Multi-Agent Concentrative Coordination with Decentralized Task Representation

Lei Yuan[1,4,*], **Chenghe Wang**[1,*], **Jianhao Wang**[2], **Fuxiang Zhang**[1], **Feng Chen**[1], **Cong Guan**[1], **Zongzhang Zhang**[1], **Chongjie Zhang**[2], **Yang Yu**[1,3,†]
[1]National Key Laboratory for Novel Software Technology, Nanjing University
[2]Institute for Interdisciplinary Information Sciences, Tsinghua University
[3]Peng Cheng Laboratory
[4]Polixir Technologies
{yuanl, wangch, zhangfx, guanc}@lamda.nju.edu.cn, wjh19@mails.tsinghua.edu.cn,
chenf@smail.nju.edu.cn, chongjie@tsinghua.edu.cn, {zzzhang, yuy}@nju.edu.cn

Abstract

Value-based multi-agent reinforcement learning (MARL) methods hold the promise of promoting coordination in cooperative settings. Popular MARL methods mainly focus on the scalability or the representational capacity of value functions. Such a learning paradigm can reduce agents' uncertainties and promote coordination. However, they fail to leverage the task structure decomposability, which generally exists in real-world multi-agent systems (MASs), leading to a significant amount of time exploring the optimal policy in complex scenarios. To address this limitation, we propose a novel framework Multi-Agent Concentrative Coordination (MACC) based on task decomposition, with which an agent can implicitly form local groups to reduce the learning space to facilitate coordination. In MACC, agents first learn representations for subtasks from their local information and then implement an attention mechanism to concentrate on the most relevant ones. Thus, agents can pay targeted attention to specific subtasks and improve coordination. Extensive experiments on various complex multi-agent benchmarks demonstrate that MACC achieves remarkable performance compared to existing methods.

1 Introduction

Multi-Agent Reinforcement Learning (MARL) has attracted widespread attention [Du and Ding, 2021] and been applied in numerous domains, including sensor networks [Zhang and Lesser, 2011], autonomous vehicle teams [Zhou *et al.*, 2020], and traffic signal control [Du *et al.*, 2021]. To address the multi-agent system problems, a popular paradigm named *Centralized Training with Decentralized Execution* (CTDE) [Oliehoek *et al.*, 2008a] is widely adopted, where agents' policies are trained with access to extra global information but executed only based on local information, including both policy gradient methods [Lowe *et al.*, 2017; Wang *et al.*, 2020] and value-based methods [Sunehag *et al.*, 2018; Rashid *et al.*, 2018; Wang *et al.*, 2021a], which demonstrate state-of-the-art performance on challenging tasks (e.g., StarCraft unit micromanagement [Samvelyan *et al.*, 2019]). Despite their success, previous value-based MARL methods struggle in complex scenarios as they mainly consider designing a delicate mixing network but neglect the nearly decomposable structure of complex environments [Simon, 1991; Zhang, 2011; Sarkar and Debnath, 2012], such as agent-wise coordination patterns and agent-task relationships.

Efficient utilizing the environment structure plays a promising role in relieving the mentioned issue. Some researchers decompose the interaction structure among agents to improve coordination. CollaQ [Zhang *et al.*, 2020] decomposes the local observation into different parts and employs a transformer architecture to capture the relationships between the ego agent and its teammates. RODE [Wang *et al.*, 2021b] learns action representations and applies clustering methods to decompose joint action into restricted role spaces to reduce the policy search space. The mentioned methods of learning structure decomposition among agents can improve the coordination ability among agents but are still restricted as they seldom consider the task structure decomposability, a general property of real-world multi-agent systems [Oliehoek *et al.*, 2008b; Krüger, 2020]. For example, in the Factored Firefighting problem [Oliehoek *et al.*, 2008b], several firefighters are assigned to some buildings on fire. The fire of different intensities are the factored state that can be taken as subtasks. Current methods ignore the task structure's decomposability and tacitly assume that the decentralized local policy can autonomously capture the relationship between agents and subtasks, resulting in low coordination efficiency. For example, in the StarCraft II micromanagement tasks [Samvelyan *et al.*, 2019], agents with a sub-optimal policy may attack different enemies, while the optimal approach is to focus fire on a specific enemy to weaken the enemy forces rapidly. By learning useful representations for subtasks, agents can implicitly infer their teammates' intentions and form local groups to focus their energy on mutually beneficial subtasks.

Towards utilizing the task structure decomposability for efficient cooperation, we propose Multi-Agent Concentrative Coordination (MACC), a value-based MARL framework that

[*]Equal contribution.
[†]Corresponding author: Yang Yu.

follows the CTDE paradigm, with which agents can concentrate on most relevant subtasks, forming an implicit coordination pattern of agents based on local information. In MACC, an agent first learns a compact representation for each subtask via a subtask representation model, incorporating its belief about subtasks. During centralized training, the model is optimized to reconstruct the true state history based on the agent's local information. An attention mechanism is then used to obtain targeted weights for different subtasks. During the decentralized execution phase, MACC can build an integrated concentrative representation for subtasks based on local information and use it to augment the local policy.

We conduct extensive experiments on various cooperative multi-agent benchmarks, including level-based foraging [Papoudakis *et al.*, 2021], predator-prey [Boehmer *et al.*, 2020], and the StarCraft II unit micromanagement benchmark [Samvelyan *et al.*, 2019], and compare our method against previous approaches, strong baselines, and ablations of our method. Experimental results show that MACC significantly improves the coordination ability among agents in complex scenarios, and visualization experiments furthermore indicate that MACC can learn meaningful coordination patterns. Careful ablation results further validate the efficacy of each component of our method.

2 Cooperative MAS with Task Decomposition

This paper considers a fully cooperative decomposable multi-agent task where agents only have access to partial observations, which can be modeled as an extended factored Dec-POMDP [Pajarinen and Peltonen, 2011]. The model is defined as $\mathcal{M} = \langle \mathcal{N}, \mathcal{S}, \{A_i\}_{i=1}^n, P, \{O_i\}_{i=1}^n, \Omega, R, \gamma \rangle$, where $\mathcal{N} = \{1, \dots, n\}$ is the set of agents, $s \in \mathcal{S}$ is the state from which agents can get local observation $O_i \in \Omega$. With regard to decomposability, we consider the setting state can be represented as $\mathcal{S} = \{[\mathcal{S}_i]_{i\in\mathcal{N}}, [\mathcal{U}_j]_{j=1}^k, \mathcal{S}_{\text{env}}\}$, where $\mathcal{S}_i$ is the set of features of agent i, $\mathcal{U}_j$ is the set of features of subtask j (e.g., positions) and k is the number of subtasks. $\mathcal{S}_{\text{env}}$ stands for the information irrelevant of agents and subtasks. In order to relieve the partial observability problem, we add an RNN module, GRU [Cho *et al.*, 2014], called agent trajectory encoder, to encode the history $(o_i^1, a_i^1, \dots, o_i^{t-1}, a_i^{t-1}, o_i^t)$ into τ_i, with $a_i^t \in A_i, o_i^t \in O_i$ stand for the action and observation of agent i at time t. At each timestep, each agent selects an action $a_i \in A$, forming a joint action $\boldsymbol{a} \in A = \prod_{i=1}^n A_i$, leading to the next state $s_{t+1} \sim P(s_{t+1}|s_t, \boldsymbol{a})$ and getting a shared reward $R(s_t, \boldsymbol{a})$. The formal objective is to find a joint policy $\boldsymbol{\pi}(\boldsymbol{a}|\boldsymbol{\tau})$ to maximize the global value function $Q_{\text{tot}}^{\boldsymbol{\pi}}(s, \boldsymbol{a}) = \mathbb{E}_{s_t, \boldsymbol{a}_t}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \boldsymbol{a}_t) \mid s_0 = s, \boldsymbol{a_0} = \boldsymbol{a}, \boldsymbol{\pi}]$ with γ indicates the discount factor.

Given a cooperative multi-agent task $\mathcal{M} = \langle \mathcal{N}, \mathcal{S}, \{A_i\}_{i=1}^n, P, \{O_i\}_{i=1}^n, \Omega, R, \gamma \rangle$, a subtask m_j contains $\langle \mathcal{G}_j, \mathcal{S}_j, \{A_i\}_{i\in\mathcal{G}_j}, P, \{O_i\}_{i\in\mathcal{G}_j}, \Omega, R, \gamma \rangle$, where $\mathcal{G}_j \subseteq \mathcal{N}$ is the group of agents assigned to the subtask j, $\mathcal{S}_j = \{[\mathcal{S}_i]_{i\in\mathcal{G}_j}, \mathcal{U}_j, \mathcal{S}_{\text{env}}\}$, and $\{A_i\}_{i\in\mathcal{G}_j}, \{O_i\}_{i\in\mathcal{G}_j}$ are the action and observation spaces for the group of agents, respectively. In our setting, the overall task can be represented as different subtasks but the corresponding optimal groups of agents are usually unknown. If agents can learn to form different groups correctly and efficiently, the search space of targeted policies could be reduced [Zhang, 2011] and the learning efficiency can be improved.

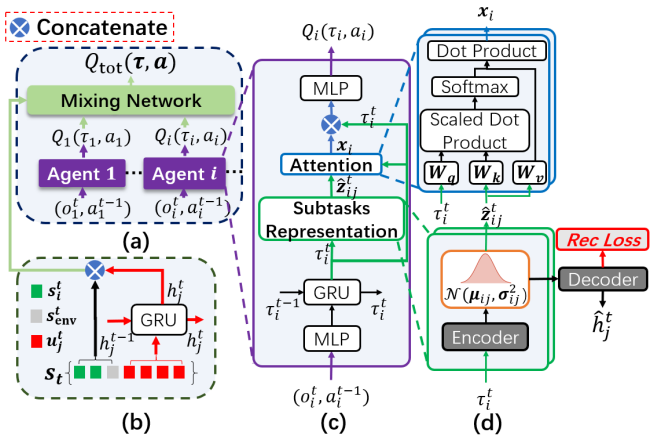


Figure 1: Structure of MACC. (a) The overall architecture. (b) Subtask state encoder. (c) The detailed structure of each agent's network. (d) The structure of the subtask representation module and the subtask attention module.

3 Method

In order to realize the mentioned goal, we propose MACC, a novel value-based MARL framework that follows the CTDE paradigm to learn the proper coordination structure of agents and relieve the MARL miscoordination problem. Figure 1 provides an overview of our MACC framework.

As with most of the existing value-based MARL algorithms, MACC includes individual networks and a mixing network (Figure 1(a)). The individual networks are shown in Figure 1(c), and Figure 1(d) shows the detailed structure of the referred modules. We will introduce the state encoder (Figure 1(b)) module which is only used in centralized training in Section 3.1. Section 3.2 introduces the subtasks representation module that agents use in decentralized execution to capture the information about each subtask. Section 3.3 presents the module with which agents can learn a concentrative learned representation for subtasks and pay targeted attention to different subtasks so that agents can form an implicit coordination pattern to better cooperate.

3.1 Subtask State Encoding

During the centralized training process, we can decompose state s_t into different parts and adopt a GRU [Cho *et al.*, 2014] module called subtask state encoder to encode subtask state history $(u_j^1, \dots, u_j^t)$ into h_j^t. MACC then takes the combination of the environment features s_{env}^t, the current state of each agent s_i^t, and h_j^t as input (Figure 1(a)) instead of the original state s_t as QMIX [Rashid *et al.*, 2018] and QPLEX [Wang *et al.*, 2021a] into the mixing network. During the decentralized execution process, the mixing network and subtask state encoder will be removed. Agents then use the learned subtask representations to anticipate each subtask's true history information based on local information.

3.2 Subtasks Representation Learning

To learn the relationship between the trajectory of a learning agent and each subtask j properly, we assume there exist some latent variables (or embedding) $\boldsymbol{z}_j^t$ that contain information about each subtask j, as well as an generative model $p_{\boldsymbol{\theta}}(h_j^t \mid \boldsymbol{z}_j^t)$ that can reconstruct h_j^t from $\boldsymbol{z}_j^t$, with unknown parameters $\boldsymbol{\theta}$.

Then we seek to approximate the true posterior of $p(\boldsymbol{z}_j^t \mid h_j^t)$ from local information. However, agents have no access to subtask j's accurate history information h_j^t during decentralized execution, so we seek to find a way to get the embedding vectors using only agents' local information. Benefiting from the strong ability of the deep neural networks to identify informative features from data, it is feasible for each agent to build a representation for each subtask from its local information even if there are times when some subtasks are invisible. Notice that agent i has an action-observation history until timestep t, denoted by τ_i^t, we then leverage τ_i^t to anticipate $\boldsymbol{z}_j^t$ with $\hat{\boldsymbol{z}}_{ij}^t$, which is the representation of agent i for subtask j. Formally, agent i inputs τ_i^t into $q_{\boldsymbol{w}}$, a subtask encoder with multiple fully connected layers parameterized with $\boldsymbol{w}$, to learn $\{\hat{\boldsymbol{z}}_{ij}^t\}_{j=1}^k$ for subtasks. We denote the encoder as $q_{\boldsymbol{w}}(\hat{\boldsymbol{z}}_{ij}^t \mid \tau_i^t)$ and use it to approximate the true posterior of $p(\boldsymbol{z}_j^t \mid h_j^t)$. Then the KL divergence of the true posterior from its approximation is calculated as:

$$D_{\mathrm{KL}}(q_{\boldsymbol{w}}(\hat{\boldsymbol{z}}_{ij}^t \mid \tau_i^t) \parallel p(\boldsymbol{z}_j^t \mid h_j^t)). \tag{1}$$

The subtask encoder $q_{\boldsymbol{w}}$ outputs parameters of an n-multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{ij}^t, \boldsymbol{\sigma}_{ij}^t)$, where $\boldsymbol{\mu}_{ij}^t$ and $\boldsymbol{\sigma}_{ij}^t$ are mean and standard deviation, respectively. We then use the reparameterization trick to sample $\boldsymbol{\epsilon}$ from a standard normal distribution, and then shift the sampled $\boldsymbol{\epsilon}$ by the latent distribution's mean $\boldsymbol{\mu}_{ij}^t$ and scale it by the latent distribution's variance $\boldsymbol{\sigma}_{ij}^t$. That is to say, $\hat{\boldsymbol{z}}_{ij}^t = \boldsymbol{\mu}_{ij}^t + \boldsymbol{\sigma}_{ij}^t \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \in \mathcal{N}(0, \boldsymbol{I})$ and $\odot$ represents the Hadamard product.

By following the variational inference theory [Kingma and Welling, 2014], Eq. 1 equals to minimize the representation loss function:

$$\begin{aligned}\mathcal{L}(\tau_i^t, h_j^t; \boldsymbol{\theta}, \boldsymbol{w}) = &-\mathbb{E}_{\hat{\boldsymbol{z}}_{ij}^t \sim q_{\boldsymbol{w}}(\hat{\boldsymbol{z}}_{ij}^t \mid \tau_i^t)}[\log p_{\boldsymbol{\theta}}(h_j^t \mid \hat{\boldsymbol{z}}_{ij}^t)] \\ &+D_{\mathrm{KL}}(q_w(\hat{\boldsymbol{z}}_{ij}^t \mid \tau_i^t) \parallel p(\boldsymbol{z}_j^t)),\end{aligned} \tag{2}$$

where $p_{\boldsymbol{\theta}}$ is the decoder with multiple fully connected layers to approximate the true generative model of subtasks. In the above equation, we can observe that the variational distribution depends only on local information. Under the CTDE paradigm, only the subtask encoder is required to generate representations of the subtasks during the testing period. The first term of Eq. 2 is the reconstruction loss (Rec Loss in Figure 1(d)) of the subtask state history information h_j^t. The second term is the KL-divergence that ensures $q_{\boldsymbol{w}}(\hat{\boldsymbol{z}}_{ij}^t \mid \tau_i^t)$ is similar to the prior distribution $p(\boldsymbol{z}_j^t)$, where $p(\boldsymbol{z}_j^t)$ is a standard normal distribution. This term is used to keep the representations of similar data points close together rather than separated in different regions of the encoding space. Then we sample the subtask representation $\hat{\boldsymbol{z}}_{ij}^t$ from the multivariate Gaussian distribution with parameters $\boldsymbol{\mu}_{ij}^t$ and $\boldsymbol{\sigma}_{ij}^t$ during the training process.

3.3 Concentrative Subtask Representation

To cooperate more efficiently, each agent is expected to build a representation for each subtask and concentrate on relevant ones to adjust its policy accordingly and implicitly form a coordination pattern. We then adopt a simple attention mechanism [Vaswani *et al.*, 2017] to select the most relevant subtasks for each agent. Formally, we calculate a weighted combination of the subtask representations:

$$\boldsymbol{x}_i = \sum_{j=1}^{n} \alpha_{ij} \boldsymbol{v}_{ij} = \sum_{j=1}^{n} \alpha_{ij} \boldsymbol{W}_v \hat{\boldsymbol{z}}_{ij}^t. \tag{3}$$

Here, the sum of the weights $\sum_{j=1}^{n} \alpha_{ij}$ is equal to one, and the value $\boldsymbol{v}_{ij}$ is the result of a linear transformation of the according representation by a shared value matrix $\boldsymbol{W}_v$. The attention weight α_{ij} computes the relevance between the historical information τ_i^t and the related subtask representation $\hat{\boldsymbol{z}}_{ij}^t$, and we use the softmax operator to compute it:

$$\alpha_{ij} = \frac{\exp(\lambda \boldsymbol{W}_q \tau_i^t \boldsymbol{W}_k \hat{\boldsymbol{z}}_{ij}^t)}{\sum_{j=i}^{k} \exp(\lambda \boldsymbol{W}_q \tau_i^t \boldsymbol{W}_k \hat{\boldsymbol{z}}_{ij}^t)}, \tag{4}$$

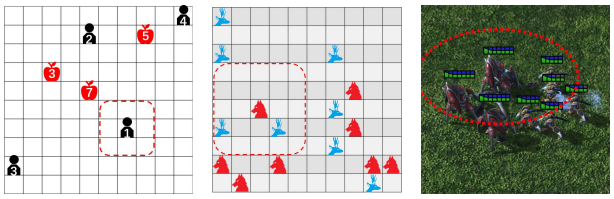
where $\lambda \in \mathbb{R}^+$ is the temperature parameter set to 1 by default, $\boldsymbol{W}_q$ is a shared query matrix to transform τ_i^t into "query", and $\boldsymbol{W}_k$ is a shared key matrix to transform $\hat{\boldsymbol{z}}_{ij}^t$ into "key". Then, we use a Fully Connected (FC) layer which takes the concatenation of τ_i^t and $\boldsymbol{x}_i$ as its input to calculate the individual Q-value function $Q_i(\tau_i, a_i)$.

3.4 Overall Optimization Objective

For the training object of our method, as MACC is agnostic to any value based MARL methods follow CTDE paradigm, here we take our original version based on QMIX to illustrate the optimization objective. MACC can be trained in an end-to-end way, thus all parameters in the framework are updated with the gradients induced from the standard TD loss of reinforcement learning and the representation loss. To compute the global TD loss, individual utilities are fed into a mixing network whose output is the estimation of joint action-value Q_{tot}, and the parameters of the mixing network are generated by a hyper-network conditioned on state s_t. Note that MACC decomposes it and encodes subtask state history information with a GRU. Hence the final learning objective to minimize is:

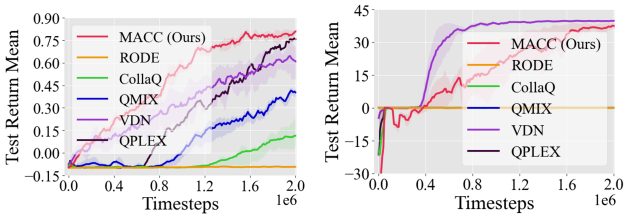
$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_{\mathrm{TD}}(\boldsymbol{\theta}) + \lambda_I \sum_{i=1}^{n} \sum_{j=1}^{k} \mathcal{L}(\tau_i^t, h_j^t; \boldsymbol{\theta}, \boldsymbol{w}). \tag{5}$$

Here, $\mathcal{L}_{\mathrm{TD}}(\boldsymbol{\theta})$ is the temporal difference loss, which equals to $[r + \gamma \max_{\boldsymbol{a}'} Q_{\mathrm{tot}}(s', \boldsymbol{a}'; \boldsymbol{\theta}^-) - Q_{\mathrm{tot}}(s, \boldsymbol{a}; \boldsymbol{\theta})]^2$, where $\boldsymbol{\theta}^-$ are parameters of a periodically updated target network. λ_I is an adjustable hyperparameter to achieve a trade-off between the temporal difference loss and the summation of all agents' subtask representation learning loss. In our CTDE framework, the mixing network will be removed after the centralized training, thus only the learned subtask representation encoder, the attention module, and individual utility networks are used in decentralized execution.



(a) LBF (b) PP (c) SMAC

Figure 2: Scenarios of the three environments that we carry out our experiments on, where the area surrounded by the dotted line indicates the vision range. (a) Level-based foraging (LBF). (b) Predator-prey (PP). (c) StarCraft II Micromanagement Benchmark (SMAC).



(a) Test return on LBF (b) Test return on PP

Figure 3: Test return mean for different methods on LBF and PP.

4 Experiments

In this section, we evaluate MACC in several complex multi-agent tasks[1] to answer the following questions: (1) Can MACC be applied to multiple existing MARL methods and improve their performance in complex scenarios? (Section 4.2) (2) What are the coordination structures learned by our method? (Section 4.3) (3) How do different components of MACC influence team cooperation? (Section 4.4)

For evaluation, we compare MACC with two value-based MARL baselines, VDN [Sunehag *et al.*, 2018] and QMIX [Rashid *et al.*, 2018], and three state-of-the-art multi-agent algorithms, CollaQ [Zhang *et al.*, 2020], QPLEX [Wang *et al.*, 2021a], and RODE [Wang *et al.*, 2021b]. To ensure fair evaluation, we carry out all the experiments with five random seeds, and the results are shown with a 95% confidence interval. Detailed network architecture and hyperparameters choices are shown in Appendix. Note MACC is implemented on QMIX if not specified based on PyMARL[2].

4.1 Environment

We evaluate the proposed method under environments where subtasks have different strategies (one immobile, one random moving strategy, and one fixed unknown strategy), including level-based foraging (LBF) [Papoudakis *et al.*, 2021], predator-prey (PP) [Boehmer *et al.*, 2020], and StarCraft II unit micromanagement benchmark (SMAC) [Samvelyan *et al.*, 2019]. More detailed descriptions of these environments can be found in Appendix.

Level-Based Foraging (LBF)

LBF (Figure 2(a)) is a cooperative partially observable grid-world game, where the agents and the immobile foods (subtasks) are assigned with different levels and random positions at the beginning, the action space of each agent consists of the movement in four directions, loading food and a "none" action. Agents can coordinate to collect the food only if the sum of their levels is no less than the level of the food, then receive a normalized reward correlated to the level of the food.

Predator-Prey (PP)

PP (Figure 2(b)) is another partially observable grid-world task where m predators (agents) are trained to capture n randomly moving preys (subtasks). Agents get rewards only when there are no empty grids around the prey and at least two predators adjacent to it execute "catch" action concurrently, and miscatching would lead to a punishment of -2, this punishment makes it a challenging benchmark for current MARL methods.

StarCraft II Micromanagement Benchmark (SMAC)

SMAC (Figure 2(c)) is a popularly used combat scenario of StarCraft II unit micromanagement tasks, where we train the ally units to beat enemy units (subtasks) controlled by the built-in AI with an unknown strategy. At each timestep, agents can move or attack any enemies and get a global reward equal to the total damage done to enemy units. We test our method on 15 different maps. The detailed descriptions of this environment can be found in Appendix.

4.2 Competitive Results

Experiments are conducted in this part to show that MACC has superiority over multiple approaches on various benchmarks and is agnostic to multiple existing MARL methods.

Overall Performance Comparison

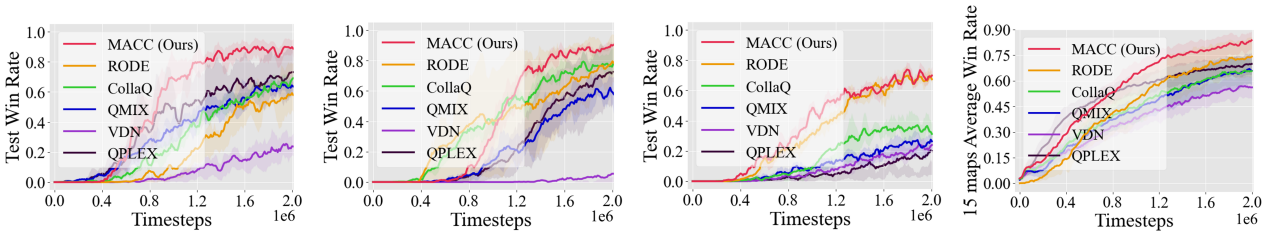
Figures 3 and 4 show the learning curves of different methods in the three multi-agent environments.

As shown in Figure 3(a), MACC outperforms all baselines in LBF. Our method yields a faster convergence speed than QPLEX and higher asymptotic performance than VDN on this benchmark where subtasks are immobile, demonstrating that MACC can efficiently help agents find targeted subtasks and achieve high cooperation ability.

Figure 3(b) exhibits that when we penalize the miscoordination action in a more complex environment with randomly moving subtasks, only VDN and MACC succeed in finding the optimal policy, VDN learns faster than MACC in this environment because the symmetry and linearity of VDN's mixing network make it good at tasks with similar structures. RODE fails in LBF and PP, indicating inappropriate role and subtask assignment may injure the coordination process. CollaQ works worse than QMIX in LBF and fails in PP, because the wrong value attribution decomposition causes inefficient coordination. In addition, QMIX and QPLEX's failures on PP indicate it is of no use for settling some miscoordination problems only with a complex mixing network. MACC

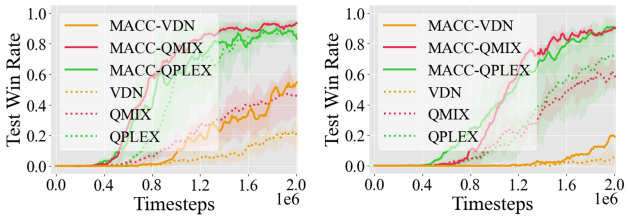
[1]Code available at https://github.com/DrZero0/MACC

[2]Our experiments are all based on the PyMARL framework, which uses SC2.4.6.2.6923. Note that performance is not always comparable among versions.



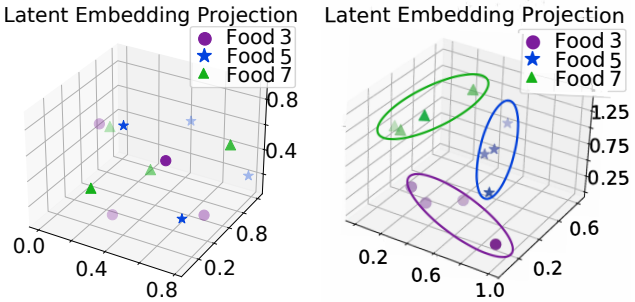
(a) 1c3s5z_vs_1c3s6z (b) MMM2 (c) 10m_vs_12m (d) Averaged test win rates

Figure 4: Episodic win rates with 95% confidence interval of the six evaluated methods during training on SMAC.



(a) 2c_vs_64zg (b) MMM2

Figure 5: Performance comparison when integrating MACC with different baselines on 2c_vs_64zg and MMM2.



(a) t-SNE projection at the start (b) t-SNE projection at the end

Figure 6: The t-SNE projection on LBF, where different colors represent different subtask vectors an agent builds, the same subtasks have the same color, and the number indicates subtask's level.

works on all the two benchmarks, manifesting that MACC is not restricted by these problems, and building a concentrative representation for subtasks can help improve coordination.

For more complex scenarios, we evaluate all methods on the mentioned SMAC environment. From Figure 4(d), we can find VDN gets the lowest averaged win rates, showing a simple addition of individual value functions cannot handle such complex settings. MACC gets lower win rates than QPLEX before 0.8M timesteps, which is because QPLEX's improved network representation capability makes it good at easy and medium maps at the beginning of training. Still, MACC outperforms all other methods gradually and exceeds nearly 10% median test win rate averaged across all 15 maps finally. We can also find that RODE specializes in super hard maps such as MMM2 (Figure 4(b)) and fails in some easy maps, but MACC works well on maps of all difficulty levels. CollaQ only has a slight advantage over QMIX on performance, indicating only local observation cannot capture the relationship between agents on complex maps like 1c3s5z_vs_1c3s6z (Figure 4(a)). More results can be found in Appendix.

Integrating MACC into Different Methods

MACC is agnostic to specific algorithms, and we can integrate it into existing value function factorization methods. We apply MACC to VDN (MACC-VDN), QMIX (MACC-QMIX), and QPLEX (MACC-QPLEX) and test them on maps including 2c_vs_64zg and MMM2. The results are shown in Figure 5. As we can see, on the two maps, MACC-QMIX, MACC-VDN, and MACC-QPLEX all outperform their corresponding vanilla methods, indicating the learned concentrative representation for subtasks of MACC can significantly enhance coordination for various value-based MARL methods.

	Start	Middle	End
LBF	0.973	0.743	**0.285**
PP	1.000	0.860	**0.623**
MMM2	0.961	0.773	**0.615**

Table 1: Relative reconstruction errors at different stages of an episode in three environments.

4.3 Coordination Structure Analysis

This section will show the subtask representation and the dynamic coordination pattern learned by our method.

Subtask Representations Visualization

We analyze the subtask representations learned by the encoder in the LBF environment, where four agents with levels 1, 2, 3, and 4, coordinate to collect three food with levels 3, 5, and 7 (Figure 2(a)). Figure 6 shows the results when we project the representation vectors on the three-dimensional plane using the t-SNE method [Van der Maaten and Hinton, 2008]. We can find the representations are chaotically distributed at the beginning but gradually belong to three clusters as agents better understand the subtasks at the end, indicating MACC can learn useful representations for different subtasks based on local information.

We further consider the reconstruction errors of our method in different settings (MMM2 for SMAC). Table 1 shows the averaged reconstruction errors of all agent-subtask pairs at

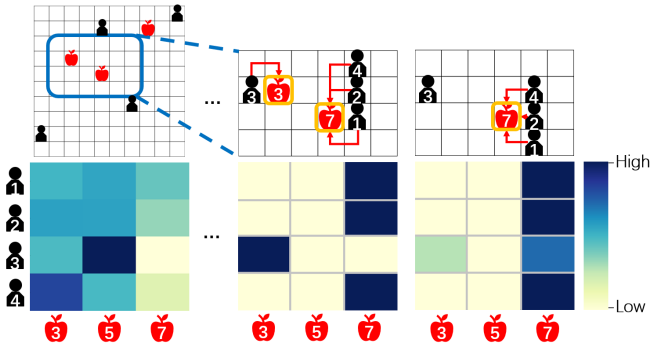


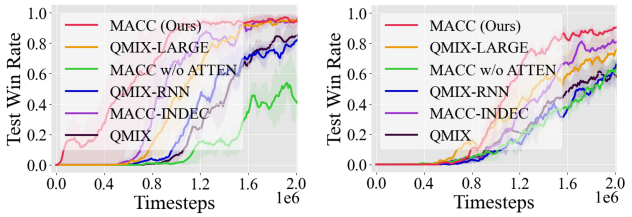
Figure 7: Coordination pattern learned by MACC during one episode. The higher the attention weight, the darker the color of the corresponding position in the heatmap.

specific time points in one episode, where "Start" means timestep $t = 5$, "End" means the timestep t_{end} that the game ends, and "Middle" means $t = \lfloor \frac{t_{\text{end}}}{2} \rfloor$. The reconstruction error is defined as the mean square errors between the true history information and the history information our method predicts of each subtask, and we use the maximum one in an episode to regularize the errors. It can be seen that at the beginning of each episode, the agents are unfamiliar with the subtasks, resulting in a relatively higher reconstruction error. As the game progresses, the agents can have more comprehensive information about the subtasks and reconstruct more precisely. In addition, the reconstruction errors are relatively low in LBF, especially at the end of an episode, which is because the subtasks in LBF are fixed during the game. On the contrary, the errors are higher in PP and MMM2. We guess it is because the subtasks are moving, making it challenging to model precisely.

Coordination Structure Learned by MACC

To investigate the coordination structure MACC learned, Figure 7 shows a visualization analysis on the attention weights of the agents in LBF. We use the same setting in the Subtask Representations Visualization part to keep the consistency of visualization experiments.

In the starting stage of one episode, as the representations are imprecise, agents pay equal attention to all subtasks. After some interactions with the environment, agents have more knowledge about the environment and move forward to collect the foods they are paying attention to (as shown in the middle), forming two different groups for two subtasks. Finally, the agent with level 3 completes the food collection with level 3 and starts to give attention to its next nearest food with level 7. The other three agents are still on their way to collect the food with level 7. The learned implicit coordination pattern validates the effect of MACC. After each agent builds representations for subtasks, it would use its action-observation history and the representations to calculate weights for the subtasks. The action-observation history contains information about its teammates so that the agent can utilize the information to infer which subtasks its teammates are heading for and concentrate on the subtasks that are most suitable for the teamwork, which helps agents group properly and achieve more efficient coordination patterns.



(a) 3s_vs_5z (b) MMM2

Figure 8: Performance comparison for ablation studies.

4.4 Ablation Studies

Finally, we design ablation studies for components of MACC on maps 3s_vs_5z and MMM2 to answer the following questions: (1) Whether the effectiveness of MACC is due to the increase in the number of parameters? (2) Does the attention mechanism make sense? (3) Is the GRU that we add in the mixing network the reason our method works? (4) How does MACC perform when facing tasks can not be decomposed? The results are illustrated in Figure 8. To answer the question (1), when QMIX has an exact number of parameters as MACC (denoted as QMIX-LARGE), QMIX-LARGE improves performance someway but is still not comparable with MACC, proving that the increase in parameters is not the direct reason why MACC works. Moreover, removing the attention module (MACC w/o ATTEN replies to the question (2)) makes MACC fail to learn a good policy because, without an attention module, too many representation vectors enlarge the local policy and increase the learning complexity. QMIX-RNN corresponds to question (3), and the results illustrate that only using the history information in the mixing network can hardly improve performance. To investigate the adaptability of MACC (question (4)), we name this version MACC-INDEC as it learns a whole belief for all subtasks. We find MACC-INDEC can surpass QMIX but is inferior to MACC. It demonstrates that even when the task is indecomposable, our method also makes sense, showing strong adaptability. See Appendix for more ablation studies.

5 Conclusion

We propose MACC, a novel framework that utilizes the task structure decomposability to learn a decentralized concentrative subtasks representation for cooperative MARL, with which agents can concentrate on the most relevant subtasks for efficient coordination. Empirical results in multiple multi-agent cooperative tasks show that our method significantly outperforms various baselines and can be integrated with existing value-based methods to improve coordination. We further design visualization experiments to investigate the coordination pattern learned by MACC, showing it can learn the dynamic coordination pattern of agents, enhancing coordination. Ablation studies validate the effectiveness of each component of MACC, and it can still facilitate coordination even faced with some scenarios where tasks are indecomposable. Further research on automatically decomposing the overall task and building more accurate representations for subtasks under POMDPs would be of great value.

Acknowledgements

We would like to thank Rong-jun Qin, Lanjihong Ma, Yu Fan, Yun-Tian Zhang, Cong Fu, and the anonymous reviewers for their helpful discussions and support. This work is supported by National Key Research and Development Program of China (2020AAA0107200), NSFC (61876077, 61876119), the Major Key Project of PCL (PCL2021A12), and Alibaba Group through Alibaba Research Fellowship Program.

References

[Boehmer *et al.*, 2020] Wendelin Boehmer, Vitaly Kurin, and Shimon Whiteson. Deep coordination graphs. In *ICML*, pages 980–991, 2020.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

[Du and Ding, 2021] Wei Du and Shifei Ding. A survey on multi-agent deep reinforcement learning: from the perspective of challenges and applications. *Artificial Intelligence Review*, 54(5):3215–3238, 2021.

[Du *et al.*, 2021] Xin Du, Jiahai Wang, Siyuan Chen, and Zhiyue Liu. Multi-agent deep reinforcement learning with spatio-temporal feature fusion for traffic signal control. In *ECML*, pages 470–485, 2021.

[Kingma and Welling, 2014] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[Krüger, 2020] Tobias Krüger. *Locating the Source of a WLAN Signal Using Multiple Robots and Dec-POMDP Planning*. PhD thesis, Universität Hamburg, 2020.

[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. In *NIPS*, 2017.

[Oliehoek *et al.*, 2008a] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

[Oliehoek *et al.*, 2008b] Frans A Oliehoek, Matthijs TJ Spaan, Nikos Vlassis, and Shimon Whiteson. Exploiting locality of interaction in factored Dec-POMDPs. In *AAMAS*, pages 517–524, 2008.

[Pajarinen and Peltonen, 2011] Joni Pajarinen and Jaakko Peltonen. Efficient planning for factored infinite-horizon Dec-POMDPs. In *IJCAI*, pages 325–331, 2011.

[Papoudakis *et al.*, 2021] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *NeurIPS*, 2021.

[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 *ICML*, pages 4295–4304, 2018.

[Samvelyan *et al.*, 2019] Mikayel Samvelyan, Tabish Rashid, Christian Schröder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob N. Foerster, and Shimon Whiteson. The Starcraft multi-agent challenge. In *AAMAS*, pages 2186–2188, 2019.

[Sarkar and Debnath, 2012] Anirban Sarkar and Narayan C. Debnath. Measuring complexity of multi-agent system architecture. In *INDIN*, pages 998–1003, 2012.

[Simon, 1991] Herbert A Simon. The architecture of complexity. In *Facets of systems science*, pages 457–476. Springer, 1991.

[Sunehag *et al.*, 2018] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *AAMAS*, pages 2085–2087, 2018.

[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):2579–2605, 2008.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[Wang *et al.*, 2020] Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. Shapley Q-value: A local reward approach to solve global reward games. In *AAAI*, pages 7285–7292, 2020.

[Wang *et al.*, 2021a] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: Duplex dueling multi-agent Q-learning. In *ICLR*, 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. *ICLR*, 2021.

[Zhang and Lesser, 2011] Chongjie Zhang and Victor Lesser. Coordinated multi-agent reinforcement learning in networked distributed POMDPs. In *AAAI*, 2011.

[Zhang *et al.*, 2020] Tianjun Zhang, Huazhe Xu, Xiaolong Wang, Yi Wu, Kurt Keutzer, Joseph E Gonzalez, and Yuandong Tian. Multi-agent collaboration via reward attribution decomposition. *arXiv:2010.08531*, 2020.

[Zhang, 2011] Chongjie Zhang. *Scaling multi-agent learning in complex environments*. PhD thesis, 2011.

[Zhou *et al.*, 2020] Ming Zhou, Jun Luo, Julian Villella, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, et al. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. *arXiv:2010.09776*, 2020.