SEQUENTIAL COMMUNICATION IN MULTI-AGENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Coordination is one of the essential problems in multi-agent reinforcement learning. Communication provides an alternative for agents to obtain information about others so that coordinated behaviors can be learned. Some existing work lets agents communicate predicted future trajectories with others, hoping to get clues about what others would do. However, circular dependencies can inevitably occur when agents are treated equally so that it may not be possible to coordinate decision-making. In this paper, we propose a novel communication scheme *Sequential Communication* (SeqComm) for better coordination. SeqComm treats agents unequally (the upper-level agents make decisions prior to the lower-level) and has two communication phases. In the negotiation phase, agents share observations with others and evaluate their intentions by the environmental model. Agents determine the priority of decision-making by comparing the value of intention. In the launching phase, the upper-level agents take the lead in making decisions and share their actions with the lower-level agents so as to avoid circular dependencies. Empirically, we show that SeqComm outperforms existing communication methods in a variety of multi-agent cooperative tasks.

1 INTRODUCTION

We have witnessed the prosperous development of multi-agent reinforcement learning (MARL) in various applications, such as smart grid (Yang et al., 2018), autonomous driving (Zhou et al., 2020), and intelligent traffic control (Xu et al., 2021). However, partial observability and stochasticity inherent to the nature of multi-agent systems can easily impede the cooperation among agents and lead to catastrophic miscoordination (Ding et al., 2020). Communication has been exploited to help agents obtain extra information during both training and execution to mitigate such problems (Foerster et al., 2016; Sukhbaatar et al., 2016; Peng et al., 2017; Jiang & Lu, 2018). Specifically, agents are entitled to share their information with others by a trainable communication channel. It has been shown that communication enables agents to develop better cooperative strategies compared to communication-free methods.

In cooperative MARL, although a centralized Q-function can be learned to evaluate the joint action of agents, the policies of agents are essentially independent. Therefore, a coordination problem arises. That is, agents may make sub-optimal actions by mistakenly assuming others' actions when there exist multiple optimal joint actions (Busoniu et al., 2008). Communication provides an alternative for agents to obtain information about others. However, most existing work only focuses on communicating messages conditioned on agents' observations or historical trajectories (Jiang & Lu, 2018; Singh et al., 2019; Das et al., 2019; Ding et al., 2020). It is impossible for an agent to acquire other's action before making decision since the game is usually symmetric, *i.e.*, agents make decisions simultaneously. Recently, intention or imagination, depicted by the combination of predicted actions and observations of many future timesteps, has been proposed as part of messages (Kim et al., 2021; Pretorius et al., 2021). However, circular dependencies can inevitably occur so that it may be impossible to coordinate decision-making.

A general approach to solving the coordination problem is to make sure that ties between equally good actions are broken by all agents. One mechanism is to know exactly what others will do and adjust the behavior accordingly under a unique ordering of agents and actions (Busoniu et al., 2008). Inspired by this, we reconsider the cooperative game from an asymmetric perspective. In

other words, each agent is assigned a priority of decision-making at each timestep in both training and execution, and thus Stackelberg equilibrium (Von Stackelberg, 2010) is naturally set up as the learning objective. Under this asymmetric setting, we assume the upper-level agents make decisions prior to the lower-level, which means it is possible for the lower-level agents to acquire the actual actions of the upper-level by communication and make their decisions conditioned on what the upper-level would do. However, *how to decide the priority of decision-making for each agent?* It reminds us that people always negotiate about division of labor, by sharing and evaluating their intentions, before launching the mission. To this end, we adopt multi-round communication scheme so that agents can reach a consensus on the priorities. This is corroborated by many previous methods (Das et al., 2019; Singh et al., 2019; Hoshen, 2017) which believe collaborative strategies require multiple rounds of back-and-forth interactions between agents.

In this paper, we propose a novel multi-round communication scheme for cooperative MARL, *Sequential Communication* (SeqComm), to enable agents explicitly coordinate with each other. More specifically, SeqComm has two-phase communication, negotiation phase and launching phase. In negotiation phase, agents communicate their observations with others simultaneously. Then they are able to generate multiple predicted trajectories, denoted as *intention*, by modeling the environmental dynamics and other agents' actions. In addition, the priority of decision-making is determined by communicating and comparing the corresponding values of agents' intentions. This is because the value of each intention represents the rewards obtained by letting that agent take the upper-level position of decision-making. The sequence of others follows the same procedure as aforementioned with the upper-level agents fixed. In launching phase, the upper-level agents take the lead in decision-making and share their actions with the lower-level agents.

SeqComm is instantiated on PPO (Schulman et al., 2017). The critic and actor of SeqComm agent only takes input as its own observation and received messages. In addition, attention mechanism (Vaswani et al., 2017) is used to accommodate non-predetermined size on input for all the modules in SeqComm. These make SeqComm more practical, since in real-world applications the number of agents may vary over time. We evaluate SeqComm in four multi-agent cooperative tasks in multi-agent particle environment (Lowe et al., 2017) and StarCraft multi-agent challenge (Samvelyan et al., 2019). In all these tasks, we empirically demonstrate that SeqComm outperforms existing communication methods. By ablation studies, we confirm that treating agents unequally is an effective way to promote coordination and SeqComm can provide the proper priority of decision-making for agents to develop better coordination.

2 Related Work

Communication Learning to communicate in MARL has been greatly advanced recently. DIAL (Foerster et al., 2016), CommNet (Sukhbaatar et al., 2016) and BiCNet (Peng et al., 2017) can be considered as first attempts to train a workable communication channel enabling agents to obtain more information for decision-making. However, agents can be overwhelmed by the redundant information as the number of agents grows since not every agent can provide useful messages. The subsequent studies (Jiang & Lu, 2018; Kim et al., 2019; Singh et al., 2019; Das et al., 2019; Zhang et al., 2019; Jiang et al., 2020; Ding et al., 2020) in this realm thus mainly focus on how to extract more valuable messages. ATOC (Jiang & Lu, 2018) and IC3Net (Singh et al., 2019) utilize gate mechanism to decide when to communicate with other agents. TarMAC (Das et al., 2019) employs attention module (Vaswani et al., 2017) to pilot agents to focus on more important message and ignore the irrelevant. I2C (Ding et al., 2020) only communicates with agents that are relevant and influential determined by causal inference. I2C firstly adopts one-to-one communication and it can cut off the useless messages from the source. However, all these methods focus on how to exploit valuable information from current or past partial observations effectively and properly.

More recently, some studies (Kim et al., 2021; Du et al., 2021; Pretorius et al., 2021) begin to answer the question: can we favour cooperation beyond sharing partial observation? They allow agents to imagine their future states with forward model and communicate that with others. IS (Pretorius et al., 2021), as the representation of this line of research, enables each agent to share its intention with other agents in the form of encoded imagined trajectory and use attention module to figure out the importance of received intention. However, two concerns arise. On one hand, circular dependencies can lead to inaccurate predicted future trajectories as long as the multi-agent system treats agents equally. On the other hand, MARL struggles in extracting useful information from

numerous messages, not to mention more complex and dubious messages conditioned on predicted future trajectories.

Unlike existing work, we treat the agents from an asymmetric perspective therefore circular dependencies issue can be naturally resolved. Furthermore, agents only send actions to lower-level agents besides partial observation to make sure the messages are compact as well as informative.

Reinforcement Learning in Stackelberg Game Many previous studies (Könönen, 2004; Sodomka et al., 2013; Greenwald et al., 2003; Zhang et al., 2020) have investigated reinforcement learning in finding Stackelberg equilibrium. Bi-AC (Zhang et al., 2020) is a bi-level actor-critic method that allows agents to have different knowledge base so that Stackelberg equilibrium (SE) is possible to find. The actions still can be executed simultaneously and distributedly. It is claimed that SE is likely to be Pareto superior to the average Nash equilibrium (NE) in games with high cooperation level. AQL (Könönen, 2004) updates the Q-value by solving the SE in each iteration and can be regarded as the value-based version of Bi-AC.

Existing work mainly focuses on two-agent settings and their order is fixed in advanced. However, fixed order can hardly be an optimal solution especially when it comes to large-scale homogeneous agents scenarios. To address this issue, we exploit agents' intentions to dynamically determine the priority of decision-making along the way of interacting with each other.

Multi-Agent Path Finding (MAPF) MAPF aims to plan collision-free paths for multiple agents on a given graph from their given start vertices to target vertices. In MAPF, prioritized planning is deeply coupled with collision avoidance (Van Den Berg & Overmars, 2005; Ma et al., 2019), where collision is used to design constraints or heuristics for planning.

Unlike MAPF, our method couples the priority of decision-making with the learning objective, thus is more general. In addition, the different motivations and problem settings may lead to the incompatibility of the algorithms in the two fields.

3 Sequential Communication for Multi-Agent Coordination

We consider fully cooperative multi-agent tasks that are modeled as Dec-POMDPs augmented with communication. At each timestep t, each agent i gets its own observation o_t^i from the global state s_t, and messages $\boldsymbol{m}_t^{-i}$ from all other agents, denoted as $\boldsymbol{m}_t^{-i} = \{m_t^1, \ldots, m_t^{i-1}, m_t^{i+1}, \ldots, m_t^n\}$. Then, it takes action a_t^i following its policy π_i and receives a shared reward $r(s_t, \boldsymbol{a}_t)$ where $\boldsymbol{a}_t = \{a_t^1, \ldots, a_t^n\}$ is the joint action of all agents. The state transitions to next state s_{t+1} according to the transition probability $\mathcal{T}(s_{t+1}|s_t, \boldsymbol{a}_t)$. Agents are fully cooperative and aim to maximize the expected return $\sum_{t=1}^{T} \gamma^{t-1} r_t$, where γ is the discount factor and T is the episode time horizon.

Sequential communication (SeqComm) is currently instantiated as an extension of Proximal Policy Optimization (PPO) (Schulman et al., 2017), but it can also be realized using any value-based methods. Each SeqComm agent consists of a policy, a critic and a world model, as illustrated in Figure 1. Besides, SeqComm adopts broadcast and multi-round communication mechanism, *i.e.*, agents are allowed to communicate with all other agents in multiple rounds. Importantly, communication is sepa-

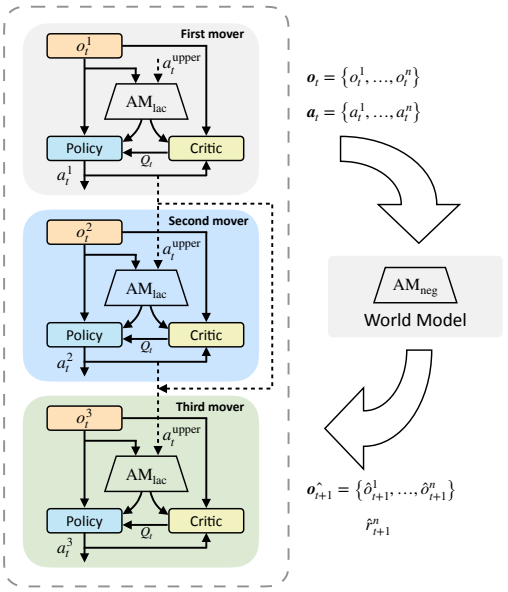


Figure 1: Network architecture. The critic and policy of each agent take input as its own observation and received message. The message includes the observations of all other agents and actions of upper-level agents. Note that the world model takes as input the received joint observation and estimated joint actions.

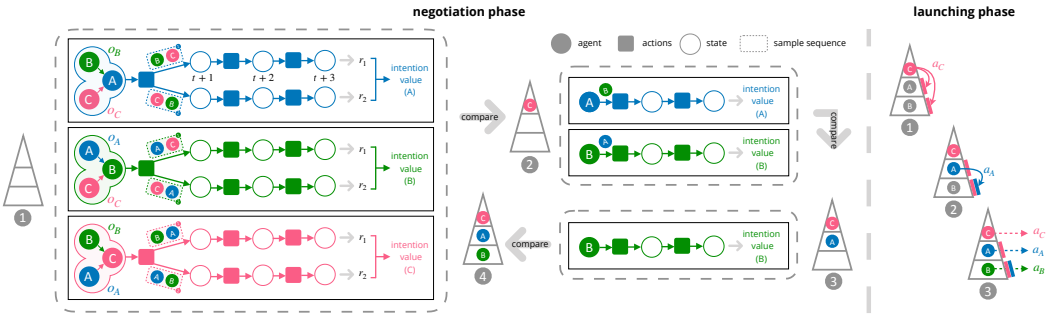


Figure 2: Sequential communication. There are two communication phases, negotiation phase (*left*) and launching phase (*right*). In the negotiation phase, agents communicate observations with others and obtain their own intentions. The priority of decision-making is determined by sharing and comparing the value of all the intentions. In the launching phase, the agents hold the upper-level positions will make decision prior to the lower-level agents. Besides, their actions will be shared to anyone that has not yet made a decision.

rated into two phases serving for different purposes. One is *negotiation* phase for agents to determine the priority of decision-making. Another is *launching* phase for agents to act conditioned on actual actions upper-level agents will take to implement *explicit coordination*.

3.1 Negotiation Phase

In negotiation phase, agents determine the priority of decision-making by intention which is established and evaluated based on the world model as illustrated in Figure 2.

World Model The world model is needed to predict and evaluate future trajectories. Note that the observation predictor of each agent in previous studies (Kim et al., 2021; Du et al., 2021; Pretorius et al., 2021) only conditions on agent's own observation o_t and action a_t to obtain next observation o_{t+1}. However, the transition of environmental state and global reward depends on the joint action of all agents, therefore it may not be possible to accurately predict next observation and reward merely based on agent's own information. SeqComm, unlike previous work, allows agents to share their observations with others through broadcast. Once agent can access to other agents' observations, it shall have adequate information to estimate their actions since all agents are homogeneous and parameter-sharing. Therefore, the world model $\mathcal{M}(\cdot)$ takes as input the joint observation $\boldsymbol{o}_t = \{o_t^1, \ldots, o_t^n\}$ and action $\boldsymbol{a}_t$, and predicts the next joint observation and reward,

$$\hat{\boldsymbol{o}}_{t+1}, \hat{r}_{t+1} = \mathcal{M}_i(\mathrm{AM}_{\mathrm{neg}}(\boldsymbol{o}_t, \boldsymbol{a}_t)), \tag{1}$$

where $\mathrm{AM}_{\mathrm{neg}}$ is the attention module. The reason that we adopt the attention module to process observation-action pairs is to entitle the world model to be generalizable in the scenario where additional agents are introduced or existing agents are removed.

Priority of Decision-Making Intention is the key element to determine the priority of decision-making. The notion of intention is described as agent's future behaviour in previous work (Rabinowitz et al., 2018; Raileanu et al., 2018; Kim et al., 2021). However, we define *intention* as agent's future behaviour *without considering others*, with one more attributive. As mentioned before, agent's intention considering others can lead to circular dependencies and cause miscoordination. By our definition, the intention of an agent should be depicted to all future trajectories by considering itself as first-mover and ignoring others. However, there are many possible future trajectories as the priority of the rest agents is *unfixed*. It is unlikely to enumerate and evaluate all. Therefore, we use Monte Carlo method to evaluate intention.

Taking agent i at timestep t to illustrate, it firstly considers itself as first-mover and produces its action only based on the joint observation, $\hat{a}_t^i \sim \pi_i(\cdot|\boldsymbol{o}_t)$. For the sequence order of lower-level agents, we randomly sample a set of sequence orders from unfixed agents. Assume agent j is the second-mover, agent i models j's action by considering the upper-level action following its own

policy $\hat{a}_t^j \sim \pi_i(\cdot|\boldsymbol{o}_t, \hat{a}_t^i)$. It is the same to predict the actions of all other agents following the sampled sequence order. Combined with the joint observation, the next joint observation $\hat{\boldsymbol{o}}_{t+1}$ and corresponding reward $\hat{r}_{t+1}$ can be predicted by the world model. The length of predicted future trajectory is H and it can then be written as $\tau^t = \{\hat{\boldsymbol{o}}_{t+1}, \hat{\boldsymbol{a}}_{t+1}, \ldots, \hat{\boldsymbol{o}}_{t+H}, \hat{\boldsymbol{a}}_{t+H}\}$ by repeating the procedure aforementioned and the value of one trajectory is defined as the return of that trajectory $v_{\tau^t} = \sum_{t'=t+1}^{t+H} \gamma^{t'-t-1} \hat{r}_{t'}/H$. In addition, the intention value is defined as the average value of F future trajectories with different sampled sequence orders.

After all the agents have computed its own intention and corresponding value, they again broadcast their intention values to other agents and agents would compare and choose the agent with the highest value to be the first-mover. The priority of lower-level decision-making follows the same procedure with the upper-level agents fixed. Note that some agents are required to communicate intention value with others multiple times until the priority of decision-making is finally determined. However, only intention values are needed to transfer multiple times which will not incur too much communication overhead in real applications.

3.2 Launching Phase

As for the launching phase, agents communicate for obtaining messages to make decisions. Apart from the received observations from last phase, we allow agents to get what *actual* actions the upper-level agents will take in execution, while other studies can only infer others' actions by opponent modeling (Rabinowitz et al., 2018; Raileanu et al., 2018) or communicating intention (Kim et al., 2021). Therefore, collision can be avoided and better coordination is possible since lower-level agents can adjust their own behaviours accordingly. The lower-level agent makes its decision following the policy $\pi(a_t|o_t, \mathrm{AM}_{\mathrm{lac}}(\boldsymbol{o}_t, \boldsymbol{a}_t^{upper}))$, where $\boldsymbol{a}_t^{upper}$ means the received actions from all upper-level agents. Besides, we again use an attention module $\mathrm{AM}_{\mathrm{lac}}$ to handle the input. As long as the agent has decided its action, it will send its action to all other lower-level agents by communication channel. Note that the actions are executed simultaneously and distributedly in execution, though agents make decisions sequentially.

3.3 Attention Module

Attention module (AM) is applied to process messages in the world model, critic network, and policy network. AM consists of three components: query, key, and values. The output of AM is the weighted sum of values, where the weight of value is determined by the dot product of the query and the corresponding key.

For AM in the world model denoted as $\mathrm{AM}_{\mathrm{neg}}$, agent i gets messages $\boldsymbol{m}_t^{-i} = \boldsymbol{o}_t^{-i}$ from all other agents at timestep t in negotiation phase, and predicts a query vector q_t^i following $\mathrm{AM}_{\mathrm{neg},q}^i(o_t^i)$. The query is used to compute a dot product with keys $\boldsymbol{k}_t = [k_t^1, \cdots, k_t^n]$. Note that k_t^j is obtained by the message from agent j following $\mathrm{AM}_{\mathrm{neg},k}^i(m_t^j)$ for $j \neq i$, and k_t^i is from $\mathrm{AM}_{\mathrm{neg},k}^i(o_t^i)$. Besides, it is scaled by $1/\sqrt{d_k}$ followed by a softmax to obtain attention weights α for each value vector:

$$\alpha_i = \mathrm{softmax}\left[\frac{{q_t^i}^T k_t^1}{\sqrt{d_k}} \cdots \underbrace{\frac{{q_t^i}^T k_t^j}{\sqrt{d_k}}}_{\alpha_{ij}} \cdots \frac{{q_t^i}^T k_t^n}{\sqrt{d_k}}\right] \tag{2}$$

The output of attention module is defined as: $c_t^i = \sum_{j=1}^n \alpha_{ij} v_t^j$, where v_t^j is obtained from messages or agent own observation following $\mathrm{AM}_{\mathrm{neg},v}^i(\cdot)$.

As for AM in the policy and critic network denoted as $\mathrm{AM}_{\mathrm{lac}}$, agent i gets additional messages from upper-level agent in launching phase. The message of upper-level and lower-level agent can be expanded as $m_t^{upper} = [o_t^{upper}, a_t^{upper}]$ and $m_t^{lower} = [o_t^{lower}, 0]$, respectively. In addition, the query depends on agent own observation o_t^i, but keys and values are only from messages of other agents.

3.4 TRAINING

The training of SeqComm is an extension of PPO. The critic and policy network are parameterized by θ_v, θ_π and takes as input the agent own observation and received messages. Besides, attention module $\mathrm{AM}_{\mathrm{lac}}$ is parameterized by θ_l and takes as input the agent own observation, the messages (observations of other agents) in negotiation phase and messages (the actions of upper-level agents) in launching phase. Let $\mathcal{D} = \{\tau_k\}_{k=1}^{K}$ be a set of trajectories by running policy in the environment. Note that we drop time t in the following notations for simplicity.

The value function is fitted by regression on mean-squared error:

$$\mathcal{L}(\theta_v) = \frac{1}{KT} \sum_{\tau \in \mathcal{D}} \sum_{t=0}^{T} \left\| V(o, \mathrm{AM}_{\mathrm{lac}}^{\theta_l}(\boldsymbol{o}, \boldsymbol{a}^{upper})) - \hat{R} \right\|_2^2 \tag{3}$$

where $\hat{R}$ is the discount rewards-to-go.

We update the policy by maximizing the PPO-Clip objective:

$$\mathcal{L}(\theta_\pi) = \frac{1}{KT} \sum_{\tau \in \mathcal{D}} \sum_{t=0}^{T} \min\left(\frac{\pi(a|o, \mathrm{AM}_{\mathrm{lac}}^{\theta_l}(\boldsymbol{o}, \boldsymbol{a}^{upper}))}{\pi_{old}(a|o, \mathrm{AM}_{\mathrm{lac}}^{\theta_l}(\boldsymbol{o}, \boldsymbol{a}^{upper}))} A_{\pi_{old}}, g(\epsilon, A_{\pi_{old}})\right) \tag{4}$$

where $g(\epsilon, A) = \begin{cases} (1+\epsilon)A & A \geq 0 \\ (1-\epsilon)A & A \leq 0 \end{cases}$, and $A_{\pi_{old}}(\boldsymbol{o}, \boldsymbol{a}^{upper}, a)$ is computed using the GAE method. Note that θ_l is learned via equation 3 and equation 4 by backpropagation.

The world model $\mathcal{M}$ is parameterized by θ_b is trained as a regression model using the training data set $\mathcal{S}$, which is obtained by running any pre-trained policy (or collected during policy learning). It updates with the loss:

$$\mathcal{L}(\theta_b) = \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{o}, \boldsymbol{a}, \boldsymbol{o}', r \in \mathcal{S}} \left\| (\boldsymbol{o}', r) - \mathcal{M}(\boldsymbol{o}, \boldsymbol{a}) \right\|_2^2. \tag{5}$$

4 EXPERIMENTS

We evaluate SeqComm in four cooperative tasks: predator prey (PP), cooperative navigation (CN), and keep away (KA) in multi-agent particle environment (MPE) (Lowe et al., 2017), and one map in StarCraft multi-agent challenge (SMAC) (Samvelyan et al., 2019).

We compare SeqComm against the following methods.

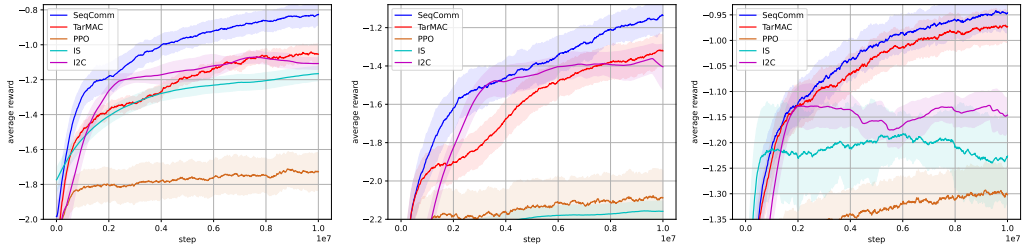
- IS (Kim et al., 2021), where agents communicate predicted future trajectories.
- TarMAC (Das et al., 2019), where agents use attention model to focus more on important incoming messages.
- I2C (Ding et al., 2020), where agents infer one-to-one communication to reduce message redundancy.
- PPO (Schulman et al., 2017), the base algorithm of SeqComm, communication-free.

In the experiments, SeqComm and baselines are parameter-sharing for fast convergence (Gupta et al., 2017; Terry et al., 2020). Moreover, to ensure the comparison is fair, their basic hyperparameters are the same and their sizes of network weights are also similar. Please refer to Appendix for the hyperparameter settings. All results are presented in terms of mean and standard deviation of five runs with different random seeds.

4.1 MPE

4.1.1 SETTINGS

Predator Prey. 5 predators (agents) try to capture 3 preys, all with random initial locations. Each predator observes the relative positions of three nearest preys, and is allowed to communicate with all other predators. Preys have a pre-defined area of activity and move in the opposite direction of the



(a) PP (b) CN (c) KA

Figure 3: Learning curves in terms of mean reward of SeqComm and baselines in PP, CN, and KA. We run $1e^7$ steps in all three MPE tasks.

closest predator. Moreover, preys move faster than predators, and thus it is impossible for a predator to capture a prey alone. Preys stop moving if being captured. The team reward of predators is the sum of negative distances of all the preys to their closest predators. Predators are also penalized by -0.5 for colliding with other predators. The length of each episode is 20 timesteps.

Cooperative Navigation. 5 agents try to occupy 5 landmarks, all with random initial locations. Each agent only observes the relative positions of three nearest landmarks and cannot perceive any other agents. This makes the communication more important. The team reward is based on the proximity of agents to landmarks, which is the sum of negative distances of all landmarks to their closest agents. The collision penalty is also set to -0.5. The length of each episode is 30 timesteps.

Keep Away. 3 attackers (agents) try to occupy 3 landmark, however, there are 3 defenders to push them away. The locations of landmarks are fixed and the positions for both attackers and defenders are randomly initialized. Each attacker observes the relative positions of three defenders. Defenders move in the same direction of the closest attacker but slower than attackers. Besides, collision is allowed between these two kinds so that defenders can push attackers away from landmarks. Therefore, attackers cannot hold on one landmark for a long time. The team reward of attackers is the sum of negative distances of all landmarks to their closest attackers. In addition, attackers get a penalty -0.5 for colliding with defenders and other attackers. The length of each episode is 20 timesteps.

Note that the size of agents is set to be larger than original settings in all three tasks so that collisions occur more easily, following the settings in (Kim et al., 2021).

4.1.2 Performance

Figure 3 shows the learning curves of all the methods in terms of mean reward averaged over timesteps in PP, CN, and KA. We can see that SeqComm converges to the highest mean reward compared with all baselines. The results demonstrate the superiority of SeqComm. In more detail, apart from communication-free PPO, IS performs the worst since it may access to inaccurate predicted information. The significant improvement over I2C and TarMAC shows the effectiveness of SeqComm since SeqComm allows agents to get more valuable accurate actions information. Note that the difference between SeqComm and other baselines narrows down in KA. This is because KA has less agents than previous tasks which makes setting easier, even though this setting requires more sophisticated coordination strategies, avoiding choosing the same landmark at the beginning as CN and keeping adjusting targets throughout the whole process as PP.

Figure 4 (upper panel from a to e) shows the priority order of decision making determined by SeqComm in PP. Agent 2 that is far away from others preys and predators is chosen to be the first-mover. If agents want to encircle and capture the preys, agents (*e.g.*, agent 2 and 5) that on the periphery of the encircling circle should hold upper-level position since they are able to decide how to narrow the encirclement. In addition, agent 3 makes decision prior to agent 5 so that collision can be avoided after agent 5 obtains the intention of agent 3.

For CN, as illustrated in Figure 4 (lower panel from a to e), agent 2 is far away from all the landmark and all other agents are in a better position to occupy landmark. Therefore, agents 2 is chosen to be the first-mover, which is similar with the phenomenon observed in PP. Once it has determined the target to occupy, other agent (agent 5 and 3) can adjust their actions accordingly and avoid conflict

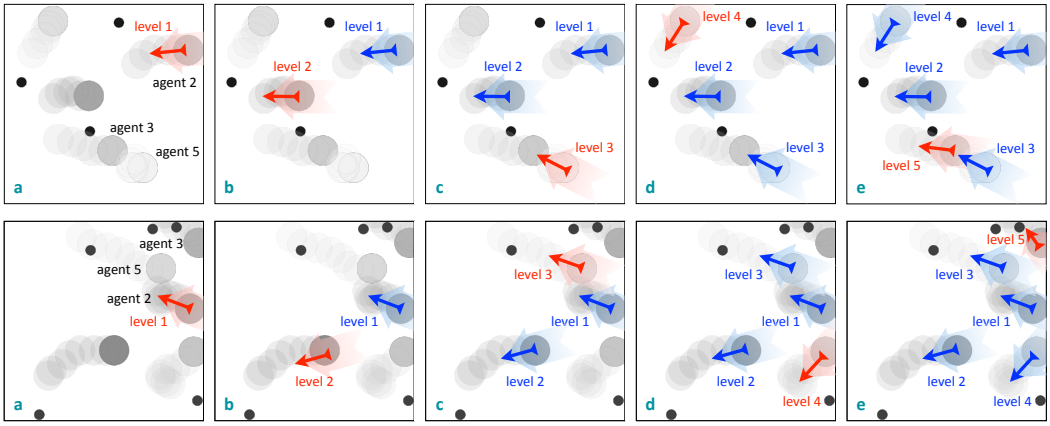


Figure 4: Illustration of learned priority of decision making in PP (*upper panel*) and CN (*lower panel*). Preys (landmarks) are viewed in black and predators (agents) are viewed in grey in PP (CN). From a to e, shown is the priority order. The smaller the level index, the higher priority of decision-making is.

of goals. Otherwise, if agent 5 makes decision first and chooses to occupy the closest landmark, then agent 2 has to approach to a further landmark which would take more steps.

4.2 SMAC

To verify the effectiveness of SeqComm in high-dimensional tasks, we evaluate SeqComm against the baselines on SMAC with one customized map, 3s_vs_4z, where we have made some minor changes to the observation part of agents to make it more difficult. Specifically, the sight range of agents is reduced from 9 to 3, and agents cannot perceive any information about their allies even if they are within the sight range. The rest settings remain the same as the default.

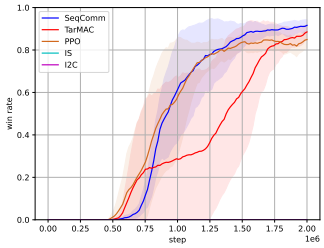


Figure 5: Learning curves in terms of win rate in 3s_vs_4z.

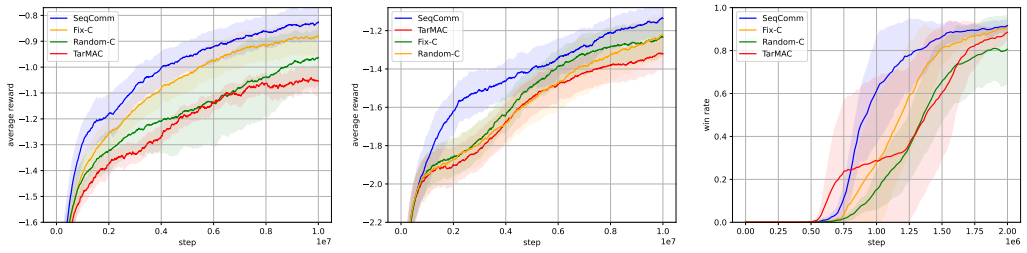
Table 1: Win rates with 100 test trails.

	3s_vs_4s
SeqComm	**0.91** ±0.04
TarMAC	0.87 ±0.05
PPO	0.84 ±0.02
IS	0
I2C	0

The learning curves of SeqComm and baselines in terms of win rate are illustrated in Figure 5, and their win rates are summarized in Table 1. IS and I2C fail in this task, because these two methods are built on MADDPG. However, MADDPG cannot work well in SMAC, especially when we reduce the sight range of agents, which is also supported by other studies (Papoudakis et al., 2021). SeqComm and TarMAC converge to better performance than PPO, which demonstrates the benefit of communication. Moreover, SeqComm outperforms TarMAC, which again verifies the gain of explicit action coordination.

4.3 Ablation Studies

Since SeqComm determines the priority of decision-making for all agents, we also compare it with two ablation baselines with only difference in the priority of decision-making: the priority of decision-making is fixed throughout one episode, denoted as Fix-C, and the priority of decision-making is determined randomly at each timestep, denoted as Random-C.



(a) PP (b) CN (c) 3s_vs_4z

Figure 6: Ablation studies on the priority of decision-making in PP, CN and 3s_vs_4z. Fix-C: the priority of decision-making is fixed throughout one episode. Random-C: the priority of decision-making is determined randomly.

As depicted in Figure 6, SeqComm achieves higher reward than Fix-C and Random-C in PP and CN. Moreover, SeqComm converges faster than the baselines even though SeqComm and Fix-C have achieved the similar win rate in 3s_vs_4z. The results demonstrate that the importance of the priority of decision-making in SE setting and it is necessary to continuously adjust it during one episode, and that SeqComm could provide the proper priority order of decision-making. Besides, Fix-C and Random-C show better performance than TarMAC in PP and CN. This result accords with the hypothesis that SE is likely to be Pareto superior to the average NE in games with high cooperation level (Zhang et al., 2020).

4.4 Generalization

Generalization to different number of agents has always been a key problem in MARL. For most algorithms in communication, once the model is trained in one scenario, it is unlikely for agents to maintain relatively competitive performance in other scenarios with different number of agents. However, as we employ attention module to process communicated messages so that agents can handle messages of different length. In addition, the module used to determine the priority of decision-making is also not restricted by the number of agents. Thus, we investigate whether SeqComm generalizes well to different number of agents in CN and PP.

Table 2: Mean reward in different tasks, averaged over timesteps, with 200 test trials.

	Fix-C	SeqComm
3-agent in CN	-0.83 ± 0.17	$\mathbf{-0.76} \pm 0.08$
7-agent in CN	-1.79 ± 0.15	$\mathbf{-1.57} \pm 0.10$
7-agent in PP	-1.89 ± 0.45	$\mathbf{-1.31} \pm 0.60$

For both tasks, SeqComm is trained on 5-agent settings. Then, we test SeqComm in 3-agent and 7-agent settings of CN and 7-agent setting of PP. We use Fix-C trained directly on these test tasks to illustrate the performance of SeqComm. Note that the quantity of both landmarks and preys is adjusted according to the number of agents in CN and PP. The test results are shown in Table 2. SeqComm exhibits the superiority in CN and PP, demonstrating that SeqComm may have a good generalization to the number of agents. A thorough study of the generalization of SeqComm is left to future work.

5 Conclusions

We have proposed SeqComm that enables agents well and explicitly coordinate with each other. SeqComm from an asymmetric perspective allows agents to make decision sequentially. Two-phase communication scheme has been adopted for determining the priority of decision-making and transferring messages accordingly. Empirically, it is demonstrated that SeqComm outperform baselines in a variety of cooperative multi-agent scenarios.

REFERENCES

Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning (ICML)*, 2019.

Ziluo Ding, Tiejun Huang, and Zongqing Lu. Learning individually inferred communication for multi-agent cooperation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Yali Du, Yifan Zhao, Meng Fang, Jun Wang, Gangyan Xu, and Haifeng Zhang. Learning predictive communication by imagination in networked system control, 2021.

Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Amy Greenwald, Keith Hall, and Roberto Serrano. Correlated q-learning. In *A comprehensive survey of multiagent reinforcement learning*, 2003.

Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2017.

Yedid Hoshen. Vain: Attentional multi-agent predictive modeling. *Advances in Neural Information Processing Systems*, 30:2701–2711, 2017.

Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. Graph convolutional reinforcement learning. In *International Conference on Learning Representation (ICLR)*, 2020.

Daewoo Kim, Sangwoo Moon, David Hostallero, Wan Ju Kang, Taeyoung Lee, Kyunghwan Son, and Yung Yi. Learning to schedule communication in multi-agent reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2019.

Woojun Kim, Jongeui Park, and Youngchul Sung. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations (ICLR)*, 2021.

Ville Könönen. Asymmetric multiagent reinforcement learning. *Web Intelligence and Agent Systems: An international journal*, 2(2):105–121, 2004.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Hang Ma, Daniel Harabor, Peter J Stuckey, Jiaoyang Li, and Sven Koenig. Searching with consistent prioritization for multi-agent path finding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks, 2021.

Peng Peng, Ying Wen, Yaodong Yang, Quan Yuan, Zhenkun Tang, Haitao Long, and Jun Wang. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*, 2017.

Arnu Pretorius, Scott Cameron, Andries Petrus Smit, Elan van Biljon, Lawrence Francis, Femi Azeez, Alexandre Laterre, and Karim Beguir. Learning to communicate through imagination with model-based deep multi-agent reinforcement learning, 2021.

Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, SM Ali Eslami, and Matthew Botvinick. Machine theory of mind. In *International Conference on Machine Learning (ICML)*, 2018.

Roberta Raileanu, Emily Denton, Arthur Szlam, and Rob Fergus. Modeling others using oneself in multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 4257–4266. PMLR, 2018.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Individualized controlled continuous communication model for multiagent cooperative and competitive tasks. In *International Conference on Learning Representations (ICLR)*, 2019.

Eric Sodomka, Elizabeth Hilliard, Michael Littman, and Amy Greenwald. Coco-q: Learning in stochastic games with side payments. In *International Conference on Machine Learning (ICML)*, 2013.

Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Justin K. Terry, Nathaniel Grammel, Ananth Hari, Luis Santos, Benjamin Black, and Dinesh Manocha. Parameter sharing is surprisingly useful for multi-agent deep reinforcement learning. *arXiv preprint arXiv:2005.13625*, 2020.

Jur P Van Den Berg and Mark H Overmars. Prioritized motion planning for multiple robots. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 430–435. IEEE, 2005.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems (NeurIPS)*, 2017.

Heinrich Von Stackelberg. *Market structure and equilibrium*. Springer Science & Business Media, 2010.

Bingyu Xu, Yaowei Wang, Zhaozhi Wang, Huizhu Jia, and Zongqing Lu. Hierarchically and cooperatively learning traffic signal control. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Yaodong Yang, Jianye Hao, Mingyang Sun, Zan Wang, Changjie Fan, and Goran Strbac. Recurrent deep multiagent q-learning for autonomous brokers in smart grid. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2018.

Haifeng Zhang, Weizhe Chen, Zeren Huang, Minne Li, Yaodong Yang, Weinan Zhang, and Jun Wang. Bi-level actor-critic for multi-agent coordination. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

Sai Qian Zhang, Qi Zhang, and Jieyu Lin. Efficient communication in multi-agent reinforcement learning via variance based control. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Ming Zhou, Jun Luo, Julian Villela, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, Aurora Chongxi Huang, Ying Wen, Kimia Hassanzadeh, Daniel Graves, Dong Chen, Zhengbang Zhu, Nhat M. Nguyen, Mohamed Elsayed, Kun Shao, Sanjeevan Ahilan, Baokuan Zhang, Jiannan Wu, Zhengang Fu, Kasra Rezaee, Peyman Yadmellat, Mohsen Rohani, Nicolas Perez Nieves, Yihan Ni, Seyedershad Banijamali, Alexander Imani Cowen-Rivers, Zheng Tian, Daniel Palenicek, Haitham Bou-Ammar, Hongbo Zhang, Wulong Liu, Jianye Hao, and Jun Wang. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. In *Conference on Robot Learning (CoRL)*, 2020.

A EXPERIMENTAL SETTINGS

In cooperative navigation, there are 5 agents and the size of each is 0.15. They need to occupy 5 landmarks with the size of 0.05. The acceleration of agents is 7. In predator prey, the number of predators (agents) and prey is set to 5 and 3, respectively, and their sizes are 0.15 and 0.05. The acceleration is 5 for predators and 7 for prey. In keep away, the number of attackers (agents) and defenders is set to 3, and their sizes are respectively 0.15 and 0.05. Besides, the acceleration is 6 for attackers and 4 for defenders. The three landmarks are located at $(0.00, 0.30)$, $(0.25, -0.15)$, and $(-0.25, -0.15)$. Note that each agent is allowed to communicate with all other agents in all three tasks. The team reward is similar across tasks. At a timestep t, it can be written as $r^t_{\text{team}} = -\sum_{i=1}^{n} d^t_i + C^t r_{\text{collision}}$, where d^t_i is the distance of landmark/prey i to its nearest agent/predator, C^t is the number of collisions (when the distance between two agents is less than the sum of their sizes) occurred at timestep t, and $r_{\text{collision}} = -1$. In addition, agents act discretely and have 5 actions (stay and move up, down, left, right).

B IMPLEMENTATION DETAILS

Our models, including SeqComm, Fix-C, and Random-C are trained based on PPO. The critic and policy network are realized by two fully connected layers. As for the attention module, key, query, and value have one fully connected layer each. The size of hidden layers is 100. Tanh functions are used as nonlinearity. For I2C and PPO, we use their official code with default settings of basic hyperparameters and networks. As there is no released code of IS and TarMAC, we implement IS and TarMAC by ourselves, following the instructions mentioned in the original papers (Kim et al., 2021; Das et al., 2019).

For the world model, observations and actions are firstly encoded by a fully connected layer. The output size for the observation encoder is 48, and the output size for the action encoder is 16. Then the outputs of the encoder will be passed into the attention module with the same structure aforementioned. Finally, we use a fully connected layer to decode. In these layers, Tanh is used as the nonlinearity.

Table 3 summarize the hyperparameters used by SeqComm and the baselines in the experiments.

Table 3: Hyperparameters for predator prey, cooperative navigation, keep away, and SMAC

Hyperparameter	SeqComm	Random-C	Fix-C	TarMAC	PPO	I2C	IS
discount (γ)	0.95,0.95,0.95,0.99						
batch size	–	–	–	–	–	800	1024
buffer capacity	–	–	–	–	–	$1e^6$	
number of processes	16,16,16,8					–	–
learning rate	$1.5e^{-5}$, $1e^{-5}$, $4e^{-5}$,$5e^{-5}$					$1e^{-2}$, $1e^{-3}$, $1e^{-3}$,$5e^{-5}$	$1e^{-2}$
H	10,10,20,20	–	–	–	–	–	–
F	$2, 2, 1, 1$	–	–	–	–	–	–