Deep Hierarchical Communication Graph in Multi-Agent Reinforcement Learning

Zeyang Liu[1] , **Lipeng Wan**[1] , **Xue Sui**[1] , **Zhuoran Chen**[1] , **Kewu Sun**[2] and **Xuguang Lan**[1*]
[1]National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, National Engineering Research Center for Visual Information and Applications, and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University
[2]Intelligent Science & Technology Academy
{zeyang.liu, wanlipeng, suixue98, zhuoran.chen}@stu.xjtu.edu.cn, sun_kewu@126.com, xglan@mail.xjtu.edu.cn

Abstract

Sharing intentions is crucial for efficient cooperation in communication-enabled multi-agent reinforcement learning. Recent work applies static or undirected graphs to determine the order of interaction. However, the static graph is not general for complex cooperative tasks, and the parallel message-passing update in the undirected graph with cycles cannot guarantee convergence. To solve this problem, we propose Deep Hierarchical Communication Graph (DHCG) to learn the dependency relationships between agents based on their messages. The relationships are formulated as directed acyclic graphs (DAGs), where the selection of the proper topology is viewed as an action and trained in an end-to-end fashion. To eliminate the cycles in the graph, we apply an acyclicity constraint as intrinsic rewards and then project the graph in the admissible solution set of DAGs. As a result, DHCG removes redundant communication edges for cost improvement and guarantees convergence. To show the effectiveness of the learned graphs, we propose policy-based and value-based DHCG. Policy-based DHCG factorizes the joint policy in an auto-regressive manner, and value-based DHCG factorizes the joint value function to individual value functions and pairwise payoff functions. Empirical results show that our method improves performance across various cooperative multi-agent tasks, including Predator-Prey, Multi-Agent Coordination Challenge, and StarCraft Multi-Agent Challenge.

1 Introduction

Recent progress of cooperative multi-agent reinforcement learning (MARL) has shown attractive prospects for various real-world applications, such as traffic control [Zhang *et al.*, 2019a], autonomous vehicles [Palanisamy, 2020], and resource optimization [Li *et al.*, 2019]. In communication-enabled MARL, learning differentiable communication protocols has become an active area [Hernandez-Leal *et al.*, 2019]. Previous work [Sukhbaatar *et al.*, 2016; Jiang and Lu, 2018; Das *et al.*, 2019] aims to learn when and with whom to share local observations, which achieves implicit coordination by aggregating messages from others. However, these methods can only represent the same policy or value space as communication-free algorithms because they use the same update methods, e.g., PPO [Schulman *et al.*, 2017] or DDPG [Lillicrap *et al.*, 2016]. As a result, they still suffer from the *non-stationarity* problem and cannot solve tasks that require significant coordination, e.g., *relative overgeneralization* pathology, where the reward for an agent gets confounded by penalties from exploratory actions of others.

Sharing intentions is an effective mechanism that improves the representational capacity of the value function, where the intention is the message that encodes each agent's future action or trajectory. Coordination graph [Guestrin *et al.*, 2002] is a graph-based value factorization method where the local observations and the actions are shared through the edge between connected agents. However, the graph is always static and complete, which has a high representational capacity of the joint value function but is not flexible in complex cooperative multi-agent tasks. To reduce communication costs, some work has been put forward to learn the relationships between agents through soft-attention mechanisms [Li *et al.*, 2021; Yang *et al.*, 2022]. In these methods, each agent makes its decision based on the weighted messages of all other agents. Namely, the relationships between agents are formulated as weighted undirected graphs. However, the parallel message-passing updates cannot guarantee convergence in undirected graphs with cycles based on the analysis of the maximum a posteriori problem in graphical models [Pearl, 1989; Wainwright *et al.*, 2002; Wainwright *et al.*, 2004].

One of the simple implementations to achieve monotonic policy improvement is to enable sequential communication and update [Wen *et al.*, 2022; Fu *et al.*, 2022]. However, these methods define the interaction order as a random permutation of agents' indexes. When the reward and state transition dependency between agents are naturally weakly coupled, these methods have a substantial computational complexity of com-

*Correpsonding Author.

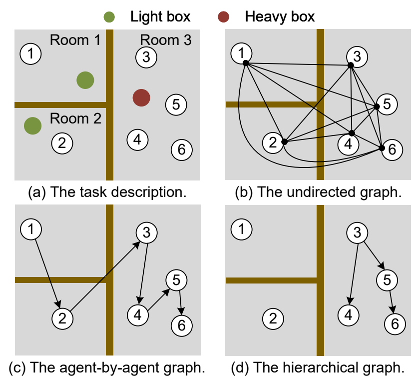

Figure 1: Different communication graphs in the *warehouse keeper* game where agents are weakly coupled.

munication and execution. For example, Fig. 1-a shows a *warehouse keeper* game, where agent 1 and 2 are assigned to two separate rooms to move the light boxes independently, but agent 3 to 6 are required to cooperate to move the heavy box. In this scenario, agent 1 and 2 do not need to propagate their intentions because their actions do not influence other agents. Fig. 1-b to d show different communication topologies. Based on this prior, it is observed that the undirected graph and the agent-by-agent graph involve many redundant edges compared to the hierarchical graph, leading to complexity in execution and potential difficulties in policy learning. Therefore, an open research question arises:

How to learn dependency relations between agents and achieve efficient sequential intention sharing in MARL with complex reward and state transition dependency?

To tackle this problem, we propose a novel graph-based communication scheme for multi-agent coordination named Deep Hierarchical Communication Graph (DHCG), that explicitly models the dependency relations between agents as a directed acyclic graph (DAG) to constrain the flowings of intentions through directed edges. We integrate the selection of the graph topology into the trial-and-error loop of reinforcement learning by regarding it as an action. During training, we apply an intrinsic reward for acyclicity constraint and use a critic to estimate the value of the communication graph at a given state. The graph is optimized by maximizing the output of the critic. In addition, we also formulate a new equivalent representation of DAG and search for its curl-free component to ensure the acyclic property. The hierarchical communication graph reduces communication costs by cutting off unrelated edges for sharing intentions and guaranteeing convergence. We propose policy-based and value-based DHCG to demonstrate the effectiveness of the learned graphs. Policy-based DHCG factorizes the joint policy in an auto-regressive manner, and value-based DHCG factorizes the joint value function to individual value functions and pairwise payoff functions. We list our main contributions as follows:

- We propose policy-based and value-based DHCG to ensure sequential intention sharing, where the dependency relations are formulated as directed acyclic graphs and learned in an end-to-end fashion.
- The empirical results show that DHCG improves performance on multiple partially observable MARL benchmarks, including Predator-Prey, Multi-Agent Coordination Challenge, and StarCraft Multi-Agent Challenge.

2 Problem Formulation and Notations

A fully cooperative multi-agent task in the partially observable setting can be formulated as a Decentralised Partially Observable Markov Decision Process (Dec-POMDP) [Oliehoek and Amato, 2016], consisting of a tuple $G = \langle A, S, \Omega, O, U, P, R, n, \gamma \rangle$, where $a \in A \equiv \{1, \ldots, n\}$ describes the set of agents, S denotes the set of states, Ω denotes the set of joint observations, and R denotes the set of rewards. At each time step, an agent obtains its observation $o \in \Omega$ based on the observation function $O(s, a) : S \times A \to \Omega$, and an action-observation history $\tau_a \in T \equiv (\Omega \times U)^*$. Each agent a chooses an action $u_a \in U$ by a stochastic policy $\pi_a(u_a|\tau_a) : T \times U \to [0, 1]$, forming a joint action $\mathbf{u} \in \mathbf{U}$, which leads to a transition on the environment through the transition function $P : S \times \mathbf{U} \times S \to [0, 1]$. All agents share the same reward function $r : S \times \mathbf{U} \to \mathbb{R}$. The goal of the task is to find the joint policy π which can maximize the joint action-value function $Q^\pi(s_t, \mathbf{u}_t) = \mathbb{E}_{s_{t+1:\infty}, \mathbf{u}_{t+1:\infty}}[R_t | s_t, \mathbf{u}_t]$, where $R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$ is the discounted return, and $\gamma \in [0, 1)$ is a discounted factor

In communication-based MARL, agents can exchange information and are still enforced to take actions simultaneously during decentralized execution on Dec-POMDP.

3 Dependency Relations

This section briefly introduces the definition of dependency relations between agents. In the dependency theory, agent i depends on agent j if agent i has the goal g_i that exceeds its capacity to reach it, and agent j has one of the necessary actions or resources to achieve g_i [Conte and Sichman, 2002]. Therefore, the dependency relationship can be interpreted as the representation of the reward and transition dependency among cooperative agents. For example, the dependency value can be quantified by the influence of the agent i's intention on agent j, i.e., the difference between the expected Q-value function of agent j and its counterfactual Q-value function without the intention of agent i. The dependency value should be zero when the cooperative agents are transition-independent [Dimakopoulou and Van Roy, 2018; Dimakopoulou *et al.*, 2018; Bargiacchi *et al.*, 2018]. One can remove the connection from agent i to j if agent j does not change its decision after receiving the intention of agent i. However, since the action space grows exponentially with the number of agents, it requires vast exploration to obtain the exact counterfactual Q-value function in multi-agent tasks.

Based on dependency relations, sharing intentions allows the agents to know the necessary actions from others, which enlarges the function expressiveness and improves coordination performance. However, the joint policy fails to converge on the optimal if the inter-agent relations contain cycles and the task has multiple optimums [Wainwright *et al.*, 2004]. In addition, it requires an intractable prior to manually setting dependency relations and eliminating cycles in complex

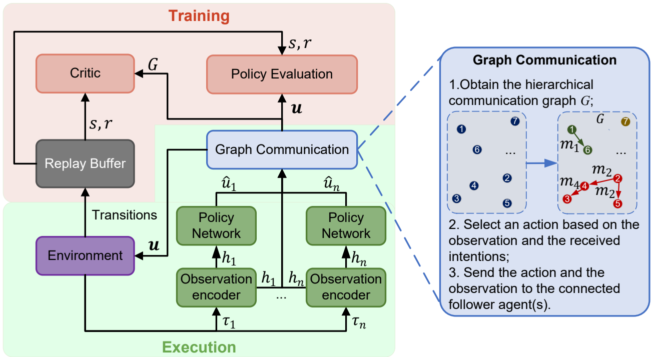

Figure 2: An overview of DHCG's framework.

multi-agent cooperative tasks involving various states. Therefore, it is crucial to learn the dynamic dependency relations in MARL with complex reward and transition dependency.

4 Method

This section presents a novel multi-agent reinforcement learning algorithm named Deep Hierarchical Communication Graph (DHCG). The dynamic dependency relations between agents are formulated as a message-dependent directed acyclic graph (DAG) $G = (V, E)$, where $V := \{1, ..., n\}$ is the set of vertices, and E is the set of directed edges. Each vertex represents an agent, and each edge describes the relationship between two connected agents.

As shown in Fig. 2, each agent i obtains the observation feature τ_i^t based on the observation-action history τ_i^{t-1} and the current local observation o_i^t at each timestep t. Then, each agent i makes its initial decision $\hat{u}_i = \pi_i(\tau_i^t)$ in the decentralized way and obtains the hierarchical communication graph $G = \pi^g(\boldsymbol{\tau}, \hat{\mathbf{u}})$. Based on the messages from its ancestors, each agent chooses the action and then sends messages to its connected agent, where the message encodes the observations and the intention. After communication, the agents take actions simultaneously and interact with the environment.

During training, the selection of the proper hierarchical communication graph G is regarded as an action aiming to maximize the discounted return. Under this interpretation, we integrate the selection of the DAG into the trial-and-error loop of reinforcement learning. We use a critic with an intrinsic reward for acyclicity to evaluate the value of the graph and train the graph by maximizing the output of the critic. Then, we project the learned graph into the admissible solution set of DAGs to eliminate cycles.

4.1 Deep Hierarchical Communication Graph

During execution, each agent i encodes the observation-action history τ_i and its initial decision $\hat{u}_i$ into a query vector $q_i \in \mathbb{R}^{d_k}$ and a key vector $k_i \in \mathbb{R}^{d_k}$, where d_k is a constant. Then, the estimated communication graph is obtained by:

$$G_i = \text{softmax}\left[\frac{q_i^\top k_1}{\sqrt{d_k}}, \frac{q_i^\top k_2}{\sqrt{d_k}}, \ldots, \frac{q_i^\top k_n}{\sqrt{d_k}}\right], \tag{1}$$

where $G_{ii} = 0$. In addition, we set $G_{ij} = 0$ if $\|G_{ij}\| < \delta$, where $\delta > 0$ denotes a fixed threshold.

The optimization of G includes two steps. First, we introduce an intrinsic reward for the acyclicity constraint and use a critic parametrized by θ_c to estimate the value of G:

$$L(\theta_c) = \mathbb{E}\left[(Q(s^t, G^t; \theta_c) - y^t)^2\right], \tag{2}$$

where $y^t = r^t + \gamma Q(s^{t+1}, G^{t+1}; \theta_c') - \lambda Z(G^t)$ is a fixed target, $Z(G^t) = \text{tr}\left[I + \exp(G^t \circ G^t)\right] - n$ is a constraint for acyclicity [Zheng *et al.*, 2018], λ denotes a fixed penalty parameter, and θ_c' is the parameters of the non-differentiable target network, which copied from θ_c every few epochs.

At each timestep t, the graph $G^t = \pi^g(\boldsymbol{\tau}^t, \hat{\mathbf{u}}^t; \theta_{\pi^g})$ is optimized by maximizing the output of the critic:

$$\nabla_{\theta_{\pi^g}} L(\theta_{\pi^g}) = \mathbb{E}\left[\nabla_G Q(s^t, G^t) \nabla_{\theta_{\pi^g}} \pi^g(\boldsymbol{\tau}^t, \hat{\mathbf{u}}^t)\right]. \tag{3}$$

Second, we search for the curl-free component of the learned graph $\hat{G}$ from Eq. (3) to ensure acyclicity by projecting it into the admissible solution set of DAGs. Inspired by DAG-Nocurl [Yu *et al.*, 2021], we reformulate an equivalent representation of a DAG with $\upsilon(W, p; \theta_\upsilon) = W \circ \text{ReLU}(\text{grad}(p))$, where $W_{ij} = -W_{ji}$ is the matrix with zero diagonal elements, grad(.) is a gradient flow, p is approximated by $\tilde{p} = -\triangle_0^\dagger \text{div}(\frac{1}{2}(C(\hat{G}) - C(\hat{G})^T))$, $C(\hat{G})$ is the connectivity matrix of $\hat{G}$, and $\triangle_0^\dagger$ is the graph Laplacian:

$$[\triangle_0^\dagger]_{ij} = \begin{cases} d-1, & \text{if } i = j \text{ and } i, j \neq n \\ -1, & \text{if } i \neq j \text{ and } i, j \neq n \\ 0, & \text{otherwise} \end{cases}. \tag{4}$$

When the estimated graph $\hat{G}$ has cycles but contains some correct ordering information, Eq. (4) ensures that the approximated value $\tilde{p}$ encodes a proper topological ordering of the agents. W is optimized with the fixed $\tilde{p}$:

$$\nabla_{\theta_\upsilon} L(\theta_\upsilon) = \mathbb{E}\left[\nabla_{\hat{G}} Q(s, \hat{G})|_{\hat{G}=\upsilon(W,\tilde{p})} \nabla_{\theta_\upsilon} \upsilon(W, \tilde{p})\right]. \tag{5}$$

Only elements in the upper triangular matrix of W are optimized to enforce the skew-symmetric property. Finally, we eliminate cycles in G by minimizing the following loss:

$$L(\theta_{\pi^g}) = \sum_{b=1}^{B} \left[\pi^g(\boldsymbol{\tau}, \hat{\mathbf{u}}; \theta_{\pi^g}) - \upsilon(W, \tilde{p})\right]^2, \tag{6}$$

where B denotes the size of the sampled minibatch.

4.2 Value-based and Policy-based DHCG

To demonstrate the adaptability and effectiveness of the learned graphs, we propose value-based and policy-based DHCG to train the agent network, respectively.

Value-based DHCG (DHCG-Q). We combine deep coordination graph [Böhmer *et al.*, 2020] with the learned communication graph $G = (V, E)$, where the joint Q-value function is factorized to the summation of individual utility functions q_i^ϑ and pairwise payoff functions q_{ij}^φ:

$$Q(s, \boldsymbol{\tau}, \boldsymbol{u}, G|\vartheta, \varphi, \omega) := \frac{1}{|V|}\sum_{i=1}^{n} q_i^\vartheta(u_i|\tau_i) + \frac{1}{|E|}\sum_{\{i,j\}\in E} q_{ij}^\varphi(u_i, u_j|\tau_i, \tau_j) + v^\omega(s), \tag{7}$$

where $v^{\omega}(s)$ is a state-based bias. DHCG-Q is a graph-based value factorization method that can deduce the contribution of each agent, where the property of DAGs reduces communication costs and guarantees to converge to a fixed point after a finite number of steps. In addition, the graph G is message-dependent and thus provides more flexibility in the task that involves multiple states.

Remark. The coordination graph always constructs the dependency relations between agents as static and undirected graphs. There are no guarantees that graphs with cycles will converge based on the analysis of the maximum a posteriori problem in graphical models [Pearl, 1989; Wainwright *et al.*, 2002; Wainwright *et al.*, 2004].

Policy-based DHCG (DHCG-P). Based on the learned graph G, each agent i can determine the set of its ancestor agents $L(i)$. The joint policy $\boldsymbol{\pi}^{\theta}(\mathbf{u}, |\boldsymbol{\tau}, \mathbf{m})$ is factorized in an auto-regressive manner:

$$\boldsymbol{\pi}^{\theta}(\mathbf{u}|\boldsymbol{\tau}, \mathbf{m}) = \prod_{i=1}^{n} \pi_i^{\theta}(u_i|\tau_i, m_i), \tag{8}$$

where θ denotes the shared parameter of the agent network, $\boldsymbol{\tau} = \{\tau_i\}_{i=1}^{n}$ denotes the joint observation-action history, $\mathbf{m} = \{m_i\}_{i=1}^{n}$ is the messages, $m_i = \cup_{j \in L(i)}\{u_j \oplus W_{ij}^{s}\tau_j\}$ contains the intention and the weighted observation-action history from its ancestors, and W^s denotes a self-attention module to aggregate the observation-action histories.

In the training stage, the agent network is optimized by minimizing the following PPO-clip objective of:

$$-\frac{1}{Tn}\sum_{i=1}^{n}\sum_{t=0}^{T-1} \min[\eta_i^t(\theta)\hat{A}^t, \mathrm{clip}(\eta_i^t(\theta), 1 \pm \epsilon)\hat{A}^t], \tag{9}$$

where $\eta_i^t(\theta) = \frac{\pi_i^{\theta}(u_i^t|\tau_i^t, m_i^t)}{\pi_i^{\theta_{\mathrm{old}}}(u_i^t|\tau_i^t, m_i^t)}$, $\hat{A}^t = \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}^{V}$ is an approximated value of the joint advantage function based on the generalized advantage estimation [Schulman *et al.*, 2016], $\delta_{t+l}^{V} = r^t + \gamma V(\boldsymbol{\tau}^{t+l+1}) - V(\boldsymbol{\tau}^{t+l})$, $V(\boldsymbol{\tau}^t) = \frac{1}{n}\sum_{i=1}^{n} V^{\phi}(\tau_i^t)$ is the joint value function, and $V^{\phi}(\tau_i^t)$ is the individual value function. $V^{\phi}(\tau_i^t)$ is optimized to minimize the following empirical Bellman error:

$$L(\phi) = \frac{1}{Tn}\sum_{i=1}^{n}\sum_{t=0}^{T-1}\left[r^t + \gamma V^{\phi'}(\tau_i^{t+1}) - V^{\phi}(\tau_i^t)\right]^2, \tag{10}$$

where ϕ' is the parameter of the non-differentiable target network. Since the outputs of all actions has already been collected in the replay buffer, the agent network can be optimized in parallel during training.

Remark. Auto-regressive learning is a minimal approximation of centralized learning. Any optimal joint policy can be factorized in an auto-regressive manner. However, the agent-by-agent optimal learned by the auto-regressive scheme may not be the global optimal [Bertsekas, 2019].

5 Related Work

In recent years, end-to-end learning with differentiable communication protocols has become an active area in cooperative multi-agent reinforcement learning (MARL). CommNet [Sukhbaatar *et al.*, 2016] uses continuous channels, averaging message vectors and sending them to each agent. ATOC [Jiang and Lu, 2018] proposes a communication model based on the hard attention mechanism to learn when to communicate and integrate information from others. TarMAC [Das *et al.*, 2019] applies a soft attention matrix to aggregate messages. GA-Comm [Liu *et al.*, 2020] argues that soft attention makes the agent still rely on irrelevant agents' messages and proposes a game abstraction mechanism to extract relationships. However, these methods mainly focus on sharing observations effectively to solve partially observable problems and use the same policy update scheme as independent learning with individual rewards. As a result, they still suffer from the non-stationarity problem in MARL.

To exploit communication in the team reward setting, VBC [Zhang *et al.*, 2019b] enables the agents to send communication requests and reply to others adaptively based on their confidence about local decisions, which reduces communication costs. NDQ [Wang *et al.*, 2019] uses message entropy and mutual information for shortening the message and reducing the uncertainty of the receiver's Q-value. MAIC [Yuan *et al.*, 2022] learns targeted teammate models, with which each agent can generate incentive messages to specific agents and bias their value functions directly. To achieve better credit assignment and coordination, these methods use mixing networks to approximate the joint Q-value function by aggregating individual Q-value functions in an additive or a monotonic way. Despite claiming their methods reduce communication costs, they can only represent the same class of joint value functions as the mixing function they used, which cannot cope with the task that an agent's ordering over its actions depends on others' actions [Rashid *et al.*, 2020; Wang *et al.*, 2020]. Consequently, they cannot solve tasks that require significant coordination within a given timestep, e.g., *relative overgeneralization* pathology.

A simple but effective way to solve this limitation is to enlarge the representational capacity of value functions by explicitly integrating the intentions into the message. IS [Kim *et al.*, 2021] allows the agents to model the environment dynamics to predict their imaginary paths and share intentions with others by an attention module. However, the softmax function in attention modules encourages agents to be fully connected, which generates redundant information for policy learning. Fu et al. [2022] propose auto-regressive policy learning where the action produced by each agent depends on its observation and all the actions from its previous agents under a specified agent-by-agent execution order. Similarly, MAT [Wen *et al.*, 2022] uses an encoder-decoder architecture and transforms multi-agent joint policy optimization into a sequence modeling process. Although sharing intentions in the agent-by-agent manner is a simple but effective method to achieve greater representational capacity compared with independent learning, it can only converge to one of the Nash equilibriums rather than the global optimum.

Coordination graph (CG) [Guestrin *et al.*, 2002] is another representative method to share intentions during communication. CG decomposes the joint Q-value into individual utilities and payoff contributions based on the intention of the agents connected by the hyper-edges. Deep coordination graph [Böhmer *et al.*, 2020] considers a static graph connecting all pairs of agents. This graph structure has high representational capacity in centralized Q-values but raises a challenge for computation in the execution phase. However, Zhang et al. [2013] suggest that the graph could also depend on states, which means each state can have its own unique CG. DICG [Li *et al.*, 2021] applies the attention mechanism to learn the appropriate message-dependent coordination graph structure with soft edge weights. CASEC [Wang *et al.*, 2022] uses the variance of payoff functions to construct context-aware sparse coordination topologies. SOP-CG [Yang *et al.*, 2022] also employs dynamic graph topology and uses structured graph classes to guarantee accuracy and computational efficiency. However, these methods model the relations between agents and undirected graphs, and the parallel message-passing update in such graphs cannot guarantee convergence.

Relationship to MAT. MAT [Wen *et al.*, 2022] and DHCG-P formulate the joint policy optimization as a sequence modeling process. MAT randomly chooses a permutation of agents as the update order, which is hard to scale to tasks with complex reward and state transitions. In contrast, we view the selection of the proper communication graph as an action and train it in an end-to-end fashion in DHCG-P, adding more flexibility in tasks involving multiple states.

Relationship to SOP-CG. SOP-CG [Yang *et al.*, 2022] and DHCG-Q apply dynamic graph into coordination graph method [Guestrin *et al.*, 2002]. SOP-CG focuses on the polynomial-time greedy policy execution and models the state-dependent graph as undirected graphs, which cannot guarantee convergence. In contrast, DHCG-Q formulates the dependency relations as directed acyclic graphs, where each edge denotes the direction of intention propagation for connected agents. The acyclic property guarantees convergence and cuts off redundant information, reducing communication costs and improving coordination performance.

6 Results

In this section, we conduct empirical experiments to answer the following questions: (1) Is Deep Hierarchical Communication Graph (DHCG) better than the existing MARL methods in scenarios with complex reward and transition dependency among cooperative agents? (2) Can DHCG outperforms the pre-defined topologies or existing graph-based methods? (3) How does DHCG differ from communication-enabled algorithms? (4) Can DHCG generate different graphs to adapt to different situations? All figures in the experiments are plotted using mean and standard deviation with confidence internal 95%. We conduct five independent runs with different random seeds for each learning curve.

6.1 Performance Comparison

In this section, we compare the performance of MAPPO [Yu *et al.*, 2022], HAPPO [Kuba *et al.*, 2022], QMIX [Rashid *et al.*, 2018], DCG [Böhmer *et al.*, 2020], CASEC, SOP-CG [Yang *et al.*, 2022], and DHCG on Predator-Prey [Son *et al.*, 2019], Multi-Agent Coordination Challenge

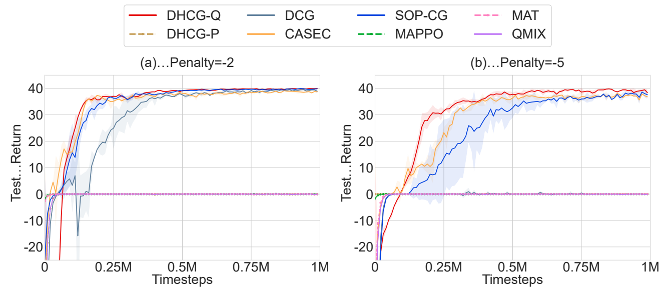

Figure 3: Performance comparisons on Predator-Prey with different penalties.

(MACO) [Wang *et al.*, 2022], and StarCraft Multi-Agent Challenge (SMAC) [Samvelyan *et al.*, 2019]. The line style is dotted with circle marks for policy-based algorithms and is solid for value-based algorithms.

Predator-Prey is a partially observable environment containing eight predators (agents) and eight prey in a 10×10 grid world. Each agent observes a 5×5 sub-grid around it and can perform five actions, i.e., up, down, left, right, and catch. When two agents surround a prey but only one tries to catch it, the team receives a miscoordination penalty $p < 0$. By contrast, they earn a bonus of 10 if they catch simultaneously. After a successful catch, the catching agents and prey will be removed from the grid.

The results on predator-prey are illustrated in Fig. 3. DHCG-Q, CASEC, SOP-CG, and DCG can learn the optimal policy when the miscoordination penalty is -2. MAT, MAPPO, and DHCG-P fail to solve this task because the agent-by-agent optimal learned in an auto-regressive manner may not be the global optimal, which is consistent with our analysis in Section 4.2. QMIX also shows negative results because it can only represent a restricted space due to the monotonic constraints on the joint Q-value function and the individual Q-value functions. CASEC and SOP-CG learn slowly and become unstable with the penalty increase, while DCG completely fails to solve the task. In contrast, DHCG-Q outperforms all baselines with a considerable gap because it ensures convergence and cuts off redundant communication edges by directed graphs. We also visualize the coordination structures learned by DHCG in Section 6.4 to show the adaptability of the deep hierarchical communication graphs.

The MACO benchmark raises challenges of partial observability and relative overgeneralization pathology. Since the reward observed by an agent is highly related to the actions of others, the learning of decentralized policies is unstable due to the exploration of other agents. The proper credit assignment is necessary to solve this issue. In Fig. 4, we can see that policy-based methods do not perform well on these tasks. Although DHCG-P cannot solve Aloha and Gather, it still outperforms both MAPPO and MAT in Disperse and Hallway by a considerable gap. In contrast, DHCG-Q achieves the best performance in all four tasks, highlighting the effectiveness of directed acyclicity in coordination graphs.

We also compare DHCG with baselines on the SMAC benchmark. We use an ϵ-greedy exploration scheme, where ϵ decreases from 1 to 0.05 over 50 thousand timesteps in

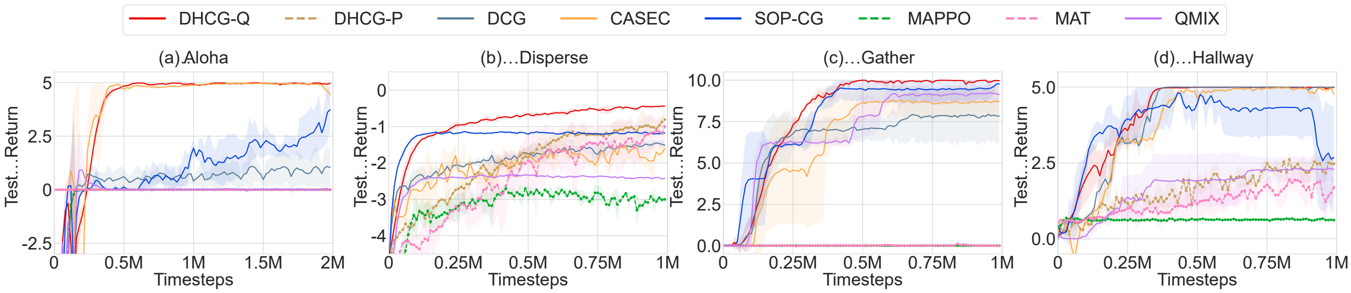

Figure 4: Performance comparisons on the Multi-Agent Coordination Challenge benchmark.

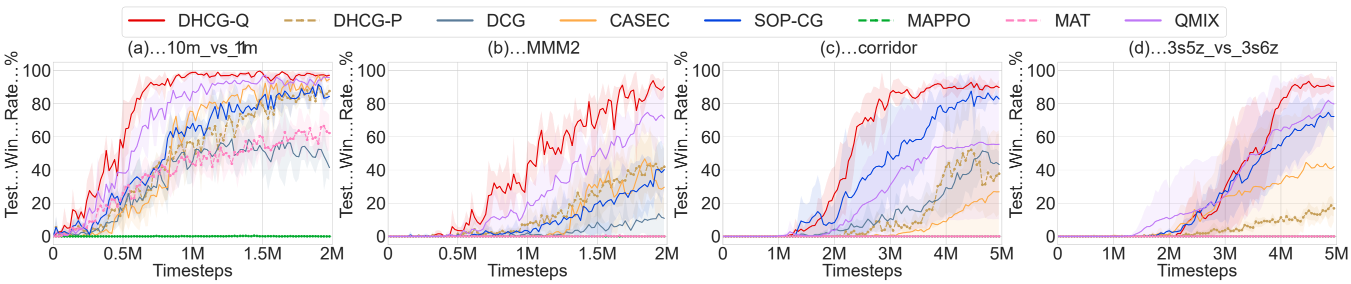

Figure 5: Performance comparisons on the StarCraft Multi-Agent Challenge benchmark.

`10m_vs_11m` and `MMM2`, and over 1 million timesteps in `corridor` and `3s5z_vs_3s6z`. Fig. 5 shows that DHCG-P performs better than MAT and MAPPO. In addition, DHCG-Q significantly outperforms all baselines, indicating that intention sharing through message-dependent directed acyclic graphs can improve learning speed and coordination performance in more complex tasks.

Line	$G := \{\{i, j\} \mid 1 \leqslant i < n, i < j \leqslant n\}$
Cycle	$G := \{\{i, (i \mod n) + 1\} \mid 1 \leqslant i \leqslant n\}$
Star	$G := \{\{1, i\} \mid 2 \leqslant i \leqslant n\}$
Soft	$G = \text{Attention}(\tau_1, ..., \tau_n)$
GA	$G = \text{G2ANet}(\tau_1, ..., \tau_n)$ [Liu *et al.*, 2020]

Table 1: Compared graph topologies.

6.2 Ablation Study

In this section, we conduct ablation studies to demonstrate the superiority of the hierarchical communication graph. We compare it with three static graphs (Line, Cycle, and Star), a random graph (Random), and two trainable topologies (Soft and GA). The definition of these topologies is shown in Tab. 1. During communication, the agents share their intentions based on these graphs.

As shown in Fig. 6, the agents in the pre-defined graphs can benefit from the intentions of the ancestors when the topologies match the ground truth execution order demands. However, when such static graphs cannot characterize the task, the structure introduces more redundant information and harms policy learning. As a result, Line does not perform well in these tasks despite having acyclicity. Since the graph with cycles cannot guarantee to converge, Cycle, Star, and Random show poor performance in `10m_vs_11m`, `5m_vs_6m`, and `MMM2`. In addition, Soft and GA perform lower than DHCG-Q with a considerable gap in `10m_vs_11m` and `MMM2` because they cannot learn acyclic graphs by the attention module without any constraints and may not converge when the task involves multiple optimums. In contrast, benefitting from the adaptability and the acyclicity, DHCG-Q cuts off redundant communication edges and outperforms these communication structures across all tasks.

6.3 Comparison with Communication Algorithms

In this section, we investigate the contribution of sharing intention with DHCG by comparing our method with more communication-enabled MARL algorithms, including CommNet [Sukhbaatar *et al.*, 2016], TarMAC [Das *et al.*, 2019], NDQ [Wang *et al.*, 2019], GA-Comm [Liu *et al.*, 2020], IS [Kim *et al.*, 2021], MAIC [Yuan *et al.*, 2022], and a leader-follower algorithm EBPG [Shi *et al.*, 2019]. Due to the team reward setting in SMAC, we combine CommNet, TarMAC, GA-Comm, IS, and EBPG with QMIX [Rashid *et al.*, 2018] to achieve credit assignment.

As shown in Fig. 7, CommNet, TarMAC, and GA-Comm perform poorly in complex coordination tasks because they ignore the importance of intentions in policy learning and have very limited expressiveness for value functions. NDQ uses mutual information to reduce non-stationarity. MAIC utilizes teammate representation to bias the other agent's Q-values, which is optimized by maximizing the mutual information between the action and the random variable of the teammate model distribution. However, their poor performance indicates that the additional entropy loss is not feasible and reliable in complex tasks. IS also fails to solve the task even though it propagates the intention through a soft-attention module, suggesting the difficulties in learning acyclic graphs by soft-attention modules without any con-

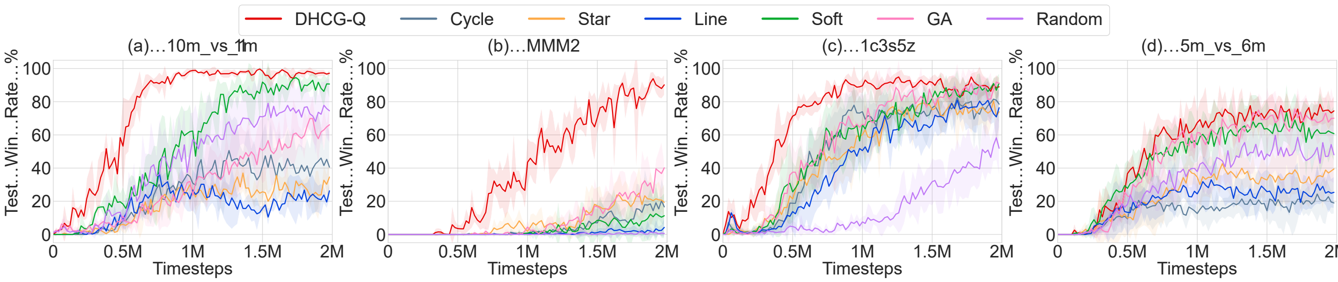

Figure 6: Ablation study on the StarCraft Multi-agent Challenge benchmark.

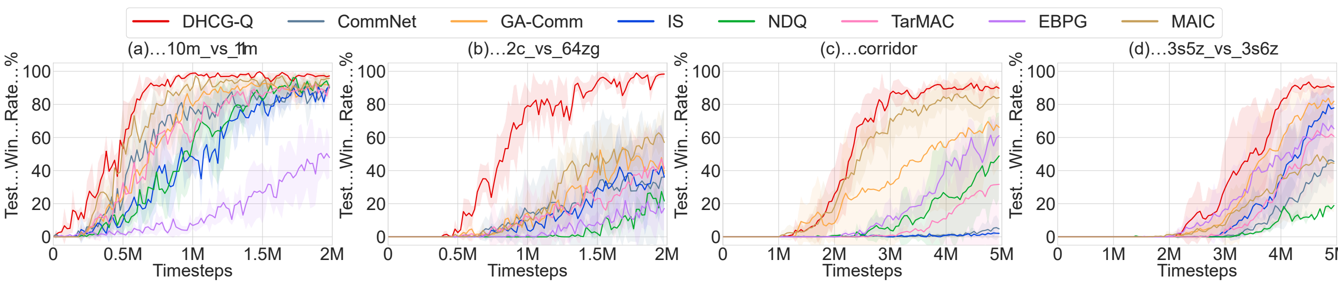

Figure 7: Performance comparisons with different communication protocols on the StarCraft Multi-agent Challenge benchmark.

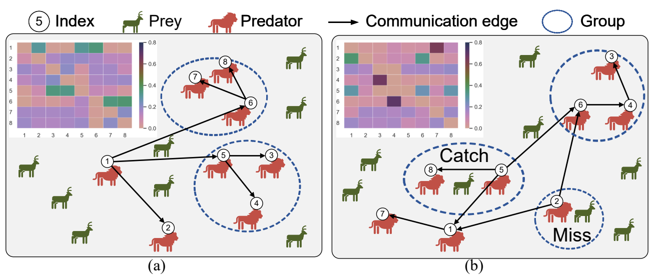

Figure 8: Communication graphs learned by DHCG-Q on Predator-Prey: (a) Grouping at initialization; (b) Sending necessary intentions at a state that requires significant coordination.

straints. In contrast, DHCG outperforms the baselines with a considerable gap in all tasks.

6.4 Visualization of Communication Graph

To show the adaptability of the deep hierarchical communication graph, we visualize the learned communication graph structures and the heatmap on Predator-Prey in Fig. 8.

We take two states in an episode as an example. Fig. 8-a presents that the predators (agents) and prey are generated randomly on the map at initialization, where agents naturally form different groups to chase prey. After several steps, as illustrated in Fig. 8-b, the prey in the center is surrounded by two predators. The follower agent would execute "catch" after knowing its ancestor intends to catch the prey and "move" otherwise. Meanwhile, the predator in the bottom right corner also propagates its intention because it encounters prey alone, and the "catch" action will lead to a miscoordination penalty. Using such dependency relations, agents know the necessary intentions at a state that requires significant coordination, i.e., the joint action would lead to various states and rewards. As a result, the joint Q-value can be factorized to individual Q-value functions and pairwise payoff functions, yielding a proper credit assignment for agents.

Compared with fully-connected graphs, the hierarchical communication graph restricts the joint Q-value function expressiveness because part of the action inputs is removed. However, Fig. 8 shows that the learned graphs approximates the comprehensible dependency relations between agents and only cuts off redundant messages, improving the communication efficiency and the algorithm's sample complexity.

7 Conclusion and Future Work

This paper proposes Deep Hierarchical Communication Graph (DHCG), a novel multi-agent graph-based method that guarantees convergence and improves coordination performance by sharing intentions through directed acyclic communication graphs. To enable end-to-end learning for the communication graph, a critic with an intrinsic reward for acyclicity is applied to evaluate the value of the graph. We also project the learned graph to an admissible set of directed acyclic graphs to eliminate cycles. To show the effectiveness and the adaptability of the learned graphs, we propose policy-based and value-based DHCG to train the agent network. The policy-based algorithm factorizes the joint policy in an auto-regressive manner. The value-based algorithm factorizes the joint Q-value function in the individual Q functions and pairwise payoff functions. The results show that DHCG can learn interpretable dependency relations and improve performance on several benchmarks. Since sharing intentions is time-consuming when the number of agents is large, predicting others' intentions based on dependency relations will be an interesting future direction.

Acknowledgments

This work was supported in part by National Key R&D Program of China under grant No. 2021ZD0112700, NSFC under grant No. 62125305, No. 62088102, and No. 61973246, the Fundamental Research Funds for the Central Universities under Grant xtr072022001.

Contribution Statement

Zeyang Liu contributed the central idea, designed the methodology, analyzed data, performed visualization, and wrote the initial draft of the paper. Lipeng Wan refined the methodology, discussed the results, and revised the manuscript. Xue Sui, Zhuoran Chen, and Kewu Sun conducted experiments and performed data curation. Xuguang Lan contributed to refining the ideas, oversight the research activity planning and execution, and finalized this paper.

References

[Bargiacchi *et al.*, 2018] Eugenio Bargiacchi, Timothy Verstraeten, Diederik Roijers, Ann Nowé, and Hado Hasselt. Learning to coordinate with coordination graphs in repeated single-stage multi-agent decision problems. In *International Conference on Machine Learning*, pages 482–490. PMLR, 2018.

[Bertsekas, 2019] Dimitri P. Bertsekas. Multiagent rollout algorithms and reinforcement learning. *CoRR*, abs/1910.00120, 2019.

[Böhmer *et al.*, 2020] Wendelin Böhmer, Vitaly Kurin, and Shimon Whiteson. Deep coordination graphs. In *International Conference on Machine Learning*, pages 980–991. PMLR, 2020.

[Conte and Sichman, 2002] Rosaria Conte and Jaime Simão Sichman. Dependence graphs: Dependence within and between groups. *Computational & Mathematical Organization Theory*, 8(2):87–112, 2002.

[Das *et al.*, 2019] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning*, pages 1538–1546. PMLR, 2019.

[Dimakopoulou and Van Roy, 2018] Maria Dimakopoulou and Benjamin Van Roy. Coordinated exploration in concurrent reinforcement learning. In *International Conference on Machine Learning*, pages 1271–1279. PMLR, 2018.

[Dimakopoulou *et al.*, 2018] Maria Dimakopoulou, Ian Osband, and Benjamin Van Roy. Scalable coordinated exploration in concurrent reinforcement learning. *Advances in Neural Information Processing Systems*, 31:4223–4232, 2018.

[Fu *et al.*, 2022] Wei Fu, Chao Yu, Zelai Xu, Jiaqi Yang, and Yi Wu. Revisiting some common practices in cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 6863–6877. PMLR, 2022.

[Guestrin *et al.*, 2002] Carlos Guestrin, Michail Lagoudakis, and Ronald Parr. Coordinated reinforcement learning. In *International Conference on Machine Learning*, pages 227–234. Citeseer, 2002.

[Hernandez-Leal *et al.*, 2019] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 750–797, 2019.

[Jiang and Lu, 2018] Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *Advances in Neural Information Processing Systems*, 31:7254–7264, 2018.

[Kim *et al.*, 2021] Woojun Kim, Jongeui Park, and Youngchul Sung. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations*, 2021.

[Kuba *et al.*, 2022] Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2022.

[Li *et al.*, 2019] Xihan Li, Jia Zhang, Jiang Bian, Yunhai Tong, and Tie-Yan Liu. A cooperative multi-agent reinforcement learning framework for resource balancing in complex logistics network. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 980–988, 2019.

[Li *et al.*, 2021] Sheng Li, Jayesh K Gupta, Peter Morales, Ross Allen, and Mykel J Kochenderfer. Deep implicit coordination graphs for multi-agent reinforcement learning. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 764–772, 2021.

[Lillicrap *et al.*, 2016] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

[Liu *et al.*, 2020] Yong Liu, Weixun Wang, Yujing Hu, Jianye Hao, Xingguo Chen, and Yang Gao. Multi-agent game abstraction via graph attention neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 34(05):7211–7218, 2020.

[Oliehoek and Amato, 2016] Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. Springer Briefs in Intelligent Systems. Springer, 2016.

[Palanisamy, 2020] Praveen Palanisamy. Multi-agent connected autonomous driving using deep reinforcement learning. In *International Joint Conference on Neural Networks*, pages 1–7. IEEE, 2020.

[Pearl, 1989] Judea Pearl. *Probabilistic reasoning in intelligent systems - networks of plausible inference*. Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann, 1989.

[Rashid *et al.*, 2018] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 4295–4304. PMLR, 2018.

[Rashid *et al.*, 2020] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 33:10199–10210, 2020.

[Samvelyan *et al.*, 2019] Mikayel Samvelyan, Tabish Rashid, Christian Schröder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob N Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 2186–2188, 2019.

[Schulman *et al.*, 2016] John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[Shi *et al.*, 2019] Zhenyu Shi, Runsheng Yu, Xinrun Wang, Rundong Wang, Youzhi Zhang, Hanjiang Lai, and Bo An. Learning expensive coordination: An event-based deep rl approach. In *International Conference on Learning Representations*, 2019.

[Son *et al.*, 2019] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 5887–5896. PMLR, 2019.

[Sukhbaatar *et al.*, 2016] Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. *Advances in Neural Information Processing Systems*, 29, 2016.

[Wainwright *et al.*, 2002] Martin J Wainwright, Tommi Jaakkola, and Alan Willsky. Exact map estimates by (hyper) tree agreement. *Advances in Neural Information Processing Systems*, 15, 2002.

[Wainwright *et al.*, 2004] Martin Wainwright, Tommi Jaakkola, and Alan Willsky. Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and computing*, 14(2):143–166, 2004.

[Wang *et al.*, 2019] Tonghan Wang, Jianhao Wang, Chongyi Zheng, and Chongjie Zhang. Learning nearly decomposable value functions via communication minimization. In *International Conference on Learning Representations*, 2019.

[Wang *et al.*, 2020] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations*, 2020.

[Wang *et al.*, 2022] Tonghan Wang, Liang Zeng, Weijun Dong, Qianlan Yang, Yang Yu, and Chongjie Zhang. Context-aware sparse deep coordination graphs. In *International Conference on Learning Representations*, 2022.

[Wen *et al.*, 2022] Muning Wen, Jakub Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. *Advances in Neural Information Processing Systems*, 35:16509–16521, 2022.

[Yang *et al.*, 2022] Qianlan Yang, Weijun Dong, Zhizhou Ren, Jianhao Wang, Tonghan Wang, and Chongjie Zhang. Self-organized polynomial-time coordination graphs. In *International Conference on Machine Learning*, pages 24963–24979. PMLR, 2022.

[Yu *et al.*, 2021] Yue Yu, Tian Gao, Naiyu Yin, and Qiang Ji. Dags with no curl: An efficient dag structure learning approach. In *International Conference on Machine Learning*, pages 12156–12166. PMLR, 2021.

[Yu *et al.*, 2022] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624, 2022.

[Yuan *et al.*, 2022] Lei Yuan, Jianhao Wang, Fuxiang Zhang, Chenghe Wang, Zongzhang Zhang, Yang Yu, and Chongjie Zhang. Multi-agent incentive communication via decentralized teammate modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 36(9):9466–9474, 2022.

[Zhang and Lesser, 2013] Chongjie Zhang and Victor Lesser. Coordinating multi-agent reinforcement learning with limited communication. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 1101–1108, 2013.

[Zhang *et al.*, 2019a] Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *The world wide web Conference*, pages 3620–3624, 2019.

[Zhang *et al.*, 2019b] Sai Qian Zhang, Qi Zhang, and Jieyu Lin. Efficient communication in multi-agent reinforcement learning via variance based control. *Advances in Neural Information Processing Systems*, 32:3230–3239, 2019.

[Zheng *et al.*, 2018] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. *Advances in Neural Information Processing Systems*, 31:9492–9503, 2018.