# Coordinated Proximal Policy Optimization


## Abstract

We present Coordinated Proximal Policy Optimization (CoPPO), an algorithm that extends the original Proximal Policy Optimization (PPO) to the multi-agent setting. The key idea lies in the coordinated adaptation of step size during the policy update process among multiple agents. We prove the monotonicity of policy improvement when optimizing a theoretically-grounded joint objective, and derive a simplified optimization objective based on a set of approximations. We then interpret that such an objective in CoPPO can achieve dynamic credit assignment among agents, thereby alleviating the high variance issue during the concurrent update of agent policies. Finally, we demonstrate that CoPPO outperforms several strong baselines and is competitive with the latest multi-agent PPO method (i.e. MAPPO) under typical multi-agent settings, including cooperative matrix games and the StarCraft II micromanagement tasks.


## 1 Introduction

Cooperative Multi-Agent Reinforcement Learning (CoMARL) shows great promise for solving various real-world tasks, such as traffic light control (Wu et al., 2020), sensor network management (Sharma and Chauhan, 2020) and autonomous vehicle coordination (Yu et al., 2019). In such applications, a team of agents aim to maximize a joint expected utility through a single global reward. Since multiple agents coexist in a common environment and learn and adapt their behaviour concurrently, the arising non-stationary issue makes it difficult to design an efficient learning method (Hernandez-Leal et al., 2017; Papoudakis et al., 2019). Recently, a number of CoMARL methods based on Centralized Training Decentralized Execution (CTDE) (Foerster et al., 2016) have been proposed, including policy-based (Lowe et al., 2017; Foerster et al., 2018; Wang et al., 2020; Yu et al., 2021) and value-based methods (Sunehag et al., 2018; Rashid et al., 2018; Son et al., 2019; Mahajan et al., 2019). While generally having more stable convergence properties (Gupta et al., 2017; Song et al., 2019; Wang et al., 2020) and being naturally suitable for problems with stochastic policies (Deisenroth et al., 2013; Su et al., 2021), policy-based methods still receive less attention from the community and generally possess inferior performance against value-based methods, as evidenced in the StarCraft II benchmark (Samvelyan et al., 2019).

The performance discrepancy between policy-based and value-based methods can be largely attributed to the inadequate utilization of the centralized training procedure in the CTDE paradigm. Unlike value-based methods that directly optimize the policy via centralized training of value functions using extra global information, policy-based methods only utilize centralized value functions for state/action evaluation such that the policy can be updated to increase the likelihood of generating higher values (Sutton et al., 2000). In other words, there is an update lag between intermediate value functions and the final policy in policy-based methods, and merely coordinating over value functions is insufficient to guarantee satisfactory performance (Grondman et al., 2012; Fujimoto et al., 2018).

To this end, we propose the *Coordinated Proximal Policy Optimization* (CoPPO) algorithm, a multi-agent extension of PPO (Schulman et al., 2017), to directly coordinate over the agents' policies by dynamically adapting the step sizes during the agents' policy update processes. We first prove a relationship between a lower bound of joint policy performance and the update of policies. Based on this relationship, a monotonic joint policy improvement can be achieved through optimizing an ideal

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

objective. To improve scalability and credit assignment, and to cope with the potential high variance due to non-stationarity, a series of transformations and approximations are then conducted to derive an implementable optimization objective in the final CoPPO algorithm. While originally aiming at monotonic joint policy improvement, CoPPO ultimately realizes a direct coordination over the policies at the level of each agent's policy update step size. Concretely, by taking other agents' policy update into consideration, CoPPO is able to achieve dynamic credit assignment that helps to indicate a proper update step size to each agent during the optimization procedure. In the empirical study, an extremely hard version of the *penalty game* (Claus and Boutilier, 1998) is used to verify the efficacy and interpretability of CoPPO. In addition, the evaluation on the StarCraft II micromanagement benchmark further demonstrates the superior performance of CoPPO against several strong baselines.

The paper is organized as follows: Section 2 provides a background introduction. Section 3 introduces the derivation process of CoPPO. Section 4 presents the experimental studies, and Section 5 reviews some related works. Finally, Section 6 concludes the paper.

## 2 Background

We model the fully cooperative MARL problem as a Dec-POMDP (Oliehoek and Amato, 2016) which is defined by a tuple $G = \langle N, S, \Omega, O, A, R, P, \gamma \rangle$. $N$ is the number of agents and $S$ is the set of true states of the environment. Agent $i \in \{1, \dots, N\}$ obtains its partial observation $o^i \in \Omega$ according to the observation function $O(s, i)$, where $s \in S$. Each agent has an action-observation history $\tau^i \in T \equiv (\Omega \times A)^*$. At each timestep, agent $i$ chooses an action $a^i \in A$ according to its policy $\pi(a^i|\tau^i)$, and we use $\boldsymbol{a}$ to denote the joint action $\{a^1, \dots, a^N\}$. The environment then returns the reward signal $R(s, \boldsymbol{a})$ that is shared by all agents, and shifts to the next state according to the transition function $P(s'|s, \boldsymbol{a})$. The joint action-value function induced by a joint policy $\boldsymbol{\pi}$ is defined as: $Q^{\boldsymbol{\pi}}(s_t, \boldsymbol{a}_t) = \mathbb{E}_{s_{t+1:\infty}, \boldsymbol{a}_{t+1:\infty}}[\sum_{t'=0}^{\infty} \gamma^{t'} R_{t+t'} | s_t, \boldsymbol{a}_t]$, where $\gamma \in [0, 1)$ is the discounted factor. We denote the joint action of agents other than agent $i$ as $\boldsymbol{a}^{-i}$, and $\boldsymbol{\pi}^{-i}$, $\boldsymbol{\tau}^{-i}$ follow a similar convention. Joint policy $\boldsymbol{\pi}$ can be parameterized by $\theta = \{\theta^1, \dots, \theta^N\}$, where $\theta^i$ is the parameter set of agent $i$'s policy. Our problem setting follows the CTDE paradigm (Foerster et al., 2016), in which each agent executes its policy conditioned only on the partially observable information, but the policies can be trained centrally by using extra global information.

**Value-based MARL** In CTDE value-based methods such as (Sunehag et al., 2018; Rashid et al., 2018; Son et al., 2019; Mahajan et al., 2019), an agent selects its action by performing an $\arg\max$ operation over the local action-value function, i.e. $\pi^i(\tau^i) = \arg\max_{a^i} Q^i(\tau^i, a^i)$. Without loss in generality, the update rule for value-based methods can be formulated as follows:

$$\Delta\theta^i \propto \mathbb{E}_{\boldsymbol{\pi}} \left[ \left( R(s, \boldsymbol{a}) + \max_{\boldsymbol{a}'} Q^{tot}(s, \boldsymbol{a}') - Q^{tot}(s, \boldsymbol{a}) \right) \frac{\partial Q^{tot}}{\partial Q^i} \nabla_{\theta^i} Q^i(\tau^i, a^i) \right], \tag{1}$$

where $\theta^i$ represents the parameter of $Q^i$, and $Q^{tot}$ is the global action-value function. The centralized training procedure enables value-based methods to factorize $Q^{tot}$ into some local action-values. Eq. (1) is essentially Q-learning if rewriting $\frac{\partial Q^{tot}}{\partial Q^i} \nabla_{\theta^i} Q^i(\tau^i, a^i)$ as $\nabla_{\theta^i} Q^{tot}$. The partial derivative is actually a credit assignment term that projects the step size of $Q^{tot}$ to that of $Q^i$ (Wang et al., 2020).

**Policy-based MARL** In CTDE policy-based methods such as (Foerster et al., 2018; Lowe et al., 2017; Wang et al., 2020; Yu et al., 2021), an agent selects its action from an explicit policy $\pi^i(a^i|\tau^i)$. The vanilla multi-agent policy gradient algorithm updates the policy using the following formula:

$$\Delta\theta^i \propto \mathbb{E}_{\boldsymbol{\pi}} \left[ Q^{tot}_{\boldsymbol{\pi}}(s, \boldsymbol{a}) \nabla_{\theta^i} \pi^i(a^i|\tau^i) \right]. \tag{2}$$

In order to reduce the variance and address the credit assignment issue, COMA (Foerster et al., 2018) replace $Q^{tot}$ with the counterfactual advantage: $A^{\boldsymbol{\pi}}(s, \boldsymbol{a}) = Q^{\boldsymbol{\pi}}(s, (a^i, \boldsymbol{a}^{-i})) - \mathbb{E}_{\hat{a}^i \sim \pi^i}[Q^{\boldsymbol{\pi}}(s, (\hat{a}^i, \boldsymbol{a}^{-i}))]$. This implies that fixing the actions of other agents (i.e. $\boldsymbol{a}^{-i}$), an agent will evaluate the action it has actually taken (i.e. $a^i$) by comparing with the average effect of other actions it may have taken.

## 3 Coordinated Proximal Policy Optimization

In policy-based methods, properly limiting the policy update step size is proven to be effective in single-agent settings (Schulman et al., 2015a, 2017). In cases when there are multiple policies, it is also crucial for each agent to take other agents' update into account when adjusting its own step size. Driven by this insight, we propose the CoPPO algorithm to adaptively adjust the step sizes during the update of the policies of multiple agents.

### 3.1 Monotonic Joint Policy Improvement

The performance of joint policy $\boldsymbol{\pi}$ is defined as: $J(\boldsymbol{\pi}) \doteq \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}, s\sim\rho^{\boldsymbol{\pi}}}\left[\sum_{t=0}^{\infty}\gamma^t R_{t+1}(s,\boldsymbol{a})\right]$, where $\rho^{\boldsymbol{\pi}}$ is the unnormalized discounted visitation frequencies when the joint actions are chosen from $\boldsymbol{\pi}$. Then the difference between the performance of two joint policies, say $\boldsymbol{\pi}$ and $\tilde{\boldsymbol{\pi}}$, can be expressed as the accumulation of the global advantage over timesteps (see Appendix A.1 for proof):

$$J(\tilde{\boldsymbol{\pi}}) - J(\boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}}, s\sim\rho^{\tilde{\boldsymbol{\pi}}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right], \tag{3}$$

where $A^{\boldsymbol{\pi}}(s,\boldsymbol{a}) = Q^{\boldsymbol{\pi}}(s,\boldsymbol{a}) - V^{\boldsymbol{\pi}}(s)$ is the joint advantage function. This equation indicates that if the joint policy $\boldsymbol{\pi}$ is updated to $\tilde{\boldsymbol{\pi}}$, then the performance will improve when the update increases the probability of taking "good" joint actions so that $\sum_{\boldsymbol{a}}\tilde{\boldsymbol{\pi}}(\boldsymbol{a}|s)A^{\boldsymbol{\pi}}(s,\boldsymbol{a}) > 0$ for every $s$. Modeling the dependency of $\rho^{\tilde{\boldsymbol{\pi}}}$ on $\tilde{\boldsymbol{\pi}}$ involves the complex dynamics of the environment, so we extend the approach proposed in (Kakade and Langford, 2002) to derive an approximation of $J(\tilde{\boldsymbol{\pi}})$, denoted as $\tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})$:

$$\tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}}) \doteq J(\boldsymbol{\pi}) + \mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}}, s\sim\rho^{\boldsymbol{\pi}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right]. \tag{4}$$

Note that if the policy is differentiable, then $\tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})$ matches $J(\tilde{\boldsymbol{\pi}})$ to first order (see Appendix A.2 for proof). Quantitatively, we measure the difference between two joint policies using the *maximum total variation divergence* (Schulman et al., 2015a), which is defined by: $D_{TV}^{\max}[\pi\|\tilde{\pi}] \doteq \max_s D_{TV}[\pi(\cdot|s)\|\tilde{\pi}(\cdot|s)]$, where $D_{TV}[\pi(\cdot|s)\|\tilde{\pi}(\cdot|s)] = \frac{1}{2}\int_{\mathcal{A}}|\pi(a|s) - \tilde{\pi}(a|s)|da$ (the definition for the discrete case is simply replacing the integral with a summation, and our results remain valid in such case). Using the above notations, we can derive the following theorem:

**Theorem 1.** *Let $\epsilon = \max_{s,\boldsymbol{a}}|A^{\boldsymbol{\pi}}(s,\boldsymbol{a})|$, $\alpha_i = \sqrt{\frac{1}{2}D_{TV}^{\max}[\pi^i\|\tilde{\pi}^i]}$, $1 \le i \le N$, and $N$ be the total number of agents, then the error of the approximation in Eq. (4) can be explicitly bounded as follows:*

$$\left|J(\tilde{\boldsymbol{\pi}}) - \tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})\right| \le 4\epsilon\left[\frac{1-\gamma\prod_{i=1}^{N}(1-\alpha_i)}{1-\gamma} - 1\right]. \tag{5}$$

*Proof.* See Appendix A.3. □

As shown above, the upper bound is influenced by $\alpha_i, N$ and $\epsilon$. By definition, we have $\alpha_i \le 1$ and $\epsilon \ge 0$, thus the upper bound will increase when $\alpha_i$ increases for any $i$, implying that it becomes harder to make precise approximation when any individual of the agents dramatically updates their policies. Also, the growth in the number of agents can raise the difficulty for approximation. As for $\epsilon$, from Eq. (5) we can roughly conclude that a larger advantage value can cause higher approximation error, and this is reasonable because $\tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})$ approximates $J(\tilde{\boldsymbol{\pi}})$ by approximating the expectation over $A^{\boldsymbol{\pi}}$. Transforming the inequality in Eq. (5) leads to $J(\tilde{\boldsymbol{\pi}}) \ge \tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}}) - 4\epsilon\left(\frac{1-\gamma\prod_{i=1}^{N}(1-\alpha_i)}{1-\gamma} - 1\right)$. Thus the joint policy can be iteratively updated by:

$$\boldsymbol{\pi}_{new} = \arg\max_{\tilde{\boldsymbol{\pi}}}\left[\tilde{J}_{\boldsymbol{\pi}_{old}}(\tilde{\boldsymbol{\pi}}) - 4\epsilon\left(\frac{1-\gamma\prod_{i=1}^{N}(1-\alpha_i)}{1-\gamma} - 1\right)\right]. \tag{6}$$

Eq. (6) involves a complete search in the joint observation space and action space for computing $\epsilon$ and $\alpha_i$, making it difficult to be applied to large-scale settings. In the next subsection, several transformations and approximations are employed to this objective to achieve better scalability.

### 3.2 The Final Algorithm

Notice that the complexity of optimizing the objective in Eq. (6) mainly lies in the second term, i.e. $4\epsilon\left(\frac{1-\gamma\prod_{i=1}^{N}(1-\alpha_i)}{1-\gamma}-1\right)$. While $\epsilon$ has nothing to do with $\tilde{\boldsymbol{\pi}}$, it suffices to control this second term only by limiting the variation divergence of agents' policies (i.e. $\alpha_i$), because it increases monotonically as $\alpha_i$ increases. Then the objective is transformed into $\tilde{J}_{\boldsymbol{\pi}_{old}}(\tilde{\boldsymbol{\pi}})$ that can be optimized subject to a trust region constraint: $\alpha_i \leq \delta, i = 1, \ldots, N$. For higher scalability, $\alpha_i$ can be replaced by the mean Kullback-Leibler Divergence between agent $i$'s two consecutive policies, i.e. $\mathbb{E}_{s\sim\rho^{\boldsymbol{\pi}}}\left[D_{KL}[\pi^i(\cdot|\tau^i)||\tilde{\pi}^i(\cdot|\tau^i)]\right]$.

As proposed in (Schulman et al., 2015a), solving the above trust region optimization problem requires repeated computation of Fisher-vector products for each update, which is computationally expensive in large-scale problems, especially when there are multiple constraints. In order to reduce the computational complexity and simplify the implementation, importance sampling can be used to incorporate the trust region constraints into the objective of $\tilde{J}_{\boldsymbol{\pi}_{old}}(\tilde{\boldsymbol{\pi}})$, resulting in the maximization of $\mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}_{old}}\left[\min\left(\boldsymbol{r}A^{\boldsymbol{\pi}_{old}}, \text{clip}\left(\boldsymbol{r}A^{\boldsymbol{\pi}_{old}}, 1-\epsilon, 1+\epsilon\right)\right)\right]$ w.r.t $\boldsymbol{\pi}$, where $\boldsymbol{r} = \frac{\boldsymbol{\pi}(\boldsymbol{a}|s)}{\boldsymbol{\pi}_{old}(\boldsymbol{a}|s)}$, and $A^{\boldsymbol{\pi}_{old}}(s,\boldsymbol{a})$ is denoted as $A^{\boldsymbol{\pi}_{old}}$ for brevity. The clip function prevents the joint probability ratio from going beyond $[1-\epsilon, 1+\epsilon]$, thus approximately limiting the variation divergence of the joint policy. Since the policies are independent during the fully decentralized execution, it is reasonable to assume that $\boldsymbol{\pi}(\boldsymbol{a}|\boldsymbol{\tau}) = \prod_{i=1}^{N}\pi^i(a^i|\tau^i)$. Based on this factorization, the following objective can be derived:

$$\underset{\theta^1,\ldots,\theta^N}{\text{maximize}} \;\; \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}_{old}}\left\{\min\left[\left(\prod_{j=1}^{N} r^j\right)A^{\boldsymbol{\pi}}, \text{clip}\left(\left(\prod_{j=1}^{N} r^j\right), 1-\epsilon, 1+\epsilon\right)A^{\boldsymbol{\pi}}\right]\right\}, \tag{7}$$

where $\theta^j$ is the parameter of agent $j$'s policy, and $r^j = \frac{\pi^j\left(a^j|\tau^j;\theta^j\right)}{\pi^j_{old}\left(a^j|\tau^j;\theta^j_{old}\right)}$. While $A^{\boldsymbol{\pi}}$ is defined as $Q^{\boldsymbol{\pi}}(s,\boldsymbol{a}) - V^{\boldsymbol{\pi}}(s)$, the respective contribution of each individual agent cannot be well distinguished. To enable credit assignment, the joint advantage function is decomposed to some local ones of the agents as: $A^{\boldsymbol{\pi}}(s,\boldsymbol{a}) = \sum_{i=1}^{N} c^i \cdot A^i(s,(a^i,\boldsymbol{a}^{-i}))$, where $A^i(s,(a^i,\boldsymbol{a}^{-i})) = Q^{\boldsymbol{\pi}}(s,(a^i,\boldsymbol{a}^{-i})) - \mathbb{E}_{\hat{a}^i}[Q^{\boldsymbol{\pi}}(s,(\hat{a}^i,\boldsymbol{a}^{-i}))]$ is the counterfactual advantage and $c^i$ is a non-negative weight.

During each update, multiple epochs of optimization are performed on this joint objective to improve sample efficiency. Due to the non-negative decomposition of $A^{\boldsymbol{\pi}}$, there is a monotonic relationship between the global optimum and the local optima, suggesting a transformation from the joint objective to local objectives (see Appendix A.4 for the proof). The optimization of Eq. (7) then can be transformed to maximizing each agent's own objective:

$$L^i(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}_{old}}\left\{\min\left[\left(\prod_{j\neq i} r^j\right)r^iA^i, \text{clip}\left(\left(\prod_{j\neq i} r^j\right)r^i, 1-\epsilon, 1+\epsilon\right)A^i\right]\right\}. \tag{8}$$

However, the ratio product in Eq. (8) raises a potential risk of high variance due to Proposition 1:

**Proposition 1.** *Assuming that the agents are fully independent during execution, then the following inequality holds:*

$$\text{Var}_{\boldsymbol{a}^{-i}\sim\boldsymbol{\pi}^{-i}_{old}}\left[\prod_{j\neq i} r^j\right] \geq \prod_{j\neq i}\text{Var}_{a^j\sim\pi^j_{old}}\left[r^j\right]. \tag{9}$$

*Proof.* See Appendix A.5. □

According to the inequality above, the variance of the product grows at least exponentially with the number of agents. Intuitively, the existence of other agents' policies introduces instability in the estimate of each agent's policy gradient. This may further cause suboptimatlity in individual policies due to the centralized-decentralized mismatch issue mentioned in (Wang et al., 2020). To be concrete, when $A^i > 0$, the external $\min$ operation in Eq. (8) can prevent the gradient of $L^i(\boldsymbol{\theta})$ from exploding when $\prod_{j\neq i} r^j$ is large, thus limiting the variance raised from other agents; but when $A^i < 0$, the gradient can grow rapidly in the negative direction, because $L^i(\boldsymbol{\theta}) = \mathbb{E}\left[\left(\prod_{j\neq i} r^j\right)r^iA^i\right]$ when

$\left(\prod_{j\neq i} r^j\right) r^i \geq 1+\epsilon$. Moreover, the learning procedure in Eq. (8) can cause a potential exploration issue, that is, different agents might be granted unequal opportunities to update their policies. Consider a scenario when the policies of some agents except agent $i$ are updated rapidly, and thus the product of these agents' ratios might already be close to the clipping threshold, then a small optimization step of agent $i$ will cause $\prod_{j=1}^{N} r^j$ to reach the threshold and thus being clipped. In this case, agent $i$ has no chance to update its policy, while other agents have updated their policies significantly, leading to unbalanced exploration among the agents. To address the above issues, we propose a double clipping trick to modify Eq. (8) as follows:

$$L^i(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}_{old}}\left\{\min\left[g(\boldsymbol{r}^{-i})r^iA^i, \text{clip}\left(g(\boldsymbol{r}^{-i})r^i, 1-\epsilon_1, 1+\epsilon_1\right)A^i\right]\right\}, \tag{10}$$

where $g(\boldsymbol{r}^{-i}) = \text{clip}\left(\prod_{j\neq i} r^j, 1-\epsilon_2, 1+\epsilon_2\right),\ \epsilon_2 < \epsilon_1$. In Eq. (8), the existence of $\prod_{j\neq i} r^j$ imposes an influence on the objective of agent $i$ through a weight of $\prod_{j\neq i} r^j$. Therefore, the clipping on $\prod_{j\neq i} r^j$ ensures that the influence from the update of other agents on agent $i$ is limited to $[1-\epsilon_2, 1+\epsilon_2]$, thus controlling the variance caused by other agents. Note that from the theoretical perspective, clipping separately on each individual probability ratio (i.e. $\prod_{j=1}^{N}\text{clip}(r^j,\cdot,\cdot)$) can also reduce the variance. Nevertheless, the empirical results show that clipping separately performs worse than clipping jointly. The detailed results and analysis for this comparison are presented in Appendix D.2.1. In addition, this trick also prevents the update step of each agent from being too small, because $r^i$ in Eq. (10) can at least increase to $\frac{1+\epsilon_1}{1+\epsilon_2}r^i$ or decrease to $\frac{1-\epsilon_1}{1-\epsilon_2}r^i$ before being clipped in each update.

In the next subsection, we will show that the presence of other agents' probability ratio also enables a dynamic credit assignment among the agents in order to promote coordination, and thus the inner clipping threshold (i.e. $\epsilon_2$) can actually function as a balance factor to trade off between facilitating coordination and reducing variance, which will be studied empirically in Section 4. A similar trick was proposed in (Ye et al., 2020a,b) to handle the variance induced by distributed training in the single-agent setting. Nonetheless, since multiple policies are updated in different directions in MARL, the inner clipping here is carried out on the ratio product of other agents instead of the entire ratio product, in order to distinguish the update of different agents. The overall CoPPO algorithm with the double clipping trick is shown in Appendix B.

### 3.3 Interpretation: Dynamic Credit Assignment

The COMA (Foerster et al., 2018) algorithm tries to address the credit assignment issue in CoMARL using the counterfactual advantage. Nevertheless, miscoordination and suboptimatlity can still arise since the credit assignment in COMA is conditioned on the fixed actions of other agents, but these actions are continuously changing and thus cannot precisely represent the actual policies. While CoPPO also makes use of the counterfactual advantage, the overall update of other agents is taken into account dynamically during the multiple epochs in each update. This process can adjust the advantage value in a coordinated manner and alleviate the miscoordination issue caused by the fixation of other agents' joint action. Note that the theoretical reasoning for CoPPO in Section 3.1 and 3.2 originally aims at monotonic joint policy improvement, yet the resulted objective ultimately realizes coordination among agents through a dynamic credit assignment among the agents in terms of coordinating over the step sizes of the agents' policies.

To illustrate the efficacy of this dynamic credit assignment, we make an analysis on the difference between CoPPO and MAPPO (Yu et al., 2021) which generalizes PPO to multi-agent settings simply by centralizing the value functions with an optimization objective of $\mathbb{E}_{\boldsymbol{\pi}_{old}}\left[\min\left[r_k^iA^i, \text{clip}\left(r_k^i, 1-\epsilon, 1+\epsilon\right)\right]\right]$, which is a lower bound of $\mathbb{E}_{\boldsymbol{\pi}_{old}}\left[r_k^iA^i\right]$ where $r_k^i$ represents the probability ratio of agent $i$ at the $k_{th}$ optimization epoch during each update. Denoting $\left(\prod_{j\neq i} r_k^j\right)A^i$ as $\tilde{A}_k^i$, the CoPPO objective then becomes approximately a lower bound of $\mathbb{E}_{\boldsymbol{\pi}_{old}}\left[r_k^i\tilde{A}_k^i\right]$. The discussion then can be simplified to analyzing the two lower bounds (see Appendix A.6 for the details of this simplification).

Depending on whether $A^i > 0$ and whether $\prod_{j\neq i} r_k^j > 1$, four different cases can be classified. For brevity, only two of them are discussed below while the rest are similar. The initial parameters of the two methods are assumed to be the same.

Case (1): $A^i > 0,\ \prod_{j\neq i} r_k^j > 1$. In this case $\left|\tilde{A}_k^i\right| > \left|A^i\right|$, thus $\left\|\tilde{A}_k^i \nabla r_k^i\right\| > \left\|A^i \nabla r_k^i\right\|$, indicating that CoPPO takes a larger update step towards increasing $\pi^i(a^i|\tau^i)$ than MAPPO does. Concretely, $A^i > 0$ means that $a^i$ is considered (by agent $i$) positive for the whole team when fixing $\boldsymbol{a}^{-i}$ and under similar observations. Meanwhile, $\prod_{j\neq i} r_k^j > 1$ implies that after this update epoch, $\boldsymbol{a}^{-i}$ are overall more likely to be performed by the other agents when encountering similar observations (see Appendix A.7 for the details). This makes fixing $\boldsymbol{a}^{-i}$ more reasonable when estimating the advantage of $a^i$, thus explaining CoPPO's confidence to take a larger update step.

Case (2): $A^i < 0,\ \prod_{j\neq i} r_k^j < 1$. Similarly, in this case $\left|\tilde{A}_k^i\right| < \left|A^i\right|$ and hence $\left\|\tilde{A}_k^i \nabla r_k^i\right\| < \left\|A^i \nabla r_k^i\right\|$, indicating that CoPPO takes a smaller update step to decrease $\pi^i(a^i|\tau^i)$ than MAPPO does. To be specific, $a^i$ is considered (by agent $i$) to have a negative effect on the whole team since $A^i < 0$, and $\prod_{j\neq i} r_k^j < 1$ suggests that after this optimization epoch, other agents are overall less likely to perform $\boldsymbol{a}^{-i}$ given similar observations. While the evaluation of $a^i$ is conditioned on $\boldsymbol{a}^{-i}$, it is reasonable for agent $i$ to rethink the effect of $a^i$ and slow down the update of decreasing the probability of taking $a^i$, thus giving more chance for this action to be evaluated.

It is worth noting that $\tilde{A}_k^i$ continues changing throughout the K epochs of update and yields dynamic adjustments in the step size, while $A^i$ will remain the same during each update. Therefore, $\tilde{A}_k^i$ can be interpreted as a dynamic modification of $A^i$ by taking other agents' update into consideration.

# 4 Experiments

In this section, we evaluate CoPPO on a modified matrix penalty game and the StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019). The matrix game results enable interpretative observations, while the evaluations on SMAC verify the efficacy of CoPPO in more complex domains.

## 4.1 Cooperative Matrix Penalty Game

The *penalty game* is a representative of problems with miscoordination penalties and multiple equilibria selection among optimal joint actions. It has been used as a challenging test bed for evaluating CoMARL algorithms (Claus and Boutilier, 1998; Spiros and Daniel, 2002). To further increase the difficulty of achieving coordination, we modify the two-player *penalty game* to four agents with nine actions for each agent. The agents will receive a team reward of 50 when they have played the same action, but be punished by -50 if any three agents have acted the same while the other does not. In all other cases, the reward is -40 for all the agents. The *penalty game* provides a verifying metaphor to show the importance of adaptive adjustment in the agent policies in order to achieve efficient coordinated behaviors. Thinking of the case when the agents have almost reached one of the optimal joint actions, yet at the current step they have received a miscoordination penalty due to the exploration of an arbitrary agent. Then smaller update steps for the three matching agents would benefit the coordinated learning process of the whole group, since agreement on this optimal joint action would be much easier to be reached than any other optimal joint actions. Therefore, adaptively coordinating over the agent policies and properly assigning credits among the agents are crucial for the agents to achieve efficient coordination in this kind of game.

We train CoPPO, COMA (Foerster et al., 2018), MAPPO (Yu et al., 2021) and DOP (Wang et al., 2020) for 10,000 timesteps, and the final results are averaged over 100 runs. The hyperparameters and other implementation details are described in Appendix C.1. From Fig. 1-upper left, we can see that CoPPO significantly outperforms other methods in terms of average rewards. Fig. 1-upper right presents an explanation of the result by showing the local advantages averaged among the three matching agents every time after receiving a miscoordination penalty. Each point on the horizontal axis represents a time step after a miscoordination penalty. While the times of penalties in a single run vary for different algorithms, we take the minimum times of 70 over all runs. The vertical axis represents $\frac{1}{3}\sum_{j\neq i} A^j$ for COMA, MAPPO and DOP, and represents the mean of $K$ epochs during one update, i.e., $\frac{1}{3}\sum_{j\neq i}\frac{1}{K}\sum_{k=1}^{K}\tilde{A}_k^j$, for CoPPO ($i$ indicates the unmatching agent). Note that CoPPO can obtain the smallest local advantages that are close to 0 compared to other methods, indicating the smallest step sizes for the three agents in the direction of changing the current action. Fig. 1-lower left shows the overall variation of the average advantages within $K$ optimization epochs after receiving a miscoordination penalty. We can see that as the number of epochs increases,

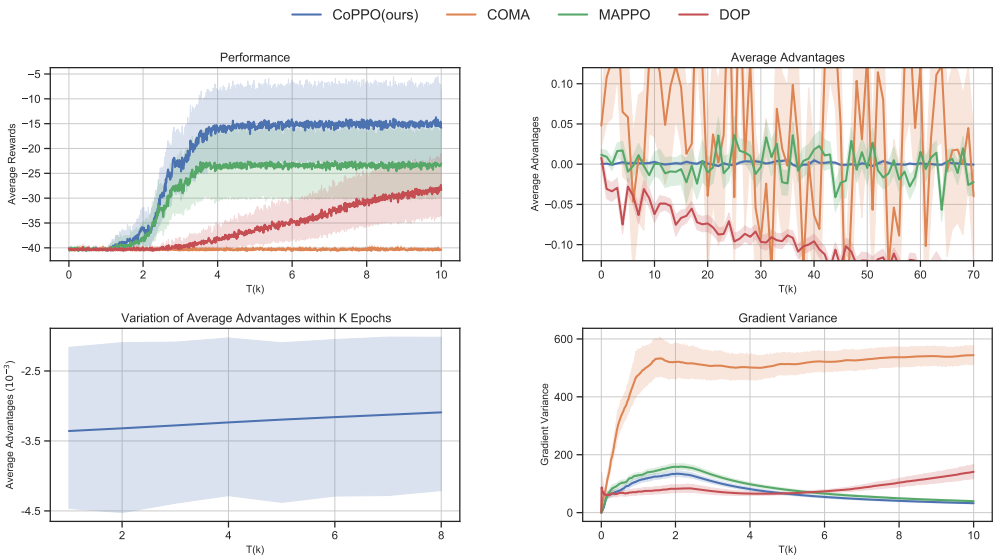


Figure 1: Upper left: average rewards; Upper right: average advantages after a penalty; Lower left: the variation of the average advantages within $K$ (here $K = 8$) optimization epochs every time after a penalty; Lower right: running policy gradient variance.

the absolute value of the average advantage of the three matching agents gradually decreases by considering the update of other agents. Since the absolute value actually determines the step size of update, a smaller value indicates a small adaptation in their current actions. This is consistent with what we have discussed in Section 3.3. Fig. 1-lower right further implies that through this dynamic process, the agents succeed in learning to coordinate their update steps carefully, yielding the smallest gradient variance among the four methods.

**Ablation study 1** Fig. 2 provides an ablation study of the double clipping trick. We can see that a proper intermediate inner clipping threshold improves the global performance, and the double clipping trick indeed reduces the variance of the policy gradient. In contrast to DOP, which achieves low gradient variance at the expense of lack of direct coordination over the policies, CoPPO can strike a nice balance between reducing variance and achieving coordination, by taking other agents' policy update into consideration. To make our results more convincing, experiments on more cooperative matrix games with different varieties are also conducted in Appendix D.1.

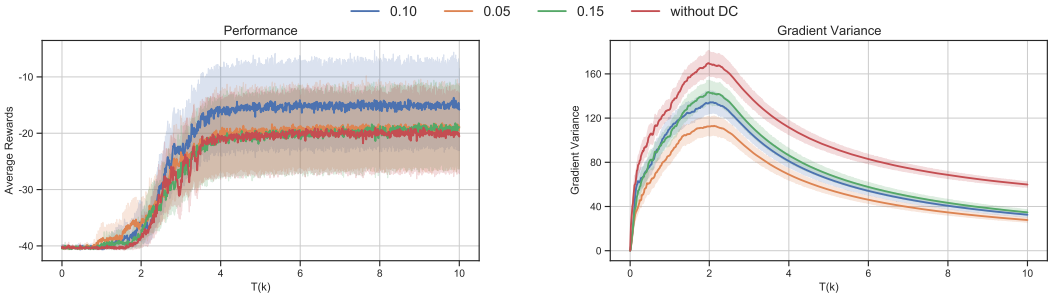


Figure 2: Ablation study of the double clipping trick. The numbers $0.05, 0.10, 0.15$ represent the inner clipping threshold, and "without DC" represents the case when the trick is not used. Left: average rewards; Right: running policy gradient variance.

## 4.2 StarCraft II

We evaluate CoPPO in SMAC against various state-of-the-art methods, including policy-based methods (COMA (Foerster et al., 2018), MAPPO (Yu et al., 2021) and DOP (Wang et al., 2020)) and

value-based methods (QMIX (Rashid et al., 2018) and QTRAN (Son et al., 2019)). The implementation of these baselines follows the original versions. The win rates are tested over 32 evaluation episodes after each training iteration. The hyperparameter settings and other implementation details are presented in Appendix C.2. The results are averaged over 6 different random seeds for easy maps (the upper row in Fig. 3), and 8 different random seeds for harder maps (the lower row in Fig. 3). Note that CoPPO outperforms several strong baselines including the latest multi-agent PPO (i.e., MAPPO) method in SMAC across various types and difficulties, especially in **Hard** (3s5z, 10m_vs_11m) and **Super Hard** (MMM2) maps. Moreover, as an on-policy method, CoPPO shows better stability across different runs, which is indicated by a narrower confidence interval around the learning curves.

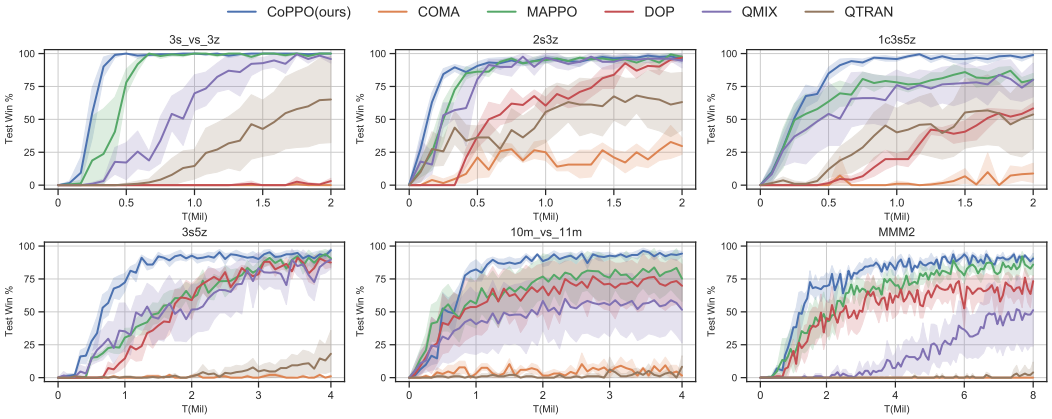


Figure 3: Comparisons against baselines on SMAC.

**Ablation study 2** The first row in Fig. 4 shows the ablation study of double clipping in SMAC, and we can see that the results share the same pattern as in Section 4.1.

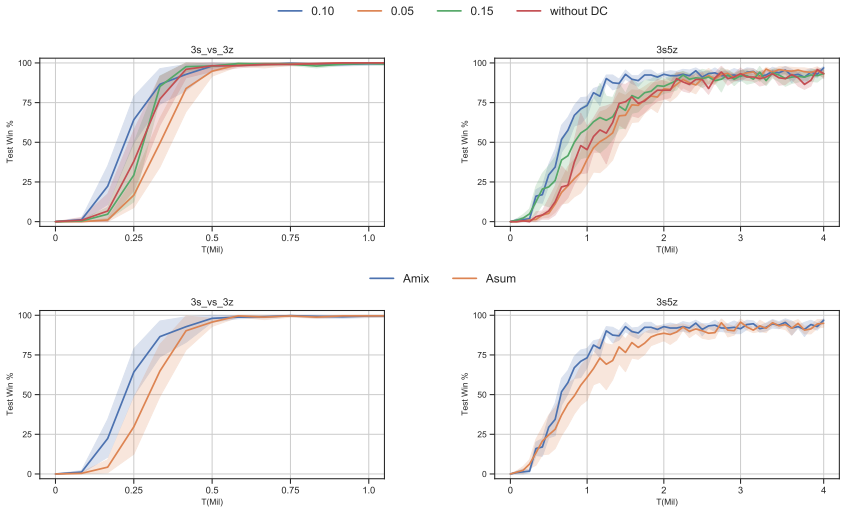


Figure 4: Ablation studies on the double clipping (the upper row) and the way of advantage decomposition (the lower row), evaluated on two maps respectively. In the upper row, the numbers "$0.05, 0.10, 0.15$" represent the values of the inner clipping threshold, and "without DC" represents the case where the double clipping trick is not utilized. In the lower row, "Amix" refers to a non-negative-weighted neural network, and "Asum" refers to an arithmetic summation.

**Ablation study 3** In Section 3.2, the global advantage is decomposed into a weighted sum of local advantages. We also compare it to a mixing network with non-negative weights and the results are

shown in Fig. 4. Similar to QMIX (Rashid et al., 2018), the effectiveness of the mixing network may largely owe to the improvement in the representational ability for the global advantage function.

## 5 Related Work

In recent years, there has been significant progress in CoMARL. Fully centralized methods suffer from scalability issues due to the exponential growth of joint action space, and applying DQN to each agent independently while treating the other agents as part of environment (Tampuu et al., 2017) suffers from the non-stationary issue (Hernandez-Leal et al., 2017; Papoudakis et al., 2019). The CTDE paradigm (Foerster et al., 2016) reaches a compromise between centralized and decentralized approaches, assuming a laboratory setting where each agent's policy can be trained using extra global information while maintaining scalable decentralized execution.

A series of work have been developed in the CTDE setting, including both value-based and policy-based methods. Most of the value-based MARL methods estimate joint action-value function by mixing local value functions. VDN (Sunehag et al., 2018) first introduces value decomposition to make the advantage of centralized training and mixes the local value functions via arithmetic summation. To improve the representational ability of the joint action-value function, QMIX (Rashid et al., 2018) proposes to mix the local action-value functions via a non-negative-weighted neural network. QTRAN (Son et al., 2019) studies the decentralization & suboptimatlity trade-off and introduces a corresponding penalty term in the objective to handle it, which further enlarges the class of representable value functions. As for the policy-based methods, COMA (Foerster et al., 2018) presents the counterfactual advantage to address the credit assignment issue. MADDPG (Lowe et al., 2017) extends DDPG (Lillicrap et al., 2015) by learning centralized value functions which are conditioned on additional global information, such as other agents' actions. DOP (Wang et al., 2020) introduces value decomposition into the multi-agent actor-critic framework, which enables off-policy critic learning and addresses the centralized-decentralized mismatch issue. MAPPO (Yu et al., 2021) generalizes PPO (Schulman et al., 2017) to multi-agent settings using a global value function.

The most relevant works are MAPPO (Yu et al., 2021), MATRPO (Li and He, 2020) and MATRL (Wen et al., 2020). MAPPO extends PPO to multi-agent settings simply by centralizing the critics. With additional techniques such as Value Normalization, MAPPO achieves promising performance compared to several strong baselines. Note that our implementation is built on the one of MAPPO (please refer to Appendix C.2 for more details).

As for MATRPO and MATRL, they both try to extend TRPO (Schulman et al., 2015a) to multi-agent settings. MATRPO focuses on fully decentralized training, which is realized through splitting the joint TRPO objective into $N$ independent parts for each agent and transforming it into a consensus optimization problem; while MATRL computes independent trust regions for each agent assuming that other agents' policies are fixed, and then solves a meta-game in order to find the best response to the predicted joint policy of other agents derived by independent trust region optimization. Different from our work, they adopt the settings where agents have separate local reward signals. By comparison, CoPPO does not directly optimize the constrained objective derived in Section 3.1, but instead incorporates the trust region constraint into the optimization objective, in order to reduce the computational complexity and simplify the implementation. CoPPO can sufficiently take advantage of the centralized training and enable a coordinated adaptation of step size among agents during the policy update process.

## 6 Conclusion

In this paper, we extend the PPO algorithm to the multi-agent setting and propose an algorithm named CoPPO through a theoretically-grounded derivation that ensures approximately monotonic policy improvement. CoPPO can properly address the issues of scalability and credit assignment, which is interpreted both theoretically and empirically. We also introduce a double clipping trick to strike the balance between reducing variance and achieving coordination by considering other agents' update. Experiments on specially designed cooperative matrix games and the SMAC benchmark verify that CoPPO outperforms several strong baselines and is competitive with the latest multi-agent methods.

## Acknowledgments and Disclosure of Funding

The work is supported by the National Natural Science Foundation of China (No. 62076259) and the Tencent AI Lab Rhino-Bird Focused Research Program (No. JR202063). The authors would like to thank Wenxuan Zhu for pointing out some of the mistakes in the proof, Xingzhou Lou and Xianjie Zhang for running some of the experiments, as well as Siling Chen for proofreading the manuscript. Finally, the reviewers and metareviewer are highly appreciated for their constructive feedback on the paper.

## References

Tong Wu, Pan Zhou, Kai Liu, Yali Yuan, Xiumin Wang, Huawei Huang, and Dapeng Oliver Wu. Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks. *IEEE Transactions on Vehicular Technology*, 69(8):8243–8256, 2020.

Anamika Sharma and Siddhartha Chauhan. A distributed reinforcement learning based sensor node scheduling algorithm for coverage and connectivity maintenance in wireless sensor network. *Wireless Networks*, 26(6):4411–4429, 2020.

Chao Yu, Xin Wang, Xin Xu, Minjie Zhang, Hongwei Ge, Jiankang Ren, Liang Sun, Bingcai Chen, and Guozhen Tan. Distributed multiagent coordinated learning for autonomous driving in highways based on dynamic coordination graphs. *IEEE Transactions on Intelligent Transportation Systems*, 21(2):735–748, 2019.

Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. A survey of learning in multiagent environments: Dealing with non-stationarity. *ArXiv*, abs/1707.09183, 2017.

Georgios Papoudakis, Filippos Christianos, Arrasy Rahman, and Stefano V. Albrecht. Dealing with non-stationarity in multi-agent deep reinforcement learning. *arXiv preprint arXiv:1906.04737*, 2019.

Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 29:2137–2145, 2016.

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

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the Conference of Association for the Advancement of Artificial Intelligence*, volume 32, page 2974–2982, 2018.

Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. Dop: Off-policy multi-agent decomposed policy gradients. In *International Conference on Learning Representations*, 2020.

Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, A. Bayen, and Yi Wu. The surprising effectiveness of mappo in cooperative, multi-agent games. *ArXiv*, abs/2103.01955, 2021.

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 *International Conference on Autonomous Agents and MultiAgent Systems*, pages 2085–2087, 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, 2018.

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, 2019.

Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multi-agent variational exploration. *Advances in Neural Information Processing Systems*, 32:7613–7624, 2019.

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*, pages 66–83, 2017.

Xinliang Song, Tonghan Wang, and Chongjie Zhang. Convergence of multi-agent learning with a finite step size in general-sum games. In *International Conference on Autonomous Agents and MultiAgent Systems*, pages 935–943, 2019.

Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and trends in Robotics*, 2(1-2):388–403, 2013.

Jianyu Su, Stephen Adams, and Peter Beling. Value-decomposition multi-agent actor-critics. In *Proceedings of the Conference of Association for the Advancement of Artificial Intelligence*, volume 35, pages 11352–11360, 2021.

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. In *International Conference on Autonomous Agents and MultiAgent Systems*, pages 2186–2188, 2019.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1063, 2000.

Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307, 2012.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596, 2018.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, 1998(746-752):2, 1998.

Frans A Oliehoek and Christopher Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015a.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, pages 267–274, 2002.

Deheng Ye, Zhao Liu, Mingfei Sun, Bei Shi, Peilin Zhao, Hao Wu, Hongsheng Yu, Shaojie Yang, Xipeng Wu, Qingwei Guo, et al. Mastering complex control in moba games with deep reinforcement learning. In *Proceedings of the Conference of Association for the Advancement of Artificial Intelligence*, pages 6672–6679, 2020a.

Deheng Ye, Guibin Chen, Wen Zhang, Sheng Chen, Bo Yuan, Bo Liu, Jia Chen, Zhao Liu, Fuhao Qiu, Hongsheng Yu, Yinyuting Yin, Bei Shi, Liang Wang, Tengfei Shi, Qiang Fu, Wei Yang, Lanxiao Huang, and Wei Liu. Towards playing full MOBA games with deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 21–632, 2020b.

K Spiros and K Daniel. Reinforcement learning of coordination in cooperative mas. In *The 18th National Conference on Artificial Intelligence*, pages 326–331, 2002.

Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395, 2017.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Hepeng Li and Haibo He. Multi-agent trust region policy optimization. *ArXiv*, abs/2010.07916, 2020.

Ying Wen, Hui Chen, Yaodong Yang, Zheng Tian, Minne Li, Xu Chen, and Jun Wang. Multi-agent trust region learning. 2020. URL `https://openreview.net/forum?id=eHG7asK_v-k`.

David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

David Pollard. Asymptopia: an exposition of statistical asymptotic theory. 2000. URL `http://www.stat.yale.edu/pollard/Books/Asymptopia`.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

# A Mathematical Details

## A.1 Difference between the performance of two joint policies

In Section 3.1, the difference between the performance of two joint policies is expressed as follows:

$$J(\tilde{\boldsymbol{\pi}}) - J(\boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}}, s\sim\rho^{\tilde{\boldsymbol{\pi}}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right], \tag{11}$$

where $\rho^{\tilde{\boldsymbol{\pi}}}$ is the unnormalized discounted visitation frequencies, i.e. $\sum_{t=0}^{\infty}\gamma^t\sum_s \Pr(s_t = s|\tilde{\boldsymbol{\pi}})$. The proof is a multi-agent version of the proof in (Kakade and Langford, 2002). Now we provide the mathematical detail formally.

*Proof.*

$$\begin{aligned}
J(\tilde{\boldsymbol{\pi}}) - J(\boldsymbol{\pi}) &= \mathbb{E}_{\tilde{\boldsymbol{\pi}}}\left[\sum_{t=0}^{\infty}\gamma^t R_{t+1} - V^{\boldsymbol{\pi}}(s_0)\right] && (12)\\
&= \mathbb{E}_{\tilde{\boldsymbol{\pi}}}\left[R_1 + \gamma V^{\boldsymbol{\pi}}(s_1) - V^{\boldsymbol{\pi}}(s_0) + \gamma[R_2 + \gamma V^{\boldsymbol{\pi}}(s_2) - V^{\boldsymbol{\pi}}(s_1)] + \cdots\right] && (13)\\
&= \mathbb{E}_{\tilde{\boldsymbol{\pi}}}\left[\sum_{t=0}^{\infty}\gamma^t A^{\boldsymbol{\pi}}(s_t,\boldsymbol{a})\right] && (14)\\
&= \sum_{t=0}^{\infty}\gamma^t\sum_s \Pr(s_t = s|\tilde{\boldsymbol{\pi}})\sum_{\boldsymbol{a}}\tilde{\boldsymbol{\pi}}(\boldsymbol{a}|s)\left[Q^{\boldsymbol{\pi}}(s,\boldsymbol{a}) - V^{\boldsymbol{\pi}}(s)\right] && (15)\\
&= \mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}}, s\sim\rho^{\tilde{\boldsymbol{\pi}}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right] && (16)
\end{aligned}$$

□

## A.2 Approximation that matches the true value to first order

In Section 3.1, we claim that $\tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})$ matches $J(\tilde{\boldsymbol{\pi}})$ to first order. Intuitively, this means that a sufficiently small update of the joint policy which improves $\tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})$ will also improve $J(\tilde{\boldsymbol{\pi}})$. Now we prove it formally.

*Proof.* We represent the policy using its parameter, i.e. $\theta$ for $\boldsymbol{\pi}$ and $\tilde{\theta}$ for $\tilde{\boldsymbol{\pi}}$. Because $\tilde{J}_{\boldsymbol{\pi}}(\boldsymbol{\pi}) = J(\boldsymbol{\pi})$, there are $\tilde{J}_{\theta}(\theta) = J(\theta)$. Furthermore, we have:

$$\begin{aligned}
\nabla_{\tilde{\theta}}\tilde{J}_{\theta}(\tilde{\theta})\big|_{\theta} &= \nabla_{\tilde{\theta}}\left(J(\theta) + \mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}}, s\sim\rho^{\boldsymbol{\pi}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right]\right) && (17)\\
&= \sum_t\gamma^t\sum_s \Pr(s_t = s|\boldsymbol{\pi})\sum_{\boldsymbol{a}}\nabla_{\tilde{\theta}}\tilde{\boldsymbol{\pi}}(\boldsymbol{a}|s)\big|_{\theta}A^{\boldsymbol{\pi}}(s,\boldsymbol{a}) && (18)\\
&= \nabla_{\tilde{\theta}}J(\tilde{\boldsymbol{\pi}})\big|_{\theta}, && (19)
\end{aligned}$$

where the last step is indicated by Theorem 1 in (Sutton et al., 2000).

□

## A.3 Upper bound for the error of joint policy approximation

**Theorem.** *Let* $\epsilon = \max_{s,\boldsymbol{a}}|A^{\boldsymbol{\pi}}(s,\boldsymbol{a})|, \alpha_i = \sqrt{\frac{1}{2}D_{TV}^{\max}[\pi^i||\tilde{\pi}^i]}, 1 \le i \le N$*, and* $N$ *be the total number of agents, then the error of the approximation in Eq. 4 can be explicitly bounded as follows:*

$$\left|J(\tilde{\boldsymbol{\pi}}) - \tilde{J}_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})\right| \le 4\epsilon\left[\frac{1-\gamma\prod_{i=1}^{N}(1-\alpha_i)}{1-\gamma} - 1\right]. \tag{20}$$

*Proof.* We first prove that for a fixed $s$, the following inequality holds:

$$|\mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right]| \le 2\epsilon\left[1 - \prod_{i=1}^{N}(1-\alpha_i)\right]. \tag{21}$$

Note that

$$\mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}}[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})] = \boldsymbol{\pi}(\boldsymbol{a}|s)\left[Q(s,\boldsymbol{a}) - V(s)\right] \tag{22}$$
$$= V(s) - V(s) \tag{23}$$
$$= 0. \tag{24}$$

Therefore,

$$\mathbb{E}_{\tilde{\boldsymbol{a}}\sim\tilde{\boldsymbol{\pi}}}[A^{\boldsymbol{\pi}}(s,\tilde{\boldsymbol{a}})] = \mathbb{E}_{(\boldsymbol{a},\tilde{\boldsymbol{a}})\sim(\boldsymbol{\pi},\tilde{\boldsymbol{\pi}})}[A^{\boldsymbol{\pi}}(s,\tilde{\boldsymbol{a}}) - A^{\boldsymbol{\pi}}(s,\boldsymbol{a})] \tag{25}$$
$$= \Pr(\boldsymbol{a}\neq\tilde{\boldsymbol{a}})\cdot\mathbb{E}_{(\boldsymbol{a},\tilde{\boldsymbol{a}})\sim(\boldsymbol{\pi},\tilde{\boldsymbol{\pi}})}[A^{\boldsymbol{\pi}}(s,\tilde{\boldsymbol{a}}) - A^{\boldsymbol{\pi}}(s,\boldsymbol{a})] \tag{26}$$
$$= \left[1-\prod_{i=1}^{N}\left(1-\Pr\left(a^i\neq\tilde{a}^{-i}\right)\right)\right]\mathbb{E}_{(\boldsymbol{a},\tilde{\boldsymbol{a}})\sim(\boldsymbol{\pi},\tilde{\boldsymbol{\pi}})}[A^{\boldsymbol{\pi}}(s,\tilde{\boldsymbol{a}}) - A^{\boldsymbol{\pi}}(s,\boldsymbol{a})] \tag{27}$$
$$\leq \left[1-\prod_{i=1}^{N}(1-\eta_i)\right]\mathbb{E}_{(\boldsymbol{a},\tilde{\boldsymbol{a}})\sim(\boldsymbol{\pi},\tilde{\boldsymbol{\pi}})}[A^{\boldsymbol{\pi}}(s,\tilde{\boldsymbol{a}}) - A^{\boldsymbol{\pi}}(s,\boldsymbol{a})] \tag{28}$$
$$\leq \left[1-\prod_{i=1}^{N}(1-\eta_i)\right]\cdot 2\max_{s,\boldsymbol{a}}|A^{\boldsymbol{\pi}}(s,\boldsymbol{a})| \tag{29}$$
$$= 2\epsilon\left[1-\prod_{i=1}^{N}(1-\eta_i)\right], \tag{30}$$

where $\eta_i = \max_{\tau^i}\Pr(a^i\neq\tilde{a}^i|\tau^i)$, and $(\boldsymbol{\pi},\tilde{\boldsymbol{\pi}})$ represents $\{(\pi^1,\tilde{\pi}^1),\ldots,(\pi^N,\tilde{\pi}^N)\}$, $(\pi^i,\tilde{\pi}^i)$ is an $\alpha_i$-coupled policy pair for $i=1,2,\ldots,N$. The definition of $\alpha_i$-coupled policy pair in (Schulman et al., 2015a) implies that $(\pi^i,\tilde{\pi}^i)$ is a joint distribution $p(a^i,\tilde{a}^i|\tau^i)$ satisfying $\Pr(a^i\neq\tilde{a}^i|\tau^i)\leq\alpha_i$.

From Proposition 4.7 in (Levin and Peres, 2017), if we have two distributions $p_X, p_Y$ that satisfy $D_{TV}(p_X\|p_Y)=\alpha$, then there exists a joint distribution $P(X,Y)$ whose marginals are $p_X, p_Y$, such that:

$$\Pr(X=Y) = 1-\alpha \tag{31}$$

Furthermore, note that there is a relationship between the total variation divergence and the KL divergence (Pollard, 2000): $D_{TV}(p\|q)^2 \leq \frac{1}{2}D_{KL}(p\|q)$. Now let $\alpha_i = \max_{\tau^i}\sqrt{\frac{1}{2}D_{KL}\left[\pi^i(\cdot|\tau^i)\|\tilde{\pi}^i(\cdot|\tau^i)\right]}$, then there exists a joint distribution $(\pi^i,\tilde{\pi}^i)$ whose marginals are $\pi^i,\tilde{\pi}^i$, satisfying:

$$\Pr(a^i=\tilde{a}^i|\tau^i)\geq 1-\alpha_i. \tag{32}$$

Thus $\eta_i\leq\alpha_i$. Since $\eta_i,\alpha_i\leq 1$, $\left[1-\prod_{i=1}^{N}(1-\eta_i)\right]$ will increase as $\eta_i$ increases. Then Eq. (21) can be derived by replacing $\eta_i$ with $\alpha_i$ in Eq. (30).

For simplification, we denote $\mathbb{E}_{\tilde{\boldsymbol{a}}\sim\tilde{\boldsymbol{\pi}}}[A^{\boldsymbol{\pi}}(s,\tilde{\boldsymbol{a}})]$ as $\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s)$ and use $n_t$ to represent the times $\boldsymbol{a}\neq\tilde{\boldsymbol{a}}$ before timestep $t$. Then there is:

$$\left|\mathbb{E}_{s_t\sim\rho^{\tilde{\boldsymbol{\pi}}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)] - \mathbb{E}_{s_t\sim\rho^{\boldsymbol{\pi}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)]\right| \tag{33}$$
$$= \Pr(n_t>0)\cdot\left|\mathbb{E}_{s_t\sim\rho^{\tilde{\boldsymbol{\pi}}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)] - \mathbb{E}_{s_t\sim\rho^{\boldsymbol{\pi}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)]\right| \tag{34}$$
$$= (1-\Pr(n_t=0))\cdot\left|\mathbb{E}_{s_t\sim\rho^{\tilde{\boldsymbol{\pi}}}|n_t>0}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)] - \mathbb{E}_{s_t\sim\rho^{\boldsymbol{\pi}}|n_t>0}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)]\right| \tag{35}$$
$$= \left(1-\prod_{t'=0}^{t}\prod_{i=1}^{N}\Pr(a_t^i=\tilde{a}_t^i|\tau^i)\right)\cdot|\cdots| \tag{36}$$
$$\leq \left(1-\prod_{i=1}^{N}(1-\alpha_i)^t\right)\cdot|\cdots| \tag{37}$$
$$\leq \left(1-\prod_{i=1}^{N}(1-\alpha_i)^t\right)\cdot 2\max_{s}\left|\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s)\right|, \tag{38}$$

where $|\cdots|$ denotes $\left|\mathbb{E}_{s_t\sim\rho^{\tilde{\boldsymbol{\pi}}}|n_t>0}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)] - \mathbb{E}_{s_t\sim\rho^{\boldsymbol{\pi}}|n_t>0}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)]\right|$ for brevity. Then, the following can be derived using Eq. (21):

$$\left|\mathbb{E}_{s_t\sim\rho^{\tilde{\boldsymbol{\pi}}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)] - \mathbb{E}_{s_t\sim\rho^{\boldsymbol{\pi}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)]\right| \tag{39}$$

$$\leq 2\left(1-\prod_{i=1}^{N}(1-\alpha_i)^t\right)\left[1-\prod_{i=1}^{N}(1-\alpha_i)\right]\max_{s,\boldsymbol{a}}|A^{\boldsymbol{\pi}}(s,\boldsymbol{a})| \tag{40}$$

$$\leq 4\epsilon\left[1-\prod_{i=1}^{N}(1-\alpha_i)\right]\left[1-\prod_{i=1}^{N}(1-\alpha_i)^t\right], \tag{41}$$

Finally we reach our conclusion:

$$|J(\tilde{\boldsymbol{\pi}}) - L_{\boldsymbol{\pi}}(\tilde{\boldsymbol{\pi}})| = \left|\mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}},s\sim\rho^{\tilde{\boldsymbol{\pi}}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right] - \mathbb{E}_{\boldsymbol{a}\sim\tilde{\boldsymbol{\pi}},s\sim\rho^{\boldsymbol{\pi}}}\left[A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right]\right| \tag{42}$$

$$= \left|\sum_s\sum_{t=0}^{\infty}\gamma^t\mathrm{Pr}(s_t=s|\tilde{\boldsymbol{\pi}})\sum_{\boldsymbol{a}}\tilde{\boldsymbol{\pi}}(\boldsymbol{a}|s)A^{\boldsymbol{\pi}}(s,\boldsymbol{a}) - \sum_s\sum_{t=0}^{\infty}\gamma^t\mathrm{Pr}(s_t=s|\boldsymbol{\pi})\sum_{\boldsymbol{a}}\tilde{\boldsymbol{\pi}}(\boldsymbol{a}|s)A^{\boldsymbol{\pi}}(s,\boldsymbol{a})\right| \tag{43}$$

$$\leq \sum_{t=0}^{\infty}\gamma^t\left|\mathbb{E}_{s_t\sim\rho^{\tilde{\boldsymbol{\pi}}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)] - \mathbb{E}_{s_t\sim\rho^{\boldsymbol{\pi}}}[\bar{A}^{\tilde{\boldsymbol{\pi}},\boldsymbol{\pi}}(s_t)]\right| \tag{44}$$

$$\leq \sum_{t=0}^{\infty}\gamma^t\cdot 4\epsilon\left[1-\prod_{i=1}^{N}(1-\alpha_i)\right]\left[1-\prod_{i=1}^{N}(1-\alpha_i)^t\right] \tag{45}$$

$$= 4\epsilon\left[1-\prod_{i=1}^{N}(1-\alpha_i)\right]\left[\frac{1}{1-\gamma}-\frac{1}{1-\gamma\prod_{i=1}^{N}(1-\alpha_i)}\right] \tag{46}$$

$$\leq 4\epsilon\left[\frac{1-\gamma\prod_{i=1}^{N}(1-\alpha_i)}{1-\gamma}-1\right]. \tag{47}$$

□

### A.4 Transformation from the joint objective into the local objectives

In Section 3.2, the joint objective is derived as:

$$\underset{\theta^1,\ldots,\theta^N}{\text{maximize}}\ \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}_{old}}\left\{\min\left[\left(\prod_{j=1}^{N}r^j\right)A^{\boldsymbol{\pi}},\text{clip}\left(\left(\prod_{j=1}^{N}r^j\right),1-\epsilon,1+\epsilon\right)A^{\boldsymbol{\pi}}\right]\right\}, \tag{48}$$

where $\theta^j$ is the parameter of agent $j$'s policy, and $r^j = \frac{\pi^j\left(a^j|\tau^j;\theta^j\right)}{\pi^j_{old}\left(a^j|\tau^j;\theta^j_{old}\right)}$. After a linear decomposition on $A^{\boldsymbol{\pi}}$ with non-negative weights (i.e. $A^{\boldsymbol{\pi}} = \sum_j c^j A^j$), the objective above then can be transformed into:

$$\underset{\boldsymbol{\theta}}{\text{maximize}}\ \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}_{old}}\left\{\min\left[\left(\prod_{j\neq i}r^j\right)r^iA^i,\text{clip}\left(\left(\prod_{j\neq i}r^j\right)r^i,1-\epsilon,1+\epsilon\right)A^i\right]\right\}, \tag{49}$$

where $i=1,\ldots,N$. Now we provide a detailed proof.

*Proof.* If

$$\min\left[\left(\prod_{j=1}^{N}r^j\right)A^{\boldsymbol{\pi}},\text{clip}\left(\left(\prod_{j=1}^{N}r^j\right),1-\epsilon,1+\epsilon\right)A^{\boldsymbol{\pi}}\right] = \text{clip}\left(\left(\prod_{j=1}^{N}r^j\right),1-\epsilon,1+\epsilon\right)A^{\boldsymbol{\pi}}, \tag{50}$$

then the objective is actually $(1-\epsilon)A^{\boldsymbol{\pi}}$ or $(1+\epsilon)A^{\boldsymbol{\pi}}$, and no gradient will be backpropagated as none of $\theta^1, \ldots, \theta^N$ is in the objective. Furthermore, there is

$$\min\left[\left(\prod_{j=1}^{N} r^j\right) A^{\boldsymbol{\pi}}, \text{clip}\left(\left(\prod_{j=1}^{N} r^j\right), 1-\epsilon, 1+\epsilon\right) A^{\boldsymbol{\pi}}\right] \tag{51}$$

$$= \min\left[\sum_i c^i \left(\prod_{j=1}^{N} r^j\right) A^i, \sum_i c^i \text{clip}\left(\left(\prod_{j=1}^{N} r^j\right), 1-\epsilon, 1+\epsilon\right) A^i\right]. \tag{52}$$

Thus, the discussion can be simplified to the case where

$$\min\left[\left(\prod_{j=1}^{N} r^j\right) A^{\boldsymbol{\pi}}, \text{clip}\left(\left(\prod_{j=1}^{N} r^j\right), 1-\epsilon, 1+\epsilon\right) A^{\boldsymbol{\pi}}\right] = \left(\prod_{j=1}^{N} r^j\right) A^{\boldsymbol{\pi}}. \tag{53}$$

While $\frac{\partial\left(\prod_{j=1}^{N} r^j\right)A^{\boldsymbol{\pi}}}{\partial\left(\prod_{j=1}^{N} r^j\right)A^i} = c^i$ and $c^i \geq 0$, there is

$$\max_{\theta^1,\ldots,\theta^N} \left(\prod_{j=1}^{N} r^j\right) A^{\boldsymbol{\pi}} = \max_{\theta^1,\ldots,\theta^N} \sum_i c^i \left(\prod_{j=1}^{N} r^j\right) A^i \tag{54}$$

$$= \sum_i c^i \max_{\boldsymbol{\theta}} \left(\prod_{j=1}^{N} r^j\right) A^i \tag{55}$$

Therefore, the transformation from Eq. (48) to Eq. (49) is proved.

□

### A.5 The potential high variance of probability ratio product

Section 3.2 mentions that there exists a risk of high variance in estimating the policy gradient when optimizing Eq. (7), due to the following proposition:

**Proposition.** *Assuming that the agents are fully independent during execution, then the following inequality holds:*

$$\mathit{Var}_{\boldsymbol{a}^{-i}\sim\boldsymbol{\pi}_{old}^{-i}}\left[\prod_{j\neq i} r^j\right] \geq \prod_{j\neq i} \mathit{Var}_{a^j\sim\pi_{old}^j}\left[r^j\right], \tag{56}$$

*where* $r^j = \frac{\pi^j(a^j|\tau^j;\theta^j)}{\pi_{old}^j(a^j|\tau^j;\theta_{old}^j)}$.

Because the agents execute the actions based only on locally observable information, it is reasonable to assume that $\pi^i$ and $\pi^j$ is independent when $i \neq j$. Now we present a detailed proof for this proposition.

*Proof.* Because the agents are fully independent during execution, there is a decomposition that $\boldsymbol{\pi}^{-i}(\boldsymbol{a}^{-i}|\boldsymbol{\tau}^{-i}) = \prod_{j\neq i}\pi^j(a^j|\tau^j)$.

Now we use mathematical induction to prove the fact. First, we assume that there are 3 agents, and let $i = 3$ without loss in generality. Then there is:

$$\text{Var}_{a^1,a^2}\left[r_1 r_2\right] = \mathbb{E}_{a^1,a^2}\left[\left(r_1 r_2\right)^2\right] - \left(\mathbb{E}_{a^1,a^2}\left[r_1 r_2\right]\right)^2 \tag{57}$$

$$= \mathbb{E}_{a^1}\left[r_1^2\right]\mathbb{E}_{a^2}\left[r_2^2\right] - \left(\mathbb{E}_{a^1}\left[r_1\right]\mathbb{E}_{a^2}\left[r_2\right]\right)^2. \tag{58}$$

Hence, there is:

$$
\begin{aligned}
&\mathrm{Var}_{a^1,a^2}\left[r_1 r_2\right] - \mathrm{Var}_{a^1}\left[r_1\right]\mathrm{Var}_{a^2}\left[r_2\right] && (59)\\
=&\mathbb{E}_{a^1}\left[r_1^2\right]\mathbb{E}_{a^2}\left[r_2^2\right] - \left(\mathbb{E}_{a^1}\left[r_1\right]\mathbb{E}_{a^2}\left[r_2\right]\right)^2 - \\
&\left[\mathbb{E}_{a^1}\left[r_1^2\right] - \left(\mathbb{E}_{a^1}\left[r_1\right]\right)^2\right]\left[\mathbb{E}_{a^2}\left[r_2^2\right] - \left(\mathbb{E}_{a^2}\left[r_2\right]\right)^2\right] && (60)\\
=&\left(\mathbb{E}_{a^1}\left[r_1\right]\right)^2\mathbb{E}_{a^2}\left[r_2^2\right] + \left(\mathbb{E}_{a^2}\left[r_2\right]\right)^2\mathbb{E}_{a^1}\left[r_1^2\right] - 2\left(\mathbb{E}_{a^1}\left[r_1\right]\mathbb{E}_{a^2}\left[r_2\right]\right)^2 && (61)\\
=&\left(\mathbb{E}_{a^1}\left[r_1\right]\right)^2\mathrm{Var}_{a^2}\left[r_2\right] + \left(\mathbb{E}_{a^2}\left[r_2\right]\right)^2\mathrm{Var}_{a^1}\left[r_1\right] \geq 0. && (62)
\end{aligned}
$$

By now we have proven $\mathrm{Var}_{a^1,a^2}\left[r_1 r_2\right] \geq \mathrm{Var}_{a^1}\left[r_1\right]\mathrm{Var}_{a^2}\left[r_2\right]$. Then if Eq. (56) holds for the case of $N$ agents, then obviously there is:

$$
\begin{aligned}
\prod_{j\neq i}^{N+1}\mathrm{Var}\left[r^j\right] &= \left(\prod_{j\neq i}^{N}\mathrm{Var}\left[r^j\right]\right)\mathrm{Var}_{a^{N+1}}\left[r^{N+1}\right] && (63)\\
&\leq \mathrm{Var}\left[\prod_{j\neq i}^{N} r^j\right]\mathrm{Var}_{a^{N+1}}\left[r^{N+1}\right] && (64)\\
&\leq \mathrm{Var}\left[\prod_{j\neq i}^{N+1} r^j\right], && (65)
\end{aligned}
$$

thus proving the proposition.

□

### A.6 The simplification in the analysis of CoPPO and MAPPO

In Section 3.3, the difference between CoPPO and MAPPO is simplified to the difference between $\mathbb{E}_{\boldsymbol{\pi}_{old}}\left[r_k^i A^i\right]$ and $\mathbb{E}_{\boldsymbol{\pi}_{old}}\left[r_k^i \tilde{A}_k^i\right]$. Now we detail the rationality of this simplification.

In each update, the value of both the two objectives start from the respective lower bounds and are updated conservatively during the optimization epochs. The objectives monotonically increase or decrease until they reach the clipping threshold. No update will be made when the objective is clipped, because $\theta^i$ is not in the clipped value (i.e. $(1-\epsilon_1)A^i$ or $(1+\epsilon_1)A^i$) and no gradient will be backpropagated then, just as discussed in Appendix A.4.

### A.7 $\prod_{j\neq i} r_k^j$ implies the variation of the probability to take $\boldsymbol{a}^{-i}$

Section 3.3 mentions that $\prod_{j\neq i} r_k^j > 1$ will cause an increase in $\boldsymbol{\pi}^{-i}(\boldsymbol{a}^{-i}|\boldsymbol{\tau}^{-i})$ and vice versa. Now we provide the details.

Similar to Appendix A.5, the decentralized policies can be viewed independently, thus $\boldsymbol{\pi}^{-i}(\boldsymbol{a}^{-i}|\boldsymbol{\tau}^{-i}) = \prod_{j\neq i}\pi^j(a^j|\tau^j)$. By definition, $\prod_{j\neq i} r_k^j = \prod_{j\neq i}\frac{\pi_k^j(a^j|\tau^j)}{\pi_{old}^j(a^j|\tau^j)}$. Synthesizing the two equations, we have $\prod_{j\neq i} r_k^j = \frac{\boldsymbol{\pi}_k^{-i}(\boldsymbol{a}^{-i}|\boldsymbol{\tau}^{-i})}{\boldsymbol{\pi}_{old}^{-i}(\boldsymbol{a}^{-i}|\boldsymbol{\tau}^{-i})}$ which suggests that if $\prod_{j\neq i} r_k^j > 1$, $\boldsymbol{a}^{-i}$ will be more likely to be jointly performed by the other agents given similar observations, and vice versa.

## B Pseudo Code

The details of our CoPPO algorithm are given in Algorithm 1.

## C Implementation Details

Experiments are conducted on NVIDIA Quadro RTX 5000 GPUs. The network architectures, optimizers, hyperparameters and environment settings in the cooperative matrix game and SMAC are described respectively in the following subsections.

**Algorithm 1** The CoPPO Algorithm

Initialize policies $\pi_{old}^1, \ldots, \pi_{old}^N$ for $N$ agents respectively;
**for** $iteration = 1, 2, \ldots$ **do**
  **for** $rollout\ thread = 1, 2, \ldots, R$ **do**
    Run policies $\pi_{old}^{1:N}$ in environment for $T$ time steps;
    Compute advantage estimates $\hat{A}_{1:T}^{\pi_{old}^j}, \ldots, \hat{A}_{1:T}^{\pi_{old}^j},\ j = 1, 2, \ldots, N$;
  **end for**
  **for** $k = 0, 1, \ldots, K-1$ **do**
    **for** $i = 1, 2, \ldots, N$ **do**
      Optimize the objective
      $L(\theta^i) = \mathbb{E}_{\boldsymbol{a} \sim \boldsymbol{\pi}_{old}} \left\{ \min \left[ g(\boldsymbol{r}^{-i}) r^i A^i, \text{clip}\left( g(\boldsymbol{r}^{-i}) r^i, 1-\epsilon_1, 1+\epsilon_1 \right) A^i \right] \right\}$
      to update the policy $\pi^i$ w.r.t. $\theta^i$;
    **end for**
  **end for**
  $\theta_{old}^j \leftarrow \theta_K^j,\ j = 1, 2, \ldots N$;
**end for**

### C.1 Cooperative matrix game

We utilize the same actor-critic network architecture for all the algorithms. The actor consists of two 18-dimensional fully-connected layers with $\tanh$ activation. For the critic, two 72-dimensional fully-connected layers are adopted with $\tanh$ activation. For the hyper network in DOP which is used to derive the weights and biases for local value mixing, we use two 36-dimensional fully-connected layers with $\tanh$ activation for both the weights and biases deriving. The optimization of both the actors and critics is conducted using RMSprop with the learning rate of $5 \times 10^{-4}$ and $\alpha$ of $0.99$. No momentum or weight decay is used in the optimizers. The discounted factor is set to $0.99$; the number of the optimization epochs (i.e. $K$) for CoPPO and MAPPO is set to $8$; the outer clipping threshold (i.e. $\epsilon$ for MAPPO and $\epsilon_1$ for CoPPO) is set to $0.20$. For the inner clipping threshold in CoPPO, we consider $\epsilon_2 \in \{0.05, 0.10, 0.15\}$ and adopt $0.10$ in the comparison with baselines. For exploration, we use $\epsilon$-greedy with $\epsilon$ annealed linearly from $0.9$ to $0.02$ over $6k$ timesteps.

### C.2 SMAC

The same actor-critic network architecture are utilized for all maps we have evaluated on. Both the actor and critic networks consist of two fully-connected layers, one GRU layer and one fully-connected layer sequentially with ReLU activation. For the mixing network mentioned in Section 3.2, we adopt the hyper network in (Rashid et al., 2018) to derive the weights and bias for local advantages, and enforce the weights to be non-negative. Similar to QMIX, the input of the hyper network is the global state. The dimensions of these layers are all set to 64, except for the 32-dimensional hidden layers of the mixing network.

For the evaluation on different maps, all the hyperparameters are fixed except for the number of optimization epochs which is set to 15 for 2s3z, 3s_vs_3z, and 1c3s5z, 10 for 3s5z and 10m_vs_11m, and 8 for MMM2. The number of epochs overall decreases as the difficulty of the map increases, ranging from $5$ to $15$. The optimization of both the actors and critics is conducted using RMSprop with the learning rate of $5 \times 10^{-4}$ and $\alpha$ of $0.99$. No momentum or weight decay is used in the optimizers. The discounted factor $\gamma$ is set to 0.99. For advantage estimation, the generalized advantage estimation (Schulman et al., 2015b) is adopted and the corresponding hyperparameter $\lambda$ is set to 0.90. Note that state value functions instead of state-action value functions are estimated in SMAC. The inner clipping threshold (i.e. $\epsilon_2$ for CoPPO) is set to $0.10$, while the outer clipping threshold (i.e. $\epsilon$ for MAPPO and $\epsilon_1$ for CoPPO) is set to $0.20$. 8 parallel environments are run for data collecting.

Overall, our implementation builds upon the one of (Yu et al., 2021). Note that MAPPO uses hand-coded states (i.e. Feature-Pruned Agent-Specific Global State) as the input of value functions, while in our implementation these states are modified into the concatenation of the Environment-Provided Global State and the Local Observation, in order to make the comparison with baselines fair. For the other baselines, we adopt the official implementations and their default hyperparameter settings that have been fine-tuned on this benchmark.

# D Additional Results

## D.1 Cooperative matrix games

Section 4.1 shows the results on a modification of the two-player penalty game. Now we present the results on other matrix games across different types and different difficulties in Fig. 5, and CoPPO outperforms the other methods in almost all the games, thus showing the general effectiveness. For evaluation, the results are also averaged over 100 runs.

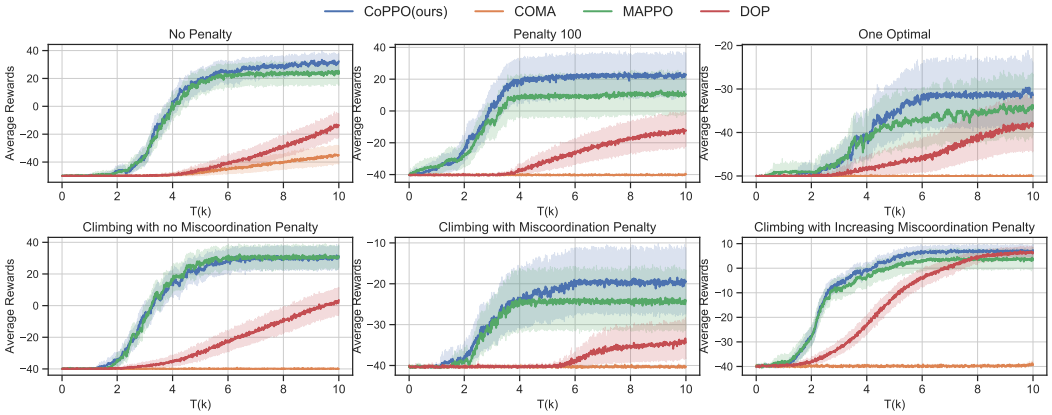


Figure 5: Performance comparisons in six matrix games.

These games are all 4-agent, 9-action cooperative games. The respective reward settings are as follows. The "miscoordination" mentioned below all refers to the case where any three agents act the same while the other does not. Fig. 5-upper left and middle are both simplifications of the penalty game presented in Section 4.1. In Fig. 5-upper left, there is no penalty for miscoordination; in Fig. 5-upper middle, the team reward becomes larger (100) when the agents play the same action. The other rewards are set the same with the one in Section 4.1. In Fig. 5-upper right, there is only one optimal joint action and the difficulty lies mainly in exploration. The agents will receive the reward of 50 if agent $i$ plays action $i$ and -50 otherwise. Fig. 5-lower left is the result on a modification of the *climbing game* that has been used as another challenging test bed for CoMARL algorithms (Claus and Boutilier, 1998), where the reward is $i \cdot 10$ if the agents all play action $i$ and -40 otherwise. Fig. 5-lower middle and right gradually increase the difficulty of the climbing game by setting obstacles in the way of climbing. In Fig. 5-lower middle, the agents will be punished by -50 for miscoordination. As for Fig. 5-lower right, the miscoordination penalty increases as the matching reward increases, i.e. $-i \cdot 10$ for miscoordination on action $i$, hence the risk will become higher and higher when the agents are "climbing" to the optimal joint action.

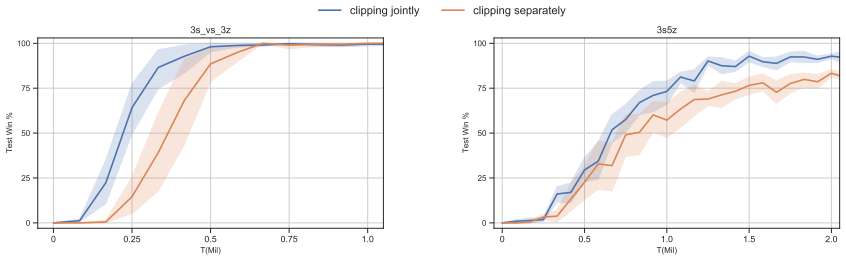


Figure 6: Ablation study on the methods of clipping.

## D.2 SMAC

### D.2.1 Comparison of clipping jointly and separately

We empirically evaluate two clipping approaches mentioned in Section 3.2, i.e. clipping jointly ($\text{clip}(\prod_{j=1}^{N} r^j, \cdot, \cdot)$) and clipping separately ($\prod_{j=1}^{N} \text{clip}(r^j, \cdot, \cdot)$). The results shown in Fig. 6 demonstrate that clipping separately performs worse than clipping jointly. To find the cause resulting in this performance discrepancy, an empirical analysis is conducted on the value of policy gradients and ratio products w.r.t. the two clipping methods, and the results are presented in Fig. 7. Obviously clipping jointly yields more stable ratio product and policy gradients than clipping separately, implying that the performance discrepancy might be owing to the stability in the policy update.

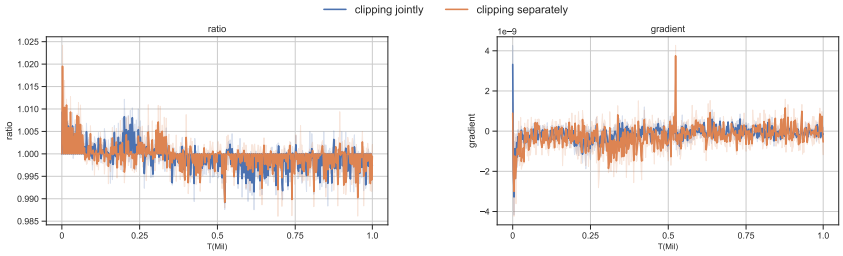


Figure 7: Comparison of two clipping methods on ratio product and mean policy gradients, evaluated on 3s_vs_3z.

### D.2.2 Results on three more maps of SMAC

Some additional results for further verification of the effectiveness of CoPPO in SMAC are given in Fig. 8. Note that CoPPO outperforms all the baselines in the maps we have evaluated on, except for the MMM map where CoPPO achieves competitive performance against MAPPO.

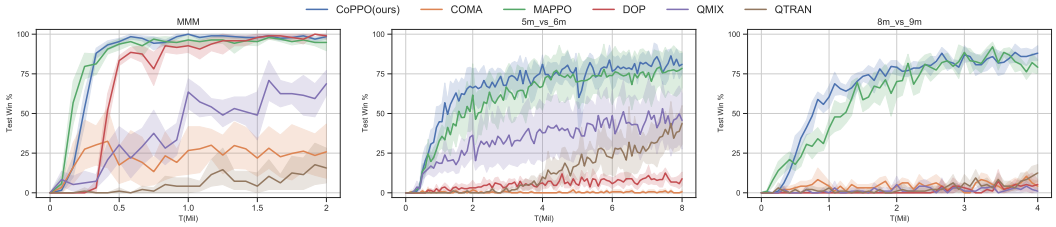


Figure 8: Additional results on SMAC.