Appendices

A Proofs

A.1 Proof of Lemma 4.1

Proof. The Lagrangian function of (4) is as follows:

$$L = \sum_{a_i} \pi^i(a_i|s) Q_i^{\boldsymbol{\pi}_{\text{old}}}(s, a_i) - \omega \sum_{a_i} \pi^i_{\text{old}}(a_i|s) f\left(\frac{\pi^i(a_i|s)}{\pi^i_{\text{old}}(a_i|s)}\right) + \lambda_s \left(\sum_{a_i} \pi^i(a_i|s) - 1\right) + \sum_{a_i} \beta^i(a_i|s)\pi^i(a_i|s),$$

where λ_s and $\beta(a_i|s)$ are the Lagrangian multiplier.

Then by the KKT condition we have

$$\frac{\partial L}{\partial \pi^i(a_i|s)} = Q_i^{\boldsymbol{\pi}_{\text{old}}}(s, a_i) - \omega f'\left(\frac{\pi^i(a_i|s)}{\pi^i_{\text{old}}(a_i|s)}\right) + \lambda_s + \beta^i(a_i|s) = 0,$$

so we can resolve $\pi^i(a_i|s)$ as

$$\frac{\pi^i(a_i|s)}{\pi^i_{\text{old}}(a_i|s)} = g\left(\frac{Q_i^{\boldsymbol{\pi}_{\text{old}}}(s, a_i) + \lambda_s + \beta^i(a_i|s)}{\omega}\right) \tag{19}$$

From the complementary slackness we know that $\beta(a_i|s)\pi^i(a_i|s) = 0$, so we can rewrite (19) as

$$\frac{\pi^i(a_i|s)}{\pi^i_{\text{old}}(a_i|s)} = \max\left\{g\left(\frac{Q_i^{\boldsymbol{\pi}_{\text{old}}}(s, a_i) + \lambda_s}{\omega}\right), 0\right\}, \tag{20}$$

$$\pi^i(a_i|s) = \max\left\{\pi^i_{\text{old}}(a_i|s) g\left(\frac{Q_i^{\boldsymbol{\pi}_{\text{old}}}(s, a_i) + \lambda_s}{\omega}\right), 0\right\}. \tag{21}$$

□

A.2 Proof of Proposition 4.2

Proof. To discuss the monotonicity of the policies p_t and q_t, let $Q_t^A(0)$ and $Q_t^A(1)$ represent the expected reward Alice will obtain by taking action u_A^0 and u_A^1 respectively. Simlilarly, we can also define $Q_t^B(0)$ and $Q_t^B(1)$ for Bob.

From the definition, we have $Q_t^A(0) = q_t \cdot a + (1 - q_t) \cdot b = b + (a - b)q_t$. Similarly we can obtain that $Q_t^A(1) = d + (c - d)q_t$, $Q_t^B(0) = c + (a - c)p_t$ and $Q_t^B(1) = d + (b - d)p_t$.

Combining (21) with the condition $g(x) \geq 0$, then we have

$$p_{t+1} = p_t g\left(\frac{(a-b)q_t + b + \lambda_t^A}{\omega}\right),\ 1 - p_{t+1} = (1 - p_t) g\left(\frac{(c-d)q_t + d + \lambda_t^A}{\omega}\right)$$

$$\Rightarrow \frac{1}{p_{t+1}} - 1 = (\frac{1}{p_t} - 1)\frac{g\left(\frac{(c-d)q_t+d+\lambda_t^A}{\omega}\right)}{g\left(\frac{(a-b)q_t+b+\lambda_t^A}{\omega}\right)}. \tag{22}$$

From (22) we can find that

$$\begin{aligned} p_{t+1} \leq p_t \quad &\Leftrightarrow \quad \frac{g\left(\frac{(c-d)q_t+d+\lambda_t^A}{\omega}\right)}{g\left(\frac{(a-b)q_t+b+\lambda_t^A}{\omega}\right)} \geq 1 \\ &\Leftrightarrow \quad (c-d)q_t + d \geq (a-b)q_t + b \\ &\Leftrightarrow \quad (b + c - a - d)q_t \geq b - d \\ &\Leftrightarrow \quad q_t \geq \hat{q}. \end{aligned} \tag{23}$$

The critical step (23) is from the combination of the condition $g(x) \geq 0$ and the property $g(x)$ is non-decreasing.

Similarly we can obtain that $p_t \geq \hat{p} \Rightarrow q_{t+1} \leq q_t$; $p_t \leq \hat{p} \Rightarrow q_{t+1} \geq q_t$; $q_t \geq \hat{q} \Rightarrow p_{t+1} \leq p_t$; and $q_t \leq \hat{q} \Rightarrow p_{t+1} \geq p_t$. $\square$

A.3 Proof of Corollary 4.3

Proof. From the iteration of $\{p_t\}$ we have

$$\frac{p_{t+1}}{1-p_{t+1}} = \frac{p_t}{1-p_t} \frac{g\left(\frac{(a-b)q_t+b+\lambda_t^A}{\omega}\right)}{g\left(\frac{(c-d)q_t+d+\lambda_t^A}{\omega}\right)}. \tag{24}$$

Let $t \to \infty$ in both side of (24), we know that

$$\frac{p^*}{1-p^*}\left(\frac{g\left(\frac{(a-b)q^*+b+\lambda_*^A}{\omega}\right)}{g\left(\frac{(c-d)q^*+d+\lambda_*^A}{\omega}\right)} - 1\right) = 0. \tag{25}$$

As $q^* > \hat{q}$, we know that $\frac{g\left(\frac{(a-b)q^*+b+\lambda_*^A}{\omega}\right)}{g\left(\frac{(c-d)q^*+d+\lambda_*^A}{\omega}\right)} < 1$. So we can rewrite (25) as $\frac{p^*}{1-p^*} = 0$ and resolve $p^* = 0$.

As for q^*, we can follow a similar idea. From the iteration of $\{q_t\}$ we have

$$\frac{1}{q_{t+1}} - 1 = (\frac{1}{q_t} - 1)\frac{g\left(\frac{(b-d)p_t+d+\lambda_t^B}{\omega}\right)}{g\left(\frac{(a-c)p_t+c+\lambda_t^B}{\omega}\right)}. \tag{26}$$

Let $t \to \infty$ in both side of (26) , we know that

$$\frac{1-q^*}{q^*}\left(\frac{g\left(\frac{(b-d)p^*+d+\lambda_*^B}{\omega}\right)}{g\left(\frac{(a-c)p^*+c+\lambda_*^B}{\omega}\right)} - 1\right) = 0. \tag{27}$$

As $p^* < \hat{p}$, we know that $\frac{g\left(\frac{(b-d)p^*+d+\lambda_*^B}{\omega}\right)}{g\left(\frac{(a-c)p^*+c+\lambda_*^B}{\omega}\right)} < 1$. Then we can rewrite (27) as $\frac{1-q^*}{q^*} = 0$ and obtain $q^* = 1$. $\square$

A.4 Proof of Lemma 4.4

Proof. For any fixed i, consider the following difference

$$\left|\sum_{\boldsymbol{a}} \boldsymbol{\pi}_{\text{new}}(\boldsymbol{a}|s) Q^{\boldsymbol{\pi}}(s,\boldsymbol{a}) - \sum_{a_i} \pi^i_{\text{new}}(a_i|s) \sum_{a_{-i}} \pi^{-i}_{\text{old}}(a_{-i}|s) Q^{\boldsymbol{\pi}}(s,a_i,a_{-i})\right|$$

$$= \left|\sum_{a_i} \pi^i_{\text{new}}(a_i|s) \sum_{a_{-i}} \left(\pi^{-i}_{\text{new}}(a_{-i}|s) - \pi^{-i}_{\text{old}}(a_{-i}|s)\right) Q^{\boldsymbol{\pi}}(s,a_i,a_{-i})\right| \tag{28}$$

$$\leq \sum_{a_i} \pi^i_{\text{new}}(a_i|s) \sum_{a_{-i}} \left|\pi^{-i}_{\text{new}}(a_{-i}|s) - \pi^{-i}_{\text{old}}(a_{-i}|s)\right| \left|Q^{\boldsymbol{\pi}}(s,a_i,a_{-i})\right| \tag{29}$$

$$\leq \frac{M}{2} \sum_{a_i} \pi^i_{\text{new}}(a_i|s) \sum_{a_{-i}} \left|\pi^{-i}_{\text{new}}(a_{-i}|s) - \pi^{-i}_{\text{old}}(a_{-i}|s)\right| \tag{30}$$

$$= \frac{M}{2} \sum_{a_{-i}} \left|\pi^{-i}_{\text{new}}(a_{-i}|s) - \pi^{-i}_{\text{old}}(a_{-i}|s)\right| \tag{31}$$

$$= \frac{M}{2} \sum_{a_{-i}} \left|\sum_{k=1,k\neq i}^{N} \pi^{1:k-1}_{\text{new}}(a_{1:k-1}|s)\pi^{k:N}_{\text{old}}(a_{k:N}|s) - \pi^{1:k}_{\text{new}}(a_{1:k}|s)\pi^{k+1\sim N}_{\text{old}}(a_{k+1:N}|s)\right| \tag{32}$$

$$\leq \frac{M}{2} \sum_{a_{-i}} \sum_{k=1,k\neq i}^{N} \left|\pi^{1:k-1}_{\text{new}}(a_{1:k-1}|s)\pi^{k:N}_{\text{old}}(a_{k:N}|s) - \pi^{1:k}_{\text{new}}(a_{1:k}|s)\pi^{k+1\sim N}_{\text{old}}(a_{k+1:N}|s)\right| \tag{33}$$

$$= \frac{M}{2} \sum_{k=1,k\neq i}^{N} \sum_{a_k} \left|\pi^{k}_{\text{new}}(a_k|s) - \pi^{k}_{\text{old}}(a_k|s)\right| \tag{34}$$

$$= M \sum_{k=1,k\neq i}^{N} D_{\text{TV}}\left(\pi^k_{\text{new}}(\cdot|s) \| \pi^k_{\text{old}}(\cdot|s)\right) \tag{35}$$

where $\pi^{1:k-1}_{\text{new}}$ denotes $\pi^1_{\text{new}} \times \pi^2_{\text{new}} \times \cdots \pi^{k-1}_{\text{new}}$ and π^i_{new} will be skipped if involved, and $a_{1:k-1}$ has similar meanings as $a_{1:k-1} = a_1 \times a_2 \times \cdots a_{k-1}$. In (29) and (33), we use the triangle inequality of the absolute value. In (30), we use the property $Q^{\boldsymbol{\pi}}(s,\boldsymbol{a}) \leq \frac{r_{\max}}{1-\gamma} = \frac{M}{2}$ from the definition of Q-function. In (32), we insert $N-1$ terms between $\pi^{-i}_{\text{new}}(a_{-i}|s)$ and $\pi^{-i}_{\text{old}}(a_{-i}|s)$ to make sure the adjacent two terms are only different in one individual policy.

By rewriting the conclusion above, for any agent i, we have

$$\begin{aligned}\sum_{\boldsymbol{a}} \boldsymbol{\pi}_{\text{new}}(\boldsymbol{a}|s) Q^{\boldsymbol{\pi}}(s,\boldsymbol{a}) \geq & \sum_{a_i} \pi^i_{\text{new}}(a_i|s) \sum_{a_{-i}} \pi^{-i}_{\text{old}}(a_{-i}|s) Q^{\boldsymbol{\pi}}(s,a_i,a_{-i}) \\ & - M \sum_{k=1,k\neq i}^{N} D_{\text{TV}}\left(\pi^k_{\text{new}}(\cdot|s) \| \pi^k_{\text{old}}(\cdot|s)\right).\end{aligned} \tag{36}$$

Then, by applying (36) to $i = 1, 2, \cdots, N$ and add all these N inequalities together, we have

$$\begin{aligned}\sum_{\boldsymbol{a}} \boldsymbol{\pi}_{\text{new}}(\boldsymbol{a}|s) Q^{\boldsymbol{\pi}}(s,\boldsymbol{a}) \geq & \frac{1}{N}\sum_{i=1}^{N} \sum_{a_i} \pi^i_{\text{new}}(a_i|s) \sum_{a_{-i}} \pi^{-i}_{\text{old}}(a_{-i}|s) Q^{\boldsymbol{\pi}}(s,a_i,a_{-i}) \\ & - \frac{(N-1)M}{N} \sum_{i=1}^{N} D_{\text{TV}}\left(\pi^i_{\text{new}}(\cdot|s) \| \pi^i_{\text{old}}(\cdot|s)\right).\end{aligned}$$

□

A.5 Proof of Proposition 4.5

Proof. By the definition of $V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}$ we have

$$
\begin{aligned}
V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s) &= \frac{1}{N}\sum_i \sum_{a_i} \pi_{\text{old}}^i(a_i|s) \sum_{a_{-i}} \rho^{-i}(a_{-i}|s) Q_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s,a_i,a_{-i}) - \omega \sum_i D_f\left(\pi_{\text{old}}^i(\cdot|s) \| \rho^i(\cdot|s)\right) \\
&\leq \frac{1}{N}\sum_i \sum_{a_i} \pi_{\text{new}}^i(a_i|s) \sum_{a_{-i}} \rho^{-i}(a_{-i}|s) Q_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s,a_i,a_{-i}) - \omega \sum_i D_f\left(\pi_{\text{new}}^i(\cdot|s) \| \rho^i(\cdot|s)\right) \quad (37) \\
&= \frac{1}{N}\sum_i \sum_{a_i} \pi_{\text{new}}^i(a_i|s) \sum_{a_{-i}} \rho^{-i}(a_{-i}|s) \left(r(s,a_i,a_{-i}) + \gamma \mathbb{E}\left[V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s')\right]\right) \\
&\quad - \omega \sum_i D_f\left(\pi_{\text{new}}^i(\cdot|s) \| \rho^i(\cdot|s)\right) \quad (38) \\
&\leq \cdots \quad \text{(expand } V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s') \text{ and repeat replacing } \pi_{\text{old}}^i \text{ with } \pi_{\text{new}}^i \text{)} \quad (39) \\
&\leq V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{new}}}(s). \quad (40)
\end{aligned}
$$

In (37), we use the definition of π_{new}^i in (11). (38) is from the definition of $Q_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s,a_i,a_{-i})$. In (39), we repeatedly expand $V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}$ according to its definition and replace π_{old}^i with π_{new}^i by the optimality of π_{new}^i like what we have done in (37). After we replace all π_{old}^i with π_{new}^i, then we obtain $V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{new}}}(s)$ according to the definition of $V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{new}}}(s)$ in (40).

With the result $V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s) \leq V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{new}}}(s)$, we know $Q_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s,\boldsymbol{a}) = r(s,\boldsymbol{a}) + \gamma \mathbb{E}[V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{old}}}(s')] \leq r(s,\boldsymbol{a}) + \gamma \mathbb{E}[V_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{new}}}(s')] = Q_{\boldsymbol{\rho}}^{\boldsymbol{\pi}_{\text{new}}}(s,\boldsymbol{a})$. ◻

A.6 Proof of Theorem 4.6

Proof. From the Proposition 4.5, we know $V_{\boldsymbol{\pi}_t}^{\boldsymbol{\pi}_{t+1}}(s) \geq V^{\boldsymbol{\pi}_t}(s)$. Thus, we just need to prove $V^{\boldsymbol{\pi}_t}(s) \geq V_{\boldsymbol{\pi}_{t-1}}^{\boldsymbol{\pi}_t}(s)$.

From the definition of $V^{\boldsymbol{\pi}_t}(s)$ we have

$$
\begin{aligned}
V^{\boldsymbol{\pi}_t}(s) =& \sum_{\boldsymbol{a}} \boldsymbol{\pi}_t(\boldsymbol{a}|s) Q^{\boldsymbol{\pi}_t}(s,\boldsymbol{a}) \\
\geq& \frac{1}{N}\sum_{i=1}^N \sum_{a_i} \pi_t^i(a_i|s) \sum_{a_{-i}} \pi_{t-1}^{-i}(a_{-i}|s) Q^{\boldsymbol{\pi}_t}(s,a_i,a_{-i}) \\
&- \omega \sum_{i=1}^N D_{\text{TV}}\left(\pi_t^i(\cdot|s) \| \pi_{t-1}^i(\cdot|s)\right) \quad (41) \\
=& \frac{1}{N}\sum_{i=1}^N \sum_{a_i} \pi_t^i(a_i|s) \sum_{a_{-i}} \pi_{t-1}^{-i}(a_{-i}|s) \left(r(s,a_i,a_{-i}) + \gamma \mathbb{E}[V^{\boldsymbol{\pi}_t}(s')]\right) \\
&- \omega \sum_{i=1}^N D_{\text{TV}}\left(\pi_t^i(\cdot|s) \| \pi_{t-1}^i(\cdot|s)\right) \quad (42) \\
\geq& \cdots \quad \text{(expand } V^{\boldsymbol{\pi}_t}(s') \text{ and repeat replacing } \pi_t^{-i} \text{ with } \pi_{t-1}^{-i} \text{)} \quad (43) \\
\geq& V_{\boldsymbol{\pi}_{t-1}}^{\boldsymbol{\pi}_t}(s). \quad (44)
\end{aligned}
$$

(41) is from Lemma 4.4, and (42) is from the definition of $Q^{\boldsymbol{\pi}_t}(s,a_i,a_{-i})$. In (43), we repeatedly expand $V^{\boldsymbol{\pi}_t}$ and replace the π_t^{-i} with π_{t-1}^{-i} by Lemma 4.4 like what we have done in (41). After we replace all π_t^{-i} with π_{t-1}^{-i}, then we obtain $V_{\boldsymbol{\pi}_{t-1}}^{\boldsymbol{\pi}_t}(s)$ in (44) according to the definition of $V_{\boldsymbol{\pi}_{t-1}}^{\boldsymbol{\pi}_t}(s)$.

From the inequalities $V_{\boldsymbol{\pi}_t}^{\boldsymbol{\pi}_{t+1}}(s) \geq V^{\boldsymbol{\pi}_t}(s) \geq V_{\boldsymbol{\pi}_{t-1}}^{\boldsymbol{\pi}_t}(s) \geq V^{\boldsymbol{\pi}_{t-1}}(s)$, we know that the sequence $\{V^{\boldsymbol{\pi}_t}\}$ improves monotonically. Combining with the condition that the sequence $\{V^{\boldsymbol{\pi}_t}\}$ is bounded, we know that $\{V^{\boldsymbol{\pi}_t}\}$ will converge to V^*. According to the definition, the sequence $\{Q^{\boldsymbol{\pi}_t}\}$ and $\{\boldsymbol{\pi}_t\}$

will also converge to Q^* and $\boldsymbol{\pi}_*$ respectively, where $\boldsymbol{\pi}_*$ satisfies the following fixed-point equation:

$$\pi_*^i = \arg\max_{\pi^i} \sum_{a_i} \pi^i(a_i|s) \sum_{a_{-i}} \pi_*^{-i}(a_{-i}|s) Q^*(s, a_i, a_{-i}) - \omega D_{\mathrm{TV}}\left(\pi^i(\cdot|s) \| \pi_*^i(\cdot|s)\right).$$

□

A.7 Proof of $D_{\mathrm{TV}}(p\|q) \leq D_{\mathrm{H}}(p\|q)$

Proof.

$$\begin{aligned} D_{\mathrm{TV}}^2(p\|q) &= \frac{1}{4}\left(\sum_i |p_i - q_i|\right)^2 = \frac{1}{4}\left(\sum_i |\sqrt{p_i} - \sqrt{q_i}|\,|\sqrt{p_i} + \sqrt{q_i}|\right)^2 \\ &\leq \frac{1}{4}\left(\sum_i |\sqrt{p_i} - \sqrt{q_i}|^2\right)\left(\sum_i |\sqrt{p_i} + \sqrt{q_i}|^2\right) \quad \text{(Cauchy–Schwarz inequality)} \\ &= \frac{1}{4} D_{\mathrm{H}}^2(p\|q)\left(2 + 2\sum_i \sqrt{p_i q_i}\right) \\ &\leq D_{\mathrm{H}}^2(p\|q). \end{aligned}$$

□

B Experimental Settings

B.1 MPE

The three tasks are based on the original Multi-Agent Particle Environment (MPE) (Lowe et al., 2017) (MIT license) and were initially used in Agarwal et al. (2020) (MIT license). The objectives of these tasks are:

- **Simple Spread:** N agents must occupy the locations of N landmarks.
- **Line Control:** N agents must line up between two landmarks.
- **Circle Control:** N agents must form a circle around a landmark.

The reward in these tasks is the distance between all the agents and their target locations. We select these tasks to maintain consistency with DPO (Su & Lu, 2022b) but set the number of agents $N = 10$ for these three tasks in our experiment.

B.2 Multi-Agent MuJoCo

Multi-agent MuJoCo (Peng et al., 2021) (Apache-2.0 license) is a robotic locomotion task featuring continuous action space for multi-agent settings. The robot is divided into several parts, each containing multiple joints. Agents in this environment control different parts of the robot. The type of robot and the assignment of joints determine the task. For example, the task "HalfCheetah-3×2" means dividing the robot "HalfCheetah" into three parts, with each part containing two joints. Details of our experiment settings in multi-agent MuJoCo are listed in Table 2. The configuration specifies the number of agents and the joints assigned to each agent. "Agent obsk" defines the number of nearest agents an agent can observe.

B.3 StarCraft2

SMAC (Samvelyan et al., 2019) (MIT license) is a widely used environment for multi-agent reinforcement learning (MARL). In SMAC, agents receive rewards when they attack or kill an enemy unit. The rewards for an episode are normalized to a maximum of 20, regardless of the number of agents, to ensure consistency across tasks. An episode is considered won if the agents kill all enemy units. The observation space for agents depends on the number of units involved in the task.

Table 2: The task settings of multi-agent MuJoCo

task	configuration	agent obsk
HalfCheetah	3×2	2
Hopper	3×1	2
Walker2d	3×2	2
Ant	4×2	2

Typically, the observation is a vector with over 100 dimensions, containing information about all units. Information about units outside an agent's field of view is represented as zero in the observation vector. More details on SMAC can be found in the original paper (Samvelyan et al., 2019). SMACv2 (Ellis et al., 2023) (MIT license) is an advanced version of SMAC. Unlike SMAC, SMACv2 allows agents to control different types of units in different episodes, where the unit types are determined by a distribution and a type list. Moreover, the initial positions of agents are randomly selected in different episodes. With these properties, SMACv2 is more stochastic and difficult than SMAC. We keep the configuration the same as the original paper (Ellis et al., 2023) among the selected tasks.

C TRAINING DETAILS

Our code of IPPO is based on the open-source code[1] of MAPPO (Yu et al., 2021) (MIT license). The original IPPO and MAPPO is actually implemented as a CTDE method with parameter sharing and centralized critics. We modify the code for individual parameters and ban the tricks used by MAPPO for SMAC. The network architectures and base hyperparameters of TVPO, DPO and IPPO are the same for all the tasks in all the environments. We use 3-layer MLPs for the actor and the critic and use ReLU as non-linearities. The number of the hidden units of the MLP is 128. We train all the networks with an Adam optimizer. The learning rates of the actor and critic are both 5e-4. The number of epochs for every batch of samples is 15 which is the recommended value in Yu et al. (2021). For IPPO, the clip parameter is 0.2 which is the same as Schulman et al. (2017). For DPO, the hyperparameter is set as the original paper (Su & Lu, 2022b) recommends. Our code of IQL is based on the open-source code[2] PyMARL (Apache-2.0 license) and we modify the code for individual parameters. The default architecture in PyMARL is RNN so we just follow it and the number of the hidden units is 128. The learning rate of IQL is also 5e-4. The architectures of the actor and critic of IDDPG are 3-layer MLPs. The learning rates of the actor and critic are both 5e-4. Our code of I2Q is from the open source code[3] of the original paper (Jiang & Lu, 2022). We keep the hyperparameter of I2Q the same as the default value of the open-source code in our experiments.

Table 3: Hyperparameters for all the experiments

hyperparameter	value
MLP layers	3
hidden size	128
non-linear	ReLU
optimizer	Adam
actor_lr	5e-4
critic_lr	5e-4
numbers of epochs	15
initial β^i	0.01
δ	1.5
ω	2
d	0.001
clip parameter for IPPO	0.2

[1] https://github.com/marlbenchmark/on-policy
[2] https://github.com/oxwhirl/pymarl
[3] https://github.com/jiechuanjiang/I2Q

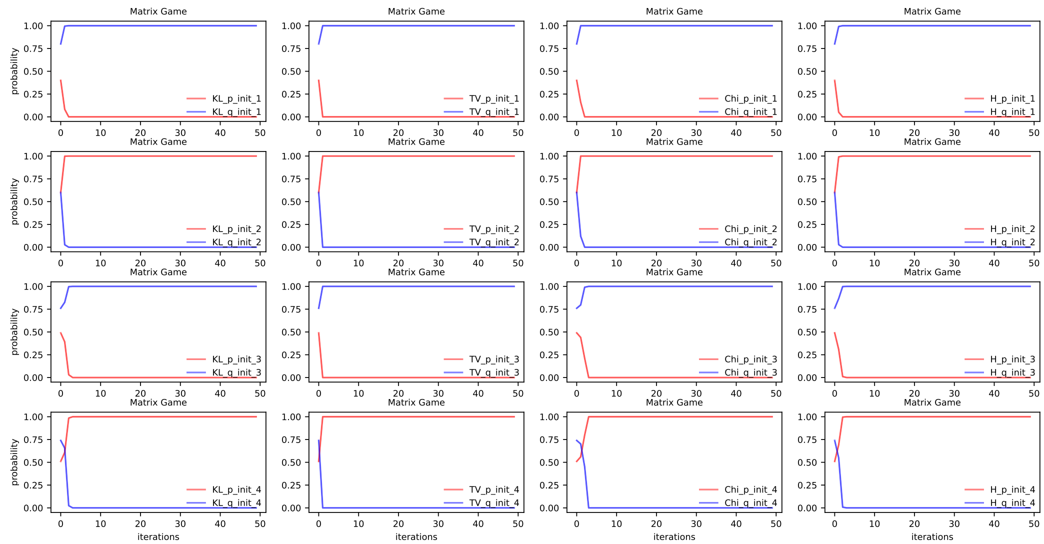

Figure 7: Learning curves of the policy p and q in the matrix game of KL-iteration, TV-iteration, χ^2-iteration, and H-iteration over four different sets of initialization. Each row corresponds to one set of initialization and each column corresponds to one type of iteration.

The version of the game StarCraft2 in SMAC is 4.10 for our experiments in all the SMAC tasks. We set the episode length of all the multi-agent MuJoCo tasks as 1000 in all of our multi-agent MuJoCo experiments. We perform the whole experiment with a total of four NVIDIA A100 GPUs. We have summarized the hyperparameters in Table 3.

D Algorithm

Algorithm 1. The practical algorithm of TVPO

for episode = 1 to M **do**
  for $t = 1$ to max_episode_length **do**
    select action $a_i \sim \pi^i(\cdot|s)$
    execute a_i and observe reward r and next state s'
    collect $\langle s, a_i, r, s' \rangle$
  end for
  Update the critic according to (17)
  Update the policy according to (15) or (18)
  Update β^i according to (16).
end for

E Additional Empirical Results

Figure 7 illustrates the learning curve of the policy p and q in the matrix game of KL-iteration, TV-iteration, χ^2-iteration, and H-iteration over four different sets of initialization. We can observe the policies of all four kinds of iterations converge.

MPE is a popular environment in cooperative MARL. MPE is a 2D environment and the objects are either agents or landmarks. Landmark is a part of the environment, while agents can move in any direction. With the relation between agents and landmarks, we can design different tasks. We use the discrete action space version of MPE and the agents can accelerate or decelerate in the direction of the x-axis or y-axis. We choose MPE for its partial observability.

The empirical results in MPE are illustrated in Figure 8. We find that TVPO obtains the best performance in all three tasks. In this environment, the policy-based algorithms, TVPO, DPO, and

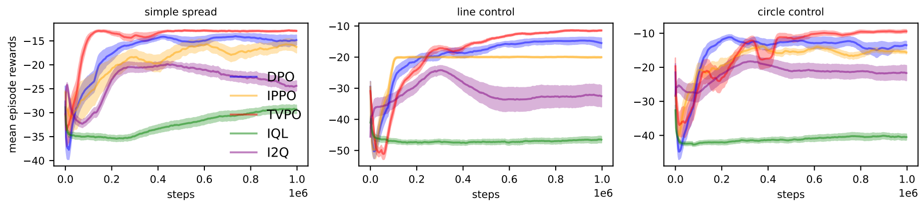

Figure 8: Learning curves of TVPO compared with IQL, IPPO, I2Q, and DPO in 10-agent simple spread, 10-agent line control, and 10-agent circle control in MPE.

IPPO, outperform the value-based algorithms, IQL and I2Q. I2Q has a better performance than IQL in all three tasks.

F Discussion

F.1 A Brief introduction of baseline algorithms

We select these four baseline algorithms as representatives of fully decentralized algorithms. IQL (Tan, 1993) is a basic value-based algorithm for decentralized learning. IPPO is a basic policy-based algorithm for decentralized learning. Both IQL and IPPO (de Witt et al., 2020) do not have convergence guarantees, to the best of our knowledge. DPO (Su & Lu, 2022b) and I2Q (Jiang & Lu, 2022) are the recent policy-based algorithm and value-based algorithm respectively, and both of them have been proved to have convergence guarantee.

IQL, IDDPG, and IPPO are relatively simple to understand, where each agent updates its policy through an independent Q-learning, DDPG, or PPO. These algorithms simply extend the single-agent RL algorithms into the MARL setting. They are heuristic algorithms without convergence guarantees in fully decentralized MARL.

The idea of DPO is to find a lower bound of the joint policy improvement objective as a surrogate which can also be optimized in a decentralized way for each agent. The formulation of DPO is as follows:

$$\pi^i_{t+1} == \arg\max_{\pi^i} \sum_{a_i} \pi^i(a_i|s) Q_i^{\boldsymbol{\pi}_t}(s,a_i) - \hat{M} \cdot \sqrt{D_{\mathrm{KL}}\left(\pi^i(\cdot|s) \| \pi^i_t(\cdot|s)\right)} - C \cdot D_{\mathrm{KL}}\left(\pi^i(\cdot|s) \| \pi^i_t(\cdot|s)\right).$$

DPO has been proven to improve monotonically and converge in fully decentralized MARL.

I2Q uses Q-learning from the perspective of QSS-value $Q_i(s, s')$. The QSS-value is updated with the following operator:

$\Gamma Q_i(s,s') = r + \gamma \max_{s'' \in \mathcal{N}(s')} Q_i(s', s'')$,

where $\mathcal{N}(s')$ is the neighbor set of state s'. In the deterministic environment and with some assumption about the transition probability, $Q_i(s, s')$ will converge to the same Q-function for each agent i, so the joint policy of agents will also converge in fully decentralized MARL.

F.2 Unary Formulation

Before proposing the f-divergence formulation, we have studied another formulation. This formulation follows the idea of entropy regularization and the extra term is only related to the policy π^i instead of the divergence between π^i and π^i_{old}. We refer to this approach as the unary formulation. Though we discovered that the unary formulation has more significant drawbacks, the properties of the unary formulation inspire us in the proof of TVPO. So we would like to provide the properties and some empirical results of the unary formulation here for discussion.

The unary formulation is

$$\pi^i_{\text{new}} = \arg\max_{\pi^i} \sum_{a_i} \pi^i(a_i|s) Q_i^{\boldsymbol{\pi}_{\text{old}}}(s,a_i) + \omega \sum_{a_i} \pi^i(a_i|s) \phi\left(\pi^i(a_i|s)\right). \tag{45}$$

This formulation (45) follows the idea of Yang et al. (2019) which discusses the regularization algorithm in single-agent RL. From the perspective of regularization, the update rule (45) can be seen as optimizing the regularized objective $J^i_\phi(\boldsymbol{\pi}) = \mathbb{E}\left[\sum_t \gamma^t \left(r_i(s,a_i) + \omega\phi\left(\pi^i(a_i|s)\right)\right)\right]$, where $r_i(s,a_i) = \mathbb{E}_{\pi^{-i}}\left[r(s,a_i,a_{-i})\right]$. The choice of ϕ is flexible, *e.g.*, $\phi(x) = -\log x$ corresponds to entropy regularization and independent SAC (Haarnoja et al., 2018); $\phi(x) = 0$ means (45) degenerates to independent Q-learning (Tan, 1993); Moreover, there are many other options for ϕ corresponding to different regularization (Yang et al., 2019). So we take (45) as the general unary formulation of independent learning, where the 'unary' means the additional terms $\sum_{a_i} \pi^i(a_i|s)\phi\left(\pi^i(a_i|s)\right)$ is only about one policy π^i.

For further discussion of (45) , we can utilize the conclusion in Yang et al. (2019) as the following lemma.

Lemma F.1. *If $\phi(x)$ in $(0,1]$ and satisfies the following conditions: (1) $\phi(x)$ is non-increasing; (2) $\phi(1) = 0$; (3) $\phi(x)$ is differentiable; (4) $f_\phi(x) = x\phi(x)$ is strictly concave, then we have that $g_\phi(x) = (f'_\phi)^{-1}(x)$ exists and $g_\phi(x)$ is decreasing. Moreover, the solution to the optimization objective (45) can be described with $g_\phi(x)$ as follows:*

$$\pi^i_{\text{new}}(a_i|s) = \max\{g_\phi\left(\frac{\lambda_s - Q_i^{\boldsymbol{\pi}_{\text{old}}}(s,a_i)}{\omega}\right), 0\}, \tag{46}$$

where λ_s satisfies $\sum_{a_i} \max\{g_\phi\left(\frac{\lambda_s - Q_i^{\boldsymbol{\pi}_{\text{old}}}(s,a_i)}{\omega}\right), 0\} = 0$.

Though it seems that $\phi(x)$ needs to satisfy four conditions, actually $\phi(x) = -\log x$ for Shannon entropy and $\phi(x) = \frac{k}{q-1}(1 - x^{q-1})$ for Tsallis entropy are still qualified.

However, unlike the single-agent setting, the update rule in Lemma F.1 may result in the convergence to sub-optimal policy or even oscillations in policy in fully decentralized MARL.

We further discuss (45) in the two-player matrix game and have the following proposition.

Proposition F.2. *Suppose that $g_\phi(x) \geq 0$ and $g_\phi(x)$ is continuously differentiable. If the payoff matrix of the two-player matrix game satisfies $b + c < a + d$, and two agents Alice and Bob update their policies with policy iteration as*

$$\pi^i_{t+1} = \arg\max_{\pi^i} \sum_{a_i} \pi^i(a_i|s) Q_i^{\boldsymbol{\pi}_t}(s,a_i) + \omega \sum_{a_i} \pi^i(a_i|s)\phi\left(\pi^i(a_i|s)\right), \tag{47}$$

then we have (1) $p_t > p_{t-1} \Rightarrow q_{t+1} > q_t$; (2) $p_t < p_{t-1} \Rightarrow q_{t+1} < q_t$; (3) $q_t > q_{t-1} \Rightarrow p_{t+1} > p_t$; (4) $q_t < q_{t-1} \Rightarrow p_{t+1} < p_t$.

Proof. To discuss the monotonicity of the policies p_t and q_t, we need the solution in Lemma F.1. Before applying the update rule (46), we need to calculate the decentralized critic given p_t and q_t. Let $Q_t^A(0)$ and $Q_t^A(1)$ represent the expected reward Alice will obtain by taking action u_A^0 and u_A^1 respectively. We can also define $Q_t^B(0)$ and $Q_t^B(1)$ for Bob.

From the definition, we have $Q_t^A(0) = q_t \cdot a + (1 - q_t) \cdot b = b + (a - b)q_t$. Similarly we could obtain that $Q_t^A(1) = d + (c - d)q_t$, $Q_t^B(0) = c + (a - c)p_t$ and $Q_t^B(1) = d + (b - d)p_t$.

With (46) and the condition $g_\phi(x) \geq 0$, we have

$$p_{t+1} = g_\phi\left(\frac{\lambda_t^A - Q_t^A(0)}{\omega}\right) = g_\phi\left(\frac{(b-a)q_t + \lambda_t^A - b}{\omega}\right),\ 1 - p_{t+1} = g_\phi\left(\frac{(d-c)q_t + \lambda_t^A - d}{\omega}\right)$$

$$g_\phi\left(\frac{(b-a)q_t + \lambda_t^A - b}{\omega}\right) + g_\phi\left(\frac{(d-c)q_t + \lambda_t^A - d}{\omega}\right) = 1$$

$$q_{t+1} = g_\phi\left(\frac{(c-a)p_t + \lambda_t^B - c}{\omega}\right),\ 1 - q_{t+1} = g_\phi\left(\frac{(d-b)p_t + \lambda_t^B - d}{\omega}\right)$$

$$g_\phi\left(\frac{(c-a)p_t + \lambda_t^B - c}{\omega}\right) + g_\phi\left(\frac{(d-b)p_t + \lambda_t^B - d}{\omega}\right) = 1.$$

We can rewrite these equations with some simplifications as follows,

$$
\begin{aligned}
&m_A(x) \triangleq \frac{(b-a)x+\lambda_A(x)-b}{\omega},\ n_A(x) \triangleq \frac{(d-c)x+\lambda_A(x)-d}{\omega},\ h_A(x)=g_\phi\left(m_A(x)\right)\\
&\text{where } \lambda_A(x) \text{ satisfies } g_\phi\left(m_A(x)\right)+g_\phi\left(n_A(x)\right)=1 \qquad (48)\\
&m_B(x) \triangleq \frac{(c-a)p_t+\lambda_B(x)-c}{\omega},\ n_B(x) \triangleq \frac{(d-b)p_t+\lambda_B(x)-d}{\omega}, h_B(x)=g_\phi\left(m_B(x)\right)\\
&\text{where } \lambda_B(x) \text{ satisfies } g_\phi\left(m_B(x)\right)+g_\phi\left(n_B(x)\right)=1.
\end{aligned}
$$

With these definitions, we know that $p_{t+1}=h_A(q_t)$, $q_{t+1}=h_B(p_t)$ and the monotonicity of p_t and q_t is determined by the property of function $h_A(x)$ and $h_B(x)$. By applying the chain rule to (48), we have:

$$
\begin{aligned}
&\frac{1}{\omega}g'_\phi\left(m_A(x)\right)\left(b-a+\lambda'_A(x)\right)+\frac{1}{\omega}g'_\phi\left(n_A(x)\right)\left(d-c+\lambda'_A(x)\right)=0\\
&\Rightarrow\ \lambda'_A(x)=-\frac{(b-a)g'_\phi(m_A(x))+(d-c)g'_\phi(n_A(x))}{g'_\phi(m_A(x))+g'_\phi(n_A(x))}. \qquad (49)
\end{aligned}
$$

Then we have:

$$
\begin{aligned}
h'_A(x)&=\frac{1}{\omega}g'_\phi\left(m_A(x)\right)\left(b-a+\lambda'_A(x)\right) \quad \text{(Apply chain rule)} \qquad (50)\\
&=\frac{1}{\omega}(b+c-a-d)\frac{g'_\phi(n_A(x))g'_\phi(m_A(x))}{g'_\phi(m_A(x))+g'_\phi(n_A(x))} \quad \text{(Substitute (49) for } \lambda'_A(x)\text{).} \qquad (51)
\end{aligned}
$$

Let $M=b+c-a-d$ and $M'=\frac{M}{\omega}$, then $h'_A(x)=M'\frac{g'_\phi(n_A(x))g'_\phi(m_A(x))}{g'_\phi(m_A(x))+g'_\phi(n_A(x))}$. From the condition and Lemma F.1 we know that $M'<0$ and $g_\phi(x)$ is decreasing which means $g'_\phi(x)<0$. Combining these conditions together, we know $h'_A(x)>0$ and $h_A(x)$ is increasing which means that $p_{t+1}=h_A(q_t)$ is increasing over q_t, which means that $q_t>q_{t-1}\ \Rightarrow\ p_{t+1}>p_t$ and $q_t>q_{t-1}\ \Rightarrow\ p_{t+1}>p_t$.

Similarly, we can obtain that $h'_B(x)=M'\frac{g'_\phi(n_B(x))g'_\phi(m_B(x))}{g'_\phi(m_B(x))+g'_\phi(n_B(x))}>0$ which could lead to the result that $p_t>p_{t-1}\ \Rightarrow\ q_{t+1}>q_t$ and $p_t<p_{t-1}\ \Rightarrow\ q_{t+1}<q_t$. ☐

Proposition F.2 actually tells us $p_{t+1}=h_A(q_t)$ is increasing over q_t and $q_{t+1}=h_B(p_t)$ is increasing over p_t when $M=b+c-a-d<0$. Intuitively, we can find two typical cases for policy iterations with Proposition F.2. In the first case, if in a certain iteration t the conditions $p_t>p_{t-1}$ and $q_t>q_{t-1}$ are satisfied, then we know that $p_{t'+1}>p_{t'}\quad q_{t'+1}>q_{t'}\quad \forall t'\geq t$. As the sequences $\{p_t\}$ and $\{q_t\}$ are both bounded in the interval $[0,1]$, we know that $\{p_t\}$ and $\{q_t\}$ will converge to p^* and q^* . The property of p^* and q^* is determined by $l_A(x)\triangleq h_B(h_A(x))$ and $l_B(x)\triangleq h_A(h_B(x))$ respectively as $p_{t+2}=h_B(h_A(p_t))$ and $q_{t+2}=h_A(h_B(q_t))$ and we have the following corollary.

Corollary F.3. $|l'_A(x)|\leq {M'}^2U_\phi^2$, $|l'_B(x)|\leq {M'}^2U_\phi^2$, *where U_ϕ is a constant determined by $\phi(x)$.*

Proof. As $g'_\phi(x)$ is continuous, let $U_A^1\triangleq\max_{x\in[0,1]}|g'_\phi(m_A(x))|$, $U_A^2\triangleq\max_{x\in[0,1]}|g'_\phi(n_A(x))|$, $U_B^1\triangleq\max_{x\in[0,1]}|g'_\phi(m_B(x))|$ and $U_B^2\triangleq\max_{x\in[0,1]}|g'_\phi(n_B(x))|$. Moreover, let $U_\phi=\max\{U_A^1,U_A^2,U_B^1,U_B^2\}$, then apply the chain rule to $l'_A(x)$ and we have

$$
\begin{aligned}
&|l'_A(x)|=|h'_B(h_A(x))h'_A(x)|\\
&={M'}^2\frac{|g'_\phi(n_B(h_A(x)))||g'_\phi(m_B(h_A(x)))|}{|g'_\phi(m_B(h_A(x)))|+|g'_\phi(n_B(h_A(x)))|}\frac{|g'_\phi(n_A(x))||g'_\phi(m_A(x))|}{|g'_\phi(m_A(x))|+|g'_\phi(n_A(x))|} \qquad (52)\\
&={M'}^2\frac{|g'_\phi(n_B(y))||g'_\phi(m_B(y))|}{|g'_\phi(m_B(y))|+|g'_\phi(n_B(y))|}\frac{|g'_\phi(n_A(x))||g'_\phi(m_A(x))|}{|g'_\phi(m_A(x))|+|g'_\phi(n_A(x))|} \quad \text{(Let } y=h_A(x)\in[0,1]\text{)}\\
&\leq {M'}^2\frac{|g'_\phi(m_B(y))|+|g'_\phi(n_B(y))|}{2}\frac{|g'_\phi(m_A(x))|+|g'_\phi(n_A(x))|}{2} \qquad (53)\\
&\leq {M'}^2U_\phi^2 \qquad (54)
\end{aligned}
$$

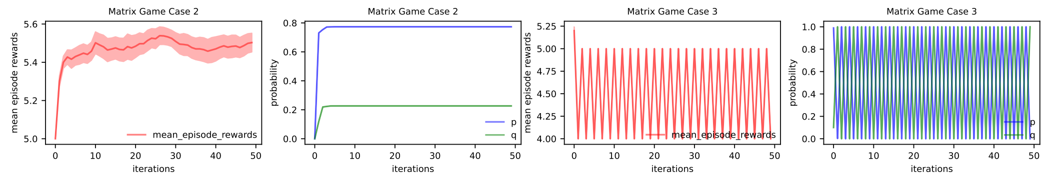

Figure 9: Learning curves of the unary formulation in two matrix game cases, where x-axis is iteration steps. The first and second figures show the performance and the policies p and q in the matrix game case 2 respectively. The third and fourth figures show the performance and the policies p and q in the matrix game case 3 respectively.

where (52) is from Proposition F.2, (53) is from the AM-GM inequality $ab \leq \frac{(a+b)^2}{2}$, and (54) is from the definition of U_ϕ. Similarly, we can obtain $|l'_B(x)| \leq {M'}^2 U_\phi^2$. $\square$

Combining Corollary F.3 and Banach fixed-point theorem, we can find that as U_ϕ is a constant, if $|M'| < \frac{1}{U_\phi}$, then we can find a constant L such that $|l'_A(x)| \leq {M'}^2 U_\phi^2 \leq L < 1$, which means that the iteration $p_{t+1} = l_A(p_t)$ is a contraction and p^* is the unique fixed-point of l_A. This conclusion can be seen as that a smaller $|M'|$ corresponds to a larger probability of convergence. In this convergence case, the converged policies p^* and q^* are usually not the optimal policy as the optimal policy is deterministic, which can be seen in our empirical results.

In the second case, which may be more general, in iteration t, $(p_t - p_{t-1})(q_t - q_{t-1}) < 0$, which means $p_t > p_{t-1}$ and $q_t < q_{t-1}$ or $p_t < p_{t-1}$ and $q_t > q_{t-1}$. Without loss of generality, we assume $p_t > p_{t-1}$ and $q_t < q_{t-1}$, then we know $p_{t+1} < p_t$ and $q_{t+1} < q_t$ from Proposition F.2. By induction we can find that for any $t' \geq t$, the sequence $\{p_{t'}\}$ and $\{q_{t'}\}$ will increase and decrease alternatively, which means that the policies may not converge but oscillate. We will show this in our experiments. As the unary formulation may result in policy oscillation, we would like to find other formulations for fully decentralized MARL.

F.3 Verification for Unary Formulation

In this section, we choose $\phi(x) = -\log x$ corresponding to the entropy regularization as the representation for the unary formulation. We build two cases to show the convergence to the sub-optimal policy and the policy oscillation. We choose $a = 5, b = 6, c = 3, d = 5$ as case 2 and $a = 7, b = 5, c = 4, d = 6$ as case 3. Both two cases satisfy the condition $b + c < a + d$ as discussed above. We keep $\omega = 0.1$ for all the experiments on these two matrix games. The empirical results are illustrated in Figure 9. We can find the policies p and q improve monotonically to the convergence $(p^*, q^*) \approx (0.773, 0.227)$ in case 2, which is a sub-optimal joint policy. However, in case 3, the policies p and q oscillate between 0 and 1 and do not converge. These results verify our discussion about the limitation of the unary formulation.

F.4 Non-Trivial Solution to Iteration (13)

In this section, we will build a two-player matrix game like Table 1 to show the non-trivial solution to iteration (13). In general, there is no closed-form solution to iteration (13). However, for the matrix game case, we can show some properties of iteration (13). With the same definitions as previous discussions, we can rewrite (13) in the matrix game as follows:

$$p_{t+1} = \arg\max_{p \in [0,1]} pQ_t^A(0) + (1-p)Q_t^A(1) - \omega|p - p_t|. \tag{55}$$

Let $f(p) = pQ_t^A(0) + (1-p)Q_t^A(1) - \omega|p - p_t|$, then $p_{t+1} = \arg\max_{p \in [0,1]} f(p)$.

We know that $f(p)$ is a linear function of p in both intervals $[0, p_t]$ and $[p_t, 1]$ and the maximums of linear function are always achieved in the endpoints of one interval. Thus, we have $p_{t+1} =$

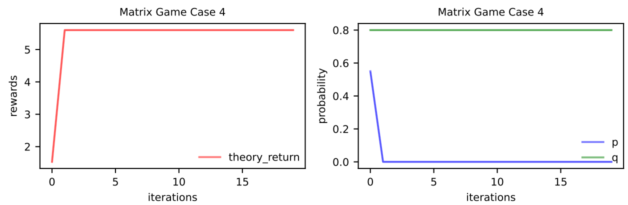

Figure 10: Learning curves of the iteration (13) in the matrix game $(a, b, c, d) = (-4, 7, 6, 4)$, where x-axis is iteration steps. The first and second figures show the expectation $J(\boldsymbol{\pi}_t)$ and the policies p and q in the matrix game case 4 respectively, where $J(\boldsymbol{\pi}_t)$ is calculated by the joint policy $\boldsymbol{\pi}_t = (p_t, q_t)$ and the payoff matrix.

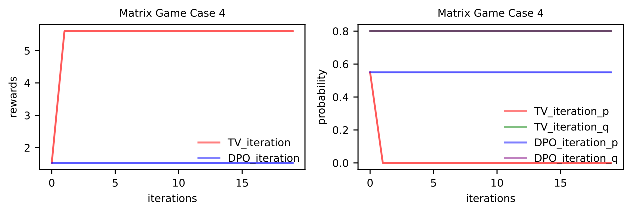

Figure 11: Learning curves of the iteration (13) and the DPO iteration in the matrix game $(a, b, c, d) = (-4, 7, 6, 4)$, where x-axis is iteration steps. The first and second figures show the expectation $J(\boldsymbol{\pi}_t)$ and the policies p and q of two iterations in the matrix game case 4 respectively, where $J(\boldsymbol{\pi}_t)$ is calculated by the joint policy $\boldsymbol{\pi}_t = (p_t, q_t)$ and the payoff matrix.

$\arg\max_{p \in \{0, p_t, 1\}} f(p)$, which means we only need to consider

$$
\begin{aligned}
f(0) &= Q_t^A(1) - \omega p_t \\
f(1) &= Q_t^A(0) - \omega(1 - p_t) \\
f(p_t) &= Q_t^A(1) + p_t(Q_t^A(0) - Q_t^A(1)).
\end{aligned}
$$

Next, we can build a matrix game with the property $b = \max\{a, b, c, d\} > c > d > 0 > a$. In this case, $M = 2\|Q\|_\infty = 2b$ and $\omega = \frac{(N-1)M}{N} = b$. Then we consider the condition $f(0) > f(p_t)$. We have

$$
\begin{aligned}
& f(0) - f(p_t) = -p_t \left(Q_t^A(0) - Q_t^A(1) + \omega\right) = -p_t \left(2b - d - (b + c - a - d)q_t\right) \\
& \Rightarrow f(0) > f(p_t) \quad \Leftrightarrow \quad q_t > \frac{2b - d}{b + c - a - d} \triangleq \tilde{q}.
\end{aligned}
$$

We need $\tilde{q} < 1$ to ensure a feasible q_t can be found, which means $b < c - a$.

Thus, for a matrix game satisfying the condition $c - a > b = \max\{a, b, c, d\} > c > d > 0 > a$, we can find a non-trivial solution to (13). To empirically verify this conclusion, we choose a matrix game with $(a, b, c, d) = (-4, 7, 6, 4)$ where $\tilde{q} = \frac{10}{13} \approx 0.769...$. For simplicity, we call this matrix game as matrix game case 4. We also choose $(p_0, q_0) = (0.55, 0.8)$ to ensure the condition $q_t > \tilde{q}$. The empirical results are illustrated in Figure 10. We can find the non-trivial update for the joint policy which verifies our conclusion discussed before.

F.5 Comparing TVPO and DPO

From the discussion in Section 4.2, we have an intuitive idea about the difference between DPO and TVPO that the bound D_{TV} of TVPO is tighter than $\sqrt{D_{\mathrm{KL}}}$ in DPO. A tighter bound means the iteration will be less influenced by the trivial update. We would like to build a matrix game to show this phenomenon. Fortunately, a previously discussed matrix game $(a, b, c, d) = (-4, 7, 6, 4)$ satisfies our requirement. The DPO iteration has no closed-form solution and we haven't found any useful properties like Section F.4. Thus, we use a numerical method to solve the DPO iteration. First, we keep the initial policy $(p_0, q_0) = (0.55, 0.8)$ for two iterations. The empirical results are included

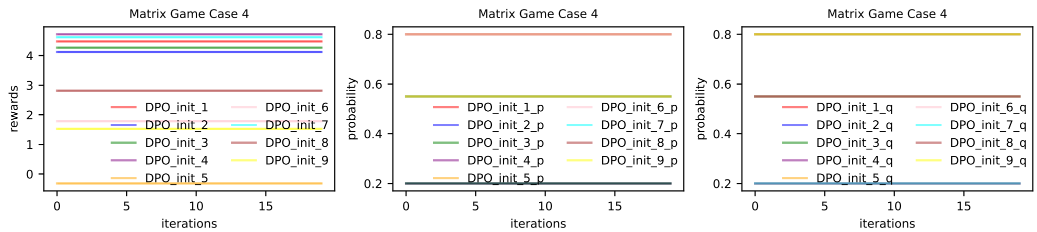

Figure 12: Learning curves of the DPO iteration with different initial policies in the matrix game $(a, b, c, d) = (-4, 7, 6, 4)$, where x-axis is iteration steps. The three figures show the expectation $J(\boldsymbol{\pi}_t)$, the policies p and q of nine different initial policies in the matrix game case 4 respectively, where $J(\boldsymbol{\pi}_t)$ is calculated by the joint policy $\boldsymbol{\pi}_t = (p_t, q_t)$ and the payoff matrix.

in Figure 11. We can find that the TVPO iteration has a non-trivial update but the DPO iteration only has trivial updates. This result can be evidence for our conclusion about the difference between TVPO and DPO.

Moreover, we study the influence of the initial policies on the DPO iteration. We select three candidate values $C = \{0.2, 0.55, 0.8\}$ for the initial policies. We traverse all the values in C for (p_0, q_0) and conclude the performances of all 9 combinations in Figure 12 and Table 4. We can find all 9 initial policies fall into the trap of the trivial update due to the regularization term $\sqrt{D_{\mathrm{KL}}}$ in DPO. These empirical results can partially exclude the impact of initial policies on the performances of the DPO iteration in this matrix game.

Table 4: The policy update types of DPO iteration with different initial policies in the matrix game $(a, b, c, d) = (-4, 7, 6, 4)$. T represents the trivial policy update and NT represents the non-trivial policy update.

p_0 \ q_0	0.2	0.55	0.8
0.2	T	T	T
0.55	T	T	T
0.8	T	T	T

F.6 Discussions about using global state s in theoretical results.

Using the global state s for theoretical analysis has been a common practice in the study of multi-agent reinforcement learning, especially in the setting of decentralized learning. There are many previous works containing theoretical results in decentralized learning, which include both cooperative settings (Jiang & Lu, 2022) and non-cooperative settings (Arslan & Yüksel, 2016; Mao et al., 2022a; Zhang et al., 2024). The main reason for this common practice is the difficulty in solving a POMDP, which has been studied for decades in Papadimitriou & Tsitsiklis (1987); Mundhenk et al. (2000); Vlassis et al. (2012). Additionally, the theoretical analysis of Dec-POMDP will be even more difficult in the multi-agent setting. If we include partial observability in the analysis, we may not obtain anything since the problem may be undecidable in Dec-POMDP (Madani et al., 1999).

G Additional experiments for Rebuttal

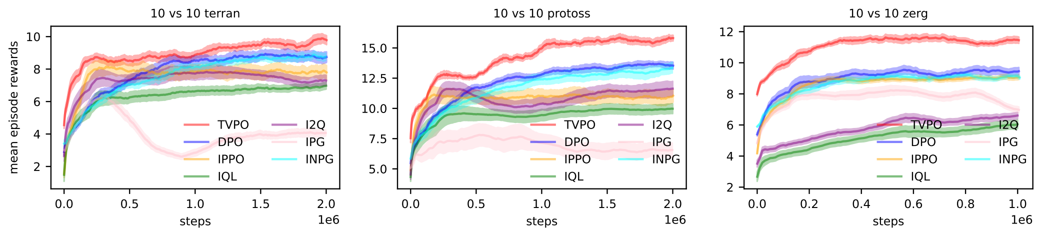

Figure 13: Learning curves of the TVPO and other baselines including IPG and INPG in the three 10_vs_10 SMAC-v2 tasks.

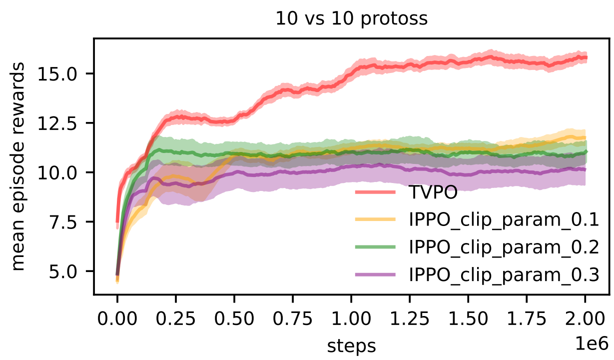

Figure 14: Learning curves of the TVPO and IPPO with different clip parameters in the 10_vs_10 protoss.

For the comparison with the baseline IPG (Leonardos et al., 2021) and INPG (Fox et al., 2022), we select three 10_vs_10 SMAC-v2 tasks. The empirical results are illustrated Figure 13. We can find that IPG's performance is not stationary and may drop with the progress of training compared with other policy based algorithms. We think the main reason is that IPG lack the constraints about the stepsize of policy iteration. We use the adaptive coefficient for INPG, and its performance is similar to DPO, which is reasonable as their policy objectives are similar except for a square root term.

We also compare the influence of the hyperparameters on IPPO's performance. We choose clip parameters with values $0.1, 0.2, 0.3$ for ablation study and select the 10_vs_10 protoss task for experiments. The empirical results are ilustrated in Figure 14. We can see that the impact of this hyperparameter is not significant.