2020 16th IEEE International Conference on Automation Science and Engineering (CASE)
August 20-21, 2020, Online Zoom Meeting

A Soft Graph Attention Reinforcement Learning for Multi-Agent Cooperation*

Huimu Wang[1] Zhiqiang Pu[1] Zhen Liu[2] Jianqiang Yi[1] and Tenghai Qiu[2]

Abstract— **The multi-agent reinforcement learning (MARL) suffers from several issues when it is applied to large-scale environments. Specifically, the communication among the agents is limited by the communication distance or bandwidth. Besides, the interactions among the agents are complex in large-scale environments, which makes each agent hard to take different influences of the other agents into consideration and to learn a stable policy. To address these issues, a soft graph attention reinforcement learning (SGA-RL) is proposed. By taking the advantage of the chain propagation characteristics of graph neural networks, stacked graph convolution layers can overcome the limitation of the communication and enlarge the agents' receptive field to promote the cooperation behavior among the agents. Moreover, unlike traditional multi-head attention mechanism which takes all the heads into consideration equally, a soft attention mechanism is designed to learn each attention head's importance, which means that each agent can learn how to treat the other agents' influence more effectively during large-scale environments. The results of the simulations indicate that the agents can learn stable and complicated cooperative strategies with SGA-RL in large-scale environments.**

I. INTRODUCTION

Cooperation in a multi-agent system has shown great successes in various fields, such as smart grid control [1], resource management [2], and games [3]. Multi-agent reinforcement learning (MARL) has been studied for a long time to promote the cooperation behavior in the multi-agent systems.

The huge potential indicated by deep reinforcement learning (DRL) [4]–[8] promotes the combination between DRL and MARL to solve complex problems in realistic large-scale environments. However, when these algorithms are applied to realistic environments, there are some limitations. First, the large number of agents causes the curse of dimensionality and the difficulty of learning a stable policy. Second, the information obtained from other agents is limited by the communications bandwidth and detective range, which especially affects the agents' cooperation behavior in large-scale environments. Finally, the number and the connection status of the agents keep changing over time, which leads to the issue of transferability.

Although a variety of MARL algorithms have been proposed to solve the issues above, they still suffer from different limitations. Some MARL algorithms [9], [10] follow a common paradigm of centralized learning with decentralized execution (CTDE) to promote the cooperation behavior among the agents. These algorithms suffer from the difficulty of transferability and scalability because they directly use the state or observation in constructing critic networks. Besides, the Mean-Field [11] can adapt to large-scale environments, but it ignores the fact that different agents' observation has different influences on their center agent. To address the limitation, the algorithms based on attention mechanisms [12]–[14] are proposed. They can effectively extract valuable information via the communication control. However, they are still limited by the communication bandwidth and ignore the underlying structure of the multi-agent system. Considering the structure of the multi-agent system, the algorithms based on graph network [15]–[17] take the graph structure into consideration, but they do not focus on the complex interaction among the agents, which makes them difficult to acquire satisfying performance in the large-scale environments.

To address the limitations mentioned above, a soft graph attention reinforcement learning (SGA-RL) is proposed. The key feature of SGA-RL lies in a communication enhanced network and a soft attention mechanism. The communication enhanced network mainly focuses on enlarging the communication field of the agents and obtaining more agents' information. It is designed based on graph attention networks (GAT) [18] to enlarge the agents receptive field or communication field through the chain propagation characteristics of graph neural networks. The soft attention mechanism mainly focuses on assigning different importance to different heads of GAT. Unlike traditional multi-head attention mechanism [19], a virtual head is designed to aggregate the states learned by the real heads. During the process of aggregation of the heads, each head is assigned with a specific importance weight, which can promote the cooperation behavior among the agents in large-scale environments.

SGA-RL is evaluated in different environments including formation control and obstacles avoidance. The simulation results demonstrate that the agents can learn stable and complicated cooperative strategies in large-scale environments.

*Research supported by the National Key Research and Development Program of China under Grant 2018AAA0102402 and Innovation Academy for Light-duty Gas Turbine, Chinese Academy of Sciences, No. CXYJJ19-ZD-02 and No. CXYJJ20-QN-05.

[1]Huimu Wang, Zhiqiang Pu and Jianqiang Yi are with Scholl of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049 China, and with Institute of Automation, Chinese Academy of Sciences, Beijing 100190 China. wanghuimu2018@ia.ac.cn, zhiqiang.pu@ia.ac.cn, jianqiang.yi@ia.ac.cn.

[2]Zhen Liu and Tenghai Qiu are with with Institute of Automation, Chinese Academy of Sciences, Beijing 100190 China. liuzhen@ia.ac.cn, tenghai.qiu@ia.ac.cn.

Authorized licensed use limited to: INSTITUTE OF AUTOMATION CAS. Downloaded on May 11,2021 at 02:10:48 UTC from IEEE Xplore. Restrictions apply.

II. PRELIMINARIES

A. Problem Definition

Let S_t denote the state of a two-dimensional environment at time t and o_i^t denote the local observation of agent i including its position, velocity. There are N agents and M obstacles in this environment. We assume that at time t, the position of agent i $p_i^t = [p_i^{t^x}, p_i^{t^y}]$, the velocity of agent i $v_i^t = [v_i^{t^x}, v_i^{t^y}]$, the formation center position $p_c^t = [p_c^{t^x}, p_c^{t^y}]$ and the position of obstacle j $p_{oj}^t = [p_{oj}^{t^x}, p_{oj}^{t^y}]$. Besides, the action space for each agent is discretized. The agent can move one step in both X and Y directions.

Moreover, the connected status among the agents can be represented in an undirected graph $G = (V, E)$. Specifically, $V = \{1, \ldots, N\}$ denotes the nodes consisting of the agents. $E \subseteq V \times V$ denotes the edge set consisting of communication status among the agents where an edge from node i to node j is denoted as $(i, j) \in E$. Besides, h is a set of node features, $h = \left\{ \overrightarrow{h}_1, \overrightarrow{h}_2, ..., \overrightarrow{h}_N \right\}, \overrightarrow{h}_i \in \mathbb{R}^F$, where F is the number of features in each node. Moreover, N_i is a set of neighbours communicating with node i in the graph. Only when the distance between agent i and agent j is less than c, agent j belongs to the set of neighbours N_i. As indicated by (1), there is an adjacency matrix A where $a_{ij} = 1$ if $j \in N_i$ otherwise $a_{ij} = 0$. Besides, the cooperation behavior is decided not only by its neighbourhoods' information but also by its own information. Therefore, there is a self-loop for each agent.

$$a_{ij} = \begin{cases} 1 & if\ dist\,(a_i, a_j) \leq c\ or\ i = j \\ 0 & if\ dist\,(a_i, a_j) > c \end{cases} \tag{1}$$

where *dist* is a 2-dimensional Euclidean norm to calculate the distance between agent i and agent j, and c represents the predefined communication threshold.

B. Reinforcement Learning

The problem in this paper is regarded as partially observable Markov Games which is an extension of the framework of Markov Games [20]. It is defined by a global state S, a set of actions $A_1, \cdots, A_N$, and a set of local observations $O_1, \ldots, O_N$. To choose actions, each agent uses a learnable policy $\pi_i : O_i \to P_a\,(A_i)$, which produces the next state according to the state transition function $T : S \times A_1 \times \cdots \times A_N \to P_t(S^{'})$. T defines the probability distribution over possible next states based on current states and actions for each agent. Each agent obtains rewards R_i from the environment after all agents take actions: $S \times A_1 \times \ldots \times A_N \to \mathbb{R}$. The agents aim to learn a policy that maximizes their expected discounted returns:

$$J_i(\pi_i) = E_{a_1 \sim \pi_1, \ldots, a_N \sim \pi_N, s \sim T} \left[\sum_{t=0}^{\infty} \gamma^t r_{it}\,(s_t, a_{1t}, \ldots, a_{Nt}) \right] \tag{2}$$

where r_{it} is the reward that agent i obtains at time t, a_{it} is the action that agent i takes at time t and s_t represents the global state S at time t. $\gamma \in [0, 1]$ is the discount factor that determines how much the policy favors immediate reward over long-term gain.

C. Proximal Policy Optimization (PPO)

Most policy gradient methods perform one gradient update per sampled trajectory, which results in high sample complexity. The proximal policy optimization algorithm (PPO) [21], which is parameterized by neural networks, is presented to address the problem. Let

$$l_t(\theta) = \frac{\pi_\theta\,(a_t\,|s_t\,)}{\pi_{\theta^k}(a_t\,|s_t\,)} \tag{3}$$

denote the likelihood ratio. θ represents the parameterized neural networks and π_{θ^k} represents the agent policy before k steps. Then PPO optimizes the objective function in the following term to learn a policy:

$$\begin{aligned} L(\theta) = E[&\min(l_t(\theta)\hat{A}_t^{\theta^k}(s_t, a_t), \\ &clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t^{\theta^k}(s_t, a_t)] \end{aligned} \tag{4}$$

where $\hat{A}_t^{\theta^k}(s_t, a_t)$ is the generalized advantage estimate and $clip\,(l_t\,(\theta), 1-\varepsilon, 1+\varepsilon)$ clips $l_t\,(\theta)$ in the interval $[1-\varepsilon, 1+\varepsilon]$.

III. METHOD

In this section, as shown in Fig. 1, SGA-RL, which is composed of three modules: graph generation, communication enhanced network (CEN) and policy optimization. As shown in Fig. 1, the observations of all the agents are fed into the multi-layer perception (MLP) to generate the basic graph. Then three CEN layers are stacked to enlarge each agent's receptive field to overcome the limitation of communication. It's worth noting that each agent takes its neighbor agents' information into consideration according to the importance learned from CEN. Finally, the obtained latent information for each agent is subsequently used to evaluate the critic network and update the actor network.

A. Communication Enhanced Network

Usually, each agent should require all the other agents' information about their observations and actions to behave cooperatively better. However, it is not always true for each agent to get information of all the other agents in large-scale environment. To address the issue, multiple graph convolution layers are stacked to enlarge the agent's receptive field. Taking agent 4 in Fig. 2 as an example, an agent can obtain the hidden states obtained from convolution layers of its neighbours by stacking one convolution layer. By stacking two layers, the agent can get the hidden states of its neighbours' neighbours. Therefore, multiple graph convolution layers can be utilized to enlarge agent 4's reception field. In this paper, the number of graph convolution layers is set to 3.

In addition to enlarging the receptive fields of agent i, the neighbour agents need to be treated differently by agent i for promoting cooperation. In the large-scale environment, the complex interaction between agent i and its nearby agents makes agent i hard to learn a stable policy. Therefore, the communication enhanced network based on [18] is designed to allow agent i to treat the other agents' states differently. It

Authorized licensed use limited to: INSTITUTE OF AUTOMATION CAS. Downloaded on May 11,2021 at 02:10:48 UTC from IEEE Xplore. Restrictions apply.

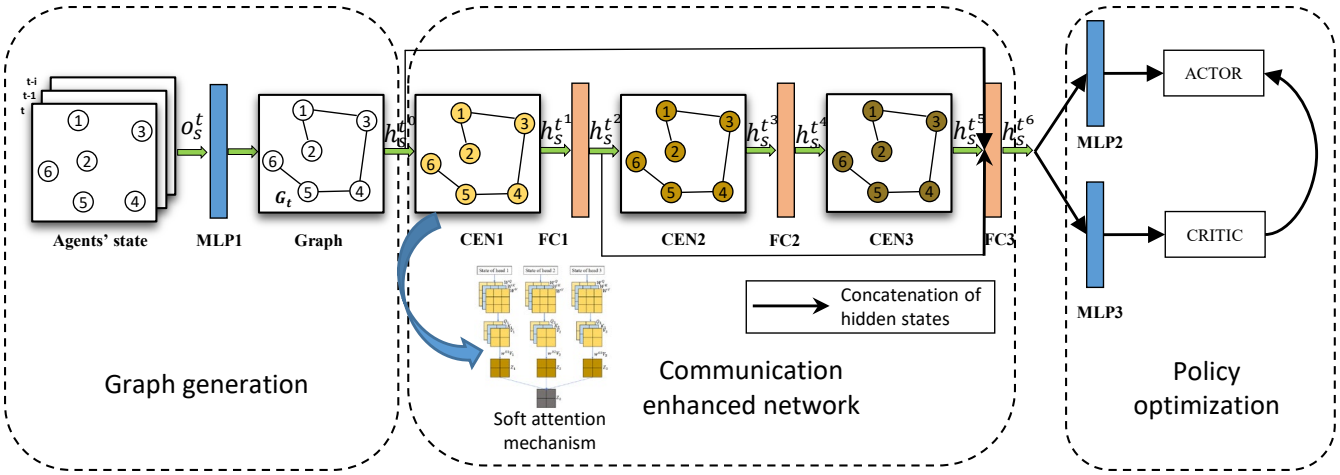


Fig. 1. Architecture of SGA-RL

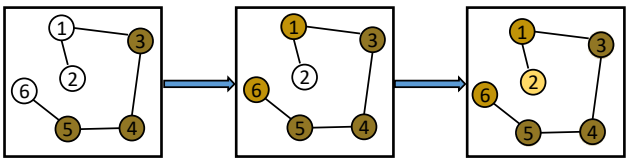


Fig. 2. Enlarging respective fields

operates on graph-structured data and computes the features of each graph node by assigning the different attentions to its neighbors, following a self-attention strategy. The hidden states of the agents are used to calculate the attention coefficients e_{ij} from agent j to agent i and its normalized form α_{ij} :

$$e_{ij} = a_G^k \left(W_G^k h_i, W_G^k h_j \right) \tag{5}$$

$$\alpha_{ij} = \mathrm{softmax}(e_{ij}) = \frac{\exp\left(\mathrm{LeakyReLU}\left(e_{ij}\right)\right)}{\sum_{k \in N_i} \exp\left(\mathrm{LeakyReLU}\left(e_{ij}\right)\right)} \tag{6}$$

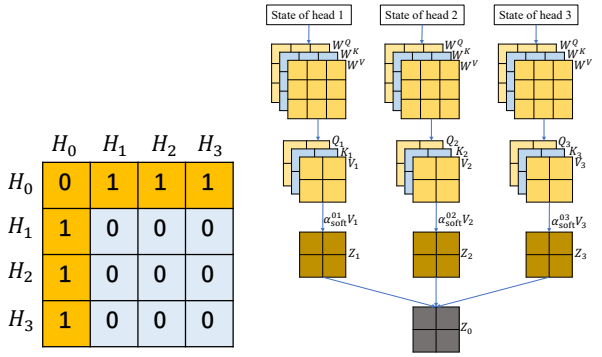
where a_G^k is a single-layer feedforward neural network, W_G^k is a learnable weight matrix and *LeakyReLU* is a nonlinear activation function. After getting the normalized attention coefficients, the output of one graph attention layer for node i at t is given by:

$$h_i^{t'} = \sigma \left(\sum_{j \in N_i} \alpha_{ij} W h_i^t \right) \tag{7}$$

where σ represents a nonlinear function. Equations (6)-(7) show how a single graph attention layer works.

Besides, after the final attention layer CEN3, the hidden states are concatenated and fed into FC3 as shown in Fig. 1. Since the hidden state could disappear during the process of graph convolution, these hidden states are concatenated in the final layer to stabilize the training process.

$$h_s^{t^5} = \sigma \left(\left[h_s^{t^5} \left\| h_s^{t^2} \right\| h_s^{t^0} \right] W_F^3 + b_F^3 \right) \tag{8}$$



(a) Soft attention adjacent matrix (b) Soft attentional gate

Fig. 3. Soft attention mechanism

where $||$ represents the concatenation, W_F^3 and b_F^3 are weight matrix and bias of fully-connected layer 3 (FC3) to learn.

B. Soft Attention Mechanism

In a large-scale environment, it's important for an agent to take the information of the other agents' into consideration according to their different importance. [18] indicates that the import of multi-head attention is beneficial to stabilize the learning process of the attention. Moreover, an agent can extract different state representation of the nearby agents from different representation subspace with multi-head setting. The output of one graph attention layer with multi-head attention for node i at t is changed from (7) to the following equation:

$$h_i^{t'} = \Big\|_{m=1}^{K} \sigma \left(\sum_{j \in N_i} \alpha_{ij}^m W^m h_i^t \right) \tag{9}$$

where K represents the number of the heads, α_{ij}^m represents the normalized attention coefficient of the m-th attention mechanism, and W^m represents the weight matrix of the m-th linear transformation. In this paper, the number of the heads is set to 3.

Authorized licensed use limited to: INSTITUTE OF AUTOMATION CAS. Downloaded on May 11,2021 at 02:10:48 UTC from IEEE Xplore. Restrictions apply.

Although the multi-head attention mechanism has the ability to explore multiple representation subspaces between each agent and its neighborhoods, some subspaces may not even exist for certain nodes and not all of these subspaces are equally important. Feeding the output of an attention head that captures a useless representation can mislead the final learned policy of the agents. And the situation will be worse for the agents in a large-scale environment.

Therefore, a soft attention mechanism is proposed to assign different importance to each head in CEN for helping the agents to learn a stable policy effectively. Different from GAT and [19], a virtual head H_0 and a soft attention adjacent matrix C are constructed, which can obtain better assignment of importance for the heads. As shown in Fig. 3 (a), H_0 is the virtual head and the rest heads are the real heads. $C_{ij}=0$ represents that there is no connection between head i and head j, otherwise there exists connection. It is observed that only H_0 is connected with the other heads and there is no connection among the real heads. Moreover, $C_{ii}=0$ represents that head i does not take its own state into consideration, which means the final output is only related to the real heads and not influenced by the virtual head H_0.

The details of the soft attention gate is shown in Fig. 3 and (10)-(13). The hidden state h_i of the agent are set as the input of the heads. They are first transformed to a different spaces including queries Q_i, keys K_i and values V_i by using linear projections matrices W^Q, W^K and W^V. After receiving query-value pair from the other heads, the soft attention coefficients α_{soft}^{ij} for head j to head i is computed. Finally, all the states of the real heads are aggregated according to the soft attention coefficients α_{soft}^{ij}.

$$\begin{aligned} Q_i &= W^Q h_i \\ K_i &= W^K h_i \\ V_i &= W^V h_i \end{aligned} \tag{10}$$

$$e_{soft}^{ij} = \left(\frac{\left(W^Q h_j\right)\left(W^K h_i\right)^T}{d_K} \right) \tag{11}$$

$$\alpha_{soft}^{ij} = \frac{\exp\left(e_{soft}^{ij}\right)}{\sum_{k=1}^{T} \exp\left(e_{soft}^{ik}\right)} \tag{12}$$

$$h_s^{t'} = \sigma\left(\sum_{j=1}^{M} \alpha_{soft}^{0j} V_j \right) \tag{13}$$

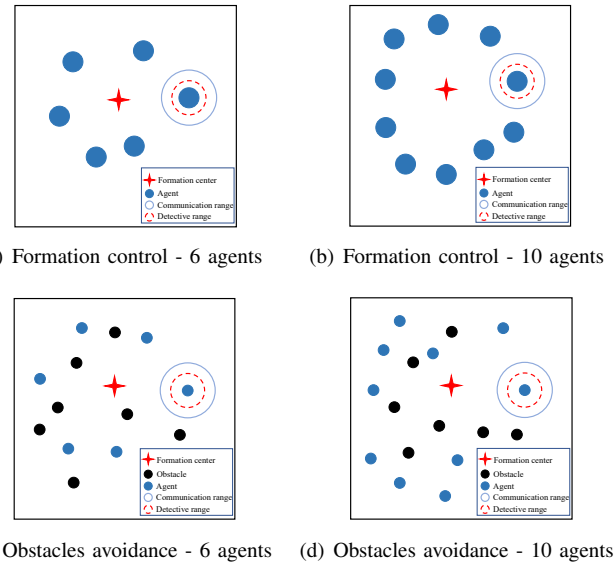
where h_i is the hidden state obtained by i-th head according to (7), d_K is the dimensionality of keys and M represents the number of the heads.

C. Policy Optimization

After the states are extracted, they are utilized to optimize the policy of the agents. As shown in Fig. 1, all the agents' information is extracted as $h_s^{t^6}$ which is a function related to all the other agents' states and its own state. After $h_s^{t^6}$ is obtained, PPO is implemented in an actor-critic framework. According to the objective function of PPO as shown in (3), it is changed as (15) after the concatenation of all the states. The changed objective function is used to optimize the policy network. Although the agents are trained with information from their nearby agents, they can obtain the information of all the other agents to promote the cooperation behaviour. Moreover, to scale up to more agents, the parameters sharing method is applied to train all the agents in a decentralized framework.

$$l_t(\theta) = \frac{\pi_\theta(a_t \left| h_s^{t^6}(O_1, O_2, \cdots, O_N)\right.)}{\pi_{\theta^k}(a_t \left| h_s^{t^6}(O_1, O_2, \cdots, O_N)\right.)} \tag{14}$$

$$\begin{aligned} L(\theta) = E[&\min(l_t(\theta)\hat{A}_t^{\theta^k}(h_s^{t^6}(O_1, O_2, \cdots, O_N)), \\ &clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t^{\theta^k}(h_s^{t^6}(O_1, O_2, \cdots, O_N))] \end{aligned} \tag{15}$$



(a) Formation control - 6 agents (b) Formation control - 10 agents

(c) Obstacles avoidance - 6 agents (d) Obstacles avoidance - 10 agents

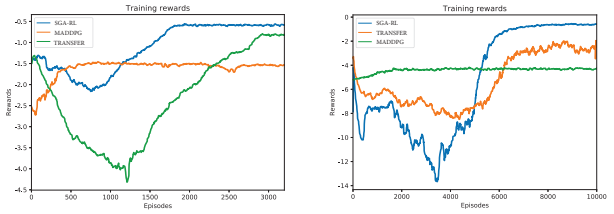
Fig. 4. The illustration of the simulation environments

IV. Simulations

A. Simulation Settings

In this section, the performance of SGA-RL is evaluated in four different scenarios as shown in Fig. 4. Scenarios (a)-(b) focusing on the formation control are designed to evaluate the effectiveness of SGA-RL. Moreover, scenarios (c)-(d) focusing on the formation control with obstacles avoidance are designed to verify the robustness of SGA-RL. For all the scenarios, the detective range for an agent is set as 0.5 and the communication range is set as 0.6. Besides, the agents can only obtain the other agents' information through the limited communication. These scenarios are implemented based on [9]. As baseline algorithms for comparing the performance, MADDPG [9] and TRANSFER [17] are taken into consideration. The former algorithm MADDPG relies on access to the states of all the agents during training instead of the partial observation state and the communication among the agents. The later algorithm ignores the complex interaction among the agents. In contrast, the partial observation state, the communication, the graph structure and the complex interaction are included in SGA-RL.

Authorized licensed use limited to: INSTITUTE OF AUTOMATION CAS. Downloaded on May 11,2021 at 02:10:48 UTC from IEEE Xplore. Restrictions apply.



(a) Rewards of scenario (a) (b) Rewards of scenario (b)

Fig. 5. The curves of training process

TABLE I

EVALUATING RESULTS OF SCENARIO (*a*)

Evaluated Algorithms	Metrics			
	Success(%)[a]	***Steps***[b]	***Rewards***	***Collision(%)***
MADDPG	0	60	-1.52	25.8
TRANSFER	98.70	13.52	-0.63	2.6
SGA-RL	**100**	**11.24**	**-0.48**	**0**

[a]The percentage of this task completed in defined steps
[b]The number of steps to finish the task.

B. Formation Control for Scenarios (a)-(b)

For the formation control tasks, all the agents are required to form the designated formation without colliding with each other. The reward for each agent is composed of the distance reward and the collision reward. Specifically, the distance reward is related to the distance from the agent to the formation center. Besides, if an agent collides with the other agents, the reward it obtains is -10. All the agents only observe the formation center location and their own states. The only way to obtain the other agents' states is through communication. Given the limitation of communication in reality, each agent communicates with up to two nearest neighboring agents only if their distance is less than a predefined threshold. SGA-RL is compared with the two algorithms mentioned above in two different formation control environments which includes scenario (a) with 6 agents and scenario (b) with 10 agents.

The learning curves of all the approaches in terms of mean rewards are presented in Fig. 5. The bigger mean rewards means that the formation formed by the agents is closer to the designated formation and fewer collisions with other agents. During the training process of scenario (a), it is observed that SGA-RL converges faster than TRANSFER with higher rewards than TRANSFER. MADDPG converges faster to the lowest value, which means that MADDPG cannot handle the complex interaction among the agents and falls into a locally optimal situation due to its need for the global state. During the training process of scenario (b), SGA-RL performs better than the other algorithms. The case in TRANSFER still does not learn a stable strategy when SGA-RL converges to a stable state. Moreover, the rewards of TRANSFER obtains is twice as much as SGA-RL (value of rewards is negative), which means that SGA-RL is more effective in large-scale environments.

In addition to the curves of the training process, the evaluation results in Table II and III present the similar results to Fig. 5. The agents trained by SGA-RL have higher rewards than MADDPG and TRANSFER in scenario (b). It is worth noting that the success rate of MAPPDG is 0 in scenarios (a) and (b), which means that the paradigm of centralized learning with decentralized execution is not suitable for the scenarios with the large number of agents. The performance difference between SGA-RL and the other algorithms demonstrates that SGA-RL can handle the complex interaction among the agents and promote the cooperation behavior.

TABLE II

EVALUATING RESULTS OF SCENARIO (*b*)

Evaluated Algorithms	Metrics			
	Success(%)	***Steps***	***Rewards***	***Collision(%)***
MADDPG	0	100	-4.06	38.40
TRANSFER	82.60	24.68	-1.78	10.20
SGA-RL	**100**	**14.36**	**-0.62**	**2**

C. Formation Control with Obstacles Avoidance for Scenarios (c)-(d)

For the formation control with obstacles avoidance tasks, the agents need to learn how to avoid dynamic obstacles represented by black circles and form the designated formation without collision. The reward is -5 for the collision between the agents and -10 for the collision between the agents and the obstacles. The other environments setting are the same with scenarios (a)-(b). The agents can obtain the position of obstacles' when they enter the detective range.

As shown in Table IV and V, SGA-RL has better results than all the baselines. Especially in scenario (d), SGA-RL converges twice faster than TRANSFER and obtains 50% higher rewards than the other methods.

To better illustrate the policy learned with SGA-RL, the process of formation is presented in Fig. 6. The final result shows that the agents have learned a reasonable cooperative strategy through SGA-RL. Moreover, the attention value distribution of different agents can be obtained in Fig. 6.

TABLE III

EVALUATING RESULTS OF SCENARIO (*c*)

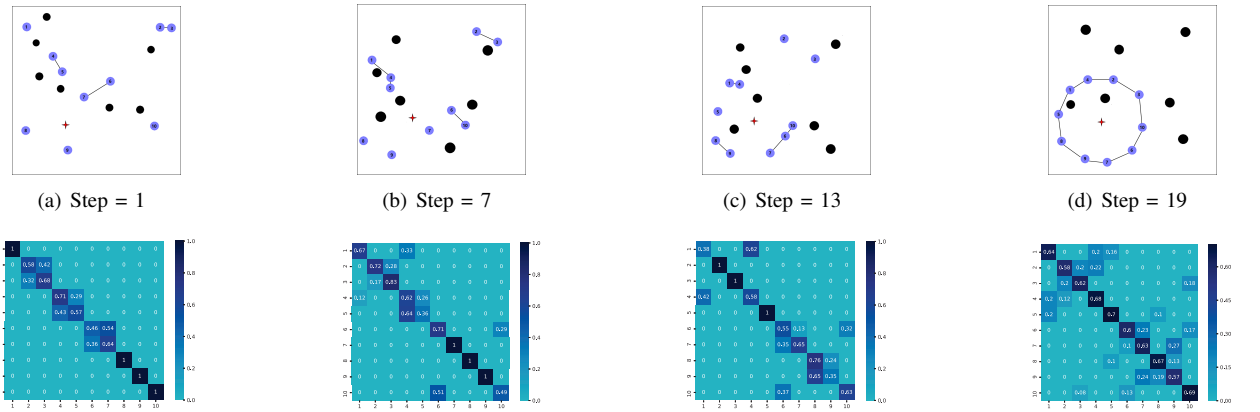
Evaluated Algorithms	Metrics			
	Success(%)	***Steps***	***Rewards***	***Collision(%)***
MADDPG	0	80	-2.78	30.40
TRANSFER	92.6	15.26	-0.84	4.80
SGA-RL	**100**	**11.24**	**-0.53**	**0**

TABLE IV

EVALUATING RESULTS OF SCENARIO (*d*)

Evaluated Algorithms	Metrics			
	Success(%)	***Steps***	***Rewards***	***Collision(%)***
MADDPG	0	120	-4.06	40.6
TRANSFER	73.4	44.20	-1.78	10.2
SGA-RL	**97.8**	**20.60**	**-0.87**	**3.4**

Authorized licensed use limited to: INSTITUTE OF AUTOMATION CAS. Downloaded on May 11,2021 at 02:10:48 UTC from IEEE Xplore. Restrictions apply.



(a) Step = 1 (b) Step = 7 (c) Step = 13 (d) Step = 19

(e) Attention value at step 1 (f) Attention value at step 7 (g) Attention value at step 13 (h) Attention value at step 19

Fig. 6. The illustration of the cooperative strategy for agent 4

For agent 4, at the beginning, it focuses on its own state as well as agent 5's state. After 6 steps, agent 4 gets close to agent 1,which means agent 4 starts taking agent 1 into consideration. Then agent 5 moves away from agent 4. The attention from agent 4 to agent 5 decrease to 0 and the attention from agent 4 to agent 1 increase from 0 to 0.42. Finally, all the agents pay more attention on itself and take the connected agents into consideration equally. It can be concluded that SGA-RL can enhance the agents' cooperative ability by assigning importance to its nearby agents properly.

V. Conclusions

In this paper, we present a novel reinforcement learning algorithm for multi-agent cooperation in large-scale environments with restricted topology. With SGA-RL, the agents' communication range is enlarged and the complex interaction among the agents is handled as well. SGA-RL is shown to perform a satisfying strategy and adapt to large-scale environments. Future work will take the time-delay phenomenon into consideration.

References

[1] B. M. Radhakrishnan and D. Srinivasan, "A multi-agent based distributed energy management scheme for smart grid applications," *Energy*, vol. 103, pp. 192–204, 2016.

[2] X. Li, J. Zhang, J. Bian, Y. Tong, and T.-Y. Liu, "A cooperative multi-agent reinforcement learning framework for resource balancing in complex logistics network," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 980–988.

[3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[4] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[5] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[6] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, 2015, pp. 1889–1897.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[8] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami, *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017.

[9] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, pp. 6379–6390.

[10] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[11] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in *International Conference on Machine Learning*, 2018, pp. 5567–5576.

[12] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Advances in Neural Information Processing Systems*, 2018, pp. 7254–7264.

[13] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International Conference on Machine Learning*, 2019, pp. 2961–2970.

[14] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau, "Tarmac: Targeted multi-agent communication," in *International Conference on Machine Learning*, 2019, pp. 1538–1546.

[15] A. Malysheva, T. T. Sung, C.-B. Sohn, D. Kudenko, and A. Shpilman, "Deep multi-agent reinforcement learning with relevance graphs," *arXiv preprint arXiv:1811.12557*, 2018.

[16] J. Jiang, C. Dun, and Z. Lu, "Graph convolutional reinforcement learning for multi-agent cooperation," *arXiv preprint arXiv:1810.09202*, vol. 2, no. 3, 2018.

[17] A. Agarwal, S. Kumar, and K. Sycara, "Learning transferable cooperative behavior in multi-agent teams," *arXiv preprint arXiv:1906.01202*, 2019.

[18] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[20] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*. Elsevier, 1994, pp. 157–163.

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

Authorized licensed use limited to: INSTITUTE OF AUTOMATION CAS. Downloaded on May 11,2021 at 02:10:48 UTC from IEEE Xplore. Restrictions apply.