ADMN: Agent-Driven Modular Network for Dynamic Parameter Sharing in Cooperative Multi-Agent Reinforcement Learning

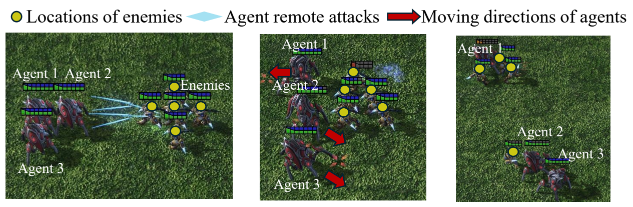
Yang Yu[1,2] , **Qiyue Yin**[1,2] , **Junge Zhang**[1,2*] , **Pei Xu**[2] , **Kaiqi Huang**[1,2,3*]
[1]School of Artificial Intelligence, University of Chinese Academy of Sciences
[2]CRISE, Institute of Automation, Chinese Academy of Sciences
[3]CAS, Center for Excellence in Brain Science and Intelligence Technology
{yuyang2019, pei.xu}@ia.ac.cn, {qyyin, jgzhang, kqhuang}@nlpr.ia.ac.cn

Abstract

Parameter sharing is a common strategy in multi-agent reinforcement learning (MARL) to make the training more efficient and scalable. However, applying parameter sharing among agents indiscriminately hinders the emergence of agents diversity and degrades the final cooperative performance. To better balance parameter sharing and agents diversity, we propose a novel Agent-Driven Modular Network (ADMN), where agents share a base network consisting of multiple specialized modules, and each agent has its own routing to connect these modules. In ADMN, modules are shared among agents to improve the training efficiency, while the combination of different modules brings rich diversity. The agent routing at different time steps is learned end-to-end to achieve a dynamic and adaptive balance. Specifically, we also propose an information-theoretical regularization between the routing of agents and their behavior to further guarantee the identifiability of different routing. We evaluated ADMN in challenging StarCraft micromanagement games and Google Research Football games, and results demonstrate the superior performance of ADMN, particularly in larger or heterogeneous cooperative tasks.

1 Introduction

Multi-agent reinforcement learning (MARL) has gained substantial attention in recent years, due to its applicability in addressing various real-world cooperative problems, including cooperative games [Berner *et al.*, 2019; Bard *et al.*, 2020], traffic control [Bazzan, 2009; Chu *et al.*, 2019], and robot fleet coordination [Cao *et al.*, 2012; Hüttenrauch *et al.*, 2019]. Nevertheless, effectively learning coordinated policies for complex cooperative tasks remains challenging, as the algorithm must explore the exponentially growing joint action-state space with the increasing number of agents. To make the training more efficient and scalable, recent MARL methods [Peng *et al.*, 2012; Wang *et al.*, 2021a; Foerster *et al.*, 2018; Yu *et al.*, 2021] predominantly employ parameter sharing, where agents share their network parameters.



(a) Early stage: remote attack (b) Mid-term stage: melee attack (c) Distracted enemies

Figure 1: Agents specialized in remote attack fight with enemies mastering melee attack. When two sides are far apart as shown in (a), all agents conduct remote attacks, and parameter sharing can be used to accelerate training. However, when facing melee attacks, non-dominant agents need to behave diversely to distract enemies. For example, one group draws fire, while the other group runs to distant positions to maintain long-range attack advantage as shown in (b,c). In this task, parameter sharing and agents diversity are required dynamically to achieve sophisticated cooperation.

Parameter sharing indeed facilitates the training in MARL, significantly reducing the total number of trainable parameters and making the training complexity tractable. More importantly, this sharing allows for the reuse of experiences by multiple agents. Although these merits make parameter sharing popular in MARL, having all agents share the same parameters also has negative effects [Li *et al.*, 2021; Hu *et al.*, 2022]. In the real world, the completion of many complex cooperative tasks requires for agents with different abilities. However, fully-shared parameters may cause agents to behave similarly, hindering the emergence of agents diversity, and limiting agents exploration capabilities, thereby degrading the final cooperative performance.

To address the lack of diversity in naive parameter sharing, there are researches encouraging no parameter sharing or selective parameter sharing in MARL [Jiang and Lu, 2021; Li *et al.*, 2021; Christianos *et al.*, 2021; Kim and Sung, 2023]. Instead of sharing parameters across all agents, these methods train individual policy networks or group-based networks to encourage the diversity across agents or agent groups. However, existing methods employ fixed parameter sharing patterns for agents during cooperation, resulting in static diversity that fails to adapt to dynamic cooperative tasks.

*Corresponding authors.

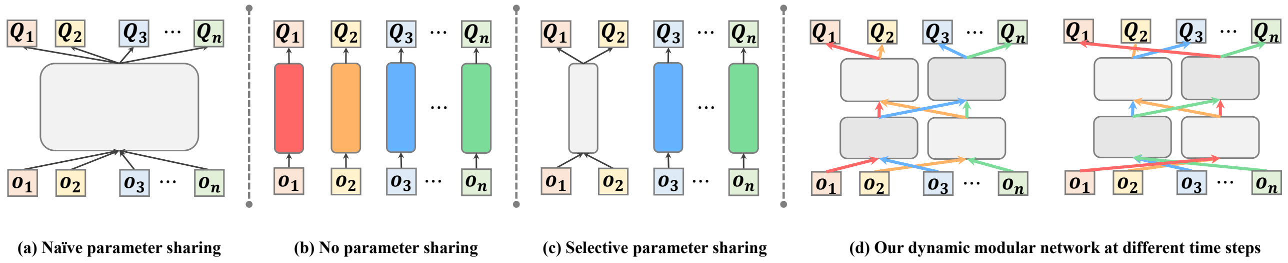

Q_1 Q_2 Q_3 ⋯ Q_n
o_1 o_2 o_3 ⋯ o_n
(a) Naïve parameter sharing
(b) No parameter sharing
(c) Selective parameter sharing
(d) Our dynamic modular network at different time steps

Figure 2: Four parameter sharing mechanisms in MARL. Gray blocks are networks shared among agents, and colored blocks are unique to each agent. At different time steps, compared to the fixed parameter sharing paradigm shown in (a, b, c), our proposed modular network shown in (d) can choose parameter sharing or agents diversity dynamically by configuring different module connections.

Note that, in sophisticated multi-agent cooperation, agents are adaptive to the environment changes, and can choose to behave diversely or not dynamically, as shown in Figure 1. Therefore, the question is how to balance the parameter sharing and agents diversity dynamically and adaptively.

In this paper, to deal with the dilemma between parameter sharing and agents diversity, and adapt to the dynamic environment or different cooperative tasks, we propose a novel framework called Agent-Driven Modular Network (ADMN). In ADMN, a modular network is introduced into the agent architecture, consisting of different specialized modules. During training, these modules are shared among agents to decrease the training complexity, while different routing, *i.e.*, the connecting weights across different modules bring rich diversity. At each time step, the agent configures its routing according to its own observation and identity, which is learned end to end and achieves a dynamic and flexible balance, without relying on human prior knowledge. Specifically, to further guarantee the identification of different routing, an information-theoretical regularization between the routing of agents and their behavior is also proposed under ADMN.

We evaluated ADMN in StarCraft Multi-Agent Challenge (SMAC) [Samvelyan *et al.*, 2019] and Google Research Football (GRF) [Kurach *et al.*, 2020]. Compared to baselines, results demonstrate superior performance of our method in challenging cooperative tasks, especially in larger or heterogeneous cooperative tasks. And visualization of agents' routing across various tasks or time steps underscores the flexibility and reasonability of learned routing policies in ADMN.

2 Related Work

To make the training in MARL efficient and scalable, many methods adopt naive parameter sharing framework, including value-based MARL methods [Rashid *et al.*, 2020; Iqbal *et al.*, 2021; Xu *et al.*, 2023b] and policy-based methods [Lowe *et al.*, 2017; Zhou *et al.*, 2020; Zhang *et al.*, 2021b; Xu *et al.*, 2023a], where all agents share the same value network or policy network, as shown in Figure 2(a). However, naive parameter sharing hinders the diversity of agents and limits the emergence of sophisticated coordinated policies. To encourage the diversity across agents or agent groups, recent approaches introduce no parameter sharing or selective parameter sharing as shown in Figure 2(b,c). For example, EOI [Jiang and Lu, 2021] avoids parameter sharing to prevent similar agent behaviors, CDS [Li *et al.*, 2021] fosters individual diversity with agent-specific heads in addition to a sharing network, and SePS [Christianos *et al.*, 2021] divides agents into groups with shared network parameters before training. Although these methods produce diverse agent policies, the introduction of agent-based or group-based networks decreases the method's scalability, whose size of training parameters still increases with the number of agents. More importantly, the diversity introduced in existing works is static, which encourages fixed parameter sharing patterns during cooperation, and cannot adapt to dynamic cooperative tasks. In contrast, the size of ADMN will not increase with the number of agents. Furthermore, as shown in Figure 2(d), the diversity in ADMN comes from the combination of different modules, which is varying at different time steps and learned end to end to achieve a dynamic and adaptive parameter sharing.

Another approach to balance efficiency and diversity in MARL is task decomposition, where the whole task is divided into subtasks, and agents are assigned to different sub-groups for various subtasks [Wooldridge *et al.*, 2000; DeLoach and Garcia-Ojeda, 2010; Lhaksmana *et al.*, 2018]. For example, RODE [Wang *et al.*, 2021b] proposes to decompose the whole task based on the decomposition of the joint-action space, where each subtask contains partial actions. However, this division is limited to specific environments, and does not apply to tasks where basic actions are needed in each subtask. More recently, LDSA [Yang *et al.*, 2022] is proposed to learn the task decomposition and agents assignment, by learning the subtask encoder and the agent trajectory encoder. Although the task decomposition in LDSA is learnable, it still needs human prior knowledge to help design the algorithm, such as the decision of the number of subtasks. On the contrary, in ADMN, the efficiency and diversity are balanced using the modular network, where the sharing of the modular knowledge across agents is learned end to end, without introducing any task-related human prior knowledge.

3 Preliminary

3.1 Problem Formulation

Multi-agent reinforcement learning is the extension of reinforcement learning to help address multi-agent sequential decision problems, especially cooperation games, which is usually modeled as Dec-POMDP, *i.e.*, $G = \langle S, U, A, P, R, Z, O, n, \gamma \rangle$ [Zhang *et al.*, 2021a]. S is the state space of the environment. At each time step t, every agent $u \in U \equiv \{1, \ldots, n\}$ chooses an action $a_u \in A$ which forms the joint action $\boldsymbol{a} \in \boldsymbol{A} \equiv A^n$. P is the state transition function which maps the current state s and the joint action $\boldsymbol{a}$ to the next state s', *i.e.*, $P(s'|s, \boldsymbol{a}) : S \times \boldsymbol{A} \times S \to [0, 1]$. All agents receive a shared reward $r \in \mathbb{R}$ according to the reward function $R(s, \boldsymbol{a}) : S \times \boldsymbol{A} \to \mathbb{R}$ and $\gamma \in [0, 1)$ is the discount factor. In a partially observable setting, each agent does not have access to the full state and instead samples observations $o_u \in O$ according to observation function $Z(s, u) : S \times U \to O$. The action-observation history for an agent u is $\tau_u \in T \equiv (O \times A)^*$, on which it conditions its policy $\pi_u(a_u|\tau_u) : T \times A \to [0, 1]$. The joint action-value function is defined as $Q^{\boldsymbol{\pi}}(s_t, \boldsymbol{a}_t) = E_{s_{t+1:\infty}, \boldsymbol{a}_{t+1:\infty}} \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t, \boldsymbol{a}_t\right]$, where $\boldsymbol{\pi}$ is the joint policy. The object of MARL is to find the joint policy $\boldsymbol{\pi}$ to maximize the expected sum of the discounted team reward.

3.2 Centralized Training and Decentralized Execution

To address the non-stationary issue and the scalability problem in cooperative MARL [Tan, 1993; Matignon *et al.*, 2012], existing cooperative MARL algorithms mainly adopt the centralized training and decentralized execution (CTDE) paradigm. CTDE means during training, the algorithm has access to the full state and the action-observation histories of all agents, while during execution each agent makes decisions conditioned on its own local history. In this paper, we also adopt the CTDE paradigm, and base our method on the popular value-based CTDE method QMIX [Rashid *et al.*, 2018]. QMIX tries to factor the joint-action value Q_{tot} into a monotonic nonlinear combination of individual utilities Q_u, where the individual utility is learned via a utility network. A mixer network with nonnegative weights is used for combining agents' utilities. The nonnegativity in the mixer network ensures that $\frac{\partial Q_{tot}(s,\boldsymbol{a})}{\partial Q_u(\tau_u, a_u)} \geq 0$, which guarantees Individual-Global-Max (IGM) Condition [Son *et al.*, 2019], *i.e.* the optimal joint actions across agents are equivalent to the collection of individual optimal actions of each agent. QMIX is effective since the maximization can be performed in $O(n|A|)$ as opposed to $O(|A|^n)$, and during execution, each agent can independently make decisions by its own utility network without any global information since IGM condition is satisfied. More details will be provided in Appendix.

4 Method

In ADMN, there is a base modular network consisting of multiple modules that can be shared by agents, and a routing network used by each agent to generate its routing, *i.e.*, connecting weights across modules at each time step. In particular, to enhance the identification of different routing, a mutual information regularization between the routing and the behavior of the agent is further proposed. In the following, we will present the network architecture of ADMN and elaborate on the mutual information regularization. The framework of ADMN is provided in Figure 3.

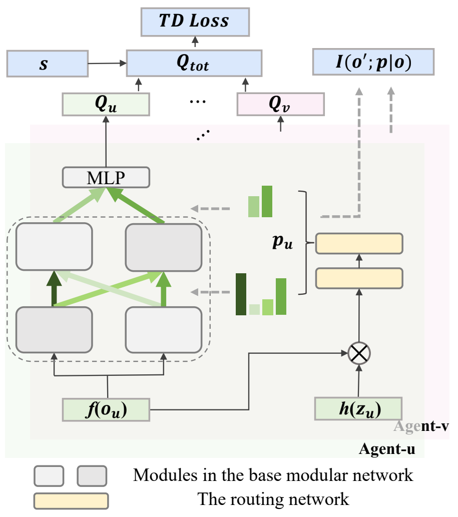


Figure 3: ADMN consists of a base modular network shown at the left center, and a routing network at the right center with the input of the agent identity and observation. Different color slices are related to routing paths of different agents. The mutual information regularization is shown on the top right to strengthen the influence of different agents routing on the agent observation transitions.

4.1 Routing Network

As depicted in Figure 3, the model of ADMN consists of a base modular network, and a routing network, where the latter assists each agent to configure the module connections in adjacent layers. Specifically, we employ soft instead of hard modularization, where the routing network outputs probabilities to weight the modules instead of determining discrete routing paths, and is trained with the base network together. As discussed in [Rosenbaum *et al.*, 2019; Yang *et al.*, 2020], the joint optimization of the hard routing network and the base network is unstable, and hard modularization introduces higher variation along with the exploration in the environment.

To learn the routing of agents with desired properties, we encode the routing in a stochastic embedding space, and the routing of agent u, denoted as, $p_u \in R^{dim}$, is drawn from a multivariate Gaussian distribution, $N(\mu_{p_u}, \sigma_{p_u})$. At each time step t, the inputs of the routing network contain the agent identity z_u, which is a one-hot vector representing each agent, and the current observation of the agent o_u^t to adapt the routing to the dynamic environment. First, the observation undergoes processing through a one-layer MLP and a GRU unit to obtain its representation, $f(o_u^t) \in R^D$, and the identity is processed by a one-layer MLP and obtain $h(z_u) \in R^D$. Then p_u^t is calculated based on the representation of its observation and identity. Assuming there are L module layers with each layer m modules, the output dimension of the routing network is $dim = (L-1) \times m \times m + m$ (the last m represents the weights of the modules in the last module layer). The output of the routing network for agent u is:

$$\begin{aligned} (\mu_{p_u}^t, \sigma_{p_u}^t) &= G(ReLU(f(o_u^t) \cdot h(z_u))), \\ p_u^t &\sim N(\mu_{p_u}^t, \sigma_{p_u}^t), \end{aligned} \tag{1}$$

where G is the parameters of fully-connected layers. Then the routing vector p_u^t is subsequently spilt into the connection weights of different modules in adjacent layers. Additionally, a softmax function is applied to normalize the weights pointed to the same module, that is:

$$\hat{p}_{i,j;u}^{l;t} = \frac{exp(p_{i,j;u}^{l;t})}{\sum_{j=1}^{m} exp(p_{i,j;u}^{l;t})}, \tag{2}$$

where $\hat{p}_{i,j;u}^{l;t}$ is the probability weighting the contribution of the jth module in the l layer to the ith module in the $l+1$ layer, utilized by agent u at time step t. With Eq. 1 and Eq. 2, we can obtain weights connecting modules in adjacent layers for each agent at each time step.

4.2 Base Modular Network

As mentioned above, the base modular network contains L layers, each with m modules, and connecting weights of modules are generated by the routing network. We denote the input for the jth module in the l layer is a d-dimensional feature $g_{j;u}^{l;t} \in R^d$, then the input feature for the ith module in the $l+1$ layer can be expressed as:

$$g_{i;u}^{l+1;t} = \sum_{j=1}^{m} \hat{p}_{i,j;u}^{l;t} \left(ReLU(W_j^l g_{j;u}^{l;t}) \right), \tag{3}$$

where W_j^l represents the parameters of the jth module in the l layer. And the output of local utility function of agent u is:

$$Q_u^t = F\left(\sum_{j=1}^{m} \hat{p}_{j;u}^{L;t} (ReLU(W_j^L g_{j;u}^{L;t})) \right), \tag{4}$$

where F is the parameter of the last layer MLP in the local utility function network, and $\hat{p}_j^L$ represents the weights of module outputs in the L layer to the last layer MLP in the local utility function network. With Eq. 3 and Eq. 4, we can calculate the weighted sum of different module outputs by the probabilities produced by the routing network. Consequently, driven by the observation and the identity of each agent, the inputs of different agents will flow into similar or different forward paths, yielding corresponding outputs, *i.e.*, their action-value utilities Q_u.

4.3 Mutual Information Regularization

In ADMN, we want agents with different routing paths to generate diverse behavior and prevent different routing from collapsing into similar behavior. To further strengthen the influence of different routing, we introduce an auxiliary objective to maximize the mutual information between the routing p_u^t and the agent's behavior.

As discussed in [Eysenbach *et al.*, 2018], different actions may yield similar effects on the environment and are indistinguishable, like different force values applied by a robotic to grasp a cup which does not move. In this paper, to enable the identification of routing based on the agent's behavior, and make the influence of the routing visible, we associate the agent routing with observable agent observation transitions rather than agent actions. Concretely, we introduce a conditional mutual information restriction between the agent routing and its next observation conditioned on the current observation of the agent, which is:

$$\begin{aligned} I(p_u^t; o_u^{t+1} | o_u^t) &= H(p_u^t | o_u^t) - H(p_u^t | o_u^{t+1}, o_u^t) \\ &= H(o_u^{t+1} | o_u^t) - H(o_u^{t+1} | p_u^t, o_u^t) \\ &= E_D \left[\log p(o_u^{t+1} | p_u^t, o_u^t) - \log p(o_u^{t+1} | o_u^t) \right], \end{aligned} \tag{5}$$

where D is the replay buffer.

Since the probabilities $p(o_u^{t+1}|p_u^t, o_u^t)$ and $p(o_u^{t+1}|o_u^t)$ are unknown, we introduce the variational distribution $q_\xi(o_u^{t+1}|p_u^t, o_u^t)$ and $q_\zeta(o_u^{t+1}|o_u^t)$ parameterized by ξ and ζ as a proxy, obtaining an approximation of the optimization objective:

$$I(p_u^t; o_u^{t+1} | o_u^t) \approx E_D \left[\log q_\xi(o_u^{t+1} | p_u^t, o_u^t) - \log q_\zeta(o_u^{t+1} | o_u^t) \right]. \tag{6}$$

To affect the training of all parameters in the agent network, we derive an auxiliary reward to optimize the above objective:

$$r_t^I = E_u \left[\log q_\xi(o_u^{t+1} | p_u^t, o_u^t) - \log q_\zeta(o_u^{t+1} | o_u^t) \right], \tag{7}$$

which will be combined with the extrinsic reward to guide agents training. More details are discussed in Appendix.

4.4 Overall Learning Objective

In this section, we introduce the overall learning objective of ADMN. Specifically, we add the auxiliary reward discussed in Eq. 7 to the extrinsic environmental rewards r, and utilize the following temporal-difference (TD) loss [Sutton and Barto, 2018] to optimize the model parameters in ADMN:

$$L_{TD} = \left(y_t^{tot} - Q_{tot}(\tau_t, \boldsymbol{a_t}) \right)^2, \tag{8}$$

where Q_{tot} is the output of the mixing network with the input of the utility of each agent Q_u^t, calculated in Eq. 4. And $y_t^{tot} = \hat{r}_t + \gamma \hat{Q}_{tot} \left(\tau_{t+1}, \arg\max_{\boldsymbol{a}_{t+1}} Q_{tot}(\tau_{t+1}, \boldsymbol{a}_{t+1}) \right)$, where $\hat{r}_t$ is the combination reward of the auxiliary reward r_t^I and the environment reward r_t with combination weight β, and $\hat{Q}_{tot}$ is the target network periodically copied from Q_{tot} for stable training as proposed in [Mnih *et al.*, 2015]. The action input to the target network is chosen by Q_{tot} as advised by [Van Hasselt *et al.*, 2016].

5 Experiments

We conduct experiments to answer the following questions: 1) Is ADMN more effective in balancing efficiency and diversity in complex cooperative tasks? 2) How does each portion contribute to ADMN, such as the design of the modular network, and the information-theoretical regularization? 3) Is the routing learned by each agent dynamic and reasonable? 4) What is the influence of different routing on the agent behavior? In the following, we will first introduce the environment used for evaluating our method and the baseline methods for comparison, then answer the questions above one by one.

5.1 Experimental Setup

We evaluate ADMN and baselines in the challenging StarCraft II micromanagement tasks (SMAC) and the Google

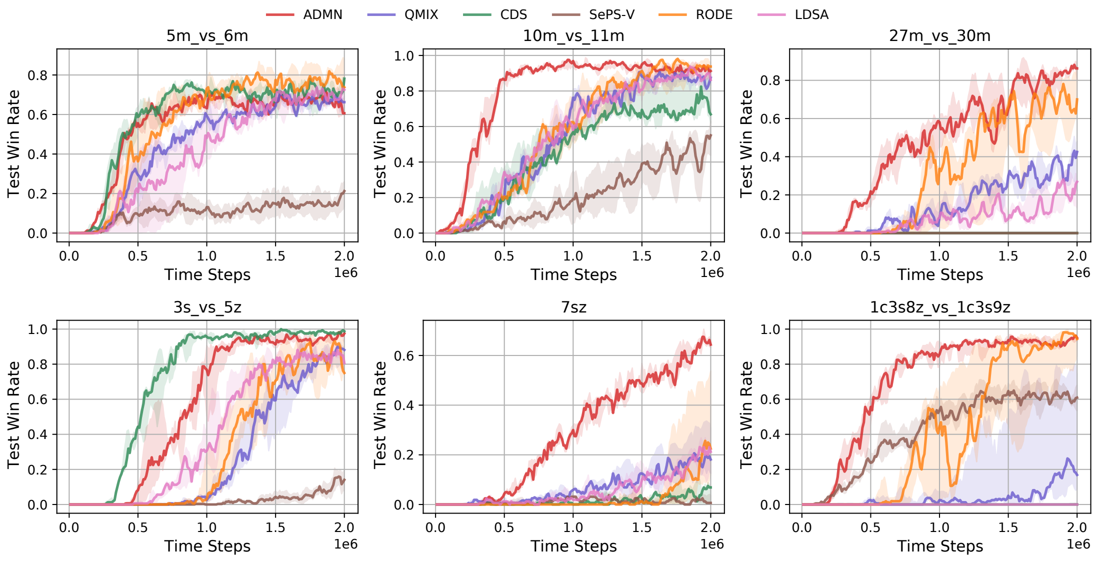


Figure 4: Training curves of different methods in SMAC tasks. In tasks with a larger number of homogeneous agents (27m_vs_30m) or heterogeneous agents (1c3s8z_vs_1c3s9z), ADMN outperforms naive parameter sharing or no parameter sharing methods, highlighting the efficacy of its dynamic balance between efficiency and diversity across tasks with different requirements of agent diversity.

Research Football (GRF). The task in SMAC is a two-team battle game, where the algorithm needs to guide controlled alley agents to eliminate all enemy units in the other team. GRF provides an environment where agents are trained to play football in a physics-based 3D simulation. We evaluated our method on 3 challenging GRF offensive scenarios, academy_3_vs_1_with_keeper, academy_counterattack_easy, and academy_counterattack_hard, where agents need to coordinate to organize the offense in different initial positions. Different from the dense rewards setting in SMAC, agents in GRF will only be rewarded when scoring goals.

To evaluate the effectiveness of ADMN, firstly, we compare it with current state-of-the-art methods that focus on the parameter sharing issue. We provide the results of QMIX [Rashid *et al.*, 2018], which adopts naive parameter sharing mechanism, CDS [Li *et al.*, 2021], which incorporates agent diversity with individual network heads, and SePS, which utilizes selective parameter sharing by grouping agents before cooperative learning. Specially, we migrate SePS from the policy-based algorithm to the value-based algorithm named SePS-V for a fair comparison, and provide the comparison among policy-based algorithms in Appendix. Additionally, we also provide the comparison with methods related to task decomposition. Specifically, we compare with RODE [Wang *et al.*, 2021b], which decomposes the task according to the division of the action space, and LDSA [Yang *et al.*, 2022], which first learns several subtask representations, then assigns agents to the subtask according to learned agent representation. Results of the baselines we compare with are obtained using the publicly available code released by their authors. Referring to existing MARL methods [Samvelyan *et al.*, 2019; Wang *et al.*, 2021b; Li *et al.*, 2021], each algorithm is evaluated using 5 independent training runs with different random seeds, and the resulting plots include the median performance shown in dark color as well as the 25%-75% percentiles shown in the shaded area.

We conduct the experiment of ADMN with the setting that the number of module layers L is 2, the number of modules in each layer m is 2, and each module outputs a $d = 32$ representation. And β weighing the environment rewards and the auxiliary rewards is 0.05. More details about the environment and training are provided in Appendix.

5.2 Performance on SMAC

To evaluate the balance of ADMN between efficiency and diversity, we evaluate ADMN and baselines in homogeneous SMAC maps with varying numbers of agents, as well as heterogeneous maps with different numbers of unit types. In homogeneous tasks, necessary parameter sharing is crucial for training efficiency, while in heterogeneous tasks, necessary diversity is essential to achieve sophisticated coordination. Results are shown in Figure 4, where in larger homogeneous or more heterogeneous cooperative tasks, such as 27m_vs_30m and 1c3s8z_vs_1c3s9z, ADMN outperforms both naive parameter sharing methods like QMIX, and diversity-based methods like CDS or SePS-V. This comparison highlights that, a dynamic and adaptive balance is effective both for tasks that demand training efficiency and for tasks that require diverse agent capabilities. Besides, we found that CDS exhibits better performance in small-scale homogeneous cooperative tasks but falls short in larger tasks. This result justifies that such fixed diversity pattern with individual networks may sacrifice the efficiency of parameter sharing, and is inefficient when scaled to tasks with a large number of agents. Results on other hard SMAC tasks are provided in Appendix, where ADMN is still comparable or even better than baselines.

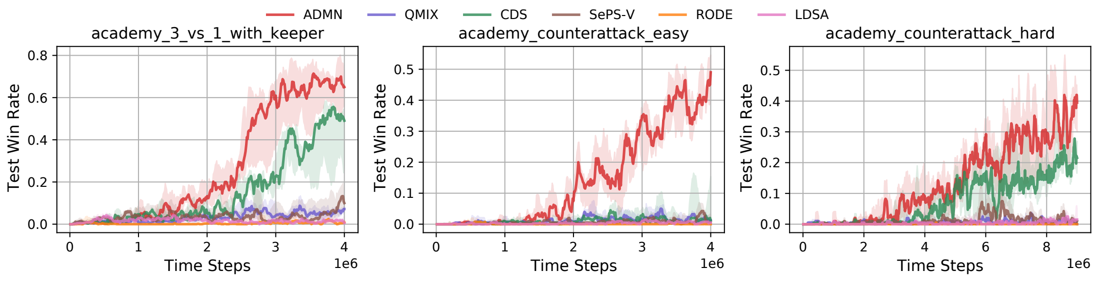


Figure 5: The training performance of various methods in GRF tasks. The consistently superior performance of ADMN underscores its effectiveness in enabling agents to strike an advanced balance, facilitating the discovery of sophisticated and coordinated policies.

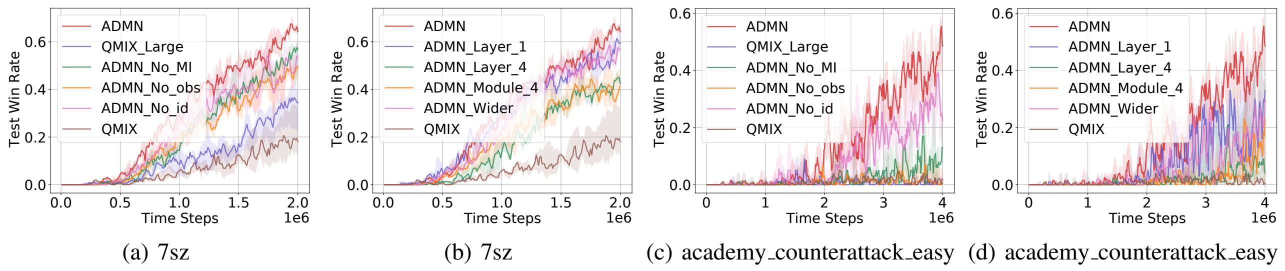


Figure 6: Ablation studies of the design in the routing network and base modular network in task 7sz and academy_counterattack_easy.

5.3 Performance on GRF

We further compare ADMN with baselines in challenging football games GRF, and results are presented in Figure 5. In Figure 5, ADMN consistently outperforms baselines in these three scenarios. Considering that tasks in GRF are in a sparse-reward setting and ask for more sophisticated coordination, ADMN's superior performance underscores its effectiveness in striking a more efficient balance, enabling agents to discover advanced coordinated policies. Besides, we notice that task decomposition methods like RODE, which performs relatively better in SMAC tasks, do not fare as well in GRF tasks. We posit that this phenomenon arises because RODE decomposes the entire task based on the decomposition of the action space, which is beneficial for SMAC tasks but does not apply to others. On the contrary, without introducing any task-related prior knowledge, ADMN finds an adaptive tradeoff between the efficiency and diversity in multiple tasks in an end-to-end manner.

5.4 Ablation Studies

To investigate the significance of different portions in ADMN, we conduct ablation studies. First, considering that the introduction of the modular network enlarges the size of the agent network, we present the results of QMIX_Large, *i.e.*, QMIX with similar size of the agent network in ADMN. Then for the routing network, we ablate the mutual information regularization in ADMN (ADMN_No_MI). Additionally, we ablate the input of the routing network, namely the agent observation and identity and obtain ADMN_no_obs and ADMN_no_id. Figure 6(a) illustrates the comparison of different ablations in SMAC task 7sz. Results indicate that the performance gains from increasing the network parameters are limited. Besides, the mutual information regularization indeed impacts the learning efficiency. Furthermore, note that both ADMN_no_obs and ADMN_no_id perform worse than ADMN, highlighting the importance of both identity and observation for the routing network in this SMAC task.

Besides, we conduct ablation studies on the base modular network design in ADMN. ADMN utilizes a modular network with 2 layers, and each layer has 2 modules with 32 neural units. We varied the network depth by replacing the number of layers with 1 (ADMN_Layer_1) and 4 (ADMN_Layer_4). Additionally, we explored configurations with 4 modules in each layer, featuring either 16 units (ADMN_Module_4, maintaining the same unit count as ADMN) or 32 units (ADMN_Wider). Figure 6(b) reveals that all ablations outperform the baseline QMIX. However, unlike modular networks with shallow layers, the deeper modular network (ADMN_Layer_4) is a little hard to train. Furthermore, ADMN_Module_4, the algorithm with a smaller number of unit sizes, is inferior to ADMN_Wider and ADMN, illustrating that the unit size in each module is crucial for ensuring adequate module specialization.

We extend above ablation studies to the GRF task academy_counterattack_easy in Figure 6(c, d). Similarly, results show that naively increasing the network parameters can not improve the performance effectively, and mutual information regularization is important. Notably, in the routing network, the absence of agent observation has a more adverse impact on performance than the absence of agent identity. This underscores the heightened importance of observation input for adaptive coordination in such dynamic tasks. In the design of the base modular network in the GRF environment, considering tasks in GRF are based on physics engine

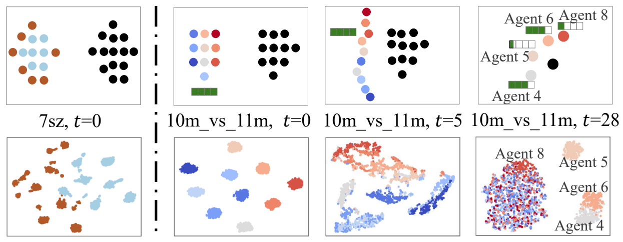


Figure 7: The visualization of agents' positions and health (top row) and their related routing embedding via t-SNE (bottom row). The first column illustrates the visualization in heterogeneous task 7sz, where ADMN learns to distinguish agents by their positions and unit types. The last 3 columns showcase the visualization in homogeneous task 10m_vs_11m at different time steps, where the routing is dynamically related to the agents' position (t=5) and health (t=28).

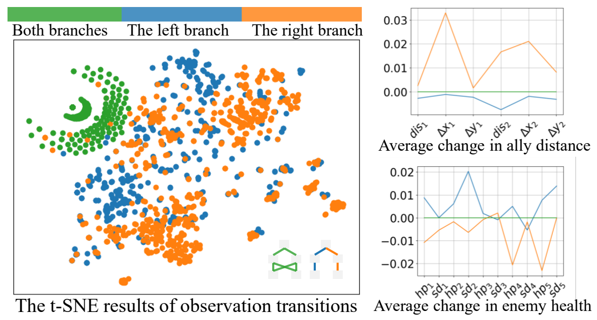


Figure 8: The visualization depicts the impact of agents' routing on agent observation transitions. Distinct routing leads to varied observation transitions, promoting diverse behaviors exhibited by agents.

and own more complex observation and action space, ADMN is relatively sensitive. While most ablations improve performance compared to QMIX, increased oscillation is observed, and a deeper modular network contributes minimally. More ablation studies are provided in Appendix.

5.5 Visualization of Routing

We analyze agents routing and visualize the probability p_u of each agent via t-SNE [Van der Maaten and Hinton, 2008]. Results are presented in Figure 7. The first column depicts initial positions and learned routing embedding of agents in task 7sz. The last three columns show the states and t-SNE of routing samples about agents in 10m_vs_11m at different time steps. Notably, colored circles in snapshots represent agents controlled by ADMN, green bars depict the health of controlled agents at different time steps, and black circles represent the enemies controlled by the environment.

In task 7sz, circles in red and blue represent two unit types (zealot and stalker) separately. Results demonstrate that ADMN enables agents to learn distinct routing policies based on their unit type. In homogeneous task 10m_vs_11m, when t is 0, our controlled agents gather together, and ADMN learns 10 separate clusters one for each agent. When t is 5, agents adopt a loose formation to gain a larger attack range, with routing policies of agents in close positions visually converging. When t is 28, with the battle goes on, agents vary in health and positions. Notably, routing of died agents is mixed. For surviving agents, routing embedding of healthier agents, like agents 4 and 6, is close yet distinct from agents with closer positions but lower health values. This observation indicates that ADMN learns to categorize agents based on both position and health, facilitating adaptable parameter sharing. The visualization of agents' routing at different time steps or tasks justifies that ADMN adeptly generates dynamic and reasonable routing for agents based on their conditions.

In addition to visualizing agents' routing, we aim to further investigate the impact of different routing strategies on agents. Considering that ADMN employs a soft routing paradigm with an infinite number of possible routing, we sample a subset and analyze their influence. Figure 8 illustrates examples of routing focusing more on the left branch (blue), the right branch (orange) and both branches (green). Note that the mutual information regularization in Eq. 5 encourages the relation of the routing and the observation transition of agents, hence we present the t-SNE results of observation changes in consecutive time steps of agents in the bottom left in Figure 8. The visualization reveals that distinct routing results in different observation transition distributions.

To delve deeper into the detailed influence of different routing, we plot the average change in observation related to these three routing on the right of Figure 8. Results indicate that the routing focusing on both branches has no impact on the agent observation, which might be adopted by dead agents. Besides, left-branch-focused routing reduces the distance between the agent and its allies, and increases the observed health of enemies. Given the partial observability of agents, the rise in enemies' health signals increased chances of agent-enemy encounters, where enemies become visible with positive health values instead of zero. And this routing may lead to behaviors resembling support for attacked allies. Conversely, right-branch-focused routing enlarges the distance to both allies and enemies, considering the changes in enemy health are nearly opposite to the left branch, suggesting behaviors akin to recuperation away from the battlefield. The visualization in Figure 8 illustrates the nuanced influence of different routing on agents, contributing to the enrichment of behavioral diversity.

6 Conclusion

In this paper, we introduce ADMN, a novel framework incorporating an agent-driven modular network to address the challenge of balancing training efficiency and agent diversity in multi-agent cooperation. ADMN enables agents to dynamically configure similar or distinct module routings in an end-to-end manner, offering adaptability to diverse cooperative tasks and dynamic environments. And the mutual information regularization within ADMN enhances the identification of different routing strategies. While ADMN demonstrates superior performance in challenging cooperative tasks, the interpretability of the learned modules remains an area for improvement. Future work will focus on investigating the specific roles and analyzing the importance of different modules in ADMN for enhanced practical applications.

Acknowledgments

This work is supported in part by the National Science and Technology Major Project (Grant No.2022ZD0116403), the National Natural Science Foundation of China (Grant No.62176255), the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No.XDA27010103), the Youth Innovation Promotion Association CAS, and the Postdoctoral Fellowship Program of CPSF (Grant No.GZC20232995).

References

[Bard *et al.*, 2020] Nolan Bard, Jakob N Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216, 2020.

[Bazzan, 2009] Ana LC Bazzan. Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multi-Agent Systems*, 18:342–375, 2009.

[Berner *et al.*, 2019] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[Cao *et al.*, 2012] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial informatics*, 9(1):427–438, 2012.

[Christianos *et al.*, 2021] Filippos Christianos, Georgios Papoudakis, Muhammad A Rahman, and Stefano V Albrecht. Scaling multi-agent reinforcement learning with selective parameter sharing. In *International Conference on Machine Learning*, pages 1989–1998. PMLR, 2021.

[Chu *et al.*, 2019] Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1086–1095, 2019.

[DeLoach and Garcia-Ojeda, 2010] Scott A DeLoach and Juan Carlos Garcia-Ojeda. O-mase: a customisable approach to designing and building complex, adaptive multiagent systems. *International Journal of Agent-Oriented Software Engineering*, 4(3):244–280, 2010.

[Eysenbach *et al.*, 2018] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

[Foerster *et al.*, 2018] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[Hu *et al.*, 2022] Siyi Hu, Chuanlong Xie, Xiaodan Liang, and Xiaojun Chang. Policy diagnosis via measuring role diversity in cooperative multi-agent rl. In *International Conference on Machine Learning*, pages 9041–9071. PMLR, 2022.

[Hüttenrauch *et al.*, 2019] Maximilian Hüttenrauch, Sosic Adrian, Gerhard Neumann, et al. Deep reinforcement learning for swarm systems. *Journal of Machine Learning Research*, 20(54):1–31, 2019.

[Iqbal *et al.*, 2021] Shariq Iqbal, Christian A Schroeder De Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. Randomized entity-wise factorization for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 4596–4606. PMLR, 2021.

[Jiang and Lu, 2021] Jiechuan Jiang and Zongqing Lu. The emergence of individuality. In *International Conference on Machine Learning*, pages 4992–5001. PMLR, 2021.

[Kim and Sung, 2023] Woojun Kim and Youngchul Sung. Parameter sharing with network pruning for scalable multi-agent deep reinforcement learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pages 1942–1950, 2023.

[Kurach *et al.*, 2020] Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4501–4510, 2020.

[Lhaksmana *et al.*, 2018] Kemas M Lhaksmana, Yohei Murakami, and Toru Ishida. Role-based modeling for designing agent behavior in self-organizing multi-agent systems. *International Journal of Software Engineering and Knowledge Engineering*, 28(01):79–96, 2018.

[Li *et al.*, 2021] Chenghao Li, Tonghan Wang, Chengjie Wu, Qianchuan Zhao, Jun Yang, and Chongjie Zhang. Celebrating diversity in shared multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:3991–4002, 2021.

[Lowe *et al.*, 2017] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, 30:6379–6390, 2017.

[Matignon *et al.*, 2012] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31, 2012.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[Peng *et al.*, 2012] Chunyi Peng, Minkyong Kim, Zhe Zhang, and Hui Lei. Vdn: Virtual machine image distribution network for cloud data centers. In *2012 Proceedings IEEE INFOCOM*, pages 181–189. IEEE, 2012.

[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.

[Rosenbaum *et al.*, 2019] Clemens Rosenbaum, Ignacio Cases, Matthew Riemer, and Tim Klinger. Routing networks and the challenges of modular and compositional computation. *arXiv preprint arXiv:1904.12774*, 2019.

[Samvelyan *et al.*, 2019] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 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.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Tan, 1993] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *International Conference on Machine Learning*, pages 330–337. PMLR, 1993.

[Van der Maaten and Hinton, 2008] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[Van Hasselt *et al.*, 2016] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[Wang *et al.*, 2021a] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations*. OpenReview, 2021.

[Wang *et al.*, 2021b] Tonghan Wang, Tarun Gupta, Anuj Mahajan, Bei Peng, Shimon Whiteson, and Chongjie Zhang. Rode: learning roles to decompose multi- agent tasks. In *Proceedings of the International Conference on Learning Representations*. OpenReview, 2021.

[Wooldridge *et al.*, 2000] Michael Wooldridge, Nicholas R Jennings, and David Kinny. The gaia methodology for agent-oriented analysis and design. *Autonomous Agents and multi-agent systems*, 3:285–312, 2000.

[Xu *et al.*, 2023a] Pei Xu, Junge Zhang, and Kaiqi Huang. Exploration via joint policy diversity for sparse-reward multi-agent tasks. In Edith Elkind, editor, *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 326–334. International Joint Conferences on Artificial Intelligence Organization, 8 2023. Main Track.

[Xu *et al.*, 2023b] Zhiwei Xu, Bin Zhang, Dapeng Li, Zeren Zhang, Guangchong Zhou, Hao Chen, and Guoliang Fan. Consensus learning for cooperative multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11726–11734, 2023.

[Yang *et al.*, 2020] Ruihan Yang, Huazhe Xu, Yi Wu, and Xiaolong Wang. Multi-task reinforcement learning with soft modularization. *Advances in Neural Information Processing Systems*, 33:4767–4777, 2020.

[Yang *et al.*, 2022] Mingyu Yang, Jian Zhao, Xunhan Hu, Wengang Zhou, Jiangcheng Zhu, and Houqiang Li. Ldsa: Learning dynamic subtask assignment in cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1698–1710, 2022.

[Yu *et al.*, 2021] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of mappo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[Zhang *et al.*, 2021a] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384, 2021.

[Zhang *et al.*, 2021b] Tianhao Zhang, Yueheng Li, Chen Wang, Guangming Xie, and Zongqing Lu. Fop: Factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 12491–12500. PMLR, 2021.

[Zhou *et al.*, 2020] Meng Zhou, Ziyu Liu, Pengwei Sui, Yixuan Li, and Yuk Ying Chung. Learning implicit credit assignment for cooperative multi-agent reinforcement learning. *Advances in neural information processing systems*, 33:11853–11864, 2020.