Self-Organized Polynomial-time Coordination Graphs

Qianlan Yang*, Weijun Dong*, Zhizhou Ren*,
Jianhao Wang, Tonghan Wang, Chongjie Zhang

Machine Intelligence Group

清華大學 交叉信息研究院
Tsinghua University Institute for Interdisciplinary Information Sciences

Common Methods for Cooperative Multi-agent RL

Fully decentralized control policies

- An agent's behavior only depends on its local observation history.
- Lack of explicit coordination.

Common Methods for Cooperative Multi-agent RL

Fully decentralized control policies

- An agent's behavior only depends on its local observation history.
- Lack of explicit coordination.

Coordination graphs

- Explicitly represent coordination relations by higher-order value factorization.
- Agents communicate to jointly optimize their actions.

Deep Coordination Graphs

DCG defines the joint value factorization upon a coordination graph G:

$$Q(\boldsymbol{\tau}^{(t)}, \boldsymbol{a}; G) = \sum_{i\in[n]} q_i(\tau_i^{(t)}, a_i) + \sum_{(i,j)\in G} q_{ij}(\tau_i^{(t)}, \tau_j^{(t)}, a_i, a_j) \tag{1}$$

where $\tau_i^{(t)}$ is the observation-action history of agent i.

Deep Coordination Graphs

DCG defines the joint value factorization upon a coordination graph G:

$$Q(\boldsymbol{\tau}^{(t)}, \boldsymbol{a}; G) = \sum_{i \in [n]} q_i(\tau_i^{(t)}, a_i) + \sum_{(i,j) \in G} q_{ij}(\tau_i^{(t)}, \tau_j^{(t)}, a_i, a_j) \tag{1}$$

where $\tau_i^{(t)}$ is the observation-action history of agent i.

Computing joint actions with maximum value can be modeled as a distributed constraint optimization problem (DCOP).

Coordiantion Graphs

- The default implementation of DCG uses complete graphs. However, the DCOPs induced by such graphs and their approximations are NP-hard problems.

Coordiantion Graphs

- The default implementation of DCG uses complete graphs. However, the DCOPs induced by such graphs and their approximations are NP-hard problems.
- DCG selects actions by a heuristic algorithm called max-sum.

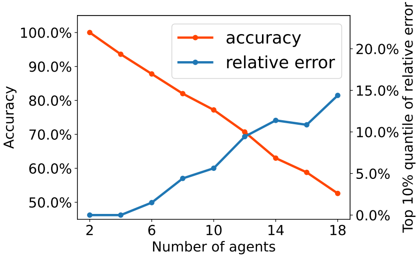

Figure 1: A motivating example with the accuracy and the relative joint Q error of max-sum algorithm w.r.t. the number of agents.

Polynomial-time Coordination Graphs Class

Definition (Polynomial-Time Coordination Graph Class)

Let n be the number of agents and $A = |\bigcup_{i=1}^{n} A^i|$. We say a graph class $\mathcal{G}$ is a *Polynomial-Time Coordination Graph Class* if there exists an algorithm that can solve any induced DCOP of any coordination graph $G \in \mathcal{G}$ in $\mathrm{Poly}(n, A)$ running time.

- The set of undirected acyclic graphs $\mathcal{G}_{\mathrm{uac}}$ is a polynomial-time coordination graph class.

Polynomial-time Coordination Graphs Class

Definition (Polynomial-Time Coordination Graph Class)

Let n be the number of agents and $A = |\bigcup_{i=1}^{n} A^i|$. We say a graph class $\mathcal{G}$ is a *Polynomial-Time Coordination Graph Class* if there exists an algorithm that can solve any induced DCOP of any coordination graph $G \in \mathcal{G}$ in $\mathrm{Poly}(n, A)$ running time.

- The set of undirected acyclic graphs $\mathcal{G}_{\mathrm{uac}}$ is a polynomial-time coordination graph class.
- However, the coordination relationship among agents may not be characterized by a static sparse coordination graph.

Polynomial-time Coordination Graphs Class

Definition (Polynomial-Time Coordination Graph Class)

Let n be the number of agents and $A = |\bigcup_{i=1}^{n} A^i|$. We say a graph class $\mathcal{G}$ is a *Polynomial-Time Coordination Graph Class* if there exists an algorithm that can solve any induced DCOP of any coordination graph $G \in \mathcal{G}$ in $\mathrm{Poly}(n, A)$ running time.

- The set of undirected acyclic graphs $\mathcal{G}_{\text{uac}}$ is a polynomial-time coordination graph class.
- However, the coordination relationship among agents may not be characterized by a static sparse coordination graph.

Use a state-dependent coordination graph!

- Given different environmental states, the joint values can be factorized with different coordination graphs chosen from a predefined graph class $\mathcal{G} \subseteq \mathcal{G}_{\text{uac}}$.

Learning Self-Organized Topology with An Imaginary Coordinator

$$Q(\boldsymbol{\tau}^{(t)}, \boldsymbol{a}; G) = \sum_{i \in [n]} q_i(\tau_i^{(t)}, a_i) + \sum_{(i,j) \in G} q_{ij}(\tau_i^{(t)}, \tau_j^{(t)}, a_i, a_j) \tag{1}$$

Consider an imaginary coordinator agent whose action space refers to the selection of graph topologies. The value factorization structure naturally serves a utility function for the coordinator agent to select graphs:

$$G^{(t)} \leftarrow \arg\max_{G \in \mathcal{G}} \left(\max_{\boldsymbol{a}} Q(\boldsymbol{\tau}^{(t)}, \boldsymbol{a}; G) \right). \tag{2}$$

Learning Self-Organized Topology with An Imaginary Coordinator

$$Q(\boldsymbol{\tau}^{(t)}, \boldsymbol{a}; G) = \sum_{i \in [n]} q_i(\tau_i^{(t)}, a_i) + \sum_{(i,j) \in G} q_{ij}(\tau_i^{(t)}, \tau_j^{(t)}, a_i, a_j) \quad (1)$$

Consider an imaginary coordinator agent whose action space refers to the selection of graph topologies. The value factorization structure naturally serves a utility function for the coordinator agent to select graphs:

$$G^{(t)} \leftarrow \arg\max_{G \in \mathcal{G}} \left(\max_{\boldsymbol{a}} Q(\boldsymbol{\tau}^{(t)}, \boldsymbol{a}; G) \right). \quad (2)$$

We design two graph classes $\mathcal{G}_P$ and $\mathcal{G}_T$, which are subsets of $\mathcal{G}_{uac}$, so that the graph selection can be computed by combinatorial optimization techniques.

Temporal Difference Learning

Update the network parameters $\boldsymbol{\theta}$ by minimizing the Q-learning TD loss:

$$\mathcal{L}_{cg}(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{(\boldsymbol{\tau},\boldsymbol{a},G,r,\boldsymbol{\tau}')\sim\mathcal{D}}\left[\left(y_{cg} - Q(\boldsymbol{\tau},\boldsymbol{a};G;\boldsymbol{\theta})\right)^2\right] \tag{3}$$

where $y_{cg} = r + \gamma \max_{(\boldsymbol{a}',G')} Q(\boldsymbol{\tau}',\boldsymbol{a}';G';\boldsymbol{\theta}^-)$ is the one-step TD target.

Empirical Results

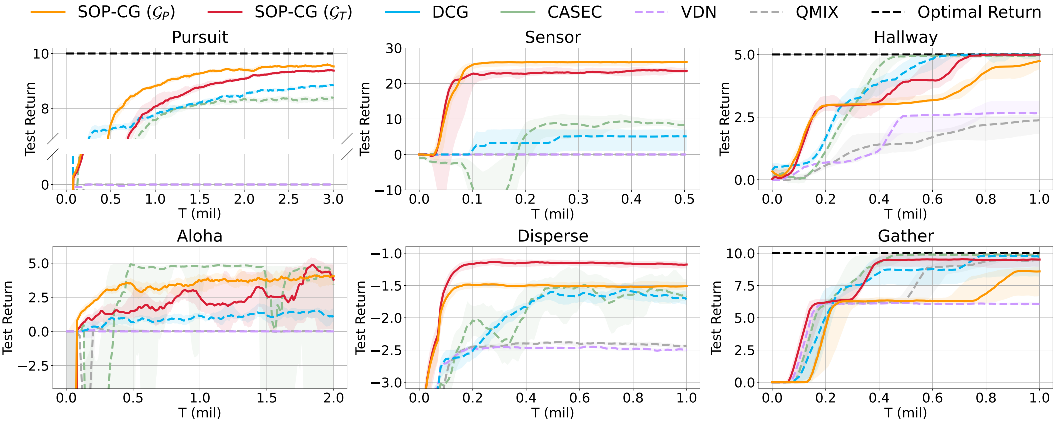

Figure 2: Learning curves on MACO benchmark.

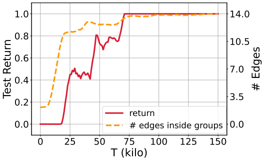

Figure 3: Illustrative example.

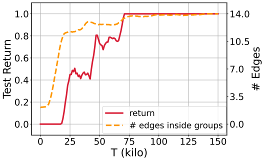

Figure 3: Illustrative example.

Graph Class	Select graph $G^{(t)}$	Select actions on a given graph
$\mathcal{G}_P$	$\surd$	$\surd$
$\mathcal{G}_T$	–	$\surd$
Complete graph	N/A	–

Figure 4: Graph Classes.

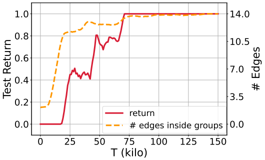

Figure 3: Illustrative example.

Graph Class	Select graph $G^{(t)}$	Select actions on a given graph
$\mathcal{G}_P$	√	√
$\mathcal{G}_T$	–	√
Complete graph	N/A	–

Figure 4: Graph Classes.

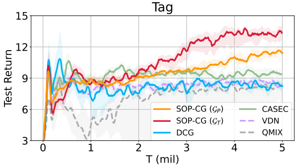

Figure 5: Evaluations on other testbeds.

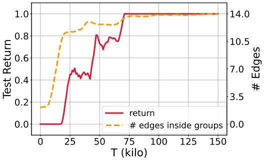

Figure 3: Illustrative example.

Graph Class	Select graph $G^{(t)}$	Select actions on a given graph
$\mathcal{G}_P$	$\surd$	$\surd$
$\mathcal{G}_T$	–	$\surd$
Complete graph	N/A	–

Figure 4: Graph Classes.

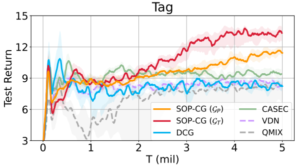

Figure 5: Evaluations on other testbeds.

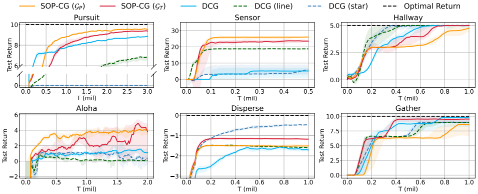

Figure 6: Ablation Study.

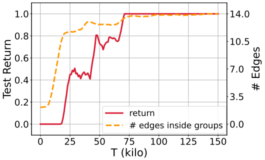

Figure 3: Illustrative example.

Graph Class	Select graph $G^{(t)}$	Select actions on a given graph
$\mathcal{G}_P$	√	√
$\mathcal{G}_T$	–	√
Complete graph	N/A	–

Figure 4: Graph Classes.

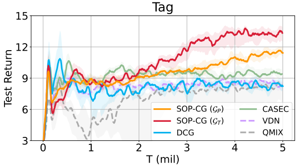

Figure 5: Evaluations on other testbeds.

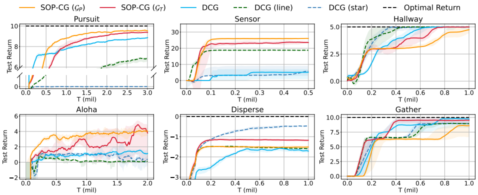

Figure 6: Ablation Study.

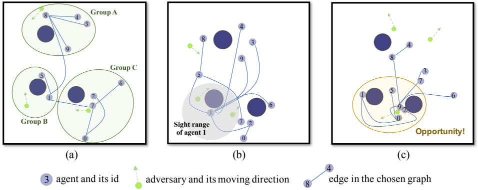

Figure 7: Visualization of Self-Organized Coordination.

Thanks for Listening!