Supplementary Material for Multi-Agent First Order Constrained Optimization in Policy Space

Anonymous Author(s)
Affiliation
Address
email

Appendix A Proof of the Optimization problem

According to the problem formulation, we give a definition of the "surrogate" cost, which aligns with what is employed in MACPO [1]:

Definition. Let $\boldsymbol{\pi}$ be a joint policy, and $\bar{\pi}^i$ be some other policy of agent i.Then for any of its costs of index $j \in 1, \cdots, m^i$, we define

$$L^i_{j,\boldsymbol{\pi}}(\bar{\pi}^i) = E_{s\sim\rho_{\boldsymbol{\pi}}, a^i\sim\bar{\pi}^i}[A^i_{j,\boldsymbol{\pi}}(s, a^i)].$$

In this way, consider $\boldsymbol{\pi}$ and $\bar{\boldsymbol{\pi}}$ be joint policies, $i \in \mathcal{N}$ be an agent and $j \in 1, \cdots, m^i$ be an index of one of its costs. From the proof of Theorem 1 in TRPO [2], (in particular, equations (41) $\sim$ (45)), applying it to joint policies $\boldsymbol{\pi}$ and $\bar{\boldsymbol{\pi}}$, we can conclude that

$$J^i_j(\bar{\boldsymbol{\pi}}) \leq J^i_j(\boldsymbol{\pi}) + E_{s\sim\rho_{\boldsymbol{\pi}}, \boldsymbol{a}\sim\bar{\boldsymbol{\pi}}}[A^i_{j,\boldsymbol{\pi}}(s, a^i)] + \frac{4\alpha^2\gamma max_{s,a^i}|A^i_{j.\boldsymbol{\pi}}(s, a^i)|}{(1-\gamma)^2}, \tag{1}$$

where $\alpha = D^{max}_{TV}(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}}) = max_s D_{TV}(\boldsymbol{\pi}(\cdot|s), \bar{\boldsymbol{\pi}}(\cdot|s))$. According to the definition of total variance divergence, defined by $D_{TV}(p||q) = \frac{1}{2}\sum_i |p_i - q_i|$, we can know that $D_{TV}(p||q) = D_{TV}(q||p)$. Using Pinsker's inequality $D_{TV}(p||q)^2 \leq \frac{D_{KL}(p||q)}{2}$ [3], we can change the order of policy in the divergence computation and obtain:

$$J^i_j(\bar{\boldsymbol{\pi}}) \leq J^i_j(\boldsymbol{\pi}) + E_{s\sim\rho_{\boldsymbol{\pi}}, \boldsymbol{a}\sim\bar{\boldsymbol{\pi}}}[A^i_{j,\boldsymbol{\pi}}(s, a^i)] + \frac{2\gamma max_{s,a^i}|A^i_{j.\boldsymbol{\pi}}(s, a^i)|}{(1-\gamma)^2} D^{max}_{KL}(\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}). \tag{2}$$

It's to be noted that $E_{s\sim\rho_{\boldsymbol{\pi}}, \boldsymbol{a}\sim\bar{\boldsymbol{\pi}}}[A^i_{j,\boldsymbol{\pi}}(s, a^i)] = E_{s\sim\rho_{\boldsymbol{\pi}}, a^i\sim\bar{\pi}^i}[A^i_{j,\boldsymbol{\pi}}(s, a^i)]$ as the actions of other agents than i do not change the value of the variable inside of the expectation. Furthermore, $D^{max}_{KL}(\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}) = max_s D_{KL}(\bar{\boldsymbol{\pi}}(\cdot|s), \boldsymbol{\pi}(\cdot|s)) = max_s(\sum_{l=1}^n D_{KL}(\bar{\pi}^l(\cdot|s), \pi^l(\cdot|s))) \leq \sum_{l=1}^n max_s D_{KL}(\bar{\pi}^l(\cdot|s), \pi^l(\cdot|s))) = \sum_{l=1}^n D^{max}_{KL}(\bar{\pi}^l, \pi^l)$. Setting $\nu^i_j = \frac{2\gamma max_{s,a^i}|A^i_{j,\boldsymbol{\pi}}(s,a^i)|}{(1-\gamma)^2}$, we can finally obtain:

$$J^i_j(\bar{\boldsymbol{\pi}}) \leq J^i_j(\boldsymbol{\pi}) + L^i_{j,\boldsymbol{\pi}}(\bar{\pi}^i) + \nu^i_j \sum_{l=1}^n D^{max}_{KL}(\bar{\pi}^l, \pi^l) \tag{3}$$

The aforementioned equation is similar to Lemma 2 in MACPO, with the only distinction being the order of policies in the Kullback-Leibler (KL) divergence term. However, this variation does not impact the subsequent derivations. To this end, we can establish the ultimate optimization problem presented in our work as follows:

$$\underset{\pi^{i_h}_\theta}{maximize}\, E_{s\sim\rho_{\boldsymbol{\pi}_{\boldsymbol{\theta_k}}}, a^{i_{1:h-1}}\sim\pi^{i_{1:h-1}}_{\theta_{k+1}}, a^{i_h}\sim\pi^{i_h}}[A^{i_h}_{\boldsymbol{\pi}_{\boldsymbol{\theta_k}}}(s, a^{i_{1:h-1}}, a^{i_h})] \tag{4}$$

$$s.t. J^{i_h}_j(\boldsymbol{\pi}_{\boldsymbol{\theta_k}}) + E_{s\sim\rho_{\boldsymbol{\pi}_{\boldsymbol{\theta_k}}}, a^{i_h}\sim\pi^{i_h}}[A^{i_h}_{j,\boldsymbol{\pi}_{\boldsymbol{\theta_k}}}(s, a^{i_h})] \leq c^{i_h}_j, \forall j \in 1, \cdots, m^{i_h} \tag{5}$$

$$\bar{D}_{KL}(\pi^{i_h}_\theta, \pi^{i_h}_{\theta_k}) \leq \delta. \tag{6}$$

where $\bar{D}_{KL}(\pi^{i_h}_\theta, \pi^{i_h}_{\theta_k}) \triangleq E_{s\sim\rho_{\boldsymbol{\pi}_{\boldsymbol{\theta_k}}}}[D_{KL}(\pi^{i_h}_{\theta_k}(\cdot|s), \pi^{i_h}_\theta(\cdot|s))]$.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

Appendix B Proof of Theorem 1

We first demonstrate the optimization problem to be solved when finding optimization problem within nonparameterized policy space:

$$\underset{\pi^{i_h}}{maximize}\, E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_{1:h-1}}\sim\pi_{\theta_{k+1}}^{i_{1:h-1}},a^{i_h}\sim\pi i_h}[A^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_{1:h-1}},a^{i_h})] \tag{7}$$

$$s.t. J_j^{i_h}(\boldsymbol{\pi_{\theta_k}}) + E_{s\sim\rho_{\pi_{\theta_k}},a^{i_h}\sim\pi^{i_h}}[A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h})] \leq c_j^{i_h}, \forall j \in 1,\cdots,m^{i_h} \tag{8}$$

$$\bar{D}_{KL}(\pi^{i_h},\pi^{i_h}_{\theta_k}) \leq \delta \tag{9}$$

Proof. We initiate our analysis by demonstrating the convexity of Problem (7-9) is convex *w.r.t.* π^{i_h}. Because $\boldsymbol{\pi_{\theta_k}}$ and $\theta_{k+1}^{i_{1:h-1}}$ is given, it can be noted that the objective function is linear *w.r.t.* π^{i_h}. Since $J_j^{i_h}(\boldsymbol{\pi_{\theta_k}})$ remains constant *w.r.t.* π^{i_h}, constraint 8 is also linear. Concerning constraint 9, it can be rewritten as $\sum_s \rho_{\boldsymbol{\pi_{\theta_k}}}(s) D_{KL}(\pi^{i_h},\pi^{i_h}_{\theta_k})[s] \leq \delta$. Notably, KL divergence is convex *w.r.t.* its first argument, hence constraint 9 can be represented as a linear combination of convex functions, confirming its convexity as well. As $\pi^{i_h}_{\theta_k}$ fulfills constraint 8 and serves as an interior point within the set defined by constraint 9, therefore Slater's constraint qualification holds and strong duality holds.

Based on above discussion, we can solve for the optimal value for the problem (7 - 9) p^* by solving the corresponding dual problem. We define $b_j^{i_h} = c_j^{i_h} - J_j^{i_h}(\boldsymbol{\pi_{\theta_k}})$, then

$$\begin{aligned} L(\pi,\lambda_j,\nu_j) = \lambda_j\delta + \nu_j b_j^{i_h} + E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}}}[E_{a^{i_{1:h-1}}\sim\pi_{\theta_{k+1}}^{i_{1:h-1}},a^{i_h}\sim\pi^{i_h}}[A^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_{1:h-1}},a^{i_h})] \\ - \nu_j E_{a^{i_h}\sim\pi^{i_h}}[A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h})] - \lambda_j D_{KL}(\pi^{i_h}||\pi^{i_h}_{\theta_k})] \end{aligned} \tag{10}$$

Therefore,

$$p^* = \max_{\pi_{i_h}\in\Pi}\min_{\lambda_j,\nu_j\geq 0} L(\pi,\lambda_j,\nu_j) = \min_{\lambda_j,\nu_j\geq 0}\max_{\pi_{i_h}\in\Pi} L(\pi,\lambda_j,\nu_j) \tag{11}$$

where we invoked strong duality in the second equality. According to the theory of convex optimization [4], if $\pi^{i_h *},\lambda_j^*,\nu_j^*$ are optimal for 11, $\pi^{i_h *}$ is also optimal for Problem 7-9.

Consider the inner maximization problem in 11, we can decompose this problem into separate problems, one for each s.

$$\begin{aligned} \underset{\pi_{i_h}}{maximize} E_{a^{i_h}\sim\pi^{i_h}}[E_{a^{i_{1:h-1}}\sim\pi_{\theta_{k+1}}^{i_{1:h-1}}}[A^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_{1:h-1}},a^{i_h})] - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) \\ - \lambda_j(log\pi^{i_h}(a|s) - log\pi^{i_h}_{\theta_k}(a|s))], \sum\pi^{i_h}(a|s) = 1 \end{aligned} \tag{12}$$

As $E_{a^{i_{1:h-1}}\sim\pi_{\theta_{k+1}}^{i_{1:h-1}}}[A^{i_h}_{\pi_{\theta_k}}(s,a^{i_{1:h-1}},a^{i_h})]$ is irrelevant to π^{i_h}, we rename this term as $\eta^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h})$ for simplicity. This is clearly a convex optimization problem which can be solved using a simple Lagrangian argument. We can then get

$$G(\pi^{i_h}) = \sum_a \pi^{i_h}(a|s)[\eta^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \lambda_j(log\pi^{i_h}(a|s) - log\pi^{i_h}_{\theta_k}(a|s)) + \zeta] - \zeta \tag{13}$$

where ζ is the Lagrange multiplier associated with the constraint $\sum\pi^{i_h}(a|s) = 1$. Differentiating $G(\pi)$ w.r.t for some a:

$$\frac{\partial G}{\partial\pi^{i_h}(a|s)} = \eta^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \lambda_j(log\pi^{i_h}(a|s) - log\pi^{i_h}_{\theta_k}(a|s)) + \zeta \tag{14}$$

Set 14 to 0 and similar to FOCOPS, we can know

$$\pi^{i_h *}(a|s) = \frac{\pi^{i_h}_{\theta_k}(a|s)}{Z_{\lambda_j,\nu_j}(s)} exp\{\frac{1}{\lambda_j}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}))\} \tag{15}$$

where $Z_{\lambda_j,\nu_j}(s)$ is the partition function that ensures π^{i_h*} to be a probability function, *i.e.,* $\sum_a \pi^{i_h*}(a|s) = 1$. Putting this π^* back into equation 11, we can get

$$\begin{aligned}
p^* &= \min_{\lambda_j,\nu_j \geq 0} \lambda_j \delta + \nu_j b_j^{i_h} + E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}}, a^{i_h}\sim\pi^{i_h*}}[\eta^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \lambda_j(log\pi^{i_h*}(a|s) - log\pi^{i_h}_{\theta_k}(a|s))] \\
&= \min_{\lambda_j,\nu_j \geq 0} \lambda_j \delta + \nu_j b_j^{i_h} + E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}}, a^{i_h}\sim\pi^{i_h*}}[\eta^{i_h}_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \lambda_j(log\pi^{i_h}_{\theta_k}(a|s) - logZ_{\lambda_j,\nu_j} \\
&+ \frac{1}{\lambda_j}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h})) - log\pi^{i_h}_{\theta_k}(a|s))] \\
&= \min_{\lambda_j,\nu_j \geq 0} \lambda_j \delta + \nu_j b_j^{i_h} + \lambda_j E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}}, a^{i_h}\sim\pi^{i_h*}}[logZ_{\lambda_j,\nu_j}(s)]
\end{aligned}$$

What's more, we give a simple description to show that for feasible policy $\boldsymbol{\pi_{\theta_k}}$, the optimal policy update π^{i_h*} has an upper bound for worst-case guarantee for cost constraint satisfaction. For agent i_h, according to Equation 3, after getting the optimal joint update policy for all agents, $J_j^i(\boldsymbol{\pi^*}) \leq J_j^i(\boldsymbol{\pi_{\theta_k}}) + L^i_{j,\boldsymbol{\pi_{\theta_k}}}(\pi^{i_h*}) + \nu_j^{i_h}\sum_{l=1}^n D_{KL}^{max}(\pi^{l*}, \pi^l_{\theta_k})$ can be obtained. According to the definition of $L^i_{j,\boldsymbol{\pi}}(\bar{\pi}^i)$ and the constraint 5 in the optimization problem, we can know that $J_j^i(\boldsymbol{\pi_{\theta_k}}) + L^i_{j,\boldsymbol{\pi_{\theta_k}}}(\pi^{i_h*}) \leq c_j^{i_h}$, thus leading to $J_j^i(\boldsymbol{\pi^*}) \leq c_j^{i_h} + \nu_j^{i_h}\sum_{l=1}^n D_{KL}^{max}(\pi^{l*}, \pi^l_{\theta_k})$. In addition, we can know that the kl divergence between update policy and π_{θ_k} for each agent l has an upper bound, which we call δ^l. To this end, we achieve $J_j^i(\boldsymbol{\pi^*}) \leq c_j^{i_h} + \frac{2\gamma max_{s,a^i}|A^i_{j,\boldsymbol{\pi}}(s,a^i)|}{(1-\gamma)^2}\sum_{l=1}^n \delta^l$, which is the upper bound for worst-case guarantee for cost constraint satisfaction. According to the result, we can know that with more agents, the upper bound for worst-case guarantee is higher, which means that optimization for more agents is more challenging, consistent with our intuition.

Appendix C Proof of Corollary 1

Corollary 1. The gradient of $L(\theta)$ takes the form

$$\nabla_\theta L(\theta) = E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}}}[\nabla_\theta D_{KL}(\pi^{i_h}_\theta || \pi^{i_h*})[s]] \tag{16}$$

where

$$\nabla_\theta D_{KL}(\pi^{i_h}_\theta || \pi^{i_h*})[s] = \nabla_\theta D_{KL}(\pi^{i_h}_\theta || \pi^{i_h}_{\theta_k}) - \frac{1}{\lambda_j} E_{a\sim\pi^{i_h}_{\theta_k}}[\frac{\nabla_\theta \pi^{i_h}_\theta(a|s)}{\pi^{i_h}_{\theta_k}(a|s)}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}))] \tag{17}$$

Proof. Using the definition of KL divergence, we note that

$$D_{KL}(\pi^{i_h}_\theta || \pi^{i_h*}) = -\sum_a \pi^{i_h}_\theta(a|s) log\pi^{i_h*}(a|s) + \sum_a \pi^{i_h}_\theta(a|s) log\pi^{i_h}_\theta(a|s) = H(\pi^{i_h}_\theta, \pi^{i_h*})[s] - H(\pi^{i_h}_\theta)[s] \tag{18}$$

where $H(\pi^{i_h}_\theta)[s]$ is the entropy and $H(\pi^{i_h}_\theta, \pi^{i_h*})[s]$ is the cross-entropy. We expand the cross-entropy term which gives us:

$$\begin{aligned}
H(\pi^{i_h}_\theta, \pi^{i_h*})[s] &= -\sum_a \pi^{i_h}_\theta(a|s) log\pi^{i_h*}(a|s) \\
&= -\sum_a \pi^{i_h}_\theta(a|s) * log(\frac{\pi^{i_h}_{\theta_k}(a|s)}{Z_{\lambda_j,\nu_j}} exp\{\frac{1}{\lambda_j}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}))\}) \\
&= -\sum_a \pi^{i_h}_\theta(a|s) * log\pi^{i_h}_{\theta_k}(a|s) + logZ_{\lambda_j,\nu_j}(s) - \frac{1}{\lambda_j}\sum_a \pi^{i_h}_\theta(a|s) * (\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A^{i_h}_{j,\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}))
\end{aligned}$$

Then put this term back into Equation18:

$$D_{KL}(\pi_\theta^{i_h}||\pi^{i_h*})[s] = -\sum_a \pi_\theta^{i_h}(a|s) * log\pi_{\theta_k}^{i_h}(a|s) + \sum_a \pi_\theta^{i_h}(a|s) log\pi_\theta^{i_h}(a|s) + logZ_{\lambda_j,\nu_j}(s)$$
$$- \frac{1}{\lambda_j}\sum_a \pi_\theta^{i_h}(a|s) * (\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))$$
$$= D_{KL}(\pi_\theta^{i_h}||\pi_{\theta_k}^{i_h}) + logZ_{\lambda_j,\nu_j}(s) - \frac{1}{\lambda_j}E_{a\sim\pi_{\theta_k}^{i_h}}[\frac{\pi_\theta^{i_h}(a|s)}{\pi_{\theta_k}^{i_h}(a|s)}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))]$$

In this way, take the gradient on both sides and we can get:

$$\nabla_\theta D_{KL}(\pi_\theta^{i_h}||\pi^{i_h*})[s] = \nabla_\theta D_{KL}(\pi_\theta^{i_h}||\pi_{\theta_k}^{i_h}) - \frac{1}{\lambda_j}E_{a\sim\pi_{\theta_k}^{i_h}}[\frac{\nabla_\theta\pi_\theta^{i_h}(a|s)}{\pi_{\theta_k}^{i_h}(a|s)}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))] \tag{19}$$

Appendix D Proof of Corollary 2

Corollary 2. The derivative of $L(\pi^{i_h*},\lambda_j,\nu_j)$ *w.r.t* ν_j is

$$\frac{\partial L(\pi^{i_h*},\lambda_j,\nu_j)}{\partial\nu_j} = b_j^{i_h} - E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h*}(a|s)}[A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})] \tag{20}$$

Proof. From the definition of $L(\pi^{i_h*},\lambda_j,\nu_j)$ and above discussion, we can know that

$$L(\pi^{i_h*},\lambda_j,\nu_j) = \min_{\lambda_j,\nu_j\geq 0}\lambda_j\delta + \nu_j b_j^{i_h} + \lambda_j E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h*}}[logZ_{\lambda_j,\nu_j}(s)] \tag{21}$$

The first two terms is an affine function for ν_j we focus on the expectation in the last term.

$$\frac{\partial\pi^{i_h*}(a|s)}{\partial\nu_j} = \frac{\pi_{\theta_k}^{i_h}(a|s)}{Z_{\lambda_j,\nu_j}^2(s)}[Z_{\lambda_j,\nu_j}(s) * \frac{\partial exp(\frac{1}{\lambda_j}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})))}{\partial\nu_j}$$
$$- exp(\frac{1}{\lambda_j}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))) * \frac{\partial Z_{\lambda_j,\nu_j}(s)}{\partial\nu_j}]$$

For simplicity, we record $exp(\frac{1}{\lambda_j}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})))$ as $e(x)$, so $\pi^{i_h*}(a|s) = \frac{\pi_{\theta_k}^{i_h}(a|s)}{Z_{\lambda_j,\nu_j}(s)} * e(x)$. In this way,

$$\begin{aligned}\frac{\partial\pi^{i_h*}(a|s)}{\partial\nu_j} &= \frac{\pi_{\theta_k}^{i_h}(a|s)}{Z_{\lambda_j,\nu_j}^2(s)}[-\frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})}{\lambda_j}Z_{\lambda_j,\nu_j}(s)e(x) - e(x)\frac{\partial Z_{\lambda_j,\nu_j}(s)}{\partial\nu_j}] \\ &= -\frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})}{\lambda_j}\pi^{i_h*}(a|s) - \pi^{i_h*}(a|s)\frac{\partial logZ_{\lambda_j,\nu_j}(s)}{\partial\nu_j}\end{aligned} \tag{22}$$

Therefore, the derivative of the expectation in the last term of $L(\pi^{i_h*},\lambda_j,\nu_j)$ can be written as:

$$\frac{\partial}{\partial\nu_j}E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h*}}[logZ_{\lambda_j,\nu_j}(s)] = E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi_{\theta_k}^{i_h}}[\frac{\partial}{\partial\nu_j}(\frac{\pi^{i_h*(a|s)}}{\pi_{\theta_k}^{i_h}(a|s)}logZ_{\lambda_j,\nu_j}(s))]$$
$$= E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi_{\theta_k}^{i_h}}[\frac{1}{\pi_{\theta_k}^{i_h}(a|s)}(\frac{\partial\pi^{i_h*}(a|s)}{\partial\nu_j}logZ_{\lambda_j,\nu_j}(s) + \pi^{i_h*}(a|s)\frac{\partial logZ_{\lambda_j,\nu_j}(s)}{\partial\nu_j})]$$
$$= E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi_{\theta_k}^{i_h}}[\frac{\pi^{i_h*}(a|s)}{\pi_{\theta_k}^{i_h}(a|s)}(-\frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})}{\lambda_j}logZ_{\lambda_j,\nu_j}(s) - \frac{\partial logZ_{\lambda_j,\nu_j}(s)}{\partial\nu_j}logZ_{\lambda_j,\nu_j}(s)) + \frac{\partial logZ_{\lambda_j,\nu_j}(s)}{\partial\nu_j})]$$
$$= E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h*}(a|s)}[-\frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})}{\lambda_j}logZ_{\lambda_j,\nu_j}(s) - \frac{\partial logZ_{\lambda_j,\nu_j}(s)}{\partial\nu_j}logZ_{\lambda_j,\nu_j}(s)) + \frac{\partial logZ_{\lambda_j,\nu_j}(s)}{\partial\nu_j}]$$

In addition, according to the definition of Z_{λ_j,ν_j}, we can get:

$$\begin{aligned}\frac{\partial Z_{\lambda_j,\nu_j(s)}}{\partial \nu_j} &= \frac{\partial}{\partial \nu_j}(\sum_a \pi_{\theta_k}^{i_h}(a|s) exp\{\frac{1}{\lambda_j}(\eta_{\boldsymbol{\pi_{\theta_k}}}(s,a^{i_h}) - \nu_j A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))\} \\ &= -\sum_a \pi_{\theta_k}^{i_h}(a|s)\frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))}{\lambda_j} e(x) = -\sum_a \frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))}{\lambda_j}\frac{\pi_{\theta_k}^{i_h}(a|s)}{Z_{\lambda_j,\nu_j}(s)} e(x) Z_{\lambda_j,\nu_j}(s) \\ &= -\sum_a \frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))}{\lambda_j}\pi^{i_h *}(a|s) Z_{\lambda_j,\nu_j}(s) \\ &= -\frac{Z_{\lambda_j,\nu_j}(s)}{\lambda_j} E_{a^{i_h}\sim\pi^{i_h *}}[A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))]\end{aligned}$$

What's more,

$$\frac{\partial log Z_{\lambda_j,\nu_j}(s)}{\partial \nu_j} = \frac{\partial Z_{\lambda_j,\nu_j}(s)}{\partial \nu_j}\frac{1}{Z_{\lambda_j,\nu_j}(s)} = -\frac{1}{\lambda_j} E_{a^{i_h}\sim\pi^{i_h *}}[A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h}))] \tag{23}$$

Putting this result to above equation, we can get

$$\begin{aligned}&\frac{\partial}{\partial \nu_j} E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h *}}[log Z_{\lambda_j,\nu_j}(s)] \\ &= E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h *}(a|s)}[-\frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})}{\lambda_j} log Z_{\lambda_j,\nu_j}(s) + \frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})}{\lambda_j} log Z_{\lambda_j,\nu_j}(s) - \frac{A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})}{\lambda_j}] \\ &= -\frac{1}{\lambda_j} E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h *}(a|s)}[A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})]\end{aligned}$$

To sum up, the derivative of ν_j to function $L(\pi^{i_h *},\lambda_j,\nu_j)$ can be written:

$$\frac{\partial L(\pi^{i_h *},\lambda_j,\nu_j)}{\partial \nu_j} = b_j^{i_h} - E_{s\sim\rho_{\boldsymbol{\pi_{\theta_k}}},a^{i_h}\sim\pi^{i_h *}(a|s)}[A_{j,\boldsymbol{\pi_{\theta_k}}}^{i_h}(s,a^{i_h})] \tag{24}$$

where $b_j^{i_h} = c_j^{i_h} - J_j^{i_h}(\boldsymbol{\pi_{\theta_k}})$. In this way, we can update ν_j by $\nu_j \leftarrow proj_{\nu_j}[\nu_j - \alpha(c_j^{i_h} - J_j^{i_h}(\boldsymbol{\pi_{\theta_k}}))]$

Appendix E Procedure of MAFOCOPS

In this section, we describe the procedure of our algorithm, outlined in Algorithm 1. To be noted, hyperparameters for each agent are identical throughout the algorithm.

Appendix F Experiment Environment Introduction

In this section, we introduce the environments that we adopt in the experiments.

F.1 Safe MAMuJoCo

This environment is an extension of MAMuJoCo [5], maintaining the background environment, agents, physics simulator, and the reward function. However, in the Safe MAMuJoCo setting, additional obstacles such as walls or pitfalls are introduced, and the environment emits cost with the increasing risk of an agent stumbling upon an obstacle. Here, we mainly introduce the scenarios that we employ in our work and present them in Figure 1.

ManyAgent Ant task & Ant task The corridor in the environment is bounded by two walls, with a width of 9 m for ManyAgent Ant and 10 m for Ant. The environment emits the cost of 1 for an agent, if the distance between the robot and the wall is less than 1.8 m, or when the robot topples over, which can be described as

$$c_t = \begin{cases} 1, & 0.2 \leq \boldsymbol{z}_{torso,t+1} \leq 1.0, z_{rot} > -0.7, ||\boldsymbol{x}_{torso,t+1} - \boldsymbol{x}_{wall}||_2 \geq 1.8 \\ 0, & otherwise \end{cases}, \tag{25}$$

Algorithm 1 MAFOCOPS

Require: number of agents n, number of updates K, minibatch size B, temperature $\{\lambda_j\}_{1\leq j\leq m^i}$, initial cost constraint parameter $\{\nu_j\}_{1\leq j\leq m^i}$, cost constraint parameter bound v_{max}, learning rate for cost constraint parameter α_ν, trust region bound δ, cost bound b_j

Initialize, policy networks $\{\pi^i_{\theta_0}, i \in \mathcal{N}\}$, global value network $\{\phi_0\}$ and cost value networks $\{\phi^i_{j,0}\}^{i\in\mathcal{N}}_{1\leq j\leq m^i}$, replay buffer $\mathcal{B}$
for $k = 0, 1, \ldots$ **do**
  Generate trajectories $\tau \sim \boldsymbol{\pi_{\theta_k}}$, save the data into the buffer and sample a batch of data;
  Estimate the C-returns $\hat{J}_C$ by averaging over the cost return for all episodes.
  Compute the advantage functions $\hat{A}_{\boldsymbol{\pi}_{\theta_k}}(s, \boldsymbol{a})$ and $\hat{A}^i_{j,\boldsymbol{\pi}_{\theta_k}}(s, a^i)$ using GAE;
  Draw a permutation $i_{1:n}$ of agents at random.
  Set $M^{i_1}(s, \boldsymbol{a}) = \hat{A}_{\boldsymbol{\pi}_{\theta_k}}(s, \boldsymbol{a})$
  for agent $i_h = i_1, i_2, \cdots, i_n$ **do**
    Update ν_j by ν_j by $\nu_j \leftarrow proj_{\nu_j}[\nu_j - \alpha(c^{i_h}_j - \hat{J}^{i_h}_{C,j}(\boldsymbol{\pi_{\theta_k}}))], \forall j = 1, \cdots, m^{i_h}$
    for K epochs **do**
      for each minibatch data of size B **do**
        Update value networks (and cost value networks analogously) by minimizing the MSE loss $\phi_{k+1} = argmin_\phi \sum^T_{t=0}(V_{\phi_k}(s_t) - \hat{R}_t)^2$, where $\hat{R}$ is the target return.
        Update policy network by the derived equation of $\nabla_\theta L(\theta)$, where $\hat{\eta}_{\boldsymbol{\pi}_{\boldsymbol{\theta_k}}}(s, a^{i_h})$ is estimated by $M^{i_{1:h}}(s, \boldsymbol{a})$.
      end for
      if $\bar{D}_{KL}(\pi^{i_h}, \pi^{i_h}_{\theta_k}) \leq \delta$ **then**
        Break
      end if
    end for
    Compute $M^{i_{1:h+1}}(s, \boldsymbol{a}) = \frac{\pi^{i_h}_{\theta^{i_h}_{k+1}}(a^{i_h}|o^{i_h})}{\pi^{i_h}_{\theta^{i_h}_{k}}(a^{i_h}|o^{i_h})} M^{i_{1:h}}(s, \boldsymbol{a})$, unless $h = n$
  end for
end for

where $\boldsymbol{z}_{torso,t+1}$ and $\boldsymbol{x}_{torso,t+1}$ is the robot's torso's z-coordinate and x-coordinate at time $t+1$, z_{rot} is the robot's rotation's z-coordinate and $\boldsymbol{x}_{wall}$ denotes the x-coordinate of the wall.

HalfCheetah task In these maps, the HalfCheetah agents move inside a corridor (which constraints their movement, but does not induce costs). Concurrently, there are pitfalls within the corridor that also move. When an agent is too close to a pitfall, specifically when the distance between an agent and a pitfall is less than 9 m, a cost of 1 will be emitted.

$$c_t = \begin{cases} 1, & ||\boldsymbol{y}_{torso,t+1} - \boldsymbol{y}_{obstacle}||_2 \geq 9 \\ 0, & otherwise \end{cases}, \tag{26}$$

where the y-coordinate of the robot's torso is represented by $\boldsymbol{y}_{torso,t+1}$ and $\boldsymbol{y}_{obstacle}$ denotes the y-coordinate of the moving obstacles.

F.2 Safe Multi-Agent Isaac Gym

This environment builds upon Issac Gym platform [6], renowned for its GPU-accelerated capabilities, and leverages the powerful Nvidia PhysX engine. Extending from the existing framework of DexterousHands [7], Safe MAIG requires agents to control the robot hands while optimizing both the reward and safety performance. Similarly, we also give an introduction of the specific scenarios in our experimental evaluations.

ShadowHandOver This task revolves around a dual-hand setup, with each hand occupying a fixed position. The primary objective entails the first hand, holding an object, navigating a suitable trajectory to transfer the item to the second hand while the second hand aims to acquire a successful grasp of the object. To be noted, this task incorporates safety constraints pertaining to the range of

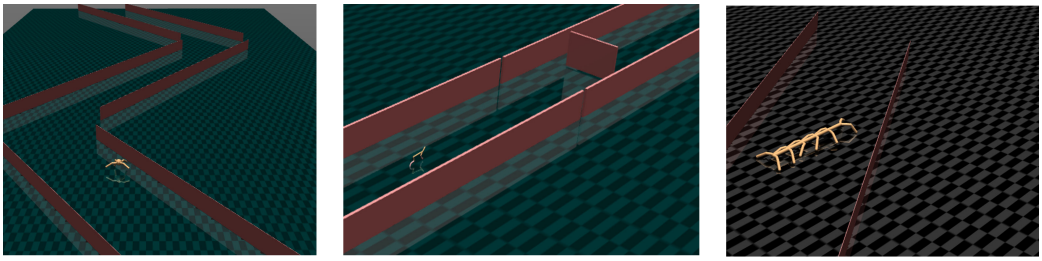

Figure 1: Specific tasks in Safe MAMuJoCo. (a): Ant Task: Ant 4x2 with three folding Jagged (30°) line walls, (b): HalfCheetah Task: HalfCheetah 2x3 with the moving obstacles, (c): ManyAgent Ant Task: ManyAgent Ant 2x3 inside one folding line walls (corridor width is 9 m).

motion of one of the fingers on the first hand. Formally, the cost function can be expressed as follows:

$$c_t = \begin{cases} 1, & ||\boldsymbol{F}_{a4,t+1}|| \geq 0.1 \\ 0, & otherwise \end{cases}, \tag{27}$$

where $\boldsymbol{F}_{a4,t+1}$ is the first hand's fourth fingers's motion degree.

ShadowHandReOrientation Within the context of this task, both hands are equipped with two items. The fundamental objective for the agents is to execute rotational movements between these two items around each other and the safety constraints remain the same as Equation 27.

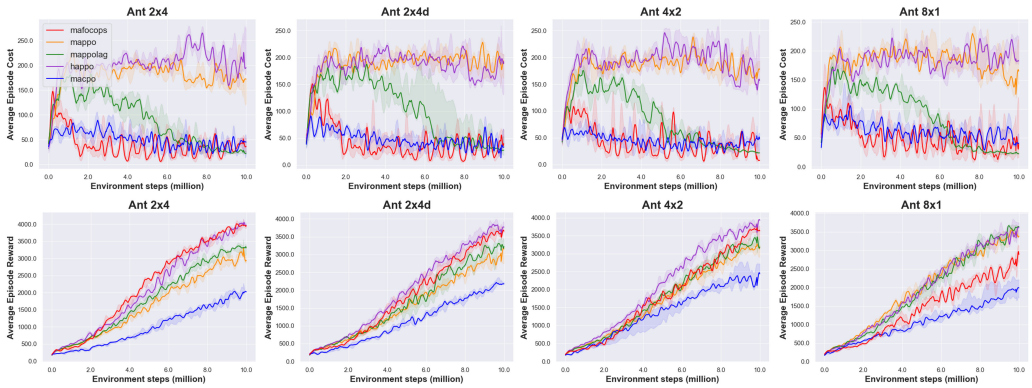

Figure 2: Performance comparisons on tasks of Ant 2x4, 2x4d, 4x2 and 8x1. The safety bound is 50, except for Ant 8x1 whose cost threshold is set as 70. The solid line shows the median performance across 5 seeds and the shaded areas correspond to the 25-75% percentiles.

Appendix G Performance on Safe MAMuJoCo

In this section, we present the comprehensive results of experiments in Safe MAMuJoCo environment in Figure 2-5. It can be observed that our proposed MAFOCOPS consistently demonstrates superior overall performance across all tasks. Even when our method achieves similar performance compared to the other two algorithms in HalfCheetah scenarios, it still exhibits faster learning, demonstrating the advantages of our approach. As is discussed in the Experiment section, MAPPO-L algorithm always achieves the similar performance as MAPPO, except in HalfCheetah scenarios where the cost threshold is significantly smaller compared to cost achieved by HAPPO and MAPPO. This may be due to that MAPPO-Lagrangian being built upon Lagrangian multiplier combined with standard MARL algorithms, leading to a performance more similar to safety-unaware MARL algorithms. Regarding other two hard constraint algorithm, their performance would degrade with the increasing number of agents. However, MAFOCOPS consistently outperforms MACPO, proving the effectiveness of our method. What's more, we provide additional videos of the trained policies of both our algorithm and MACPO.

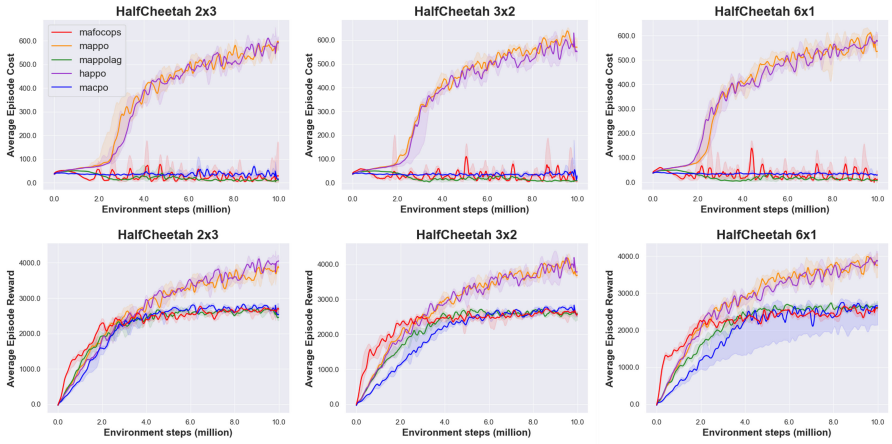

Figure 3: Performance comparisons on tasks of HalfCheetah 2x3, 3x2 and 6x1. The safety bound is 30. The solid line shows the median performance across 5 seeds and the shaded areas correspond to the 25-75% percentiles.

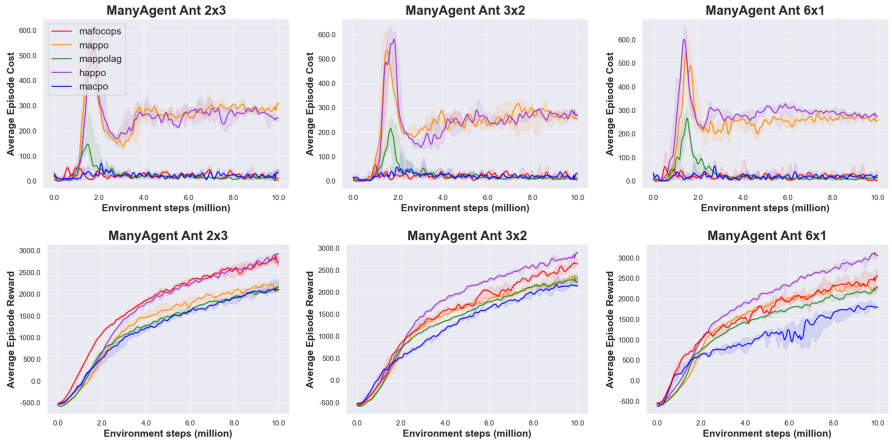

Figure 4: Performance comparisons on tasks of ManyAgent Ant 2x3, 3x2 and 6x1. The safety bound is 25. The solid line shows the median performance across 5 seeds and the shaded areas correspond to the 25-75% percentiles.

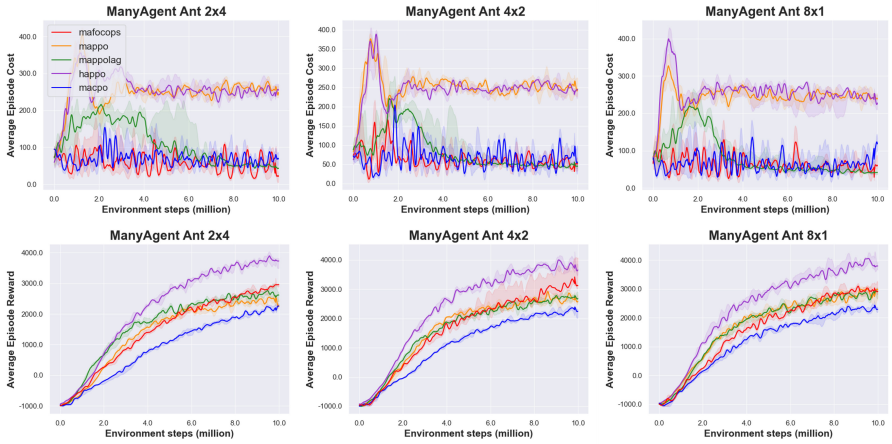

Figure 5: Performance comparisons on tasks of ManyAgent Ant 2x4, 4x2 and 8x1. The safety bound is 80. The solid line shows the median performance across 5 seeds and the shaded areas correspond to the 25-75% percentiles.

Appendix H Efficiency Analysis

In this section, we evaluate the training efficiency, which is measured by frame per second (FPS), and memory cost between MACPO and our MAFOCOPS. To be specific, we record the time and samples spent for each update to calculate average FPS and employ memory monitor tools to track memory utilization after 200000 samples. To ensure a fair comparison, both algorithms are executed on the same GPU device, thereby minimizing the influence of other variables. The results obtained from these evaluations are presented in Table 1 and Table 2 with a precision of two decimal places. Based on the obtained results, it is evident that an increase in the number of agents leads to a noticeable escalation in computational cost for MACPO. Whereas, our algorithm showcases substantial improvement in computational efficiency and demonstrates the ability to effectively conserve memory resources, especially in scenarios involving a larger number of agents.

Scenarios	Ant Task				HalfCheetah Task		
Config / FPS	2x4d	2x4	4x2	8x1	2x3	3x2	6x1
MACPO	231	218	130	73	298	192	106
MAFOCOPS	322	270	160	115	340	229	162
Improvement(%)	**39.39**	**23.85**	**23.08**	**57.53**	**14.09**	**19.27**	**52.83**
Scenarios	ManyAgent Ant Task						
Config / FPS	2x3	3x2	6x1	–	2x4	4x2	8x1
MACPO	244	167	98	–	232	135	73
MAFOCOPS	271	249	149	–	253	193	115
Improvement(%)	**11.07**	**49.10**	**52.04**	–	**9.05**	**42.96**	**57.53**

Table 1: Average FPS between MACPO and MAFOCOPS and the bold results demonstrate the improvement brings by our algorithm.

Scenarios	Ant Task				HalfCheeath Taks		
Config / Memory (MiB)	2x4d	2x4	4x2	8x1	2x3	3x2	6x1
MACPO	18.85	23.60	31.24	66.25	16.54	30.20	52.08
MAFOCOPS	18.97	21.82	24.23	56.99	19.34	27.26	39.15
Saved Memory	-0.12	**1.77**	**7.01**	**9.26**	-2.80	**2.93**	**12.93**
Scenarios	ManyAgent Ant Task						
Config / Memory (MiB)	2x3	3x2	6x1	–	2x4	4x2	8x1
MACPO	25.32	32.64	55.27	–	27.31	38.90	65.02
MAFOCOPS	24.45	30.88	44.38	–	24.62	34.73	60.71
Saved Memory	**0.87**	**1.76**	**10.89**	–	**2.69**	**4.17**	**4.31**

Table 2: Memory cost of MACPO and MAFOCOPS and the bold results demonstrate the memory saved by our algorithm.

Appendix I Sensitivity Analysis

We test the sensitivity of our algorithm to hyperparameters, *i.e.,* λ_j and ν_{max}, as well as the safety bound. To be noted, because the benchmarks that we adopt only involve a single cost, we only need to set one value for λ_j and ν_{max}. In future works, we may explore that the performance of our method in environments with multiple costs. We choose several scenarios in Safe MAMuJoCo to conduct the ablation studies.

The sensitivity to the hyperparameters are evaluated across several different values for λ_j and ν_{max} while keeping all other parameters fixed. For ease of comparison, we normalized the results based on the return and cost achieved by [8], namely if our method yields a return of x and HAPPO achieves a return of y, the normalized result is reported as $\frac{x}{y}$. The results report the final performance of the models after training for 10 million steps and are showcased in Table 3 and Table 4 with a precision of three decimal places. Given the complexity inherent in multi-agent environments, it is difficult to delineate the correlation between the performance of our method and the hyperparameters λ_j and ν_{max}. Nonetheless, it can be observed that our approach's effectiveness is relatively insensitive to variations in these hyperparameter values. Notably, even setting $\nu_{max} = \infty$ does not significantly affect the reward achieved by our method, only resulting in an average degradation of less than 10%.

	Ant 2x4		HalfCheetah 2x3		ManyAgent Ant 2x3		ManyAgent Ant 2x4		All envs	
λ	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
1	0.913	0.212	0.599	0.049	0.889	0.137	0.781	0.082	0.796	0.120
2	0.975	0.188	0.658	0.056	0.946	0.131	0.882	0.212	0.865	0.147
2.2	0.964	0.091	0.668	0.049	0.947	0.090	0.802	0.166	0.845	0.099
3	0.983	0.183	0.699	0.073	0.958	0.059	0.879	0.178	0.880	0.123
5	1.004	0.113	0.694	0.078	0.871	0.070	0.784	0.267	0.838	0.132

Table 3: Performance of MAFOCOPS for different λ and the "all envs" column presents the averaged performance across these four scenarios.

Furthermore, we select some maps to examine the sensitivity of our algorithm to the safety bound. To be mentioned, hyperparameters in this experiment keep unchanged. The results, as depicted in Figure 6, indicate that although the reward performance of MAFOCOPS diminishes as the safety constraints become more stringent, the algorithm's overall effectiveness remains consist across different safety levels.

	Ant 2x4		HalfCheetah 2x3		ManyAgent Ant 2x3		ManyAgent Ant 2x4		All envs	
ν_{max}	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
1	1.042	0.074	0.680	0.029	0.981	0.067	0.859	0.328	0.891	0.125
1.3	0.964	0.091	0.668	0.049	0.947	0.090	0.802	0.166	0.845	0.099
2	0.908	0.222	0.627	0.040	0.962	0.137	0.910	0.206	0.852	0.151
3	0.923	0.097	0.579	0.042	0.936	0.077	0.844	0.123	0.821	0.085
5	0.793	0.175	0.606	0.063	1.013	0.102	0.800	0.256	0.803	0.149
∞	0.873	0.209	0.588	0.048	0.959	0.100	0.749	0.146	0.792	0.126

Table 4: Performance of MAFOCOPS for different ν_{max} and the "all envs" column presents the averaged performance across these four scenarios.

Appendix J Details of Settings for Experiments

The majority of settings have been described in detail in the Experiments section; however, we provide some additional information here. As our implementation is based on codebase provided by MACPO [1], and thus most hyperparameters remain consistent with their original values For MAFOCOPS, the Lagrange multipliers, namely λ and ν_{max}, we utilize is 2.2 and 1.3, respectively, which can founded in Table 3 and 4. For other two safe MARL algorithms, MACPO and MAPPO-L, we modify the relevant hyperparameters to ensure their compatibility with the safety bound As is mentioned in the Experiments section, for the two benchmarks, we adopt distinct hyperparameters for MAPPO-L in different categories of tasks, as the safety bound is relative to the cost achieved by standard MARL algorithms. However, MAFOCOPS and MACPO both use unchanged parameters, indicating robustness of these two methods. We present the specific hyperparameters that we use in our experiments in Table 5 (as most parameters are unchanged, we only report the changed ones or unique parameters in our algorithm).

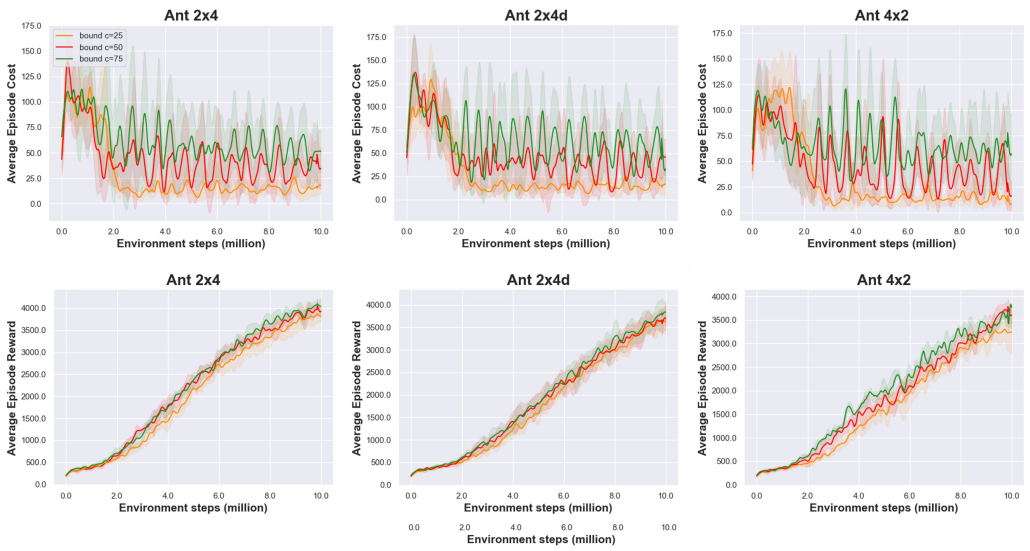

Figure 6: Performance comparisons on Ant 2x4, 2x4d, 4x2 with different safety bound.

Safe MAMuJoCo	MACPO	MAPPO-L	MAFOCOPS
kl-threshold	0.008	/	0.0125
lambda lagr	/	$[0.38^a, 0.46^b, 0.59^c, 0.52^d]$	/
λ	/	/	2.2
ν_{max}	/	/	1.3
ν lr	/	/	0.00005
fraction coef	0.3	/	/
minibatch size	/	/	256
update numbers	/	/	5
Safe MAIG	**MACPO**	**MAPPO-L**	**MAFOCOPS**
kl-threshold	0.009	/	0.01
lambda lagr	/	$[0.14^a, 0.68^b]$	/
lagrangian coef rate	/	$[1e-7^a, 9e-7^b]$	/
λ	/	/	2
ν_{max}	/	/	1.4
ν lr	/	/	0.001
fraction coef	0.26	/	/
minibatch size	/	/	8192
update numbers	/	/	3

Table 5: Different hyperparameters used for MACPO, MAPPO-L and MAFOCOPS. As MAPPO-L employs different hyperparameters, the changed ones are represented in the list. In Safe MAMuJoCo domains, a means Ant tasks, b corresponds to HalfCheetah tasks, c represents ManyAgent Ant 2x3 tasks and d represents denotes ManyAgent Ant 2x4 tasks. In Safe MAIG domains, a represents ShadowHandOver task and b denotes ShadowHandReOrientation task.

References

[1] S. Gu, J. G. Kuba, Y. Chen, Y. Du, L. Yang, A. Knoll, and Y. Yang, "Safe multi-agent reinforcement learning for multi-robot control," *Artificial Intelligence*, p. 103905, 2023.

[2] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[3] I. Csiszár and J. Körner, *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press, 2011.

[4] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[5] C. S. de Witt, B. Peng, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, "Deep multi-agent reinforcement learning for decentralized continuous cooperative control," *arXiv preprint arXiv:2003.06709*, vol. 19, 2020.

[6] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[7] Y. Chen, T. Wu, S. Wang, X. Feng, J. Jiang, Z. Lu, S. McAleer, H. Dong, S.-C. Zhu, and Y. Yang, "Towards human-level bimanual dexterous manipulation with reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5150–5163, 2022.

[8] J. G. Kuba, R. Chen, M. Wen, Y. Wen, F. Sun, J. Wang, and Y. Yang, "Trust region policy optimisation in multi-agent reinforcement learning," in *International Conference on Learning Representations*, 2022.