Efficient and Scalable Reinforcement Learning for Large-scale Network Control

Chengdong Ma[1,*], Aming Li[2,5,*], Yali Du[3,*], Hao Dong[4], and Yaodong Yang[1,†]

[1]Institute for Artificial Intelligence, Peking University, Beijing, China.
[2]Center for Systems and Control, College of Engineering, Peking University, Beijing, China.
[3]King's College London, London, UK.
[4]CFCS, School of CS, Peking University, Beijing, China.
[5]Center for Multi-Agent Research, Institute for Artificial Intelligence, Peking University, Beijing, China.
[*]These authors contributed equally to this work.
[†]Corresponding author with yaodong.yang@pku.edu.cn

ABSTRACT

The primary challenge in the development of large-scale AI systems lies in achieving scalable decision-making – extending the AI models while maintaining sufficient performance. Existing research indicate that distributed AI can improve scalability by decomposing complex tasks and distributing them across collaborative nodes. However, prior technologies suffered from compromised real-world applicability and scalability due to the massive requirement of communication and sampled data. Here we develop Model-based Decentralized Policy Optimization framework, which can be efficiently deployed in multi-agent systems. By leveraging local observation through the agent-level topological decoupling of global dynamics, we prove that this decentralized mechanism achieves accurate estimations of global information. Importantly, we further introduce model learning to reinforce the optimal policy for monotonic improvement with a limited amount of sampled data. Empirical results on diverse scenarios demonstrate the superior scalability of our approach, particularly in real-world systems with hundreds of agents, thereby paving the way for scaling up AI systems.

Introduction

Achieving scalable decision-making becomes a critical challenge when deploying AI models into large-scale systems[1]. Firstly, this requires effective information exchange among system entities to help an agent perceive the state of the environment and other agents[2]. However, due to constraints and high costs associated with communication[3], achieving comprehensive information sharing across the entire system becomes unfeasible (Fig. 1a). For example, in traffic networks, frequent and extensive communication between traffic lights can cause a significant power loss (Fig. 1b). In some systems involving user information, centralized information collection also increases the risk of privacy leakage[4]. Secondly, efficient sample acquisition and learning from limited data are necessary because the cost of agent-environment interaction rises exponentially with the scale of the system[5], and in some scenarios, only limited interactions are permitted. Therefore, developing an effective communication mode and a sample-efficient approach is crucial for achieving scalable decision-making.

As an advanced paradigm of distributed AI, Multi-Agent Reinforcement Learning (MARL) provides a possible solution[6,7] and has made progress in various scenarios, including autonomous driving[8–12], wireless communications[13,14], multi-player games[15,16], power systems[17,18] and urban transportation[19–24]. The advantage of MARL lies in its ability to perform nonlinear fitting solely by data and attain efficient inference. In contrast, traditional methods such as Model Predictive Control (MPC) require precise system dynamics[25,26], which are often difficult to obtain in complex systems. Even with accurate dynamics, traditional methods often rely on system linearization[27], which inevitably harms performance by disregarding numerous intricate nonlinear factors and system perturbations inherent in complex systems. Additionally, deficiencies in computational efficiency, numerical stability and communication costs are also difficult to avoid for traditional methods[28]. Therefore, we focus on MARL approaches to realize large-scale AI systems.

However, despite its numerous advantages, developing efficient and scalable MARL algorithm applicable to real-world scenarios remains challenging[29–36]. Previous works towards alleviating information exchange have been shown insufficient because they rely on impractical assumptions and redundant communication mechanisms, resulting in inferior adaptability to complex systems compared to our framework. Specifically, one direction involves connecting agents through a network topology and executing decentralized control by neighboring information[21,37–39]. Based on this setting, existing methods have achieved policy learning convergence by assuming linear approximation of the value functions[40], or assuming that the system dynamics are fully independent and decouplable[39]. By contrast, we discard this assumption to adapt to more generalized systems. Additionally, some insightful research adopt the truncated policy gradient method to optimize local policies and provide guarantees for policy improvement where communication is limited[41,42]. We further extend the concept of truncated Q-learning[41] from tabular case to deep MARL, thereby factoring global Markov Decision Processes (MDPs) into marginal transition to reduce system complexity[43]. This extension provides theoretical guarantees for policy improvement at each step in general systems. Another direction involves learning hidden information through computational communication protocols[21,44–47], whereas our approach only requires neighbor states. This indicates that each agent needs less information for learning and inference[21].

Furthermore, high sample efficiency is also crucial yet unaddressed, rendering current methods only effective in small-scale or simulated environments[48–51]. Model-based methods are widely studied as a promising approach for improving sample efficiency[52–57], as they can leverage estimated dynamic models to execute efficient interactions in for acquisition of policy learning data[53,58]. This is particularly helpful for the application of MARL in large-scale systems with constrained interactions (Fig. 1b). However, prior research on model-based MARL is primarily restricted to specific scenarios, e.g. two-player zero-sum games[59,60], the pursuit-evasion game[61], or tabular tasks[62]. Some works formulate opponent models or dynamics in a decentralized structure[63], but these approaches assume full state observability, and their scalability has not been sufficiently verified. Subsequently, some researchers extended model-based prioritized sweeping to a MARL scenario[62] but were only limited to the tabular MARL setting, thus unable to address more general tasks. In summary, existing MB-MARL methods are difficult to achieve scalable decision-making due to the following limitations[64]: (1) Despite being rooted in local models, they still rely on a global critic, undermining their effectiveness in communication-constrained scenarios[65–67]. (2) Theoretical analysis and bounded estimation for multi-agent model learning are lacking, resulting in an insufficient understanding of the model's impact on MARL[66,68]. (3) The integration of model learning and policy learning in MARL scenarios, a potential iterative process in real-world scenarios, has not been achieved[66].

Here we surpass the existing research bottleneck and take a significant step toward a decision paradigm that is scalable to large-scale systems. Firstly, we propose the ξ-dependent networked MDP, which possesses the Markovian property of transition. This is a form of Decentralized Partially Observable MDP (Dec-POMDP), and its advantage lies in its ability to deal with more general partially observable cooperative multi-agent tasks. This concept bridges the gap between standard network systems and general multi-agent systems, providing the necessary theoretical framework and analytical tools for research into multi-agent systems. Secondly, building upon ξ-dependent network systems, we propose the first decentralized model-based framework for networked systems, called Model-based Decentralized Policy Optimization. Our method is efficient and scalable, enabling overall performance improvement of the general system even when individual agents have severely limited access to information retrieval (our method can achieve the minimum local message size $O(|s_i|)$). In our method, we couple localized model learning with decentralized policy optimization. Specifically, we use well-learned localized models to predict future states and use local communication to broadcast their predictions. To alleviate the issue of compounding model error, we adopt a branching strategy[55,69] by replacing few long-horizon rollouts with many short-horizon rollouts to reduce compounding errors in model-generated rollouts. In the policy optimization part, we perform decentralized Proximal Policy Optimization (PPO)[70] using data generated by localized models and incorporating extended value function. Theoretically, we prove that the policy gradient computed from the extended value function is a close approximation to the true gradient, establishing the theoretical bridge between model learning and monotonic policy improvement. Our theoretical analysis begins with extending the fully independent systems[41,71] to more realistic and general multi-agent systems, and expanding single-agent model learning theory[55] to multi-agent systems.

Empirically, we evaluate our algorithms in highly realistic simulators and real-world scenarios in a progressive manner, encompassing various systems such as transportation, electric energy and social healthcare. Specifically, the evaluation scenarios involved: (1) Cooperative Adaptive Cruise Control (CACC)[21] and connected autonomous vehicle control (Flow)[72], (2) Adaptive Traffic Signal Control (ATSC), including the construction of a traffic network using real maps of Monaco[21] and New York, (3) IEEE Power-Grid[73] and the real power systems constructed using real-time electricity consumption data from Portugal[74], and (4) the pandemic networks constructed by epidemiologists based on real epidemiological dynamics, COVID-19 prevention and control policies, and infection status from real Swedish government data[75]. These tasks accurately and comprehensively simulate the complexities and challenges that existed in large-scale networked systems, with agent counts reaching 199 and 436. Importantly, although some MARL methods can handle tasks involving large-scale multi-agent systems (500 agents and above)[76–78], these approaches often simplify the complexity of large-scale problems through agent grouping[77,78] or mean-field approximations[76]. The problem setting remains approximately centralized and cannot address decentralized multi-agent learning in communication-limited scenarios. In contrast, our method is decentralized. Therefore, it is necessary to qualify our statements to indicate that our method significantly surpasses previous decentralized MARL methods in terms of the number of agents it can handle.

Model-based Decentralized Policy Optimization

In the general framework of our algorithm (Algorithm 1), three key components need to be specified: (1) how to perform decentralized policy optimization, (2) how to query localized models, and (3) to what extent can we approximate the true policy gradients? To train the models and policies, we maintain two experience buffers, $\mathscr{D}^E$ for trajectories generated by true environment p and $\mathscr{D}^M$ for data generated by the model. The trajectories for agent i consist of $(s_i, a_i, s_i', r_i, d_i)$, $i \in \mathscr{V}$, where $\mathscr{V} = \{1, ..., n\}$ is the set of agents, s_i is state, a_i is action, s_i' is next state, r_i is reward and d_i is binary, indicating whether the task is completed or reaches maximum episode length. The architecture of our framework is presented in Fig. 2a. Below we present the details of model learning and policy improvement.

For each agent i, let π_i^* denote the optimal policy and $V_i(s)$ denote the critic. The parameterized policy π^{θ_i} and critic V^{ϕ_i} are used to approximate these functions. Let $\{(s_{i,\tau}, a_{i,\tau}, r_{i,\tau})\}_{i\in\mathscr{V},\tau\in\mathscr{B}}$ represent a mini-batch of samples from $\mathscr{D}^M$ generated by policies $\pi^{\theta_i}, i \in \mathscr{V}$, where τ is the collected interaction trajectory and $\mathscr{B}$ represents a mini-batch sampled from the experience buffer. As it is not easy to obtain the true value $V_i(s)$, we adopt the extended value function[39], which is defined as

$$V_i(s_{N_i^\kappa}) = \mathbb{E}_{s_{N_{-i}^\kappa}} \left[\sum_{t=0}^{\infty} r_i^t | s_{N_i^\kappa}^0 = s_{N_i^\kappa} \right], i \in \mathscr{V}. \tag{1}$$

Let N_i^κ denote the κ-hop neighbor of agent i.. Note that $V_i(s_{N_i^\kappa})$ is a good approximation of $V_i(s)$, with the discrepancy decreasing exponentially with κ (according to Theorem 3). To generate the objective for the extended value function, or return R_i, we use reward-to-go technique[79]. The target of $V_i(s_{N_i^\kappa}^t)$ is

$$R_i^t = \sum_{l=0}^{T-t-1} \gamma^l r_i^{t+l} + V^{\phi_i}(s_{N_i^\kappa}^T). \tag{2}$$

where $\gamma \in (0,1)$ is the temporal discount factor and T is the length of each episode. The loss for learning value function approximation is defined as

$$\mathscr{L}(\phi_i) = \frac{1}{|\mathscr{B}|} \sum_{\tau\in\mathscr{B}} \left(V^{\phi_i}(s_{N_i^\kappa}^\tau) - R_i^\tau \right)^2. \tag{3}$$

Empirically, we use communication within κ-hop neighbors to generate an estimation of the global value function, denoted as

$$\tilde{V}_i(s_{N_i^\kappa}^t) = \frac{1}{n} \sum_{j\in N_i^\kappa} V_j(s_{N_j^\kappa}^t) \tag{4}$$

The advantage is thus defined as $\hat{A}_t(s_{N_i^\kappa}^t, a_i^t) = \mathbb{E}_{s_{N_i^\kappa}^{t+1}} \left[r_i^t(s_{N_i^\kappa}^t, a_i^t) + \gamma \tilde{V}_i(s_{N_i^\kappa}^{t+1}) - \tilde{V}_i(s_{N_i^\kappa}^t) \right]$. We adopt a PPO agent[70] in implementations. The loss function of an agent in our method is defined as:

$$\mathscr{L}(\theta_i) = \frac{1}{|\mathscr{B}|} \sum_{\tau\in\mathscr{B}} \big(- \frac{\pi^{\theta_i}(a_{i,\tau}|s_{N_i,\tau})}{\pi^{\theta_i^{\text{old}}}(a_{i,\tau}|s_{N_i,\tau})} \hat{A}_{i,\tau} + \beta H(\pi^{\theta_i}) \big). \tag{5}$$

where $H(\pi^{\theta_i})$ represents the policy entropy function used to compute the entropy of the policy π^{θ_i}, and β is the constant coefficient corresponding to the policy entropy.

In general, larger κ leads to higher performance on policy training, but also more communication costs. In algorithms, computation cost cannot be overlooked, as larger κ results in more complex models or policy architectures, thereby complicating the training process. We discuss more about this problem in the experiments section.

Update Model. To perform decentralized model-based learning, each agent maintains a localized model. This localized model observes the state of its κ-hop neighbors and its own action, with the goal of predicting the next timestep's information, including state and reward. Let $\widehat{s}_{i,t+1} = p^{\psi_i}(s_{N_i,t}, a_i)$. In practice, each agent stores the data locally. The models are updated by minimizing the following objective function:

$$\mathscr{L}(\psi_i) = \frac{1}{|\mathscr{B}|} \sum_{\tau\in\mathscr{B}} \|\widehat{s}_{i,\tau+1} - s_{i,\tau+1}\|^2. \tag{6}$$

Scaling model-based methods to real tasks can lead to decreased performance, even if the model itself is relatively accurate. A primary reason for this is the compounding modeling error during long model rollouts, where errors accumulate along the rollout trajectory, ultimately making it inaccurate. To mitigate the negative effects of model error, we adopt a branched rollout scheme[55,69]. In branched rollout, the model rollout does not start from an initial state, but from a state randomly selected from the most recent environmental trajectory τ. Additionally, the model rollout length is fixed to T. The model training steps are described in lines 4-7 of Algorithm 2.

Results

In this section, we will introduce the results of our method and baselines (more details in Supplementary Table 1) in the control of large-scale networked systems (more description in Supplementary Section 2). We have solved many control problems in simulators and real-world systems (more details in Supplementary Table 2). Baselines consist of two main categories: model-based algorithm such as MAG[65], Independent Model-based Policy Optimization (IMPO), MPO$_{na}$ and model-free algorithms CommNet[47], IC3Net[80], DIAL[45], NeurComm[21], ConseNet[40], FPrint[81], CPPO and DPPO, where IMPO is a fully independent MB-MARL method, MPO$_{na}$ ablated disconnected agents in our method (Fig. 1b) and our method with linear models (Ours-lin) ablated linear models in our method (more details in Supplementary Fig. 1). Overall, our method has decentralized models, policies, and value functions, with the minimum local message size. Especially compared to the centralized methods, the communication costs of our method in different scenarios is only about 5%-35% of that in centralized methods (more details in Supplementary Table 3).

Training Results and Ablation Study

Learning process. We first focus on the convergence of the episode reward and sample efficiency in the learning process. Figure 3e shows that our method significantly improves sample efficiency and achieves highest sample efficiency, converging within $1e5$ training steps, exceedingly outperforms baselines (more results in Supplementary Fig. 2). The reason is that the usage of the model will greatly increase the amount of experience to support training and improve the exploration efficiency (corresponding to line 9-12 in Algorithm 2). Our algorithm in Fig. 3d generates more extensive state-visited, further confirming this point. Therefore, the experimental results are consistent with the theoretical results. From the results in different tasks, we observe that the performance of CPPO does not exceed that of the algorithms that use extended value functions. In this way, we conclude that by using an extended value function, a centralized algorithm can be decomposed into a decentralized algorithm, but the performance would not drop significantly. According to Theorem 3 and Theorem 4, the extended value function is accurate enough to evaluate the overall multi-agent system and support the training process. Therefore, the experimental results are consistent with the theoretical results. Compared to the two model-based baselines IMPO and MPO$_{na}$, we conclude that the existence of the extended value increased the estimation accuracy of the global value function and gradient. Although theoretical analysis of our method in Theorem 3 and Theorem 4 does not provide more restrictions on whether the neighbor selection is adjacent, in the actual experiment, we recommend using information of adjacent neighbors in the topology of the networked systems to predict the global value and gradient, which can reduce the variance of the learning process and improve the performance. It is worth noting that MAG models the learning process of model errors as MARL tasks, resulting in lower model errors in certain scenarios. However, MAG still relies on global modeling information during the learning phase. In the model prediction phase, the traditional control algorithm MPC is utilized in the model prediction phase, thus having a higher computational complexity and communication cost.

In terms of optimal performance, Figure 3a shows that our method reaches the highest performance with the minimum local message size among all the baselines. Due to the fact that CPPO is a fully centralized algorithm, where each agent obtains global complete information, it has the highest theoretical performance and performs well on ATSC Monaco. However, due to its dependence on large network size and computing resources, it is difficult to truly achieve theoretical optimal performance. MAG also has high performance in multiple scenarios but leads to high computational burden and poor scalability.

Rollout length. In order to verify the relationship of rollout length k to Theorem 1 and Theorem 2, we designed relevant experiments to observe the effects of different choices of rollout length on the convergence of the episode reward and sample efficiency. Model rollout length represents the step size of using the model forward to predict the state and reward. Figure 3b report more details in experimental results. We choose four representative environments: CACC Catch-up, Ring Attenuation, ATSC Monaco and Power-Grid with the set of rollout length: 1, 25, 50, 100 and 200, to validate how sensitive the accuracy of environment model is to rollout length. From the final episode reward and sample efficiency in Ring Attenuation under the rollout lengths of 1, 25 and 100, it can be concluded that the environment model is accurate enough to support normal training. However, the accuracy of model drops significantly when the rollout length is 200, which is reflected in the final episode reward and sample efficiency are lower than shorter rollout lengths. According to Theorem 1 and Theorem 2, a shorter rollout length k leads to a lower model error and better convergence of episode reward. Therefore, the experimental results are consistent with the theoretical results.

κ-hop neighbors. In order to further verify the relationship of κ to Theorem 3 and Theorem 4, we designed relevant experiments to observe the effect of different choices of κ on the convergence of the episode reward, sample efficiency and the model error which is defined as the MSE loss under the fixed network structures in our framework. From the Fig. 3c, we conclude that neighbor information is accurate enough for a model to predict the next state in these environments, but the effect of different κ on state prediction is different. When the network structure of is fixed, the sampling efficiency will increase and the error of our model in predicting the state will increase with the increase of κ. But when κ is greater than a certain value, the sampling efficiency will decrease and the error of our model in predicting the state will increase due to the limited fitting ability of the network. The optimal choice of κ is 4 under our network structure in Ring Attenuation and 3 in CACC Catch-up. There is almost no difference between different choice of κ in Ring Attenuation, which indicates that the information of neighbors under $\kappa = 1$ is sufficient for policy learning. According to Theorem 3 and Theorem 4, a larger κ leads to a lower model error, higher sample efficiency and better convergence of episode reward. Therefore, the experimental results are consistent with the theoretical results.

Execution Results

Connected vehicle control. Figure 4a-b show the execution performance of the trained policies in Flow and CACC. In CACC, both the velocity and headway of vehicles with the same interval in the queue can be stably controlled around the target value by our method. In Flow, We conclude that our decentralized controller improves the efficiency of the overall traffic flow while ensuring safety (more results in Supplementary Fig. 3).

Traffic signal control. Figure 5a-b show scenarios description and execution performance in ATSC Grid, Monaco and New York. The average queue length and average intersection delay at intersections are important indicators to measure the level of traffic congestion. Compared to the baselines, our method significantly reduces the average queue length and intersection delay. Our solution still effectively solves traffic congestion with low communication costs, achieving the maximum number of arrived vehicles, average velocity, and the minimum number of halting vehicles. The reason why our method in New York City road network with 436 controllable intersections has a lower average velocity after 1800s is that in the later stage, our method has already cleared most vehicles to the destination, and only a few vehicles are driving at this time.

Pandemic network control. In the large-scale spread of the epidemic in society, real-time local decision-making is very important. As the infection situation gradually worsens, the decisions of different decision-making institutions (education departments, medical systems, social service systems, etc.) usually rely on the local information possessed by the current institutions (Fig. 4c). We observe the overall infection of different infectious disease prevention and control strategies in a society with a growing population in three months. Figure 4c reports the relationship between critical cases and the maximum capacity of hospitals under different policies, If the number of critical cases exceeds the maximum capacity of the hospital, it indicates that the current virus transmission is very serious. Our approach can effectively control severe cases below the maximum capacity of the hospital and alleviate the pressure on medical staff. Figure 4c shows the relationship between death cases and maximum hospital capacity under different policies. our method maintained a small number of deaths in multiple scenarios, controlling the mortality rate at a lower level between 0.8 % and 1.5%, while also balancing regional economic development as much as possible (According to the final performance shown in Fig. 3a, more results in Supplementary Fig. 4).

Power control. Figure 6a shows the architecture of secondary voltage control in Power-Grid. Figure 6c shows execution performance of the trained policies based on different algorithms. We can conclude that our method can maintain reliable control performance and adaptability to random disturbances while reducing the cost of communication (more results in Supplementary Fig. 5). Furthermore, We report the main results in the large-scale Real Power-Net 141-bus, 322-bus and 421-bus scenarios within 12 hours in Fig. 6b. Our method can maintain a higher safety control rate in the 141 bus scenario and achieve lower power loss in terms of q loss and total power loss (Fig. 6d). When we expand the system scale to 322-bus (22 agents) and 421-bus (199 agents), the system complexity and modeling difficulty are higher. Our method still maintains effective policies, which demonstrate its scalability. The strong baseline OPF[82] algorithm, which relies on the real dynamics of the system, and its decision-making ability will decrease in large-scale systems. Besides, solving the constrained optimization problem in OPF is time-consuming, so it is difficult to react to the rapid change of load profile in a large network. This further confirms the advantages of our solution in practical power grid problems and modeling complex non-stationary dynamic systems.

Discussion

Here we provide a new theoretical framework for scalable decision-making and propose a novel MARL framework for the control of general systems. Notably, our MARL framework remains effective even when the system size scales up to the hundreds of agents which significantly surpasses the scale of problems addressed by previous decentralized methods. Therefore, this novel method holds promise in establishing large-scale AI decision-makers for traffic, energy systems, and pandemic systems, thereby elevating the intelligence level in the real world. The core idea in this work shares an intriguing parallel with the theory of six degrees of separation[83]. Through interconnected topology, agents require only minimal information exchange to assess the global situation. In the future, we hope to explore the optimal system topology through information entropy theory and expand modules such as vision and natural language, which will comprehensively improve the ability of AI models and provide new insights for developing scalable AI methodologies.

Methods

In this section, we mainly describe the details of the problem formulation and our algorithm[84] (more details in Supplementary Code 1), as well as some basic theoretical conclusions. Firstly, we introduce networked MDP. We define Independent system[39] and ξ-dependent system, and we present diverse experimental results on algorithms and models.

Networked MDP

We consider a system with n agents as a graph. Specifically, n agents coexist in an undirected and stationary graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each agent in a system is represented as a node in an undirected and stationary graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, therefore $\mathcal{V} = \{1, ..., n\}$ is the set of agents. $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ comprises the edges that represent the connectivity of all the agents. Each agent can communicate along the edges with their adjacent agents. Let N_i denote the neighbor of the agent i including itself. Let N_i^κ denote the κ-hop neighbor of i, i.e. the nodes whose graph distance to i is less than or equal to κ. For the simplicity of notation, we also define $N_{-i}^\kappa = \mathcal{V} \setminus N_i^\kappa$. We define an adjacency matrix A with size of $n \times n$. If there is an adjacency relationship between agent i and agent j, then $A_{ij} = A_{ji} = 1$, otherwise $A_{ij} = A_{ji} = 0$. This adjacency relationship is determined by the system structure and κ.

We define its corresponding networked MDP as $(\mathcal{G}, \{\mathcal{S}_i, \mathcal{A}_i\}_{i \in \mathcal{V}}, p, r)$. Each agent i has its local state $s_i \in \mathcal{S}_i$, and performs action $a_i \in \mathcal{A}_i$. The global state is the concatenation of all local states: $s = (s_1, ..., s_n) \in \mathcal{S} := \mathcal{S}_1 \times ... \times \mathcal{S}_n$. The global action is $a = (a_1, ..., a_n) \in \mathcal{A} := \mathcal{A}_1 \times ... \times \mathcal{A}_n$. For the simplicity of notation, we define s_{N_i} to be the states of i's neighbors. The transition function is defined as: $p(s'|s,a) : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, and the state transition satisfies Markov properties. Each agent maintains a localized policy $\pi_i^{\theta_i}(a_i|s_{N_i})$ with parameter $\theta_i \in \Theta_i$, which means that the localized policy is dependent only on states of its neighbors and itself. We use $\theta = (\theta_1, ..., \theta_n)$ to denote the tuple of localized policy parameters, and $\pi^\theta(a|s) = \prod_{i=1}^n \pi_i^{\theta_i}(a_i|s_{N_i})$ denote the joint policy. The reward function for each agent only depends on local state and action: $r_i(s_i, a_i)$, and the global reward function is defined to be the average reward $r(s,a) = \frac{1}{n}\sum_{i=1}^n r_i(s_i, a_i)$. The goal of reinforcement learning is to find a policy π^θ that maximizes the accumulated reward:

$$\pi^{\theta^*} = \arg\max_{\pi^\theta} \mathbb{E}_{\pi^\theta}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right], \tag{7}$$

where $\gamma \in (0,1)$ is the temporal discount factor. The value function is defined as

$$V_{\pi^\theta}(s) = \mathbb{E}_{\pi^\theta}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s^0 = s\right] = \frac{1}{n}\sum_{i=1}^{n} V_i(s). \tag{8}$$

In the last step, we have defined $V_i(s)$, which is the value function for individual reward r_i.

Model-based Learning Let $\eta[\pi]$ denote the return of the policy in the true environment:

$$\eta[\pi] = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right] \tag{9}$$

Let $\hat{\eta}[\pi]$ denote the returns of the policy under the approximated model. To analyze the difference between $\eta[\pi]$ and $\hat{\eta}[\pi]$, we need to construct a bound

$$\eta[\pi] \geq \hat{\eta}[\pi] - C(p, \hat{p}, \pi, \pi_D), \tag{10}$$

where C is a non-negative function, and π_D is the data-collecting policy. According to (10), if every policy update ensures an improvement of $\hat{\eta}[\pi]$ by at least C, $\eta[\pi]$ will improve monotonically. This inequality was first presented in single agent domain[55].

ξ-dependent Networked System

Networked system may have some extent of locality, meaning in some cases, local states and actions do not affect the states of distant agents. In such systems, environmental transitions can be factorized, and agents are able to maintain local models to predict future local states. We define Independent Networked System (INS) and ξ-dependent Networked System as follows. Figure 1b shows different topological structure in networked systems. In addition, exponential decay has been shown to hold in networked MARL when the network is static[41,71]. It means that with the increase of the topological distance, the influence between agents will gradually decrease in networked systems. We have proved the properties of these networked systems through experiments in the Ring Attenuation (more results in Supplementary Fig. 6).

Definition 1. *An environment is an Independent Networked System (INS) if:*

$$p(s'|s,a) = \prod_{i=1}^{n} p_i(s_i'|s_{N_i^\kappa}, a_i). \tag{11}$$

INS might be an assumption that is too strong to hold. However, for the dynamics that cannot be factorized, we can still use an INS to approximate it. Let D_{TV} denote the total variation distance between distributions, we have the following definition:

Definition 2. *(ξ-dependent) Assume there exists an Independent Networked System $\bar{p}$ such that $\bar{p}(s'|s,a) = \prod_{i=1}^{n} p_i(s'_i|s_{N_i^\kappa}, a_i)$. An environment is ξ-dependent, if:*

$$\sup_{s,a} D_{TV}\left(p(s'|s,a)\|\bar{p}(s'|s,a)\right) = \sup_{s,a} \frac{1}{2}\sum_{s'\in\mathscr{S}} |p(s'|s,a) - \bar{p}(s'|s,a)| \leq \xi.$$

Denote $\widehat{p}(s'|s,a) = \prod_{i=1}^{n} \widehat{p}_i(s'_i|s_{N_i^\kappa}, a_i)$. By using $\widehat{p}(s'|s,a)$ to approximate the true dynamics, larger κ leads to better approximation of model p, but also heavier computation overhead. The universal model error $D(p\|\hat{p})$ can be divided into two parts: dependency bias $D(p\|\bar{p})$ and independence-approximation error $D(\bar{p}\|\hat{p})$ with

$$D(p\|\hat{p}) \leq D(p\|\bar{p}) + D(\bar{p}\|\hat{p}). \tag{12}$$

More details of networked MDP and distances among $\bar{p}, \hat{p}$ and $\widehat{p}$ are shown in Fig. 2b-d. Then for a ξ-dependent system, when model becomes very accurate, meaning $D(\bar{p}\|\hat{p}) \approx 0$, $\sup D(p\|\hat{p}) \approx \sup D(p\|\bar{p}) = \xi$. While D can be any appropriate distance metric, we use the TV-distance hereafter for the ease of presentation. In section **Methods**, we give analysis under both independent and ξ-dependent networked systems. In the following experiments, the systems used to evaluate the algorithms also possess properties of ξ-dependent system.

Let π indicate a collective policy $\pi = [\pi_1, ..., \pi_n]$, and the model $\hat{p}$ be an INS $\hat{p}(s'|s,a) = \prod_{i=1}^{n} \hat{p}_i(s'_i|s_{N_i}, a_i)$ that approximating the true MDP. Denote the data-collecting policy as π_D. For decentralized learning and control, each agent learns a localized model $\hat{p}_i$, policy $\pi_i(\cdot|s_{N_i})$. For each agent i, we solve the following problem:

$$\pi_i^{k+1}, p_i^{k+1} = \underset{\pi_i, p_i}{\operatorname{argmax}} \quad \hat{\eta}[\pi] - C(p, \hat{p}, \pi, \pi_D). \tag{13}$$

Below, we first lay out the monotonic improvement property of independent and ξ-independent systems. Then we describe our framework in Algorithm 1, and Fig. 2e shows the model learning process. Experience data of state transition and rewards from the interaction between the environment and the agent are used to train the environment model. Then our method can use the large amount of data generated by the interaction between the model and the agent to improve the sampling efficiency in the training process.

Monotonic model-based improvement

In model-based learning, different rollout schemes can be chosen. The *vanilla rollout* assumes that models are used in an infinite horizon. The *branched rollout* performs a rollout from a state sampled by a state distribution of previous policy π_D, and runs T steps in $\hat{\pi}$ according to π. Based on different rollout schemes, we can construct two lower bounds. Under *vanilla rollout*, real return and model return can be bounded by model error and policy divergence. Formal results are presented in Theorem 1. The detailed proof is deferred to Supplementary Section 4.2.

Theorem 1. *Consider an independent networked system. Denote local model errors as:*

$$\varepsilon_{m_i} = \max_{s_{N_i}, a_i} D_{TV}[p_i(s'_i|s_{N_i}, a_i)\|\hat{p}_i(s'_i|s_{N_i}, a_i)]$$

and divergences between the data-collecting policy and evaluated policy as:

$$\varepsilon_{\pi_i} = \max_{s_{N_i}} D_{TV}[\pi_D(a_i|s_{N_i})\|\pi(a_i|s_{N_i})]$$

Assume the upper bound of rewards of all agents is $r_{\max}$. Let $\eta^p[\pi_1, ..., \pi_n]$ denote the real returns in the environment. Also, let $\eta^{\hat{p}}[\pi_1, ..., \pi_n]$ denote the returns estimated in the model trajectories, and the states and actions are collected with π_D. Then we have:

$$|\eta^p[\pi_1, ..., \pi_n] - \eta^{\hat{p}}[\pi_1, ..., \pi_n]| \leq \frac{2r_{\max}}{1-\gamma}\sum_{i=1}^{n}\left[\frac{\varepsilon_{\pi_i}}{n} + (\varepsilon_{m_i} + 2\varepsilon_{\pi_i}) \cdot \sum_{k=0}^{\infty}\gamma^{k+1}\frac{|N_i{}^k|}{n}\right]. \tag{14}$$

Intuitively, the term $\sum_{k=0}^{\infty}\gamma^{k+1}\frac{|N_i{}^k|}{n}$ would be in the same magnitude as $\frac{1}{1-\gamma}$, which might be huge given the choice of γ, making the bound too loose to be effective. To make tighter the discrepancy bound in Theorem 1, we adopt the *branched rollout* scheme. The *branched rollout* enables a effective combination of model-based and model-free rollouts. For each rollout, we begin from a state sample from d_{π_D}, and run T steps in each localized $\hat{\pi}_i$. When branched rollout is applied in an INS, Theorem 2 gives the returns bound.

Algorithm 1 A General Model-Based Learning framework

```
1: Initialize policy $\pi(a \mid s) = \{\pi_i\}_{i=1}^n$, predictive model $p_\psi(s' \mid s, a) = \{p_{\psi_i}(s_i \mid s_{N_i^\kappa}, a_i)\}_{i=1}^n$, empty dataset $\{\mathscr{D}_i\}_{i=1}^n$.
2: for $N$ epochs do
3:     Collect data with $\{\pi_i\}_{i=1}^n$ in real environment: $\mathscr{D}_i = \mathscr{D}_i \cup \left\{\left(s_i^t, a_i^t, s_i^{t+1}, r_i^t\right)\right\}_t$ for all agents
4:     Train model $p_{\psi_i}$ on dataset $\mathscr{D}_i$ via maximum likelihood: $\psi_i \leftarrow \mathrm{argmax}_{\psi_i} \mathbb{E}_{\mathscr{D}_i}\left[\log p_{\psi_i}(s_i \mid s_{N_i^\kappa}, a_i)\right]$
5:     Optimize policy under predictive model: $\pi_i^{k+1} \leftarrow \mathrm{argmax}_{\pi_i'} \quad \hat{\eta}[\pi'] - C(p, \hat{p}, \pi', \pi_D)$.
6: end for
```

Theorem 2. *Consider an independent networked system. Denote local model errors as:*

$$\varepsilon_{m_i} = \max_{s_{N_i}, a_i} D_{TV}[p_i(s_i'|s_{N_i}, a_i) \| \hat{p}_i(s_i'|s_{N_i}, a_i)]$$

and divergences between the data-collecting policy and evaluated policy as:

$$\varepsilon_{\pi_i} = \max_{s_{N_i}} D_{TV}[\pi_D(a_i|s_{N_i}) \| \pi(a_i|s_{N_i})]$$

Assume the upper bound of rewards of all agents is $r_{\max}$. Let $\eta^p[\pi_1, ..., \pi_n]$ denote the real returns in the environment. Also, let $\eta^{branch}[\pi_1, ..., \pi_n]$ denote the returns estimated via T-step branched rollout scheme. Then we have:

$$|\eta^p[\pi_1, ..., \pi_n] - \eta^{branch}[\pi_1, ..., \pi_n]| \leq \frac{2r_{\max}}{1-\gamma} \sum_{i=1}^n \left[\varepsilon_{m_i} \cdot \left(\sum_{k=0}^{T-1} \gamma^{k+1} \frac{|N_i^k|}{n}\right) + \varepsilon_{\pi_i} \cdot \left(\sum_{k=T}^{\infty} \gamma^{k+1} \frac{|N_i^k|}{n}\right)\right]$$

Comparing the results in Theorem 1 and 2, we can see that branched rollout scheme reduced the coefficient before ε_{m_i} from $\sum_{k=0}^{\infty} \gamma^{k+1} \frac{|N_i^k|}{n} \leq \frac{\gamma}{1-\gamma}$ to $\sum_{k=0}^{T-1} \gamma^{k+1} \frac{|N_i^k|}{n} \leq \sum_{k=0}^{T-1} \gamma^{k+1} = \frac{\gamma(1-\gamma^T)}{1-\gamma}$. This reduction explains that empirically, branched rollout brings better asymptotic performance. Also, if we set $T = 0$, this bound turn into a model-free bound. This indicates that when ε_{m_i} is lower than ε_{π_i} allowed by our algorithm, a model might increase the performance.

Incorporating Dependency Bias

In reality, not every system satisfies the definition of INS. Yet we can generalize Theorem 2 into a ξ-dependent system.

Corollary 1. *Consider an ξ-dependent networked system. Denote local model errors as:*

$$\varepsilon_{m_i} = \max_{s_{N_i}, a_i} D_{TV}[p_i(s_i'|s_{N_i}, a_i) \| \hat{p}_i(s_i'|s_{N_i}, a_i)]$$

and divergences between the data-collecting policy and evaluated policy as:

$$\varepsilon_{\pi_i} = \max_{s_{N_i}} D_{TV}[\pi_D(a_i|s_{N_i}) \| \pi(a_i|s_{N_i})]$$

Assume the upper bound of rewards of all agents is $r_{\max}$. Let $\eta^p[\pi_1, ..., \pi_n]$ denote the real returns in the environment. Also, let $\eta^{branch}[\pi_1, ..., \pi_n]$ denote the returns estimated via T-step branched rollout scheme. Then we have:

$$|\eta^p[\pi_1, ..., \pi_n] - \eta^{branch}[\pi_1, ..., \pi_n]| \leq \frac{2r_{\max}\gamma}{(1-\gamma)^2}\xi + \frac{2r_{\max}}{1-\gamma} \sum_{i=1}^n \left[\varepsilon_{m_i} \cdot \left(\sum_{k=0}^{T-1} \gamma^{k+1} \frac{|N_i^k|}{n}\right) + \varepsilon_{\pi_i} \cdot \left(\sum_{k=T}^{\infty} \gamma^{k+1} \frac{|N_i^k|}{n}\right)\right]$$

The proof can also be found in Supplementary Information. Compared to Theorem 2, Corollary 1 is more general, as it applies to multi-agent systems that are not fully independent. Intuitively, if a networked system seems nearly independent, local models will be effective enough. The bound indicates that when the policy is optimized in a trust region where $D(\pi, \pi_D) \leq \varepsilon_{\pi_i}$, the bound would also be restricted, making monotonic update more achievable.

Compared with the theorems of model-based policy optimization in the single-agent domain[55], we extend them to the multi-agent domain, especially for completely independent networked systems or even more general and real-world systems named ξ-dependent networked systems. Further, to solve the problems of the increase of computation complexity, communication cost and system instability caused by the multi-agent system, we propose a novel communication mechanism and analyze the accuracy of the estimation of value functions and policy gradients through the following theoretical proofs, which is completely different from the general model-based algorithms.

Algorithm 2 Model-based Decentralized Policy Optimization

```
1: Initialize the model buffer $\mathscr{D}_i^M$ and the environment buffer $\mathscr{D}_i^E$, where $i \in \mathscr{V}$.
2: Initialize the rollout length $T$ and adjacency matrix $A$ through $\kappa$.
3: Initialize the decentralized model $p^{\psi_i}$, actor $\pi^{\theta_i}$ and critic $V^{\phi_i}$ by adjacency matrix $A$, where $i \in \mathscr{V}$.
4: for many epochs do
5:   Sample action according to decentralized $\pi^{\theta_i}$ and local observation $s_{N_i^\kappa}$ in environment to collect trajectories $\tau_i^E$.
6:   Store trajectories to environment buffer $\mathscr{D}_i^E$ for each agent $i$, $\mathscr{D}_i^E = \mathscr{D}_i^E \cup \{\tau_i^E\}$.
7:   Minimize the objective in Eq.(6) to update model $p^{\psi_i}$ through trajectories from $\mathscr{D}_i^E$.
8:   for many steps do
9:     for many rollouts do
10:      Sample $s_i^t$ from $\mathscr{D}_i^E$ as an initial state.
11:      Simulate $T$-step trajectories $\tau_i^M$ from $s_i^t$ by decentralized policy $\pi^{\theta_i}$ and decentralized model $p^{\psi_i}$.
12:      Store trajectories $\tau_i^M$ to model buffer $\mathscr{D}_i^M$ for each agent $i$, $\mathscr{D}_i^M = \mathscr{D}_i^M \cup \{\tau_i^M\}$.
13:    end for
14:    for many gradient steps do
15:      Update the extended value functions $V^{\phi_i}$ on trajectories $\tau_i^M$ sampled from $\mathscr{D}_i^M$ according to Eq. (3)
16:      Update the decentralized policies $\pi^{\theta_i}$ on trajectories $\tau_i^M$ sampled from $\mathscr{D}_i^M$ according to Eq.(5).
17:    end for
18:  end for
19: end for
```

Therefore, the challenges for our algorithm and theorems are mainly two-fold in these ξ-dependent networked systems with fewer communications. One is that local communication and approximation dependency bias $D(p\|\bar{p})$ between independent networked systems and ξ-dependent networked systems will increase the generalization error ε_{m_i} of the environment model and policy shift ε_{π_i} between π and π_D, which will further increase the lower bounds $C(p,\hat{p},\pi,\pi_D)$ in Theorem 1 and Theorem 2. The results of experiments fully prove that our method can reduce the influence of these errors on the montonic improvement of the model-based learning through the following algorithm design and hyperparameter selection. The other is that our method needs to provide accurate policy gradients and value functions for the multi-agent system to guide the policy optimization, which will be explained with more details in Theorem 3 and Theorem 4.

Analysis of Policy Gradients Approximation

In this section, we show that the policy gradient induced by Eq. (5) is an asymptotic approximation of the true policy gradients. We start from discussing that the extended value function $V_i(s_{N_i^\kappa})$ is a good approximation of the real value function. We formally state the result in Theorem 3 and defer the proof to Supplementary Section 4.5.

Theorem 3. *Define $V_i(s_{N_i^\kappa}) = \mathbb{E}_{s_{N_{-i}^\kappa}}[\sum_{t=0}^\infty r_i^t(s_t,a_t)|s_{N_i^\kappa}^0 = s_{N_i^\kappa}]$, and $V_i(s) = \mathbb{E}[\sum_{t=0}^\infty r_i^t|s^0 = s]$, then*

$$|V_i(s) - V_i(s_{N_i^\kappa})| \leq \frac{r_{\max}}{1-\gamma}\gamma^\kappa. \tag{15}$$

Remark 1. *Recall that $V(s) = \frac{1}{n}\sum_{i=1}^n V_i(s)$. From Eq. (15), it is easy to obtain the following result,*

$$|V(s) - \frac{1}{n}\sum_{i=1}^n V_i(s_{N_i^\kappa})| \leq \frac{r_{\max}}{1-\gamma}\gamma^\kappa, \tag{16}$$

which indicates that the global value function $V(s)$ can be approximated by the average of all extended value functions.

In policy optimization, value functions are used for calculating advantages $\hat{A}^{(t)}$, and we have shown that $V(s)$ can be estimated with the average of extended value functions $\frac{1}{n}\sum_{i=1}^n V_i(s_{N_i^\kappa})$. In practice, an agent might not get the value function of distant agents and can only access the value function of its κ-hop neighbors. However, we can prove that $\tilde{V}_i = \frac{1}{n}\sum_{j\in N_i^\kappa} V_j(s_{N_j^\kappa})$ is already very accurate for calculating the policy gradient for agent i. This theoretical result has been extended to general multi-agent systems based on previous research[39]. Theorem 4 formally states this result and the proof is deferred to Section 4.6.

Theorem 4. *Let $\hat{A}_t = r^{(t)} + \gamma V(s^{(t+1)}) - V(s^{(t)})$ be the TD residual, and $g_i = \mathbb{E}[\hat{A}\nabla_{\theta_i}\log\pi_i(a|s)]$ be the policy gradient. If $\tilde{A}_t$ and $\tilde{g}_i$ are the TD residual and policy gradient when value function $V(s)$ is replaced by $\tilde{V}_i(s) = \frac{1}{n}\sum_{j\in N_i^\kappa} V_j(s_{N_i^\kappa})$, we have:*

$$|g_i - \tilde{g}_i| \leq \frac{\gamma^{\kappa-1}}{1-\gamma}[1-(1-\gamma^2)\frac{N_i^\kappa}{n}]r_{\max}g_{\max}, \tag{17}$$

where $r_{\max}$ and $g_{\max}$ denote the upper bound of the absolute value of reward and gradient, respectively.

Remark 2. *Theorem 4 justifies that the policy gradients computed based on the sum of the neighboring extended value functions is a close approximation of true policy gradients. The power of this theorem is that the extended value function $V_i(s_{N_i}^{\kappa})$ requires only the neighboring information, thus easier to approximate and scalable. Despite the reduction in computation, the difference between the approximated and true gradient in Eq. (17) is small.*

Based on the difficulties and challenges proposed in the above theoretical analysis, we designed corresponding solutions in the algorithm and the specific implementation details can be found in our code. In Theorem 1 and Theorem 2, the probability to monotonic improvement of the model-based learning in the algorithm will be increase when the lower bound $C(p, \hat{p}, \pi, \pi_D)$ that consists of two parts: generalization model error ε_{m_i} and policy shift ε_{π_i} is tight enough to constrain (10). In order to reduce the generalization model error ε_{m_i} caused by dependency bias $D(p\|\hat{p})$ in (12), we design a stage of warm-up in our algorithm to pre-train the model until an acceptable accuracy of the model is obtained and a method to select data from the true MDP or model with a probability to guide the usage of the model, and we further adopted a k-branched rollout method to improve the accuracy of model. According to (12), when model becomes very accurate, meaning $D(\bar{p}\|\hat{p}) \approx 0$, $\sup D(p\|\hat{p}) \approx \sup D(p\|\bar{p}) = \xi$. Then we design a distribution of the neighbors for each agent, including the quantity and topology of neighbors, to reduce the model error ε_{m_i} caused by dependency bias $D(p\|\bar{p})$ in the experiments. Lastly, in order to constrain the policy shift ε_{π_i} between π and π_D, We use the method of proximal policy optimization to limit the difference in policies.

Data availability

The SUMO data generated during other experimental processes, as well as the power-grid and pandemic data used, are all included in the code repository and can be used directly https://github.com/CDM1619/Networked-MB-MARL[84]. The specific usage methods are further explained in README.md. The real data of Real Power-Net are available in https://drive.google.com/file/d/1-GGPBSolVjX1HseJVblNY3KoTqfblmLh/view, This is a publicly available dataset from previous research work[74]. For more details on how to use this data to run the experiments, please refer to https://github.com/CDM1619/Networked-MB-MARL.

Code availability

An implementation of our method is available at https://github.com/CDM1619/Networked-MB-MARL. We provide detailed installation tutorials and running examples in https://github.com/CDM1619/Networked-MB-MARL[84].

Acknowledgements

We appreciate Dr. Kai Du's suggestions for revising the manuscript and response letter. We thank Yifan Zhong for assistance in language polishing and Dr. Jianhong Wang for providing suggestions in experiments with real-world power nets.

Author contributions

C.M., A.L., Y.D., H.D. and Y.Y. designed the project. C.M., A.L., Y.D. and Y.Y. conducted the experiments and analyzed the results. C.M., A.L., Y.D., H.D. and Y.Y. wrote the manuscript and response letter. C.M., A.L., Y.D. and Y.Y. analyzed the theoretical results. C.M., A.L., Y.D., H.D. and Y.Y. participated in the discussion of the revised content.

Competing interests

The authors declare no competing interests.

Figure Captions (for main text figures)

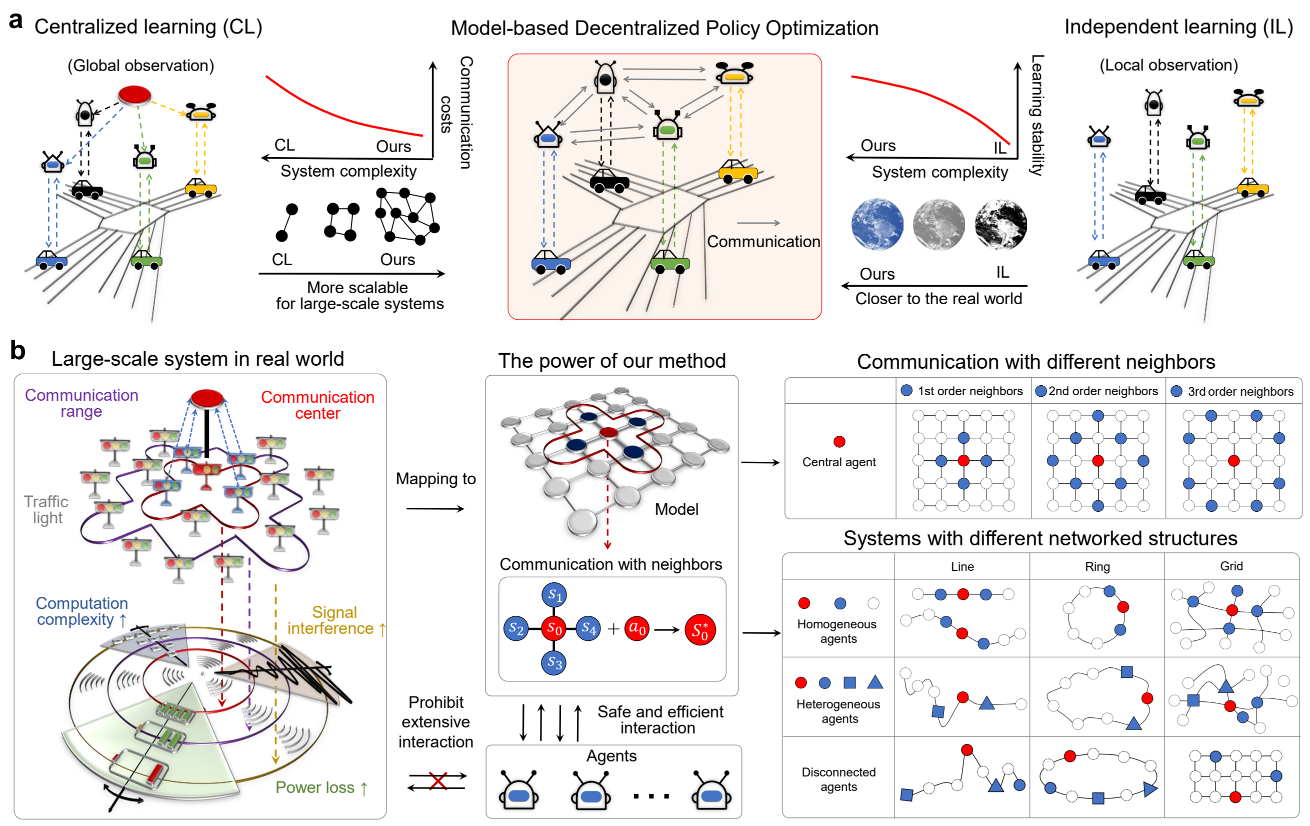

Figure 1. Research motivation and relationship of networked agents. a, When comparing our method with two main learning paradigms, centralized learning (left) requires each agent to have global observations, significantly increasing algorithm complexity and communication costs, and reducing scalability. In independent learning (right), each agent learns in a single-agent fashion through local observations, which reduces system and algorithm complexity. However, the learning process is unstable, leading to poor decision-making performance and mismatch with real-world scenarios. Our method (middle) learns based on local communication between topologically connected agents, avoiding the drawbacks of these two learning paradigms and achieving superior performance with low observation costs. **b,** When controlling traffic lights on a traffic system, obtaining information from many other traffic lights to make decisions for many other traffic lights can lead to power loss with each communication step, thereby reducing the endurance of the traffic lights. Additionally, frequent and large-scale communication operations increase the probability of signal interference and raise computational complexity, which are drawbacks of extensive communication operations. Our approach relies on local communication between topologically adjacent agents to reduce communication costs, power consumption, and computational complexity. Each agent receives the state observation from its neighbors and aggregates them with its state and action to obtain the final decision-dependent information. Furthermore, our framework maps the real system to a model, allowing it to learn policies through safe and efficient interactions between agents and the model. On the right, we describe the relationships between agents and neighbors of different orders in our methodology, where the order corresponds to κ, and a larger κ implies a broader communication range. In our method, each agent accurately estimates the global value and policy gradient solely through its neighbors' information, aiding policy learning. Another advantage of our solution is its ability to handle heterogeneous agents and systems of multiple types. In our experiments, CACC represents a linear-type system, Flow represents a ring-type system, and ATSC, Power Grid, Real Power Net, and Pandemic Networks represent grid-type systems. The agents in CACC, Flow, and ATSC Grid are homogeneous, while agents in ATSC Monaco, ATSC New York, Real Power Network, and Pandemic Networks are heterogeneous. Additionally, in the non-adjacent setting, some disconnected agents existed in systems.

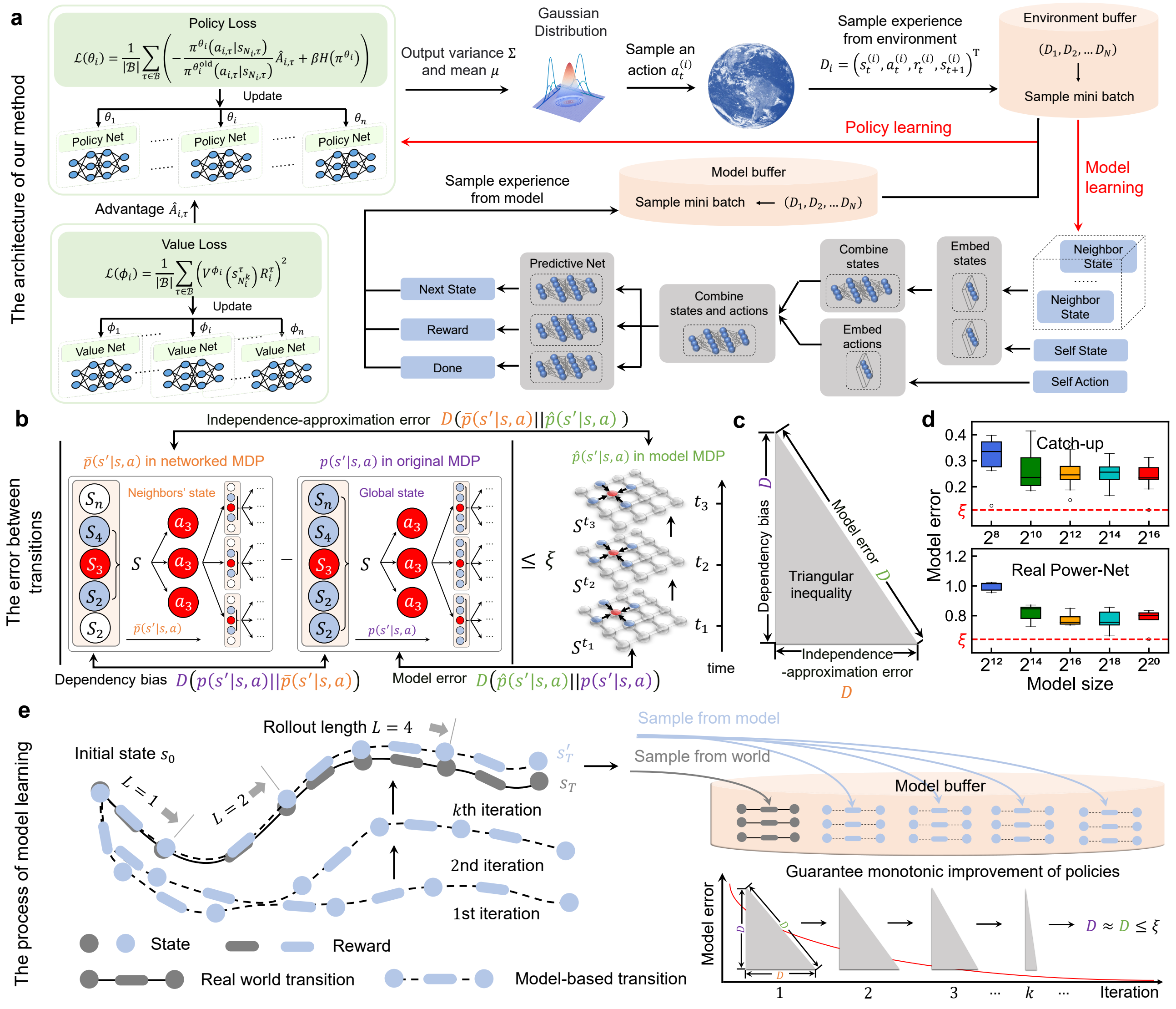

Figure 2. The architecture of our method, distinctions, and connections among different MDPs, and the model learning process. a, Our algorithm comprises decentralized policy, value function, model, model buffer and environment buffer. Bold red arrows highlight primary steps, involving experience sampling from both the model and the real environment for policy and model learning. The model framework incorporates embedding layers for state and action feature extraction, followed by merging based on graph network topology. **b,** Distinction and connection of different p in networked MDP, original MDP, and model MDP as depicted in inequality 12. The differences between different p constitute dependency bias, model error, and independence-approximation error, respectively. **c,** These different D constitute the triangular inequality relationship in inequality 12. The main objective of model learning is to minimize the independence-approximation error $D(\bar{p}\|\hat{p})$ to reduce the difference between model error $D(p\|\hat{p})$ and dependency bias $D(p\|\bar{p})$. **d,** In real-world complex systems, we prove that the values of ξ are all small positive values ($\xi \geq 0.2$ in Catch-up and $\xi \geq 0.8$ in Real Power-Net), which proves the authenticity and correctness of our assumptions about ξ-dependent systems in Definition 2. The data ($n = 5$) are presented as median values (the central line of each box) along with the 25th and 75th percentiles (the bottom and top edges of each box), minima and maxima (the whiskers attached to each box), as well as outliers (outside the box and whiskers). **e,** The process of model learning in our approach involves sampling trajectories from the model buffer for multiple iterations of learning. As a result, the model's predictions of transition gradually approach the real world, with $\sup D(p\|\hat{p}) \approx \sup D(p\|\bar{p}) = \xi$, which promotes a monotonic improvement of the policies.

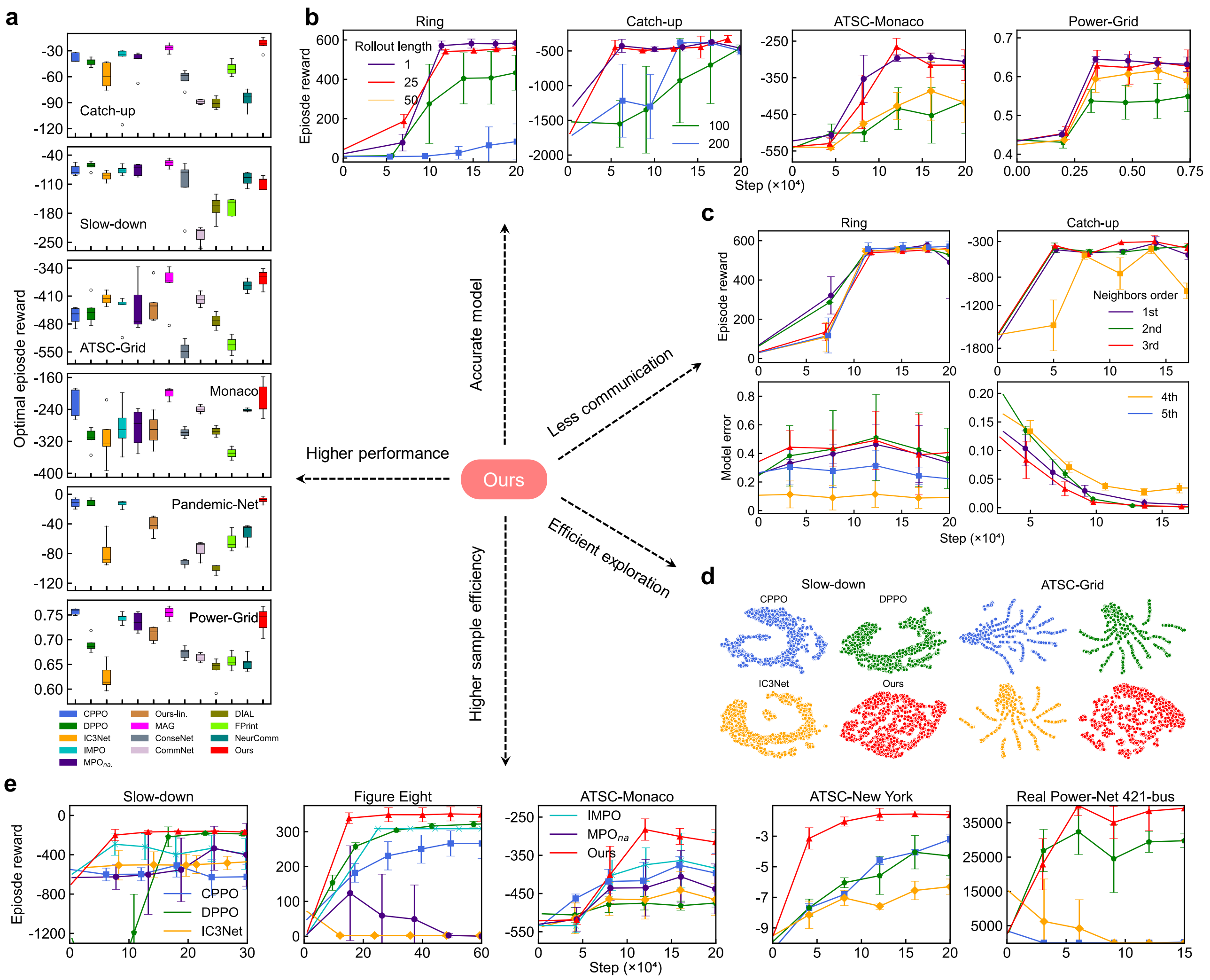

Figure 3. The main training results and ablation studies. These results demonstrate the advantages of our method, including high performance, accurate model learning, low communication cost, efficient exploration ability, and higher sample efficiency. **a,** The performance upper bound of all the baselines in different scenarios with different random seeds. The color red illustrates the performance of our algorithm. We compared it with model-based and model-free baselines. Our approach achieves superior performance in terms of optimality, as demonstrated by the upper edge of the red boxplot being at its highest position in the figure. Additionally, our method exhibits excellent performance in the upper quartile, median, and lower quartile, as shown in the plot. Furthermore, our algorithm demonstrates enhanced training stability, indicating the absence of outliers within the red boxplot. The data ($n = 5$) are presented as median values (the central line of each box) along with the 25th and 75th percentiles (the bottom and top edges of each box), minima and maxima (the whiskers attached to each box), as well as outliers (outside the box and whiskers). **b,** Training reward under different rollout lengths in Ring, Catchup, ATSC-Gtid and Real Power-Net 141-bus ($n = 5$). **c,** Training rewards and model error ($n = 5$) under different κ in Ring and Catch-up. Different κ corresponding to different neighbors order (Fig. 1b). **d,** The states visited by different algorithms in Slow-down and ATSC-Grid. The distance between points represents the difference between states[85]. Due to the existence of the model, our approach outperforms other algorithms in exploration efficiency, thereby increasing sample efficiency. This augmentation is visually represented by the more even and expansive spatial coverage of the red data points within the depicted space. **e,** Training reward increment during the training process under different scenarios and different algorithms. Solid lines represent the mean rewards, and error bars correspond to the standard deviation ($n = 5$).

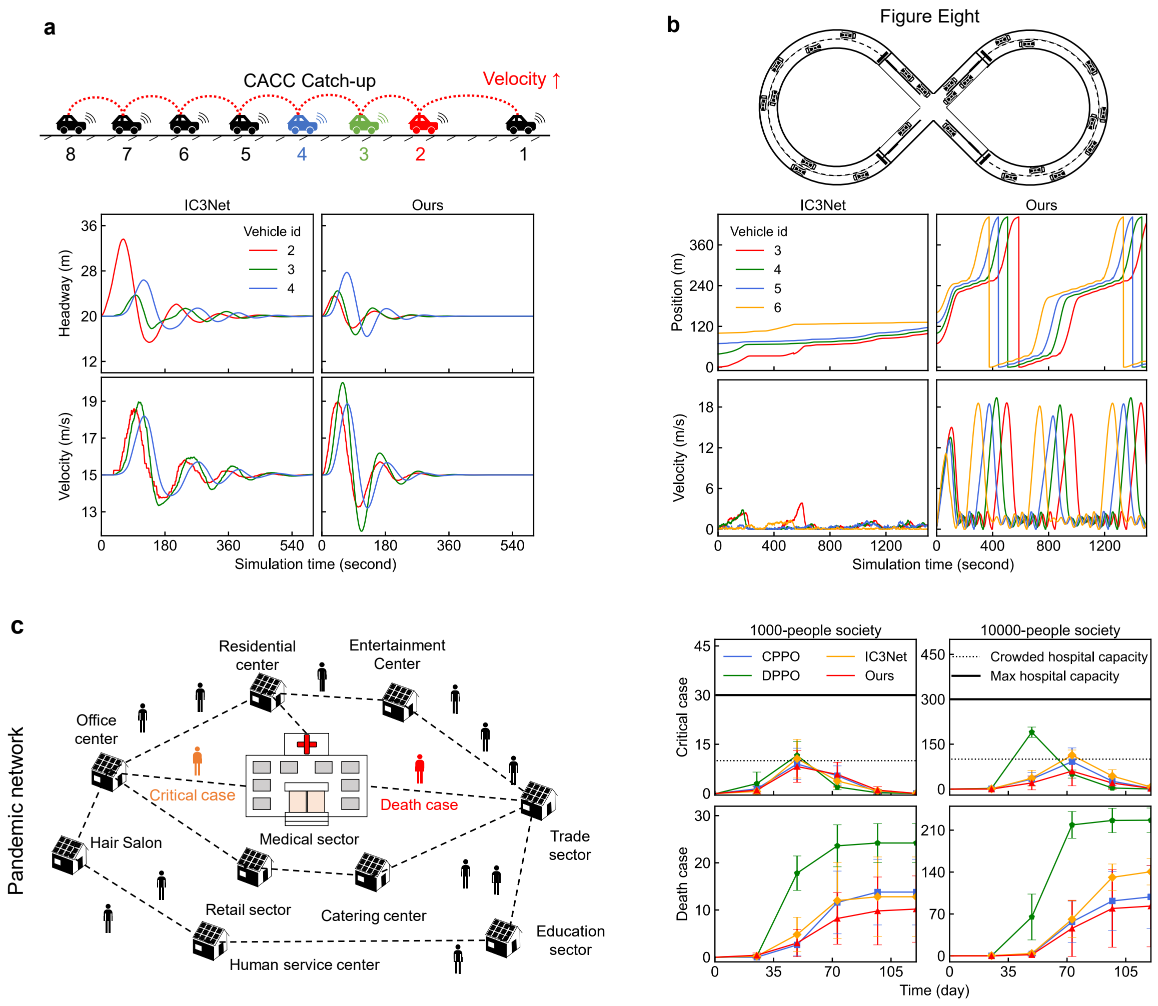

Figure 4. The evaluation results on vehicle control on CACC, Flow and pandemic network. a, Description of velocity and headway of vehicles 1,5,8 in Catch-up. The headway stabilizes at 20 m which matches the target headway. The velocity stabilizes at 15 m/s, matching the target velocity. **b,** Description of velocity and position of vehicles 1-4 in Figure Eight. The visualization of tasks is adapted from existing work[72]. Our approach controls the formation of queues to cross the intersection with an orderly process of accelerating to the target velocity and then decelerating to the safe velocity near 0 m/s. **c,** Description of pandemic network, where various societal units adjust control policies to manage the critical cases and death cases. The right side shows the strategies provided by different algorithms for handling critical cases and death cases with different population sizes. The data are presented as mean values $+/-$ SD. The center line represents the mean values and the error bar represents the standard deviation ($n = 5$).

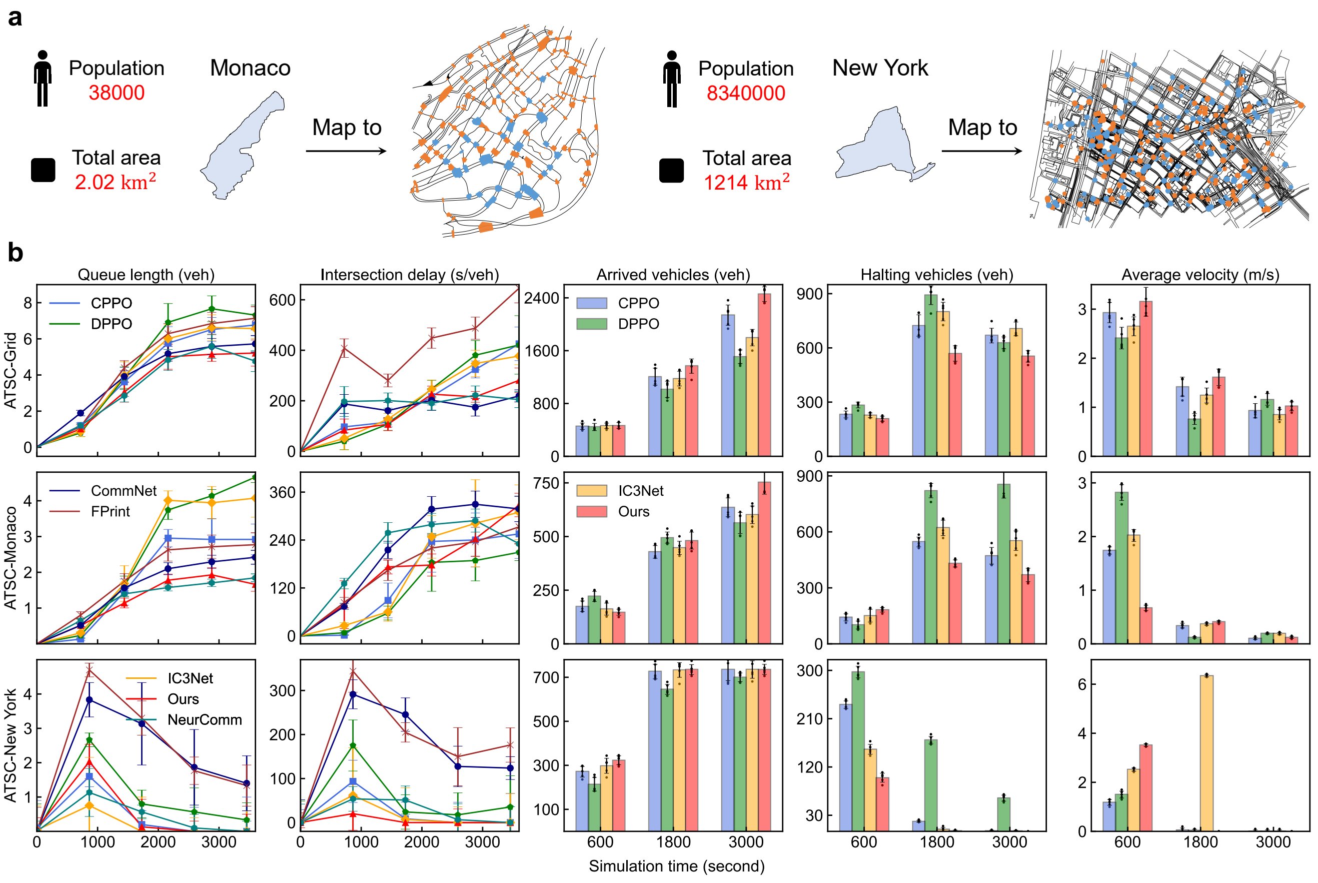

Figure 5. The evaluation results on large-scale traffic control from ATSC. a, Monaco traffic network (a part of Monaco, adapted from existing research[21]) with 28 heterogeneous traffic lights and New York traffic network with 432 traffic lights (a part of New York). **b,** Key evaluation indicators for ATSC tasks. The data are presented as mean values $+/-$ SD. The center represents the mean values and the error bar represents the standard deviation ($n = 5$). During the first 3000s, the system will continue to load vehicles, causing the traffic pressure at the intersections to gradually increase, and the average queue length will gradually increase. However, our method clears the traffic congestion well, so that the maximum average queue length does not exceed about 6 vehs in Grid, 2 vehs in Monaco and 1 vehs in New York. This indicator is lower than other algorithms. After 3000s of vehicles loading, our method can maintain an average intersection delay at about 250 s/veh in Grid, 300 s/veh in Monaco and 10 s/veh in New York. Our method prevents it from increasing. At different times, our decentralized policies achieved the most arrived vehicles, the least halting vehicles, and stable average velocity. It should be noted that in New York scenario, the actual total evaluation time is 1000s. to unify the time with all scenarios, we have standardized it to within 3600s.

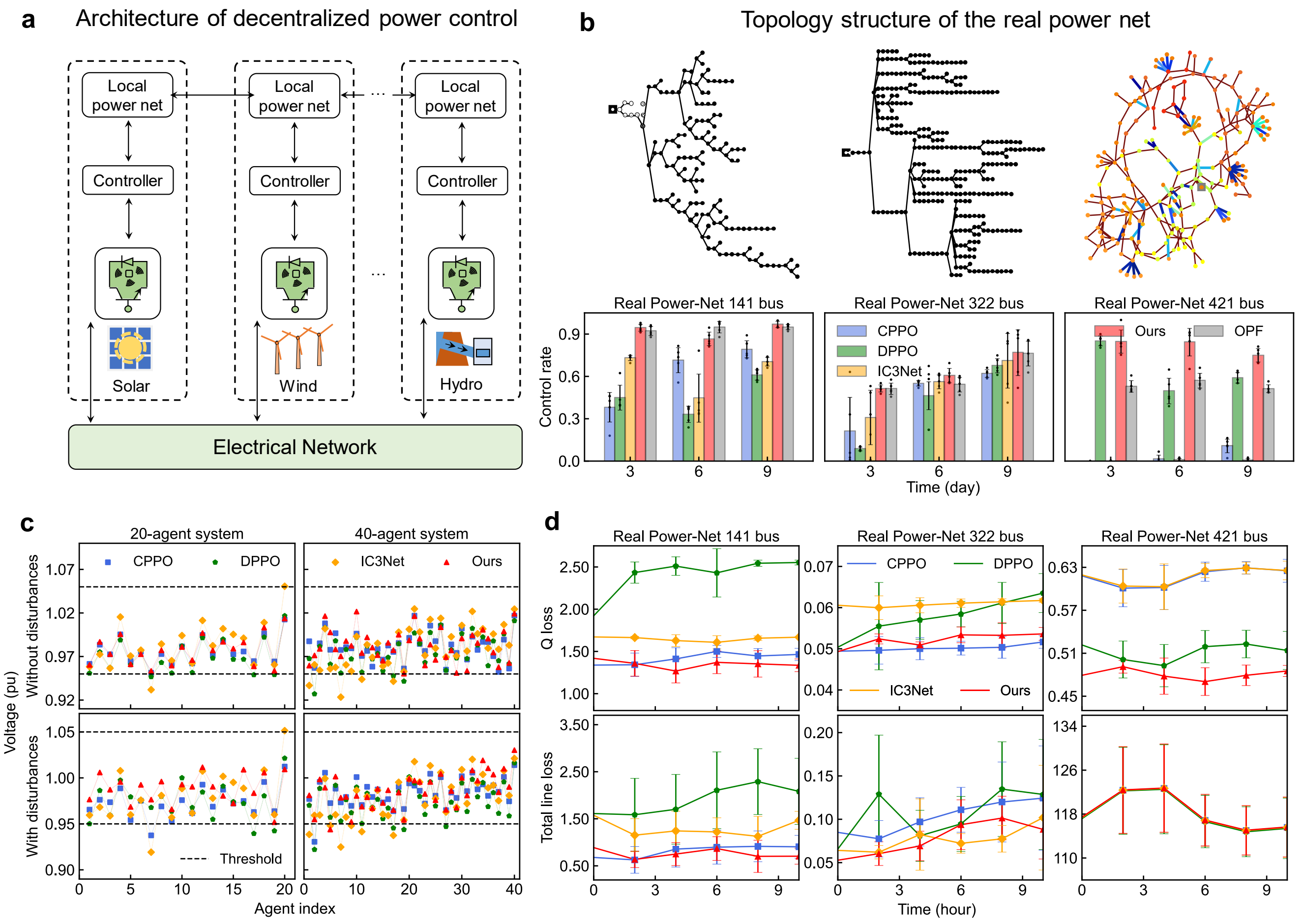

Figure 6. The evaluation results on large-scale power control from Power-Grid and Real Power-Net. a, The architecture of secondary voltage control in Power-Grid with the objective of stabilizing the output voltages of the distributed generators at a pre-defined reference value. 0.95 *pu* to 1.05 *pu* is the range of reference value. This system structure is adapted from existing research[73]. **b,** The structures of 141-bus, 322-bus and 421-bus in Real-Power Net. We report the capabilities in terms of safety control rate, Control rate calculates the ratio of time steps. The data are presented as mean values +/− SD. The top of the bar chart represents the mean values and the error bar represents the standard deviation ($n = 5$) where all buses' voltages are under control for 12 hours. **c,** The Voltage control performance without random disturbances and with random disturbances in Power-Grid with 20 agents and 40 agents. The control performance of the baselines decreases when random disturbances are added and it is difficult to control the voltage of all distributed generators within the range of reference value. but our method can control the voltage of most distributed generators well within the range of reference value regardless of the presence of random disturbances. **d,** We report the Q loss and total line loss in Real Power-Net. The data are presented as mean values +/− SD. The center line represents the mean values and the error bar represents the standard deviation ($n = 5$). Q loss represents the average of the mean reactive power generations by agents each time step during 12 hours. total line loss represents the average of the total power loss overall buses per time step during 12 hours.

References

1. Barmer, H. *et al.* Scalable ai. *CMU.edu* (2021).
2. Zhang, K., Yang, Z. & Başar, T. Multi-agent reinforcement learning: A selective overview of theories and algorithms (2019). 1911.10635.
3. Qin, J., Chow, Y., Yang, J. & Rajagopal, R. Distributed online modified greedy algorithm for networked storage operation under uncertainty. *IEEE Transactions on Smart Grid* **7**, 1106–1118 (2015).
4. Huo, X. & Liu, M. Privacy-preserving distributed multi-agent cooperative optimization—paradigm design and privacy analysis. *IEEE Control. Syst. Lett.* **6**, 824–829 (2021).
5. Gronauer, S. & Diepold, K. Multi-agent deep reinforcement learning: a survey. *Artif. Intell. Rev.* 1–49 (2022).
6. Busoniu, L., Babuska, R. & De Schutter, B. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Syst. Man, Cybern. Part C (Applications Rev.* **38**, 156–172 (2008).
7. Ruan, J. *et al.* Gcs: Graph-based coordination strategy for multi-agentreinforcement learning. In *AAMAS* (2022).
8. Zhou, M. *et al.* Smarts: An open-source scalable multi-agent rl training school for autonomous driving. In *Conference on Robot Learning*, 264–285 (PMLR, 2021).
9. Li, Y. *et al.* V2x-sim: Multi-agent collaborative perception dataset and benchmark for autonomous driving. *IEEE Robotics Autom. Lett.* **7**, 10914–10921 (2022).
10. Lim, S., Yu, H. & Lee, H. Optimal tethered-uav deployment in a2g communication networks: Multi-agent q-learning approach. *IEEE Internet Things J.* **9**, 18539–18549 (2022).
11. Qiu, X., Xu, L., Wang, P., Yang, Y. & Liao, Z. A data-driven packet routing algorithm for an unmanned aerial vehicle swarm: A multi-agent reinforcement learning approach. *IEEE Wirel. Commun. Lett.* **11**, 2160–2164 (2022).
12. Lian, Z. & Deshmukh, A. Performance prediction of an unmanned airborne vehicle multi-agent system. *Eur. J. Oper. Res.* **172**, 680–695 (2006).
13. Feriani, A. & Hossain, E. Single and multi-agent deep reinforcement learning for ai-enabled wireless networks: A tutorial. *IEEE Commun. Surv. & Tutorials* **23**, 1226–1252 (2021).
14. Naderializadeh, N., Sydir, J. J., Simsek, M. & Nikopour, H. Resource management in wireless networks via multi-agent deep reinforcement learning. *IEEE Transactions on Wirel. Commun.* **20**, 3507–3523 (2021).
15. Samvelyan, M. *et al.* The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 2186–2188 (2019).
16. Vinyals, O. *et al.* Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature* **575**, 350–354 (2019).
17. Kamboj, S., Kempton, W. & Decker, K. S. Deploying power grid-integrated electric vehicles as a multi-agent system. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, 13–20 (2011).
18. Li, W., Logenthiran, T., Phan, V.-T. & Woo, W. L. Intelligent multi-agent system for power grid communication. In *2016 IEEE Region 10 Conference (TENCON)*, 3386–3389 (IEEE, 2016).
19. Ghanadbashi, S. & Golpayegani, F. Using ontology to guide reinforcement learning agents in unseen situations: A traffic signal control system case study. *Appl. Intell.* **52**, 1808–1824 (2022).
20. Noaeen, M. *et al.* Reinforcement learning in urban network traffic signal control: A systematic literature review. *Expert. Syst. with Appl.* 116830 (2022).
21. Chu, T., Chinchali, S. & Katti, S. Multi-agent reinforcement learning for networked system control. In *International Conference on Learning Representations (ICLR)* (2020).
22. Jin, I. G. & Orosz, G. Dynamics of connected vehicle systems with delayed acceleration feedback. *Transp. Res. Part C: Emerg. Technol.* **46**, 46–64 (2014).
23. Wu, C., Kreidieh, A., Vinitsky, E. & Bayen, A. M. Emergent behaviors in mixed-autonomy traffic. In *Conference on Robot Learning*, 398–407 (PMLR, 2017).
24. Bando, M., Hasebe, K., Nakayama, A., Shibata, A. & Sugiyama, Y. Dynamical model of traffic congestion and numerical simulation. *Phys. review E* **51**, 1035 (1995).
25. Simpson-Porco, J. W. *et al.* Secondary frequency and voltage control of islanded microgrids via distributed averaging. *IEEE Transactions on Ind. Electron.* **62**, 7025–7038 (2015).

26. Garcia, C. E., Prett, D. M. & Morari, M. Model predictive control: Theory and practice—a survey. *Automatica* **25**, 335–348 (1989).

27. Lai, J. *et al.* Distributed voltage control for dc mircogrids with coupling delays & noisy disturbances. In *IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society*, 2461–2466 (IEEE, 2017).

28. Wang, S. *et al.* A data-driven multi-agent autonomous voltage control framework using deep reinforcement learning. *IEEE Transactions on Power Syst.* **35**, 4644–4654 (2020).

29. Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N. & Whiteson, S. Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)* (2018).

30. Lowe, R. *et al.* Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 6379–6390 (2017).

31. Du, Y. *et al.* Liir: Learning individual intrinsic reward in multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)* (2019).

32. Yu, C. *et al.* The surprising effectiveness of ppo in cooperative multi-agent games. *Adv. Neural Inf. Process. Syst.* **35**, 24611–24624 (2022).

33. Zhong, Y. *et al.* Heterogeneous-agent reinforcement learning. *J. Mach. Learn. Res.* **25**, 1–67 (2024).

34. Sunehag, P. *et al.* Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2085–2087 (2018).

35. Rashid, T. *et al.* Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*, 4292–4301 (2018).

36. Son, K., Kim, D., Kang, W. J., Hostallero, D. & Yi, Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)* (International Conference on Machine Learning Organizing Committee, 2019).

37. Lin, Y., Qu, G., Huang, L. & Wierman, A. Multi-agent reinforcement learning in stochastic networked systems. *Adv. Neural Inf. Process. Syst.* **34**, 7825–7837 (2021).

38. Li, T. & Zhang, J.-F. Consensus conditions of multi-agent systems with time-varying topologies and stochastic communication noises. *IEEE Transactions on Autom. Control.* **55**, 2043–2057 (2010).

39. Du, Y. *et al.* Scalable model-based policy optimization for decentralized networked systems. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 9019–9026 (IEEE, 2022).

40. Zhang, K., Yang, Z., Liu, H., Zhang, T. & Başar, T. Fully decentralized multi-agent reinforcement learning with networked agents. In *International Conference on Machine Learning (ICML)*, 5872–5881 (2018).

41. Qu, G., Lin, Y., Wierman, A. & Li, N. Scalable multi-agent reinforcement learning for networked systems with average reward. *Adv. Neural Inf. Process. Syst. (NeurIPS)* **33** (2020).

42. Simao, T. D. & Spaan, M. T. Safe policy improvement with baseline bootstrapping in factored environments. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, vol. 33, 4967–4974 (2019).

43. Guestrin, C., Koller, D. & Parr, R. Multiagent planning with factored mdps. In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 1, 1523–1530 (2001).

44. Du, Y. *et al.* Learning correlated communication topology in multi-agent reinforcement learning. In *AAMAS*, 456–464 (2021).

45. Foerster, J., Assael, I. A., de Freitas, N. & Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. In *NeurIPS*, 2137–2145 (2016).

46. Zhang, C. & Lesser, V. Coordinating multi-agent reinforcement learning with limited communication. In *AAMAS*, 1101–1108 (2013).

47. Sukhbaatar, S., Szlam, A. & Fergus, R. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2252–2260 (2016).

48. Mnih, V. *et al.* Human-level control through deep reinforcement learning. *Nature* **518**, 529 (2015).

49. Vinyals, O. *et al.* Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature* **575**, 350–354 (2019).

50. Silver, D. *et al.* A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science* **362**, 1140–1144 (2018).

51. Han, L. *et al.* Grid-wise control for multi-agent reinforcement learning in video game ai. In *ICML*, 2576–2585 (PMLR, 2019).

52. Kaelbling, L. P., Littman, M. L. & Moore, A. W. Reinforcement learning: A survey. *J. artificial intelligence research* **4**, 237–285 (1996).

53. Deisenroth, M. & Rasmussen, C. E. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 465–472 (Citeseer, 2011).

54. Luo, Y. *et al.* Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. In *International Conference on Learning Representations (ICLR)* (2019).

55. Janner, M., Fu, J., Zhang, M. & Levine, S. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems (NeurIPS)* (2019).

56. Schrittwieser, J. *et al.* Mastering atari, go, chess and shogi by planning with a learned model. *Nature* **588**, 604–609 (2020).

57. Morgan, A. S. *et al.* Model predictive actor-critic: Accelerating robot skill acquisition with deep reinforcement learning. In *ICRA*, 6672–6678 (IEEE, 2021).

58. Deisenroth, M. P., Neumann, G., Peters, J. *et al.* A survey on policy search for robotics. *Foundations trends Robotics* **2**, 388–403 (2013).

59. Zhang, K., Kakade, S., Basar, T. & Yang, L. Model-based multi-agent rl in zero-sum markov games with near-optimal sample complexity. *Adv. Neural Inf. Process. Syst. (NeurIPS)* **33** (2020).

60. Brafman, R. I. & Tennenholtz, M. A near-optimal polynomial time algorithm for learning in certain classes of stochastic games. *Artif. Intell.* **121**, 31–47 (2000).

61. Bouzy, B. & Métivier, M. Multi-agent model-based reinforcement learning experiments in the pursuit evasion game. *academia.edu* (2007).

62. Bargiacchi, E., Verstraeten, T. & Roijers, D. Cooperative prioritized sweeping. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021)*, 160–168 (IFAAMAS, 2021).

63. Zhang, W., Wang, X., Shen, J. & Zhou, M. Model-based multi-agent policy optimization with adaptive opponent-wise rollouts. In Zhou, Z. (ed.) *IJCAI*, 3384–3391 (ijcai.org, 2021).

64. Zhang, K., Yang, Z. & Başar, T. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handb. Reinf. Learn. Control.* 321–384 (2021).

65. Wu, Z., Yu, C., Chen, C., Hao, J. & Zhuo, H. H. Models as agents: optimizing multi-step predictions of interactive local models in model-based multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 10435–10443 (2023).

66. Wang, R. *et al.* Model-based reinforcement learning for decentralized multiagent rendezvous. In *Conference on Robot Learning*, 711–725 (PMLR, 2021).

67. Kim, W., Park, J. & Sung, Y. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations* (2020).

68. Pretorius, A. *et al.* Learning to communicate through imagination with model-based deep multi-agent reinforcement learning. In *2020 openreview.net* (2020).

69. Sutton, R. S. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, 216–224 (Elsevier, 1990).

70. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

71. Qu, G., Wierman, A. & Li, N. Scalable reinforcement learning of localized policies for multi-agent networked systems. In *Learning for Dynamics and Control (L4DC)*, 256–266 (PMLR, 2020).

72. Vinitsky, E. *et al.* Benchmarks for reinforcement learning in mixed-autonomy traffic. In *Conference on robot learning*, 399–409 (PMLR, 2018).

73. Chen, D. *et al.* Powernet: Multi-agent deep reinforcement learning for scalable powergrid control. *IEEE Transactions on Power Syst.* **37**, 1007–1017 (2021).

74. Wang, J., Xu, W., Gu, Y., Song, W. & Green, T. C. Multi-agent reinforcement learning for active voltage control on power distribution networks. *Adv. Neural Inf. Process. Syst.* **34**, 3271–3284 (2021).

75. Kompella, V. *et al.* Reinforcement learning for optimization of covid-19 mitigation policies. In *2020 AAAI Fall Symposium on AI for Social Good, AI4SG 2020* (2020).

76. Hao, Q., Huang, W., Feng, T., Yuan, J. & Li, Y. Gat-mf: Graph attention mean field for very large scale multi-agent reinforcement learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 685–697 (2023).

77. Zhou, M. *et al.* Factorized q-learning for large-scale multi-agent systems. In *Proceedings of the first international conference on distributed artificial intelligence*, 1–7 (2019).

78. Liu, Y. *et al.* Gplight: grouped multi-agent reinforcement learning for large-scale traffic signal control. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 199–207 (2023).

79. Sutton, R. S. & Barto, A. G. Reinforcement learning: An introduction. *Robotica* **17**, 229–235 (1999).

80. Singh, A., Jain, T. & Sukhbaatar, S. Learning when to communicate at scale in multiagent cooperative and competitive tasks. In *International Conference on Learning Representations* (2018).

81. Foerster, J. *et al.* Stabilising experience replay for deep multi-agent reinforcement learning. In *International conference on machine learning*, 1146–1155 (PMLR, 2017).

82. Gan, L., Li, N., Topcu, U. & Low, S. H. Optimal power flow in tree networks. In *52nd IEEE Conference on Decision and Control*, 2313–2318 (IEEE, 2013).

83. Guare, J. Six degrees of separation. In *The Contemporary Monologue: Men*, 89–93 (Routledge, 2016).

84. Chengdong Ma, Y. D., Aming Li & Yang, Y. Official implementation of model based decentralized policy optimization, DOI: https://zenodo.org/doi/10.5281/zenodo.11549522 (2024).

85. Van der Maaten, L. & Hinton, G. Visualizing data using t-sne. *J. machine learning research* **9** (2008).