more efficient planning and decision-making, conditional information is incorporated into the diffusion process to ensure that the generated trajectories align with the desired attributes. One possible solution is classifier-based guidance [60], which trains a classifier independently to predict condition $\boldsymbol{y}(\tau)$ from noisy trajectories $\boldsymbol{x}_k(\tau)$, and the gradients from the classifier are used to guide the sampling process of an unconditional diffusion model. However, this method introduces additional computational overhead and heightened sensitivity to classifier performance, as training a classifier is equivalent to estimating a Q-function to predict the discounted cumulative reward from timestep t.

Therefore, instead of training a separate classifier model, we utilize classifier-free guidance [61], which mixes the score estimates of a conditional diffusion model and a jointly trained unconditional diffusion model as $((1-\beta)\boldsymbol{y}(\tau) + \beta\varnothing, k)$. Here, the parameter β, drawn from a Bernoulli distribution, controls the interpolation between the conditional and unconditional guidance. In other words, we use a unified neural network to parameterize both models, whereas for the unconditional model, we can simply input a null token $\varnothing$. On this basis, consistent with Eq. (3), we train an unconditional denoising diffusion model $p_\theta(\boldsymbol{x}_{k-1}(\tau)|\boldsymbol{x}_k(\tau))$ parameterized through a noise estimator $\epsilon_\theta(\boldsymbol{x}_k(\tau), k)$ together with the conditional model $p_\theta(\boldsymbol{x}_{k-1}(\tau)|\boldsymbol{x}_k(\tau), \boldsymbol{y}(\tau))$ parameterized through $\epsilon_\theta(\boldsymbol{x}_k(\tau), \boldsymbol{y}(\tau), k)$.

We simultaneously train the reverse diffusion process p_θ, which can be reasonably learned from the noise model ϵ_θ Eq. (5), and the inverse dynamics model f_ϕ in a supervised manner. Given an offline dataset $\mathcal{D}$ that consists of state-action trajectories labeled with rewards, we train with the reverse diffusion loss and the inverse dynamics loss:

$$\begin{aligned}&\mathcal{L}_{\mathrm{CDMP}}(\theta, \phi)\\ &:= \mathbb{E}_{(s,a,s')\in\mathcal{D}}\left[\| a - f_\phi(s, s') \|^2\right]\\ &\quad + \mathbb{E}_{k,\tau\in\mathcal{D},\beta}\left[\| \epsilon - \epsilon_\theta(\boldsymbol{x}_k(\tau), (1-\beta)\boldsymbol{y}(\tau) + \beta\varnothing, k) \|^2\right].\end{aligned} \tag{11}$$

C. OOD Penalty Based on Distance to Data

Although offline MBRL leverages pre-collected data, which is often more cost-effective compared to online interactions, it inherently lacks real-time data exchange. This limitation hinders the model's adaptability to unfamiliar states or dynamic changes in the environment, ultimately leading to the OOD problem. Recalling the derivations in Eq. (2), we try to remedy this by introducing a constrained conservative objective for individual states as

$$\min_\theta \sum_{t'=t+1}^{t+H-1} e(s_{t'}), \tag{12}$$

where $e(s_{t'}) := \| \log p_\theta(s_{t'}) - \log q(s_{t'}) \|_2$ is defined as the true uncertainty between the true distribution q and the surrogate distribution p_θ. Eq. (12) restricts the generated states

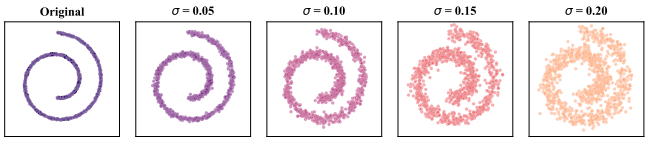


Fig. 3. Illustration of the perturbed two-dimensional swiss roll distribution.

of the trajectory to closely match the training distribution, thus making the generation more reliable.

However, directly computing likelihoods for high-dimensional data poses significant challenges. Meanwhile, the scarcity of the dataset adds to the difficulty. Consistent with prior works [62], [63], we first perturb data with random Gaussian noise to smooth the empirical distribution and construct a more comprehensive state space that includes meaningful OOD states [64]. Consider the state dataset $\mathcal{D}_{\text{stat}} := \{s^i\}_{0\le i\le M'}$ and its corresponding empirical distribution $q(s;\mathcal{D}_{\text{stat}})$, we define the perturbed empirical distribution [63] as

$$\begin{aligned} q_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}}) &:= \int q(s;\mathcal{D}_{\text{stat}})\mathcal{N}(\widetilde{s};s,\sigma^2\mathbf{I})ds \\ &= \frac{1}{M'}\sum_i \mathcal{N}(\widetilde{s^i};s^i,\sigma^2\mathbf{I}), \end{aligned} \tag{13}$$

where σ is the noise level controlling the degree of perturbation applied to the original data. As demonstrated by the two-dimensional Swiss roll distribution in Fig. 3, an increase in σ results in a progressively smoother distribution. Afterward, we try to reformulate Eq. (12) by introducing an uncertainty penalty term based on the smoothed distance to data [65]. Taking an arbitrary point in the smoothed state space $\widetilde{s}\in\mathbb{R}^n$, we formally define the smoothed distance to data as

$$\begin{aligned} d_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}})^2 &:= \text{Softmin}_\sigma \frac{1}{2}\|\widetilde{s}-s^i\|_2^2 + C_1 \\ &= -\sigma^2\log\left[\textstyle\sum_i \exp\left(-\frac{1}{2\sigma^2}\|\widetilde{s}-s^i\|_2^2\right)\right] + C_1, \end{aligned} \tag{14}$$

where C_1 is the constant to ensure positiveness of $d_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}})^2$. Different from [65], which directly optimizes actions based on policy gradient methods, we apply this "smoothed distance to data" metric to constrain the generated states for conservative planning. More precisely, the rationality of using the smoothed distance to data as an uncertainty metric can be explained in the following lemma.

Lemma 1: The negative log-likelihood of the perturbed empirical distribution $q_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}})$ is equivalent to the smoothed distance to data by the noise level σ, up to some constant that does not depend on s,

$$-\sigma^2\log q_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}}) = d_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}})^2 + C(M',n,\sigma), \tag{15}$$

where we define $C(M',n,\sigma) := \sigma^2(\log M' + n/2\log(2\pi\sigma))$.

The proof of Lemma 1 is included in Appendix A. Intuitively, Lemma 1 explicitly links the likelihood and the "smoothed distance to data" metric. Unlike aforementioned uncertainty-penalty-based methods [21], [53] that rely on model ensembles or require value function fitting, this metric offers higher computational efficiency and models uncertainty with reasonable accuracy.

Next, we mathematically establish the relationship between the "smoothed distance to data" metric and $e(s)$ in Eq. (12), by analyzing how much the metric provides an upper bound on the true uncertainty using the Lipschitz constant [66] of the model bias L_e.

Theorem 1: Let L_e be the local Lipschitz constant of the true error $e(s)$ valid over the state space $\mathcal{S}$. For any $s\in\mathcal{S}$, $e(s)$ can be bounded by

$$e(s) \le e(s_c) + \sqrt{2}L_e\sqrt{d_\sigma(s;\mathcal{D}_{\text{stat}})^2 + C_2}, \tag{16}$$

where $s_c := \arg\min_{s^i\in\mathcal{D}_{\text{stat}}} \frac{1}{2}\| s-s^i \|_2^2$, i.e., the closest data-point, and $C_2 := \sigma^2\log M' - C_1$, with C_1 defined in Eq. (14).

The proof of Theorem 1 is unveiled in Appendix B. We believe this offers benefits over directly maximizing log-likelihood, as the smoothed distance to data avoids numerical instability and does not require explicit density estimation. In general, it is difficult to obtain L_e in the absence of more structured knowledge of the empirical distribution $q(s)$. However, it is possible to obtain confidence bounds on L_e using statistical estimation with pairwise finite slopes $\|e(s^i)-e(s^j)\|/\|s^i-s^j\|$ within the dataset [67, Ch. 3] [68]. Based on Theorem 1, the "smoothed distance to data" metric can overestimate the true uncertainty $e(s)$, enabling the establishment of an error bound and serving as a safeguard against excessive deviation from the true data distribution. Moreover, leveraging the powerful distribution modeling capability of DMs, we can assume that the first term on the right side of Eq. (16) approaches zero. Therefore, we primarily focus on optimizing the second term.

Additionally, as the smoothing degree approaches zero, i.e., $\sigma\to 0$, the smoothed distance to data converges to the standard squared distance $d(s;\mathcal{D}_{\text{stat}})^2 := \min_{s^i\in\mathcal{D}_{\text{stat}}}\frac{1}{2}\|s-s^i\|_2^2$. To encourage the surrogate state distribution closer to the real noise-free data distribution, we employ this true minimum distance $d(s;\mathcal{D}_{\text{stat}})^2$ as the uncertainty metric during practical training. As a byproduct, it also enhances the safety of long-term planning and promotes policy conservativeness. Moreover, we relax this minimum distance by the deviation between the model-generated states and the input states within data batches. According to Eq. (4), the states generated by the model can be directly obtained from the reconstructed trajectories as

$$(\hat{s}_t,\hat{s}_{t+1},\cdots,\hat{s}_{t+H-1}) := \hat{\boldsymbol{x}}_0(\tau) := \frac{\boldsymbol{x}_k(\tau)-\sqrt{1-\overline{\alpha}_k}\epsilon_\theta}{\sqrt{\overline{\alpha}_k}}. \tag{17}$$

Based on this, we reformulate the conservative objective in Eq. (12) as an OOD penalty term, namely

$$\mathcal{L}_{\text{OOD}}(\theta) := \mathbb{E}_{k,\tau\in\mathcal{D}}\left[\sum_{t'=t+1}^{t+H-1}\| s_{t'}-\hat{s}_{t'}\|^2\right], \tag{18}$$

where $s_{t'}$ represents the state within the batch from the offline dataset. Since this relaxed distance is guaranteed to be greater than or equal to the true minimum distance, we can theoretically prove that this metric still bounds the true model error $e(s)$. Ultimately, we formulate the penalized loss as

$$\mathcal{L}_{\text{CDMP}-\text{pen}}(\theta,\phi) := \mathcal{L}_{\text{CDMP}}(\theta,\phi) + \zeta\mathcal{L}_{\text{OOD}}(\theta), \tag{19}$$

Algorithm 1 Offline Training of CDMP & CDMP-Pen

Input: Offline trajectory dataset $\mathcal{D}$ and state dataset $\mathcal{D}_{\text{stat}}$, planning horizon H, diffusion steps K, conditional information removal probability β, OOD penalty weight ζ
Initialize Diffusion noise model ϵ_θ and inverse dynamics model f_ϕ with random parameters θ, ϕ

for each epoch **do**
  for each step **do**
    Sample a mini-batch $\mathcal{B}$ of $\boldsymbol{x}_0(\tau)$ in $\mathcal{D}$
    Predict actions $f_\phi(s, s')$ for each state transition pair in $\mathcal{B}$
    $\boldsymbol{y}(\tau) \leftarrow \text{Return}(\tau)$ ▷ Obtain the condition information by Eq. (9)
    $k \sim \text{Uniform}(\{1, 2, \cdots, K\})$
    $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
    $\boldsymbol{x}_k(\tau) \leftarrow \sqrt{\overline{\alpha}_k}\boldsymbol{x}_0(\tau) + \sqrt{1-\overline{\alpha}_k}\epsilon$ ▷ Apply the forward diffusion process
    Predict noise $\epsilon_\theta(\boldsymbol{x}_k(\tau), (1-\beta)\boldsymbol{y}(\tau) + \beta\varnothing, k)$
    if training CDMP **then**
      Update the diffusion noise model ϵ_θ and inverse dynamics model f_ϕ by Eq. (11)
    else if training CDMP-pen **then**
      $(\hat{s}_t, \hat{s}_{t+1}, \cdots, \hat{s}_{t+H-1}) \leftarrow \left(\boldsymbol{x}_k(\tau) - \sqrt{1-\overline{\alpha}_k}\epsilon_\theta\right)/\sqrt{\overline{\alpha}_k}$
      ▷ Reconstruct trajectory
      Update the diffusion noise model ϵ_θ and inverse dynamics model f_ϕ by Eq. (19)
    end if
  end for
end for

where ζ controls the weight of OOD penalty term. The overall offline training procedure of CDMP & CDMP-pen is summarized in Algorithm 1, while the implementation process of both frameworks will be detailed in Section IV-D.

D. Implementation

We can implement the well-trained noise model and inverse dynamic model to generate trajectories for policy optimization. Formally, the trajectory generation process p_θ can be described as starting from Gaussian noise $\boldsymbol{x}_K(\tau) \sim \mathcal{N}(\mathbf{0}, \xi\boldsymbol{I})$ and diffuse $\boldsymbol{x}_k(\tau)$ into $\boldsymbol{x}_{k-1}(\tau)$ at each diffusion step k with the perturbed noise as

$$\hat{\epsilon} := \epsilon_\theta(\boldsymbol{x}_k(\tau), \varnothing, k) + \omega(\epsilon_\theta(\boldsymbol{x}_k(\tau), \boldsymbol{y}(\tau), k) - \epsilon_\theta(\boldsymbol{x}_k(\tau), \varnothing, k)), \tag{20}$$

where we manually set the condition $\boldsymbol{y}(\tau)$ as a high constant value during implementation, which serves as the conditional expected return, and the scalar ω adjusts the influence of the condition by amplifying and extracting the most relevant portions of trajectories that exhibit the desired attributes in the dataset. Meanwhile, to reduce randomness and enable the DM to focus on generating higher-likelihood trajectories from the dataset, we employ low-temperature sampling with a scaling factor $\xi \in [0, 1)$ [58]. This approach reduces the variance of the Gaussian distribution at each step of the denoising process, progressively lowering the noise intensity as the model iteratively refines the generated trajectory. By applying this scaling factor at every step, we prevent the degradation of model generation quality due to the contamination of suboptimal behaviors in the offline dataset, which helps stabilize the exploration process and ensures that the model generates more accurate and reliable trajectories. In addition, to ensure consistency between the generated trajectory and the historical data, the first state of the trajectory is always set to the currently observed state throughout all diffusion steps. After completing the final K iterations of the denoising process, we can infer the next researchable predicted state and determine the optimal action using our inverse dynamics model f_ϕ. This procedure repeats in a standard receding-horizon control loop described in Algorithm 2.

Algorithm 2 Implementation of CDMP & CDMP-Pen

Input: Diffusion noise model ϵ_θ, inverse dynamics model f_ϕ, planning horizon H, diffusion steps K, conditional guidance scale ω, condition $\boldsymbol{y}$
Initialize $t \leftarrow 0$

while not done **do**
  Observe s_t; Initialize $\boldsymbol{x}_K(\tau) \sim \mathcal{N}(\mathbf{0}, \xi\boldsymbol{I})$
  $\boldsymbol{x}_K(\tau)[0] \leftarrow s_t$ ▷Constrain consistency
  for $k = K \cdots 1$ **do**
    $\hat{\epsilon} \leftarrow \epsilon_\theta(\boldsymbol{x}_k(\tau), k) + \omega(\epsilon_\theta(\boldsymbol{x}_k(\tau), \boldsymbol{y}, k) - \epsilon_\theta(\boldsymbol{x}_k(\tau), k))$
    ▷Classifier-free guidance
    $(\mu_{k-1}, \Sigma_{k-1}) \leftarrow \text{Denoise}(\boldsymbol{x}_k(\tau), \hat{\epsilon})$
    $\boldsymbol{x}_{k-1}(\tau) \sim \mathcal{N}(\mu_{k-1}, \xi\Sigma_{k-1})$
    $\boldsymbol{x}_{k-1}(\tau)[0] \leftarrow s_t$ ▷Constrain consistency
  end for
  Extract (s_t, s_{t+1}) from $\boldsymbol{x}_0(\tau)$
  Execute $a_t = f_\phi(s_t, s_{t+1})$; $t \leftarrow t+1$
end while

E. Time Complexity and Space Complexity Analysis

1) Time Complexity Analysis: The time complexity of our algorithm at each decision timestep is primarily determined by: the diffusion steps K, the time complexity of the diffusion noise model $\mathcal{T}_\theta$ and the time complexity of the inverse dynamics model $\mathcal{T}_\phi$, which can be expressed as

$$\mathcal{T}_{\text{total}} = K \cdot \mathcal{T}_\theta + \mathcal{T}_\phi. \tag{21}$$

Due to the discrete and non-smooth nature of the action space, we employ an autoregressive strategy to generate M-dimensional action vector. Specifically, the inverse dynamics model first encodes the input state-transition pair using a 4-layer MLP into a latent representation of dimension d_{trans}. Each action dimension is then generated sequentially, conditioned on the state embedding and a d_{act}-dimensional representation of previously predicted components. This concatenated embedding is passed through an output head that produces a probability distribution over N discrete bins, while the final action is obtained by sampling from these distributions. This structure improves generalization by breaking the

high-dimensional action prediction into conditionally dependent subproblems, and its formulation over discrete bins makes it naturally compatible with non-continuous action domains, enhancing the accuracy of action synthesis. Notably, the dominant computational cost arises from this iterative generation process, while the cost for embedding state-transition pairs is negligible. Therefore, the overall time complexity of the inverse dynamics model is given by

$$\begin{aligned}\mathcal{T}_\phi &= \sum_{i=1}^{ML} \left(\mathcal{O}(i \cdot d_{\text{act}}) + \mathcal{O}((d_{\text{act}} + d_{\text{trans}}) \cdot N)\right) \\ &= \mathcal{O}\left((ML)^2 \cdot d_{\text{act}}\right) + \mathcal{O}\left(ML \cdot (d_{\text{act}} + d_{\text{trans}}) \cdot N\right) \end{aligned} \tag{22}$$

Following the design in [35], we primarily adopt the temporal U-Net architecture [69] as the diffusion noise model ϵ_θ. The network consists of 3 paired downsampling and upsampling residual blocks, each comprising two temporal convolutions. Without loss of generality, for a block with the number of input and output channels C_{in} and C_{out}, it has time complexity as

$$\mathcal{O}\left((C_{\text{in}} + C_{\text{out}}) \cdot C_{\text{out}} \cdot k_{\text{conv}} \cdot D\right), \tag{23}$$

where k_{conv} denotes the kernel size, and D is the output feature length. In our case, the first residual block maps the input of D_{state} channels to a base channel count C_{out}, and the subsequent two blocks increase this to $4C_{\text{out}}$ and $8C_{\text{out}}$, respectively. Meanwhile, the feature length starts with $D = H$, and is downsampled by half to $H/4$ through the next two blocks. The reverse upsampling path is symmetric to this downsampling process. Diffusion step and condition embeddings are generated by separate 2-layered MLPs and are concatenated before getting added to the activations of the first temporal convolution within each block. Note that the main computational cost derives from the temporal convolutions in residual blocks. Thus, by Eq. (23), the time complexity of ϵ_θ can be expressed as

$$\mathcal{T}_\theta = \mathcal{O}\left((D_{\text{state}} + C_{\text{out}}) \cdot C_{\text{out}} \cdot k_{\text{conv}} \cdot H\right) \tag{24}$$

Merging Eq. (22) and Eq. (24) into Eq. (21), we have

$$\begin{aligned}\mathcal{T}_{\text{total}} =& \mathcal{O}\left(K \cdot (D_{\text{state}} + C_{\text{out}}) \cdot C_{\text{out}} \cdot k_{\text{conv}} \cdot H\right) \\ &+ \mathcal{O}\left((ML)^2 \cdot d_{\text{act}}\right) + \mathcal{O}\left(ML \cdot (d_{\text{act}} + d_{\text{trans}}) \cdot N\right). \end{aligned} \tag{25}$$

2) Space Complexity Analysis: Next, we analyze the space complexity of CDMP & CDMP-pen. Generally, the runtime memory usage can be expressed as

$$\mathcal{S}_{\text{total}} = \mathcal{S}_\theta + \mathcal{S}_\phi, \tag{26}$$

where $\mathcal{S}_\theta$ and $\mathcal{S}_\phi$ represent the space complexity of ϵ_θ and f_ϕ, respectively. Based on the structural analysis in Section IV-E.1, $\mathcal{S}_\theta$ mainly arises from the residual blocks of the U-Net, while $\mathcal{S}_\phi$ is primarily attributed to the autoregressive generation layers. Accordingly, the overall space complexity can be approximated as

$$\begin{aligned}\mathcal{S}_{\text{total}} =& \mathcal{O}\left((D_{\text{state}} + C_{\text{out}}) \cdot C_{\text{out}} \cdot k_{\text{conv}}\right) \\ &+ \mathcal{O}\left((ML)^2 \cdot d_{\text{act}}\right) + \mathcal{O}\left(ML \cdot (d_{\text{act}} + d_{\text{trans}}) \cdot N\right), \end{aligned} \tag{27}$$

where the first term corresponds to $\mathcal{S}_\theta$, and the latter two terms collectively represent $\mathcal{S}_\phi$.

V. Performance Evaluation

In this section, we conduct numerical experiments to evaluate the performance of our algorithm and validate the effectiveness of the proposed framework. Additionally, we explore the influence of different hyperparameters on the algorithm's behavior to further demonstrate its robustness.

A. Experimental Settings

We represent the noise model ϵ_θ with a temporal U-Net [69], consisting of 6 repeated residual blocks. Timestep and condition embeddings are concatenated together and then added to the activations of the first temporal convolution within each block. The inverse dynamics model f_ϕ is developed as a self-regressive network generating *ML*-dimensional actions, where each dimension is predicted sequentially based on the previous ones.

We select OPNET as the simulation platform of a clustered wireless ad hoc network, where the environment spans a 10 km $\times$10 km area with randomly placed 16 nodes. Each node has a maximum transmission range of 6 km and can transmit packets in 4 channels at a rate of 2 Mbps. The traffic transmission stage of the frame consists of 10 time slots. We simulate diverse network loads by adjusting the ratios of high- to low-speed nodes from $4:12$, $6:10$, to $8:8$. Each node generates data packets following an exponential distribution, where low-speed nodes produce a fixed-size packet every 0.02 seconds on average, while high-speed nodes create a packet every 0.0025 to 0.0033 seconds depending on the high-to-low speed node ratio in the scenario, so as to balance overall traffic load and avoid excessive congestion in high-density settings. To collect the offline dataset, we allocate time slots based on these ratios to generate actions in multiple random communication contexts, recording environment data every frame. This yields a dataset of $1,814$ action-state trajectories labeled with rewards, each corresponding to a communication duration of 30 seconds (i.e., $6,000$ time-steps).

All algorithms including our proposed CDMP & CDMP-pen, MFRL approaches such as Behavior Cloning (BC) [31], Conservative Q-learning (CQL) [32], Implicit Q-learning (IQL) [33] and Decision Transformer (DT) [34], as well as the MBRL method Diffuser [35], are then trained on this offline dataset for 100 epochs, with each epoch comprising of $1,000$ training steps. During testing, we evaluate the training planners on the aforementioned OPNET platform under random communication scenarios and compare performance based on the average reward and QoS (e.g., average throughput, average delay, and packet loss rate) of 5-second episodes. We report the mean and the standard deviation over 3 random scenarios. Furthermore, the principal parameters used in simulations are summarized in Table II.

B. Performance Comparison

1) Numerical Results of CDMP: Firstly, we demonstrate the superiority of CDMP compared to other methods and plot the average reward curves during training, as shown in Fig. 4. Notably, the oracle represents the optimal strategy, where the ratio of high- to low-speed nodes is known in advance, and

TABLE II
LIST OF KEY PARAMETER SETTINGS FOR THE SIMULATION

Parameters Description	Value
Simulator	
Number of nodes	$N = 16$
Number of time slots in a frame	$M = 10$
Number of channels	$L = 4$
Coverage area of the simulation environment	10 km $\times$ 10 km
Maximum transmission range of each node	6 km
Packet transmission rate	2 Mbps
Ratios of high- to low-speed nodes	$4:12$, $6:10$, or $8:8$
Number of trajectories in the offline dataset	$1,814$
Communication duration of a trajectory in the offline dataset	30 s
Communication duration of an episode during testing	5 s
CDMP & CDMP-pen	
Diffusion steps	$K = 200$
Planning horizon	$H = 12$
Discount factor	$\gamma = 0.99$
Conditional guidance scale	$\omega = 1.6$
Conditional information removal probability	$\beta = 0.25$
OOD penalty weight	$\zeta = 1.0$
Neural Network Training	
Architecture of the noise model ϵ_θ	U-Net
Architecture of the inverse dynamics model f_ϕ	Self-regressive network
Dimension of diffusion step embedding in ϵ_θ	256
Dimension of condition embedding in ϵ_θ	256
Dimension of state-transition pair embedding in f_ϕ	256
Dimension of action embedding in f_ϕ	128
Base channel count of the temporal U-Net backbone	256
Kernel size of the temporal U-Net backbone	5
Batch size	128
Number of training epochs	$N_{\text{epoch}} = 100$
Number of training steps per epoch	$N_{\text{step}} = 1,000$
Learning rate	$1e-4$
Optimizer	Adam
EMA decay factor for model updates	0.995

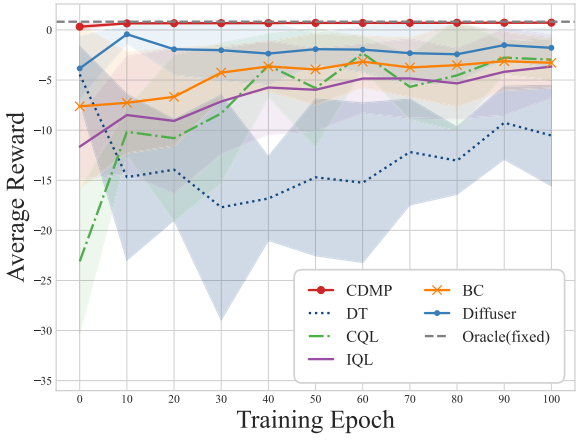


Fig. 4. Comparison of CDMP with different methods in terms of average reward.

time slots are allocated proportionally according to the ratio at the very beginning. It can be observed that CDMP achieves higher average rewards and approaches optimal performance after sufficient training. It also converges significantly faster

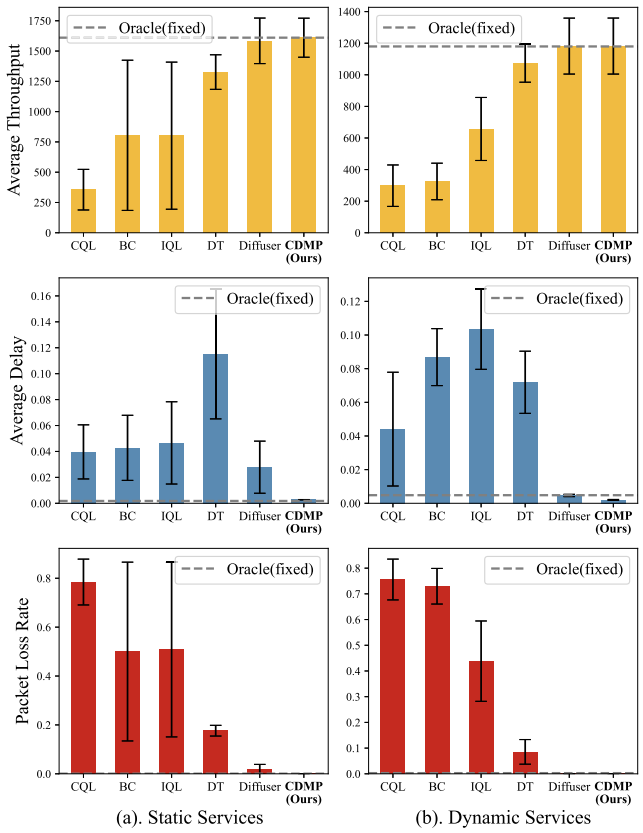


Fig. 5. Comparison of CDMP with different methods in terms of QoS. Specifically, static services represent a constant ratio of high- to low-speed nodes throughout the communication period. In contrast, dynamic services involve a changing ratio of high- to low-speed nodes over time.

with a smaller training variance, indicating a more stable training process.

Additionally, to further validate the effectiveness of CDMP, we compare the detailed QoS metrics, as shown in Fig. 5. The results for static services in Fig. 5(a) demonstrate that CDMP outperforms other methods in terms of average throughput, average delay, and packet loss rate. To better emulate real-world wireless environments characterized by service heterogeneity and non-stationary demands, we construct scenarios with dynamically shifting traffic patterns, where the high-to-low-speed node ratio changes randomly, averaging twice every 5 seconds. Fig. 5(b) indicates that under such fluctuating conditions, CDMP, which can adaptively adjust time slot allocation, even surpasses the fixed oracle strategy and more effectively accommodates the evolving traffic distributions.

2) Empirical Analysis of CDMP-Pen: To substantiate the potency of the proposed uncertainty penalty in addressing the distribution shift problem, we compare the performance of CDMP-pen and CDMP under varying interference duty cycles, as shown in Fig. 6. In the experiment, OOD scenarios are simulated by introducing random "interference" packets into two channels. The comparison reveals that as the interference duty cycle increases, both CDMP and CDMP-pen experience performance degradation. However, CDMP-pen exhibits a slower decline. Moreover, under the same interference level, CDMP-pen achieves a higher average reward, indicating

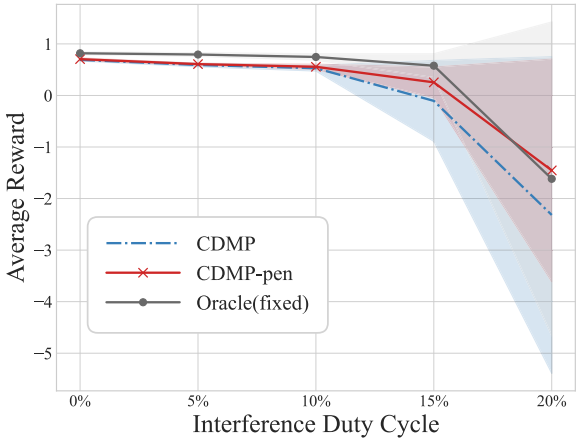


Fig. 6. Comparison of CDMP-pen and CDMP under different inference duty cycles in terms of average reward.

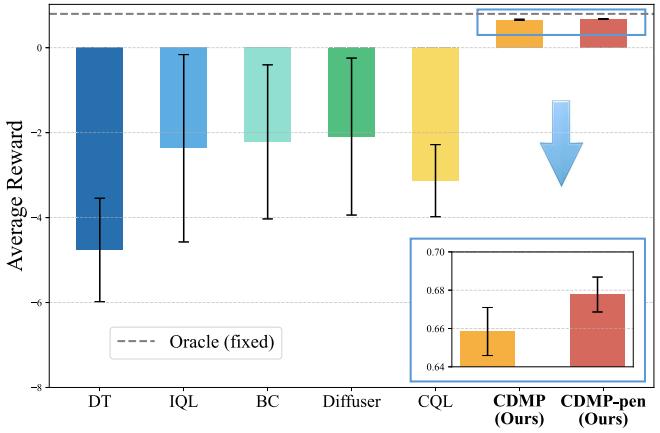


Fig. 7. Comparison of CDMP & CDMP-pen with different methods in the node mobility scenario in terms of average reward.

that the designed distance-to-data metric effectively quantifies the errors introduced by an imperfect dynamic model. This mechanism helps reduce model confidence in OOD regions, preventing high-risk decisions in unknown scenarios while enhancing policy robustness and generalization.

3) Generalization Capability Study: In the original offline dataset and prior evaluation scenarios, the positions of the nodes are predetermined and remain fixed during simulations. To further examine the algorithm's effectiveness under dynamic conditions, we divide the 16 nodes into four groups based on their initial locations and randomly assign each group a movement direction - north, south, east, or west - at a constant speed of 100 m/s. As illustrated in Fig. 7, our proposed methods consistently outperform the baseline algorithms in terms of both higher average reward and lower standard deviation, indicating superior adaptability and enhanced robustness to node mobility. Furthermore, the CDMP-pen variant demonstrates performance improvement over the original CDMP approach, providing stronger empirical evidence that the incorporation of an uncertainty-aware penalty term significantly enhances the algorithm's generalization capability in dynamic and previously unseen environments.

Additionally, we adopt the Free-Space Path Loss (FSPL) [70] model to simulate signal propagation in our wireless communication system. Besides ideal Radio Frequency (RF) conditions, where stable and reliable signal transmission could

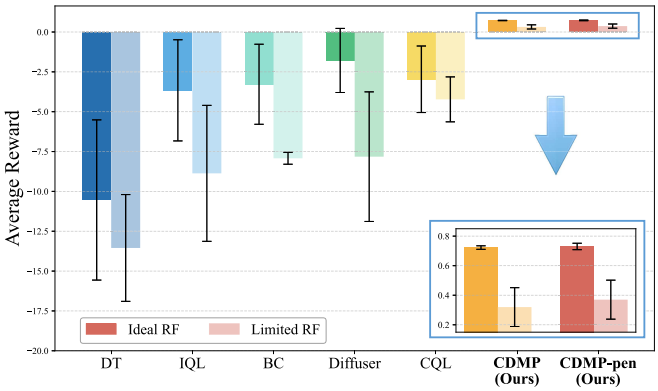


Fig. 8. Comparison of CDMP & CDMP-pen with different methods in the ideal and limited RF scenarios in terms of average reward.

be achieved, we further assess the model's generalization capability under more practical, limited RF conditions. Specifically, we increase the carrier frequency to induce higher propagation loss and simultaneously decrease the transmission power to reduce the effective communication range, thereby emulating the limited RF setting with signal degradation. As shown in Fig. 8, although all methods exhibit performance degradation in this harsher RF regime, our algorithms consistently maintain a significant performance margin over the baselines. This demonstrates the superior generalization ability across varying RF environments, highlighting the practical applicability of our algorithms in more complex and unpredictable wireless deployment scenarios.

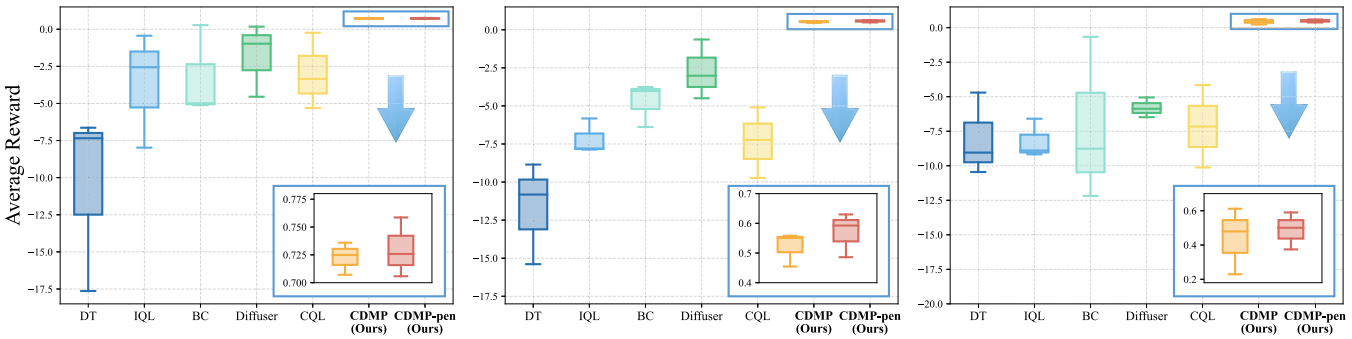
We also investigate the scalability of the proposed algorithms with respect to the number of nodes, aiming to evaluate the effectiveness in more complex and realistic network environments. We compare the performance of our methods against baseline algorithms under scenarios with 16, 20, and 24 nodes, as illustrated in Fig. 9. The results demonstrate that our algorithms consistently achieve superior performance across all settings. This highlights not only the robustness but also the strong generalization capability to varying node densities. The ability to maintain stable and effective performance as the network scales underscores the suitability of our algorithms for deployment in large-scale and dynamic wireless ad hoc networks, where adaptability and reliability are essential.

C. Ablation Studies

To evaluate the efficacy and simplicity of the reward function defined in Eq. (6), we design further experiments to investigate how different reward function definitions affect the algorithm's performance. Particularly, we re-define the QoS-weighted average reward function as

$$\mathcal{R}_t\text{-w} := \sum_{i=1}^{N} \left(\lambda \cdot u_t^{(i)} - d_t^{(i)} \right) - \eta \cdot l_t, \tag{28}$$

where $u_t^{(i)}$ represents the size of data packets successfully transmitted and received, while l_t denotes the ratio of lost packets to total sent packets in the network. The factors λ and η balance the weights of these metrics, aiming to promote high throughput while minimizing delay and packet loss across the entire communication system. The framework trained with this weighted reward is denoted as CDMP-w, and we compare



(a). $N = 16$ (b). $N = 20$ (c). $N = 24$

Fig. 9. Comparison of CDMP & CDMP-pen with different methods under varying node number scenarios in terms of average reward.

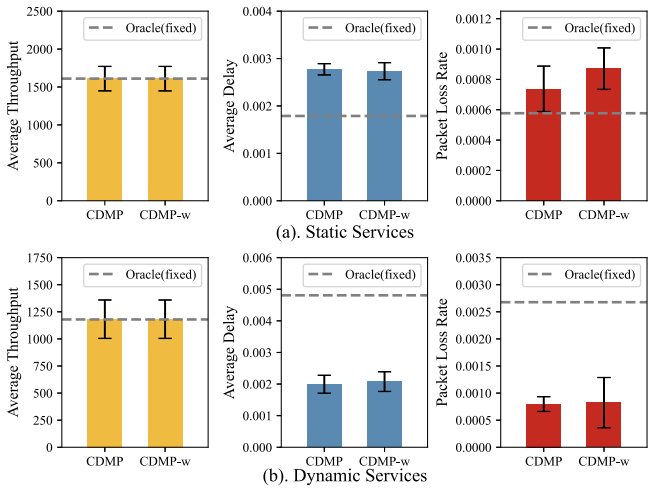


Fig. 10. Comparison of different reward function definitions in terms of QoS.

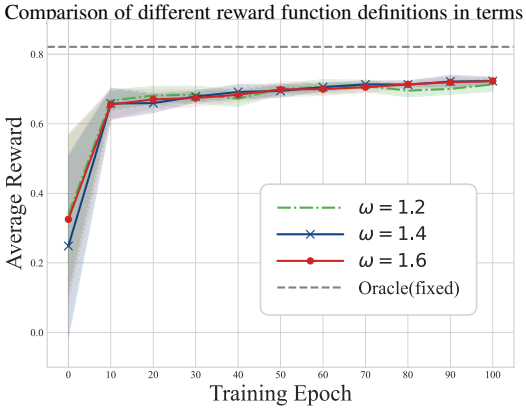


Fig. 11. Comparison of different conditional guidance scales ω in terms of average reward.

the QoS metrics with CDMP as illustrated in Fig. 10. As observed from the figure, using average delay alone as the reward function yields performance comparable to that of the QoS-weighted average reward function. Furthermore, it leads to a lower packet loss rate and a smaller standard deviation. This indicates that the simplified reward function not only circumvents the complexities of hyperparameter tuning but also contributes to improved model stability.

Next, we investigate the impact of conditional guidance scale ω on the algorithm's performance of CDMP. By varying ω, we track the average reward during training, as illustrated in Fig. 11. The results indicate that for different ω settings,

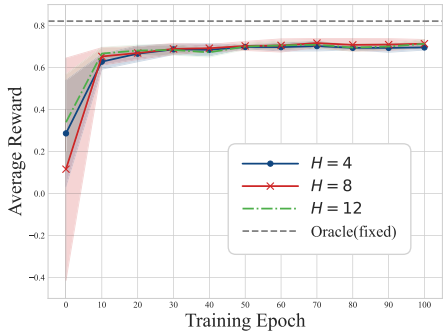


Fig. 12. Comparison of different planning horizons H in terms of average reward.

the model consistently converges to similar average reward values once completing sufficient training. This demonstrates the robustness and adaptability of the model with respect to variations in conditional guidance scalar, ensuring reliable results under varying conditions. However, small ω values may prevent the model from effectively capturing contextual cues, while excessively large ω values may cause the algorithm to rely too heavily on conditional guidance. These results suggest that a practical operational range of $\omega \in [1.2, 1.6]$ offers a good trade-off between conditioning effectiveness and model generality in our setting. Therefore, in real-world deployments, it is advisable to tune ω within this range to balance convergence speed, stability, and contextual expressiveness.

Furthermore, to verify the relationship between the planning horizon H and the performance of CDMP, we conduct relevant experiments to evaluate how different values of H affect average rewards. It can be observed from Fig. 12 that different planning horizons result in fluctuating model performance at the beginning of training. However, all settings rapidly converge to similar, high average rewards, indicating that the model maintains stable performance and strong reliability across a range of planning horizon configurations. Nonetheless, an excessively short horizon may neglect long-term rewards and future states, while longer horizons increase the computational complexity, impairing overall algorithm efficiency. These results suggest that an operational range of $H \in [4, 12]$ provides reliable performance in our scenario. In practice, the horizon length should be adapted based on the environment's temporal complexity and system resource

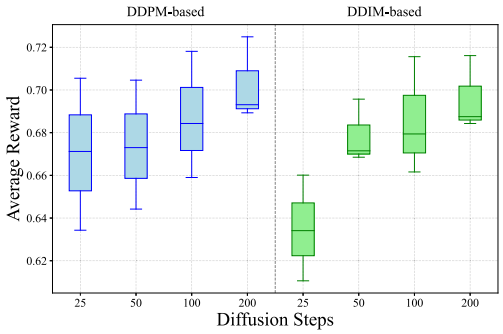


Fig. 13. Comparison of different diffusion steps K and the denoising sampling mechanisms in terms of average reward.

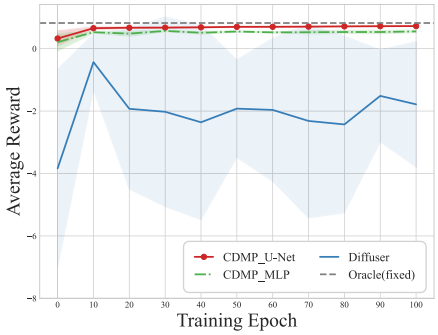


Fig. 14. Comparison of different diffusion noise model backbones in terms of average reward.

constraints to balance planning accuracy with computational efficiency.

We also conduct a more in-depth numerical analysis on the effects of diffusion steps K and the denoising sampling mechanism in CDMP, with results presented in Fig. 13. The sampling process is built upon the DDPM and DDIM [71] frameworks, where DDIM leverages a pre-trained DDPM-based noise model ($K = 200$) and follows a deterministic, non-Markovian strategy with uniform stride skipping to reduce iterations. Our comparison reveals that the DDPM-based model exhibits a steady increase in average reward as K grows, with a concurrent reduction in standard deviation, enhancing performance robustness. In contrast, the DDIM-based model results in lower average rewards with smaller K, likely due to reduced denoising refinement and incomplete noise removal. While DDIM benefits from faster sampling, training-free deployment, and consistent outputs due to its deterministic structure, this often leads to reduced sample diversity and suboptimal performance in complex decision-making tasks. Therefore, the choice of sampling framework and K should be carefully tailored to balance computational efficiency and policy performance. In our setting, we find that DDPM with $K \geq 100$ and DDIM with $K \geq 50$ offer stable and reliable results, while overly aggressive reduction in K leads to degraded QoS. To our knowledge, this is the first study comparing DDPM and DDIM sampling frameworks in decision-making applications of DMs.

Moreover, we examine how different diffusion noise model backbones affect the overall performance of our algorithm. As analyzed in Section IV-E.1, the U-Net backbone adopted in our main implementation is relatively complex. To improve decision-making efficiency, we explore a simplified variant by replacing the temporal U-Net with a 3-layer MLP, while keeping the diffusion step and condition embedding networks unchanged. The average reward during training is shown in Fig. 14. Although the MLP-based CDMP achieves slightly lower average rewards than the U-Net variant, it still approaches oracle-level performance after sufficient training. Notably, it consistently outperforms the strongest baseline, Diffuser, in terms of higher average reward, faster convergence, and lower variance, demonstrating superior reliability and stability. These results indicate that, in real-world deployments, the choice of noise model backbone should be guided by the specific trade-offs between decision quality and computational efficiency required by the application scenario.

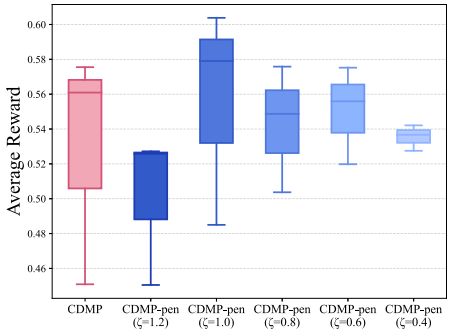


Fig. 15. Comparison of different OOD penalty weight factors ζ in terms of average reward.

In addition, we investigate the impact of the OOD penalty weight factor ζ in the CDMP-pen loss function defined in Eq. (19). Experiments are conducted under a scenario with a 10% interference duty cycle, where the average rewards of different weighting strategies are recorded, as illustrated in Fig. 15. Comparative analysis reveals that CDMP-pen models with moderate penalty weights, which impose penalties on unknown and hazardous states that deviate significantly from the offline dataset exhibit lower variance, demonstrate enhanced adaptability and robustness in dynamic environments. However, assigning excessively large weights (e.g., $\zeta = 1.2$) may lead to performance degradation, likely due to overfitting and overly conservative behavior. Conversely, small penalty values (e.g., $\zeta = 0.4$) fail to provide sufficient regularization, weakening the model's ability to avoid unsafe OOD actions. These findings highlight the importance of proper penalty calibration. Overall, our experiments suggest that an operational range of $\zeta \in [0.6, 1.0]$ offers a favorable balance between conservatism and exploration, guiding practitioners in parameter tuning for enhanced policy robustness and adaptability.

VI. Conclusion

In this paper, we have studied a centralized MBRL-based planning solution for MF-TDMA time slot scheduling strategy optimization. In particular, we have proposed the CDMP algorithm with a significant performance boost in terms of communication efficiency and effectiveness, addressing the challenges of resource allocation in clustered ad hoc networks. Specifically, we have utilized a conditional diffusion model to generate high-quality state sequences and employed an inverse dynamics model to predict the optimal actions. Afterward, we have introduced a dynamics uncertainty penalty to tackle the OOD problem in offline RL, extending it into a

CDMP-pen algorithm, which has been theoretically proven to effectively bound the true distribution error. We have demonstrated the superiority & robustness of our framework over several classical offline RL algorithms through extensive simulations.

There are many interesting directions to be addressed in the future. For example, our current simulation scenarios are limited to relatively simple and traditional MF-TDMA communication frameworks. Upon validating the proposed algorithm's effectiveness and advantages, future research could explore its integration with more sophisticated communication protocols. This would enable a more comprehensive investigation of the algorithm's generalization capabilities and adaptability, broadening its practical applications and contributing to further advancements in communication systems. Moreover, our framework relies on centralized resource scheduling. Thus, as the network scales up, it significantly increases the complexity of training. Given these challenges, it is worthwhile to explore distributed decision-making approaches. Through localized task allocation and parallel training, distributed strategies can facilitate more efficient learning of optimal policies at the individual level, ultimately alleviating the burden of centralized information exchange and accelerating convergence. In addition, since this study focuses on resource allocation within pre-defined clusters in a clustered ad hoc network, future work may consider jointly optimizing clustering strategies and resource assignment. By dynamically adjusting both the cluster structure and resource distribution, the proposed algorithm could better adapt to heterogeneous and time-varying network conditions. This line of investigation holds promise for improving the flexibility and robustness of the framework in complex and practical deployment environments.

APPENDIX A
PROOF FOR LEMMA 1

The perturbed data distribution can be written as a sum of Gaussians, since

$$\begin{aligned}\hat{q}_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}}) &:= \int \hat{q}(s;\mathcal{D}_{\text{stat}})\mathcal{N}(\widetilde{s};s,\sigma^2\mathbf{I})ds \\ &= \int \left[\frac{1}{M'}\sum_i \delta(s^i)\right]\mathcal{N}(\widetilde{s};s,\sigma^2\mathbf{I})ds \\ &= \frac{1}{M'}\sum_i \int \delta(s^i)\mathcal{N}(\widetilde{s};s,\sigma^2\mathbf{I})ds \\ &= \frac{1}{M'}\sum_i \mathcal{N}(\widetilde{s};s^i,\sigma^2\mathbf{I}). \end{aligned} \tag{29}$$

Then we consider the negative log of the perturbed data distribution multiplied by σ^2 as

$$\begin{aligned}-\sigma^2\log\hat{q}_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}}) &= -\sigma^2\log\left[\frac{1}{M'}\sum_i \mathcal{N}(\widetilde{s};s^i,\sigma^2\mathbf{I})\right] \\ &= -\sigma^2\log\left[\sum_i \mathcal{N}(\widetilde{s};s^i,\sigma^2\mathbf{I})\right] + \sigma^2\log M' \\ &= -\sigma^2\log\left[\frac{1}{\sqrt{(2\pi\sigma)^n}}\sum_i \exp\left(-\frac{1}{2\sigma^2}\|\widetilde{s}-s^i\|^2\right)\right] + \sigma^2\log M' \\ &= -\sigma^2\log\left[\sum_i \exp\left(-\frac{1}{2\sigma^2}\|\widetilde{s}-s^i\|^2\right)\right] \\ &\quad + \sigma^2\log M' + \frac{\sigma^2 n}{2}\log(2\pi\sigma) \\ &= -\sigma^2\text{LogSumExp}_i\left[-\frac{1}{2\sigma^2}\|\widetilde{s}-s^i\|^2\right] + C(M',n,\sigma) \\ &= \text{Softmin}_\sigma\left[\frac{1}{2}\|\widetilde{s}-s^i\|^2\right] + C(M',n,\sigma) \\ &= d_\sigma(\widetilde{s};\mathcal{D}_{\text{stat}})^2 + C(M',n,\sigma). \end{aligned} \tag{30}$$

Then the lemma comes. ∎

APPENDIX B
PROOF FOR THEOREM 1

Let L_e denote the local Lipschitz constant of the true error $e(s)$ valid over the state space $\mathcal{S}$, and define $s_c := \arg\min_{s^i\in\mathcal{D}_{\text{stat}}} \frac{1}{2}\| s-s^i \|_2^2$, i.e., the closest data-point. By definition, we have $e(s) \le e(s^i) + L_e\|s-s^i\|_2$. Consequently, the following holds:

$$\begin{aligned} e(s) &\le \min_{s^i\in\mathcal{D}_{\text{stat}}} \left[e(s^i) + L_e\|s-s^i\|_2\right] \\ &\le e(s_c) + L_e\|s-s_c\|_2 \\ &= e(s_c) + \sqrt{2}L_e\sqrt{\frac{1}{2}\|s-s_c\|_2^2} \\ &= e(s_c) + \sqrt{2}L_e\sqrt{\frac{1}{2}\min_{s^i\in\mathcal{D}_{\text{stat}}}\|s-s^i\|_2^2} \\ &\le e(s_c) + \sqrt{2}L_e\sqrt{-\sigma^2\log\left(\frac{1}{M'}\sum_i \exp\left(-\frac{1}{2\sigma^2}\|s-s^i\|_2^2\right)\right)} \\ &= e(s_c) + \sqrt{2}L_e\sqrt{d_\sigma(s;\mathcal{D}_{\text{stat}})^2 + C_2}, \end{aligned} \tag{31}$$

where $C_2 = \sigma^2\log M' - C_1$, with C_1 defined in Eq. (14). In the second line, we leverage the fact that as s_c is a feasible solution to the minimization problem in the first line, it serves as an upper bound for the optimal value. In the fifth line, we rely on convex analysis and the characteristics of the smoothed minimum, where for any vector $\mathbf{v} = [v_1,\cdots,v_n]^\top \in \mathbb{R}^n$, the inequality $\min\{v_1,\cdots,v_n\} \le -\frac{1}{\delta}\log\sum_{i=1}^n \exp(-\delta v_i) + \frac{\log n}{\delta}$ holds for any scaling parameter $\delta > 0$. Thus, we attain the theorem. ∎

REFERENCES

[1] M. G. Rubinstein, I. M. Moraes, M. E. M. Campista, L. H. M. K. Costa, and O. C. M. B. Duarte, "A survey on wireless ad hoc networks," in *Proc. Mobile Wireless Commun. Netw. (MWCN)*, Santiago, Chile, Aug. 2006, pp. 1–33.

[2] S. Zhang, Z. Ni, L. Kuang, C. Jiang, and X. Zhao, "Load-aware distributed resource allocation for MF-TDMA ad hoc networks: A multi-agent DRL approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 6, pp. 4426–4443, Nov. 2022.

[3] J. Guo et al., "ICRA: An intelligent clustering routing approach for UAV ad hoc networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2447–2460, Feb. 2023.

[4] X. Chen, G. Sun, T. Wu, L. Liu, H. Yu, and M. Guizani, "RANCE: A randomly centralized and on-demand clustering protocol for mobile ad hoc networks," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 23639–23658, Dec. 2022.

[5] B. Suman and L. C. Mangal, "A dynamic TDMA slot scheduling(DTSS) scheme for efficient channel allocation in tactical ad hoc networks," in *Proc. Int. Conf. Comput., Commun. Autom.*, Noida, India, May 2015, pp. 502–507.

[6] I. Rhee, A. Warrier, J. Min, and L. Xu, "DRAND: Distributed randomized TDMA scheduling for wireless ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 8, no. 10, pp. 1384–1396, Oct. 2009.

[7] J. Degrave et al., "Magnetic control of tokamak plasmas through deep reinforcement learning," *Nature*, vol. 602, no. 7897, pp. 414–419, Feb. 2022.

[8] J. Wu, Z. Huang, and C. Lv, "Uncertainty-aware model-based reinforcement learning: Methodology and application in autonomous driving," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 1, pp. 194–203, Jan. 2023.

[9] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[10] T. Lillicrap et al., "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2016.

[11] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[12] Q. Huang, "Model-based or model-free, a review of approaches in reinforcement learning," in *Proc. Int. Conf. Comput. Data Sci. (CDS)*, Stanford, CA, USA, Aug. 2020, pp. 219–221.

[13] Y. Hu, J. Fu, and G. Wen, "Safe reinforcement learning for model-reference trajectory tracking of uncertain autonomous vehicles with model-based acceleration," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 3, pp. 2332–2344, Mar. 2023.

[14] C. You, J. Lu, D. Filev, and P. Tsiotras, "Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning," *Robot. Auto. Syst.*, vol. 114, pp. 1–18, Apr. 2019.

[15] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum, "OPAL: Offline primitive discovery for accelerating offline reinforcement learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2021, pp. 14975–14998.

[16] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Proc. Adv. Neural Inf. Proces. Syst.*, vol. 31, Dec. 2018, pp. 4754–4765.

[17] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[18] S. Lange, T. Gabel, and M. Riedmiller, *Reinforcement Learning: State-of-the-Art*. Berlin, Germany: Springer, 2012.

[19] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Stockholm, Sweden, Jul. 2019, pp. 2052–2062.

[20] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," 2020, *arXiv:2005.01643*.

[21] T. Yu et al., "MOPO: Model-based offline policy optimization," in *Proc. Adv. Neural Inf. Proces. Syst.*, Dec. 2020, pp. 14129–14142.

[22] F.-M. Luo, T. Xu, X. Cao, and Y. Yu, "Reward-consistent dynamics models are strongly generalizable for offline reinforcement learning," 2023, *arXiv:2310.05422*.

[23] Y. Du and I. Mordatch, "Implicit generation and modeling with energy based models," in *Proc. Adv. Neural Inf. Proces. Syst.*, vol. 32, Dec. 2019, pp. 3603–3613.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Scottsdale, AZ, USA, May 2013.

[25] R. Labaca-Castro, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Proces. Syst. (NeurIPS)*, Montréal, QC, Canada, Dec. 2023, pp. 73–76.

[26] T. B. Brown et al., "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Proces. Syst. (NeurIPS)*, Dec. 2020, pp. 1877–1901.

[27] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, Jun. 2022, pp. 10674–10685.

[28] A. Babazadeh Darabi and S. Coleri, "Diffusion model based resource allocation strategy in ultra-reliable wireless networked control systems," *IEEE Commun. Lett.*, vol. 29, no. 1, pp. 85–89, Jan. 2025.

[29] T. Liu, X. Fang, and R. He, "Deep diffusion deterministic policy gradient based performance optimization for Wi-Fi networks," 2024, *arXiv:2404.15684*.

[30] Z. Liu et al., "DNN partitioning, task offloading, and resource allocation in dynamic vehicular networks: A Lyapunov-guided diffusion-based reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 24, no. 3, pp. 1–17, Mar. 2025.

[31] W. Farag and Z. Saleh, "Behavior cloning for autonomous driving using convolutional neural networks," in *Proc. Int. Conf. Innov. Intell. Informat., Comput., Technol. (3ICT)*, Sakhier, Bahrain, Nov. 2018, pp. 1–7.

[32] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative Q-learning for offline reinforcement learning," in *Proc. Adv. Neural Inf. Proces. Syst. (NeurIPS)*, Dec. 2020, pp. 1179–1191.

[33] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit Q-learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Apr. 2022, pp. 20876–20887.

[34] L. Chen et al., "Decision transformer: Reinforcement learning via sequence modeling," in *Proc. Adv. Neural Inf. Proces. Syst. (NeurIPS)*, Dec. 2021, pp. 15084–15097.

[35] M. Jänner, Y. Du, J. B. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jul. 2022, pp. 9902–9915.

[36] C. Zhu and M. S. Corson, "A five-phase reservation protocol (FPRP) for mobile ad hoc networks," in *Proc. 17th Annu. Joint Conf. IEEE Comput. Commun. Societies*, vol. 1, San Francisco, CA, USA, Mar. 1998, pp. 322–331.

[37] I. Jabandzic, S. Giannoulis, R. Mennes, F. A. P. De Figueiredo, M. Claeys, and I. Moerman, "A dynamic distributed multi-channel TDMA slot management protocol for ad hoc networks," *IEEE Access*, vol. 9, pp. 61864–61886, 2021.

[38] G. Wi, S. Son, and K.-J. Park, "Delay-aware TDMA scheduling with deep reinforcement learning in tactical MANET," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2020, pp. 370–372.

[39] S. Chilukuri, G. Piao, D. Lugones, and D. Pesch, "Deadline-aware TDMA scheduling for multihop networks using reinforcement learning," in *Proc. IFIP Netw. Conf. (IFIP Networking)*, Espoo and Helsinki, Finland, Jun. 2021, pp. 1–9.

[40] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki, "Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation," in *Proc. Conf. Robot Learn.*, Atlanta, Georgia, Nov. 2023, pp. 2323–2339.

[41] H. He et al., "Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning," in *Proc. Adv. Neural Inf. Proces. Syst.*, New Orleans, LA, USA, Dec. 2023, pp. 64896–64917.

[42] H. Du et al., "Enhancing deep reinforcement learning: A tutorial on generative diffusion models in network optimization," *IEEE Commun. Surveys Tuts.*, vol. 26, no. 4, pp. 2611–2646, May 2024.

[43] R. Zhang, K. Xiong, X. Tian, Y. Lu, P. Fan, and K. B. Letaief, "Inverse reinforcement learning meets power allocation in multi-user cellular networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, May 2022, pp. 1–2.

[44] Y. Liu et al., "Deep generative model and its applications in efficient wireless network management: A tutorial and case study," *IEEE Wireless Commun.*, vol. 31, no. 4, pp. 199–207, Aug. 2024.

[45] P. Ning et al., "Diffusion-based deep reinforcement learning for resource management in connected construction equipment networks: A hierarchical framework," *IEEE Trans. Wireless Commun.*, vol. 24, no. 4, pp. 2847–2861, Apr. 2025.

[46] 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. (ICML)*, Stockholm, Sweden, Jul. 2018, pp. 1861–1870.

[47] T. Rashid, M. Samvelyan, C. S. D. Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *J. Mach. Learn. Res.*, vol. 21, no. 178, pp. 1–51, Aug. 2020.

[48] T. Wang and J. Ba, "Exploring model-based planning with policy networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Apr. 2020, pp. 8647–8668.

[49] M. P. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Washington, DC, USA, Jun. 2011, pp. 465–472.

[50] M. Jänner, Q. Li, and S. Levine, "Offline reinforcement learning as one big sequence modeling problem," in *Proc. Adv. Neural Inf. Proces. Syst.*, Dec. 2021, pp. 1273–1286.

[51] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, "COMBO: Conservative offline model-based policy optimization," in *Proc. Adv. Neural Inf. Proces. Syst. (NeurIPS)*, Dec. 2021, pp. 28954–28967.

[52] K. Jiang, J.-Y. Yao, and X. Tan, "Recovering from out-of-sample states via inverse dynamics in offline reinforcement learning," in *Proc. Adv. Neural Inf. Proces. Syst.*, New Orleans, LA, USA, Dec. 2023, pp. 38815–38826.

[53] Y. Sun, J. Zhang, C. Jia, H.-C. Lin, J. Ye, and Y. Yu, "Model-Bellman inconsistency for model-based offline reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, Honolulu, HI, USA, Jul. 2023, pp. 33177–33194.
[54] M. Jänner, J. Fu, M. Zhang, and S. Levine, "When to trust your model: Model-based policy optimization," in *Proc. Adv. Neural Inf. Proces. Syst.*, Dec. 2019, pp. 12519–12530.
[55] S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks, "Deep generative modelling: A comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7327–7347, Nov. 2022.
[56] C. Luo, "Understanding diffusion models: A unified perspective," 2022, *arXiv:2208.11970*.
[57] J. Ho, A. N. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Proces. Syst. (NeurIPS)*, Dec. 2020, pp. 6840–6851.
[58] A. Ajay, Y. Du, A. Gupta, J. B. Tenenbaum, T. Jaakkola, and P. Agrawal, "Is conditional generative modeling all you need for decision-making?," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2023, pp. 29598–29620.
[59] C. Allen, N. Parikh, O. Gottesman, and G. Konidaris, "Learning Markov state abstractions for deep reinforcement learning," in *Proc. Adv. Neural Inf. Proces. Syst.*, Dec. 2021, pp. 8229–8241.
[60] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," in *Proc. Adv. Neural Inf. Proces. Syst.*, Dec. 2021, pp. 8780–8794.
[61] J. Ho and T. Salimans, "Classifier-free diffusion guidance," 2022, *arXiv:2207.12598*.
[62] M. Laskey, J. Lee, R. Fox, A. D. Dragan, and K. Goldberg, "DART: Noise injection for robust imitation learning," in *Proc. Conf. Robot Learn. (CoRL)*, Mountain View, California, Nov. 2017, pp. 143–156.
[63] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Proc. Adv. Neural Inf. Proces. Syst. (NeurIPS)*, Dec. 2019, pp. 11918–11930.
[64] H. Zhang, J. Shao, Y. Jiang, S. He, G. Zhang, and X. Ji, "State deviation correction for offline reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, Vancouver, BC, Canada, Jun. 2022, pp. 9022–9030.
[65] H. J. T. Suh, G. Chou, H. Dai, L. Yang, A. Gupta, and R. Tedrake, "Fighting uncertainty with gradients: Offline reinforcement learning via diffusion score matching," in *Proc. Conf. Robot Learn. (CoRL)*, Atlanta, Georgia, Nov. 2023, pp. 2878–2904.
[66] J. E. Marsden and M. J. Hoffman, *Elementary Classical Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1974.
[67] S. Coles, *An Introduction to Statistical Modeling of Extreme Values*. London, U.K.: Springer, 2001.
[68] C. Knuth, G. Chou, N. Ozay, and D. Berenson, "Planning with learned dynamics: Probabilistic guarantees on safety and reachability via Lipschitz constants," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5129–5136, Jul. 2021.
[69] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Med. Image Comput. Comput. Assist. Interv. (MICCAI)*, Munich, Germany, Oct. 2015, pp. 234–241.
[70] H. T. Friis, "A note on a simple transmission formula," *Proc. IRE*, vol. 34, no. 5, pp. 254–256, May 1946.
[71] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2021, pp. 14205–14225.

Kechen Meng (Student Member, IEEE) received the B.E. degree in information engineering from Zhejiang University, Hangzhou, China, in 2024, where she is currently pursuing the Ph.D. degree with the College of Information Science and Electronic Engineering. Her research interests include diffusion model, deep reinforcement learning, and wireless communication.

Sinuo Zhang (Graduate Student Member, IEEE) received the B.E. degree in electronic science and technology from Zhejiang University, Hangzhou, China, in 2023, where she is currently pursuing the M.E. degree with the College of Information Science and Electronic Engineering. Her research interests include wireless communications, ad hoc networks, and deep reinforcement learning.

Rongpeng Li (Senior Member, IEEE) is currently an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. He was a Research Engineer with the Wireless Communication Laboratory, Huawei Technologies Company, Ltd., Shanghai, China, from August 2015 to September 2016. He was a Visiting Scholar with the Department of Computer Science and Technology, University of Cambridge, Cambridge, U.K., from February 2020 to August 2020. His research interest currently focuses on networked intelligence for comprehensive efficiency (NICE). He received the Wu Wenjun Artificial Intelligence Excellent Youth Award in 2021. He serves as an Editor for *China Communications*.

Chan Wang (Member, IEEE) received the B.Eng. degree in information and communication engineering and the Ph.D. degree in microelectronics and solid state electronics from Zhejiang University in 2012 and 2017, respectively. Since October 2017, she has been working with Zhejiang University, where she is currently a Post-Doctoral Researcher with the College of Information Science and Electronic Engineering. Her research interests include channel coding, cooperative communication, machine learning for wireless communications, and ad hoc networks.

Ming Lei received the B.Eng. and Ph.D. degrees in information and communication engineering from Zhejiang University in 2006 and 2012, respectively. He is currently an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University. His research interests include anti-jamming ad hoc networks, routing strategy, channel estimation and equalization, cooperative communication, and machine learning for wireless communication.

Zhifeng Zhao (Senior Member, IEEE) received the B.E. degree in computer science, the M.E. degree in communication and information systems, and the Ph.D. degree in communication and information systems from the PLA University of Science and Technology, Nanjing, China, in 1996, 1999, and 2002, respectively. From 2002 to 2004, he was a Post-Doctoral Researcher with Zhejiang University, Hangzhou, China, where his researches were focused on multimedia next-generation networks (NGNs) and softswitch technology for energy efficiency. From 2005 to 2006, he was a Senior Researcher with the PLA University of Science and Technology, where he performed research and development on advanced energy-efficient wireless router, ad-hoc network simulator, and cognitive mesh networking test-bed. From 2006 to 2019, he was an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University. Currently, he is with Zhejiang Laboratory, Hangzhou, as the Chief Engineering Officer. His research interests include software defined networks (SDNs), wireless network in 6G, computing networks, and collective intelligence. He is the Symposium Co-Chair of ChinaCom 2009 and 2010. He is the Technical Program Committee (TPC) Co-Chair of the 10th IEEE International Symposium on Communication and Information Technology (ISCIT 2010).