Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Supplementary materials for

Yixiang REN, Zhenhui YE, Yining CHEN, Xiaohong JIANG, Guanghua SONG, 2023. Soft-HGRNs: soft hierarchical graph recurrent networks for multi-agent partially observable environments. *Front Inform Technol Electron Eng*, 24(1):117-130. https://doi.org/10.1631/FITEE.2200073

1 Detailed derivations for Section 4

1.1 Proof of Eq. (7)

We obtain the probability of the action $\pi(a_t|\mathcal{O}_t)$ by using softmax to process the Q value produced by HGRN:

$$\pi(a_t|\mathcal{O}_t)=\frac{\exp\left(\frac{1}{\alpha}Q(\mathcal{O}_t,a_t)\right)}{\sum_{a_t}\exp\left(\frac{1}{\alpha}Q(\mathcal{O}_t,a_t)\right)}=\exp\left(\frac{Q(\mathcal{O}_t,a_t)}{\alpha}-\ln\sum_{a_t}\exp\left(\frac{Q(\mathcal{O}_t,a_t)}{\alpha}\right)\right), \tag{S1}$$

where α is a temperature parameter to control the level of exploration of the model, which is proportional to the diversity of the policy's output actions. The larger the α, the more balanced the action probability distribution of the policy. To learn such an energy-based policy, the value function V is redefined as

$$V(\mathcal{O}_t)=\alpha\ln\sum_{a_t}\exp\left(\frac{Q(\mathcal{O}_t,a_t)}{\alpha}\right). \tag{S2}$$

1.2 Derivations and pseudo code of SAC-HGRN

Adapted from SAC (Haarnoja et al., 2018), we designed an actor-critic styled variant named SAC-HGRN. For SAC-HGRN, we define the value function V as

$$V(\mathcal{O}_t)=\mathbb{E}_{a_t\sim\pi(a_t|\mathcal{O}_t)}[Q(\mathcal{O}_t,a_t)-\alpha\ln\pi(a_t|\mathcal{O}_t)], \tag{S3}$$

and the actor network is updated by policy gradient (PG) with a maximum-entropy regularization term:

$$\nabla\pi=\frac{1}{S}\sum_{S}\nabla\ln\pi(a_t|\mathcal{O}_t)\cdot\Big(Q(\mathcal{O}_t,a_t)-\alpha\ln\pi(a_t|\mathcal{O}_t)\Big). \tag{S4}$$

SAC-HGRN trains an actor network and a critic network, both of which adopt the HGRN structure proposed in Section 4.1. In addition to the policy gradient equation discussed in the study, the learning objective of the actor network in Eq. (S4) is additionally deducted with an estimated baseline value b, represented as

$$b=\mathbb{E}_{a\sim\pi}[Q(\mathcal{O},a)]\approx\frac{1}{S}\sum_{a}\pi(a|\mathcal{O})Q(\mathcal{O},a), \tag{S5}$$

where S is the batch size.

During training, we found that the critic network, which provides the learning target for the actor, could not properly catch up with the update of the actor, resulting in a relatively unstable training process and slightly worse performance. To this end, we adopt a training strategy named delayed updated policy (DUP) (Fujimoto et al., 2018; Ye et al., 2022a), which updates the critic network twice as frequently as the actor network.

The pseudo code that illustrates the training phase of SAC-HGRN is given in Algorithm 1.

Algorithm 1 Training procedure of SAC-HGRN

Initialize an HGRN critic network Q with parameters θ^Q and a target critic network Q' with parameters $\theta^{Q'} \leftarrow \theta^Q$
Initialize an HGRN actor network π with parameters θ^π and a target actor network π' with parameters $\theta^{\pi'} \leftarrow \theta^\pi$
Initialize the learnable temperature $\alpha \leftarrow \alpha_0$
Set global time step $T \leftarrow 0$
for episode=1 to max-episodes **do**
  Reset the environment and the hidden states in GRU
  for local time step $t = 1$ to episode-length **do**
    $T \leftarrow T + 1$
    for agent $i = 1$ to N **do**
      Obtain the observation of agent i and its neighbors, represented as $\mathcal{O}_t^i$
      Obtain the GRU's hidden states of actor $h_t^{i;\pi}$ and critic $h_t^{i;Q}$
      Select the action: $a_t^i = \arg\max_a Q(\mathcal{O}_t^i, a; h_t^{i;\pi})$
      Execute the action a_t^i and obtain a reward r_t^i and observations $\mathcal{O}_{t+1}^i$
      Obtain the GRU's hidden states of actor $h_{t+1}^{i;\pi}$ and critic $h_{t+1}^{i;Q}$
      Obtain an experience for agent i: $(\mathcal{O}_t^i, h_t^{i;Q}, h_t^{i;\pi}, a_t^i, r_t^i, \mathcal{O}_{t+1}^i, h_{t+1}^{i;Q}, h_{t+1}^{i;\pi})$
    end for
    Integrate all agents' experience at time step t as one tuple and restore it into the replay buffer
    if T mod update-interval= 0 **then**
      Randomly sample S integrated experiences
      Split each integrated experience into N individual experiences: $(\mathcal{O}_t^i, h_t^{i;Q}, h_t^{i;\pi}, a_t^i, r_t^i, \mathcal{O}_{t+1}^i, h_{t+1}^{i;Q}, h_{t+1}^{i;\pi})$
      for $m = 1$ to 2 **do**
        Update θ^Q by minimizing the Q loss: $\left(r_t^i + V(\mathcal{O}_t^i; h_t^{i;Q}) - Q(\mathcal{O}_t^i, a_t^i; h_t^{i;Q})\right)^2$, where V is defined in Eq. (S3) in the study
      end for
      Update θ^π based on the Eq. (S4) in the study
      Update α based on the Eq. (8) in the study
    end if
    if T mod target-update-interval= 0 **then**
      Update target network Q' by: $\theta' \leftarrow \theta$
    end if
  end for
end for

2 Environment details

2.1 UAV-MBS

UAV-MBS (Ye et al., 2022b) is a cooperative task with N UAVs that served as mobile base stations to fly around a target region to provide communication services to the randomly distributed ground users, which is represented as N_{PoI} point of interests (PoIs). As can be seen in Fig. S1a, each UAV (blue circle) could only provide services to the PoI within a small coverage range R_{cov} (dashed green circle), but it could observe PoIs and other UAVs in a larger observation range R_{obs} (dashed yellow circle), and it could communicate with other UAVs within a communication range R_{com} (dashed blue circle). The UAV is rewarded by the number of PoIs exclusively covered by itself. Green squares with different shades denote the different numbers of PoIs. As the observation range is small compared with the map size, UAVs must learn to communicate with each other and memorize the history information to achieve optimal performance. We follow the continuous world setting in Ye et al. (2022b).

2.2 Surviving

Surviving (Jiang et al., 2020) is a cooperative task consisting of N agents (green circles) cooperating to explore a big map and collect the randomly distributed food to prevent starvation. The food is scattered by N_{resource} resource points (yellow squares), which are also randomly refreshed over time. Green squares with different shades denote the different numbers of food items. As can be seen in Fig. S1b, each agent could observe food, other agents, and the resource point within the observation range R_{obs}, and could communicate with other agents in the communication range R_{com}. When the agent is on a grid that has food, it will store

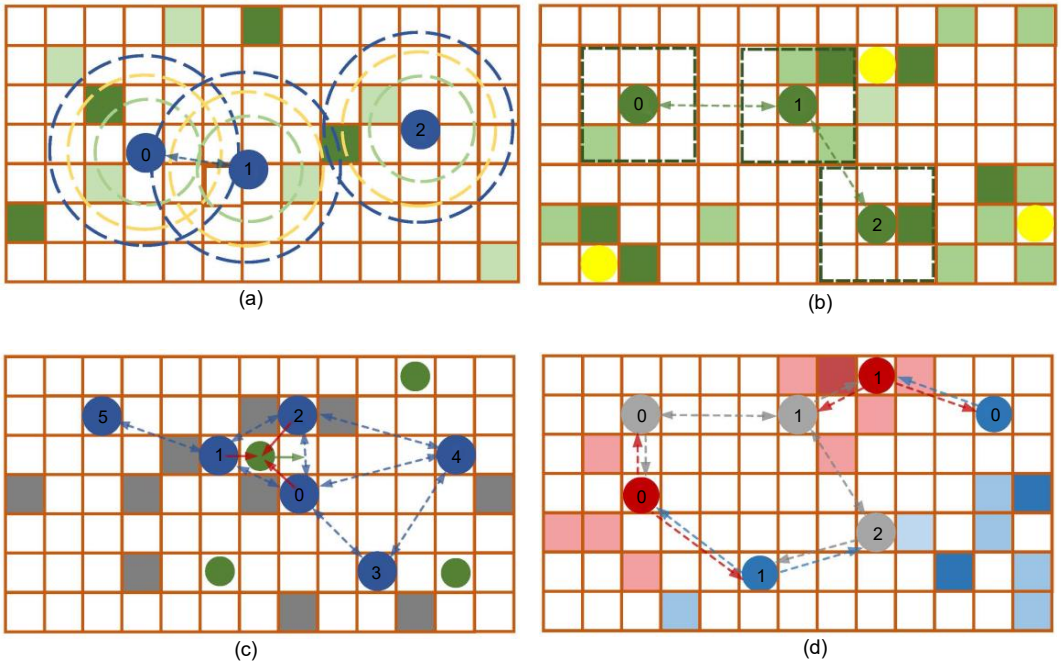

Fig. S1 Screenshots of the tested simulation environments: (a) UAV-MBS; (b) Surviving; (c) Pursuit; (d) cooperative treasure collection (CTC)

the food in its package. At each time step, the agent will consume 1 unit of food in its package, and if there is no food it will be punished with a negative reward -0.2. This environment is more challenging than UAV-MBS because the explored food will be consumed and be randomly regenerated in another position, which requires efficient communication to achieve cooperative exploration.

We tested the environment with the same settings as in Jiang et al. (2020), i.e., $N = 100$, $N_{\text{grid}} = 30$, $N_{\text{resource}} = 8$, $R_{\text{obs}} = 1$, and $R_{\text{com}} = 3$.

2.3 Pursuit

Pursuit (Zheng et al., 2018) is an adversarial environment that consists of $N_{\text{predator}} = 25$ learnable predators and $N_{\text{prey}} = 50$ pre-trained prey units. The predators (blue circles) are rewarded by attacking the prey (green circles). Different from UAV-MBS and Surviving, the predator has a large observation range $R_{\text{obs}} = 13$. As can be seen from Fig. S1c, the predators need to cooperate with nearby teammates to form stable closure to lock their prey to achieve the optimal performance. As the cooperation is only necessary among nearby agents, we set a small communication range $R_{\text{com}} = 5$.

2.4 Cooperative treasure collection

Cooperative treasure collection (CTC) (Iqbal and Sha, 2019) is a heterogeneous environment with three types of agents, N hunters (gray circles), N_{RB} red banks (red circles), and N_{BB} blue banks (blue circles). The hunter can obtain a small reward by collecting each red or blue treasure in its package and could obtain a large reward by depositing it into the bank with the correct color. The bank is also movable and could obtain a big reward when a treasure is deposited from the hunter. Each agent has the same observation range R_{obs} and communication range R_{com}. Note that communication among heterogeneous agents is available in this scenario to achieve better cooperation between the hunter and the bank. The colored dashed arrow denotes HGAT connectivity.

In this study, we set the environment with the settings of $N = 30$, $N_{\text{RB}} = 10$, $N_{\text{BB}} = 10$, $R_{\text{obs}} = 3$, $R_{\text{com}} = 5$, and $N_{\text{grid}} = 34$.

3 Experimental settings and results

3.1 Hyper-parameter settings

Hyper-parameter settings related to the training and evaluation process in the four simulation environments are listed in Table S1.

3.2 Network architecture search result

As can be seen in Fig. S2, two stacked HGAT layers outperform one HGAT layer, which is due to the larger perception field. The skip-connection (He et al., 2016) over two HGAT layers leads to faster convergence and slightly better performance. Therefore, we use two stacked HGAT layers with skip-connection as the communication structure and apply it in DGN, MAAC, Soft-HGRN, and SAC-HGRN.

Table. S1 Hyper-parameter settings of all environments

Hyper-parameters	UAV-MBS	Surviving	Pursuit	CTC
Number of agents	20	100	25	30+10+10
Number of episodes	50 000	20 000	40 000	20 000
Episode length	100	500	300	500
Buffer size	5×10^4	5×10^4	5×10^4	5×10^4
Initial ϵ	0.9	0.9	0.9	0.9
Burnin episode	500	500	500	20
ϵ decay rate	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Minimal ϵ	0.05 (deterministic policy)/0 (stochastic policy)			
Entropy target factor p_α	0.7	0.9	0.05	0.3
Update times	4	4	4	4
Update interval	100 timesteps	100 timesteps	100 timesteps	100 timesteps
Target update interval	500 timesteps	500 timesteps	500 timesteps	500 timesteps
Discount factor	0.99	0.99	0.99	0.99
Batch size	128	128	128	128
Optimizer	Adam	Adam	Adam	Adam
Learning rate	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Number of attention heads	4	4	4	4
Hidden dimensions	256	256	256	256
Activation	ReLU	ReLU	ReLU	ReLU
Initializer	Random normal	Random normal	Random normal	Random normal

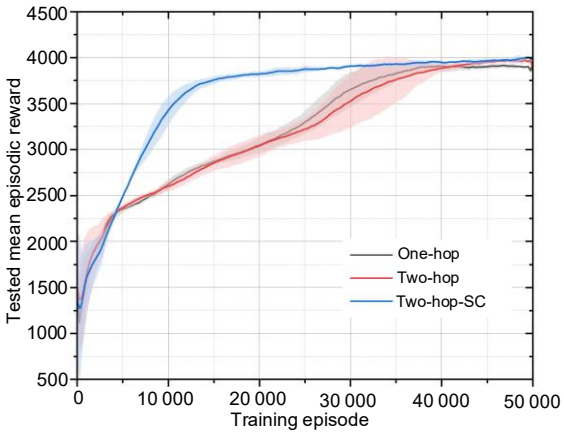

Fig. S2 Comparison of Soft-HGRN with different communication structures in UAV-MBS. One-hop, two-hop, two-hop-SC denote one HGAT layer, two stacked HGAT layers, and two HGAT layers with skip connection, respectively

3.3 Learning curves in homogeneous environments

We conducted comparison experiments and ablation studies in three homogeneous scenarios (UAV-MBS, Surviving, and Pursuit). Each model was updated until convergence. All learning curves can be found in Figs. S3–S5. Error bars denote the standard derivation over three runs.

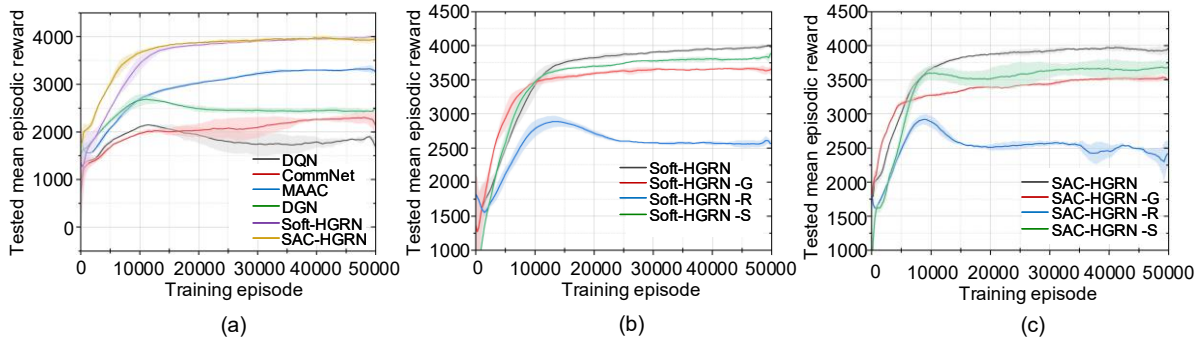

Fig. S3 Learning curves in the UAV-MBS environment: (a) comparison with baselines; (b) ablations in Soft-HGRN; (c) ablations in SAC-HGRN

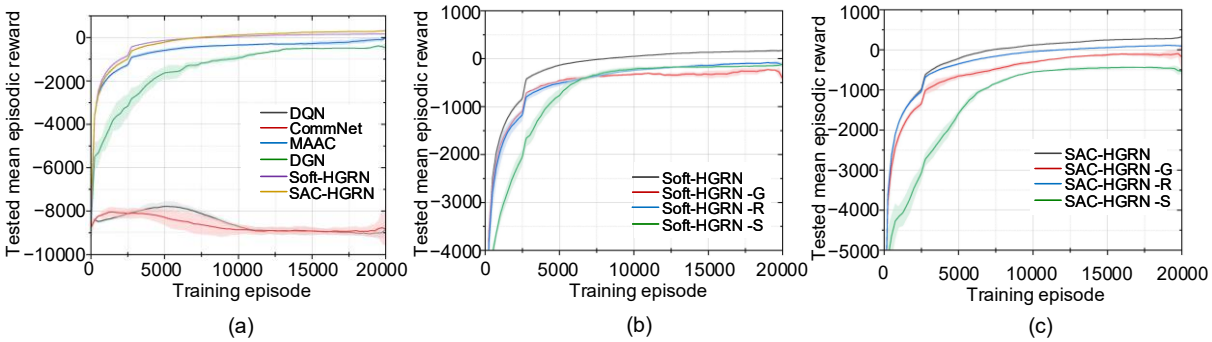

Fig. S4 Learning curves in the Surviving environment: (a) comparison with baselines; (b) ablations in Soft-HGRN; (c) ablations in SAC-HGRN

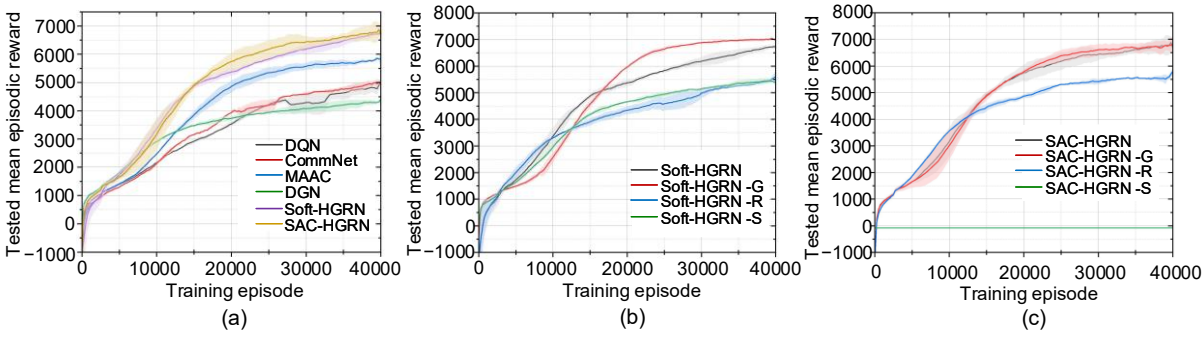

Fig. S5 Learning curves in the Pursuit environment: (a) comparison with baselines; (b) ablations in Soft-HGRN; (c) ablations in SAC-HGRN

References

Fujimoto S, Hoof H, Meger D, 2018. Addressing function approximation error in actor-critic methods. Int Conf on Machine Learning, p.1587-1596. https://doi.org/10.48550/arXiv.1802.09477

Haarnoja T, Zhou A, Abbeel P, et al., 2018. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. Proc 35th Int Conf on Machine Learning, p.1861-1870. https://proceedings.mlr.press/v80/haarnoja18b.html

He KM, Zhang XY, Ren SQ, et al., 2016. Deep residual learning for image recognition. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.770-778. https://doi.org/10.1109/CVPR.2016.90

Iqbal S, Sha F, 2019. Actor-attention-critic for multi-agent reinforcement learning. Proc 36$^{\text{th}}$ Int Conf on Machine Learning, p.2961-2970. https://proceedings.mlr.press/v97/iqbal19a.html

Jiang J, Dun C, Huang T, et al., 2020. Graph convolutional reinforcement learning. Int Conf on Learning Representations. https://openreview.net/forum?id=HkxdQkSYDB

Ye ZH, Chen YN, Jiang XH, et al., 2022a. Improving sample efficiency in multi-agent actor-critic methods. *Appl Intell*, 52(4):3691-3704. https://doi.org/10.1007/s10489-021-02554-5

Ye ZH, Wang K, Chen YN, et al., 2022b. Multi-UAV navigation for partially observable communication coverage by graph reinforcement learning. *IEEE Trans Mobile Comput*, early access. https://doi.org/10.1109/TMC.2022.3146881

Zheng LM, Yang C, Cai H, et al., 2018. MAgent: a many-agent reinforcement learning platform for artificial collective intelligence. Proc 32$^{\text{nd}}$ AAAI Conf on Artificial Intelligence, p.8222-8223.