VMAPD: Generate Diverse Solutions for Multi-Agent Games with Recurrent Trajectory Discriminators

1st Shiyu Huang
Tsinghua University
Beijing, China
hsy17@mails.tsinghua.edu.cn

2nd Chao Yu
Tsinghua University
Beijing, China
yc19@mails.tsinghua.edu.cn

3rd Bin Wang
Huawei Noah's Ark Lab
Beijing, China
wangbin158@huawei.com

4th Dong Li
Huawei Noah's Ark Lab
Beijing, China
lidong106@huawei.com

5th Yu Wang
Tsinghua University
Beijing, China
yu-wang@tsinghua.edu.cn

6th Ting Chen
Tsinghua University
Beijing, China
tingchen@tsinghua.edu.cn

7th Jun Zhu
Tsinghua University
Beijing, China
dcszj@tsinghua.edu.cn

***Abstract*—Recent algorithms designed for multi-agent tasks focus on finding a single optimal solution for all the agents. However, in many tasks (e.g., matrix games and transportation dispatching), there may exist more than one optimal solution, while previous algorithms can only converge to one of them. In many practical applications, it is important to develop reasonable agents with diverse behaviors. In this paper, we propose "variational multi-agent policy diversification" (VMAPD), an on-policy framework for discovering diverse policies for coordination patterns of multiple agents. By taking advantage of latent variables and exploiting the connection between variational inference and multi-agent reinforcement learning, we derive a tractable evidence lower bound (ELBO) on the trajectories of all agents. Our algorithm uses policy iteration to maximize the derived lower bound and can be simply implemented by adding a pseudo reward during centralized learning. And the trained agents do not need to access the pseudo reward during decentralized execution. We demonstrate the effectiveness of our algorithm on several popular multi-agent testbeds. Experimental results show that VMAPD finds more solutions with similar sample complexity compared with other baselines.**

***Index Terms*—deep reinforcement learning, multi-agent reinforcement learning, diversity, probabilistic graphical models**

I. Introduction

Recently, multi-agent reinforcement learning (MARL) shows huge advantages on various multi-agent tasks, e.g., AlphaStar achieved super-human performance in StarCraft II [30] and OpenAI Five won against the world champion in Dota II [3]. While recent algorithms achieve the state-of-the-art performance on a suite of StarCraft II benchmark tasks [28], they are designed to fit for a single optimal solution. In many multi-agent tasks, there may exist more than one optimal solution. Obtaining diverse solutions is critical in practical applications. For example, in a confrontational environment, if the strategy of an RL agent is fixed, it will be easy for its opponent to find its weakness and then beat it after playing with the agent many times. An agent with diverse strategies may help to relieve this problem. Moreover, in the game AI design, we need to develop non-player characters (NPCs) with various styles to allow the user to select a preferred one, which can improve the user experience. In an automatic driving scenario, we also need to develop agents to model diverse behaviors of social vehicles, which can improve the robustness of driving strategies. Based on this consideration, it is critical for developing new MARL algorithms which can obtain diverse solutions.

In this paper, we propose a new framework for multi-agent reinforcement learning algorithms to obtain diverse and coordinated behavior under *centralized training with decentralized execution* (CTDE). Despite recent successes of training diverse agents under single-agent scenario [5], [25], [26], [34], a challenge remains in the field of MARL: even if an MARL algorithm produces behavior like a single-agent algorithm, how can each agent coordinate well under partial observability due to decentralized execution? To solve this challenge, we formulate the MARL control problem as a probabilistic graphical model (PGM) and insert a shared latent variable to indicate the trajectory diversity. We show that the variational inference leads to the final objective of our "variational multi-agent policy diversification" (VMAPD) algorithm. Our algorithm uses policy iteration to maximize the derived evidence lower bound (ELBO) and it can be simply implemented by adding a pseudo reward during centralized training. And the trained agents do not need to access the pseudo reward during decentralized execution. VMAPD can be easily adapted to existing MARL algorithms and we implement VMAPD based on an on-policy MARL algorithm, i.e., the MAPPO algorithm [36]. We empirically show that VMAPD achieves better diversity compared with both policy-based and value-based MARL algorithms.

Our contributions are summarized as follows: (1) We represent the diverse multi-agent control problem as a unified

probabilistic graphical model and then derive an evidence lower bound (ELBO) as the optimization objective. The information bottleneck term in the ELBO can prevent the diversity from degenerating to the behavior of a single agent. (2) We introduce a modified ELBO with a Lagrange multiplier to achieve decoupling between rewards maximization and policy diversification. And we borrow the principle of PI controller [2] to tune this multiplier dynamically. (3) We thoroughly introduce a novel yet practical MARL algorithm, denoted as VMAPD, based on the theoretical objective. Our algorithm can produce diverse solutions by simply switching the latent variables with no need to change anything else. Experimental results on various benchmarks show that our method achieves competitive performance (and sample efficiency) while better diversity compared with other baselines.

II. Related Works

A. Probabilistic Graphical Model for Reinforcement Learning

Representing reinforcement learning as a probabilistic graphical model (PGM) has been studied in prior works [6], [19], [37]. Soft actor-critic (SAC) algorithm [8] formalizes reinforcement learning as probabilistic inference and optimizes the ELBO via adding an entropy term to the RL objective, which provides an implicit way to encourage exploration. The probabilistic graphical model also serves as a powerful tool for solving partially observable Markov decision process (POMDP) problems under a unified framework [13], [14], [17]. Haarnoja et al. [7] used PGM to construct a hierarchical reinforcement learning and their method can solve more complex sparse-reward tasks by learning higher-level policies on top of high-entropy skills. Hausman et al. [10] learned a multi-task policy via a variational bound and their method allows for the discovery of multiple solutions with a minimum number of distinct skills. Recently, PGM also provides some insights for designing new MARL algorithms [32], [35]. Yang et al. [35] proposed mean-field reinforcement learning to model the dynamics of interactions in the multi-agent systems and used model-free reinforcement learning methods to solve the Ising model [15]. Wen et al. [32] proposed PR2-Actor-Critic algorithm which adopted variational Bayes methods to approximate the opponents' conditional policies. In this paper, we also represent the diversity multi-agent control problem as a unified probabilistic graphical model and then derive the ELBO for further design of our MARL algorithm.

B. Diversity in Deep Reinforcement Learning

The measure of diversity has long been studied in the deep reinforcement learning community [5], [22], [23], [25]. Eysenbach et al. [5] proposed DIYAN to maximize the mutual information between states and skills, which results in a maximum entropy policy. More recently, Osa et al. [25] proposed a RL method that can learn infinitely many solutions by training a policy conditioned on a continuous or discrete low-dimensional latent variable. Their method can learn diverse solutions in continuous control tasks via variational information maximization. There is also a growing corpus

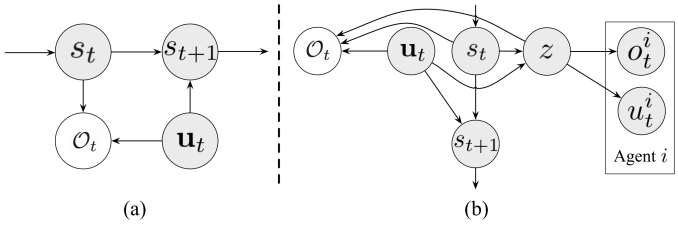


Fig. 1: (a) The graphical model of MDPs. (b) The graphical model of diverse Dec-POMDPs. Grey nodes are observed, white nodes are hidden. The $\mathcal{O}_t$ is a binary random variable, where $\mathcal{O}_t = 1$ denotes that the action is optimal at time t, and $\mathcal{O}_t = 0$ denotes that the action is not optimal.

of works studying the diversity of multi-agent reinforcement learning [11], [18], [21]. Mahajan et al. [21] proposed MAVEN, a method that overcomes the detrimental effects of QMIX's [27] monotonicity constraint on exploration via maximizing the mutual information between latent variables and trajectories. However, their method does not provide multiple solutions for a specified task, while our method can find diverse solutions during the learning process. Lee et al. [18] learned a hierarchical policy structured by latent variables to efficiently coordinate skills to solve challenging collaborative control tasks. However, the meta policy in their method needs access to the fully observed state, while our method can coordinate agents under partial information during decentralized execution. He et al. [11] investigated multi-agent algorithms for learning diverse skills using information bottlenecks with unsupervised rewards. However, their method does not train agents to solve a specific task. Instead, it just learns diverse behaviors randomly, while our method is aimed to find diverse solutions for multi-agent tasks.

III. Preliminaries

A. Cooperative Multi-Agent Reinforcement Learning

The cooperative multi-agent reinforcement learning can be formalized as a Dec-POMDP [24]. The Dec-POMDP can be represented as a tuple $(\mathcal{N}, S, U, P_T, r, O, F_G, \gamma)$, where U is the action space, S is the state space, O is the observation space, and γ is the discount factor. $\mathcal{N} \equiv \{1, ..., n\}$ is the set of n agents and each agent $i \in \mathcal{N}$ will choose an action $u^i \in U$ to make up a joint action $\mathbf{u} \equiv U^n$. And all the agents will receive a reward $r(s, \mathbf{u})$ after taking the joint action. The state-transition function $P_T(s, \mathbf{u}, s')$ defines the probability over the next state s' after taking joint action $\mathbf{u}$ at state s. The observation function $F_G(s, i) : S \times \mathcal{N} \to O$ means that each agent i has only access to observation $o \in O$ at state s. And each agent i produces its action according to the policy $\pi^i(u^i|o^i)$. The goal is to maximize the expected accumulated reward $\mathbb{E}_{s_t, \mathbf{u}_t}[\sum_t \gamma^t r(s_t, \mathbf{u}_t)]$.

B. Reinforcement Learning as Probabilistic Graphical Model

The optimal control problem can be solved as a probabilistic inference task [19]. The probabilistic graphical model (PGM) of Markov decision processes (MDP) is shown in Fig. 1(a). We

borrow the notation from [19] to illustrate the algorithm. [19] introduces a binary random variable $\mathcal{O}_t$ to the graphical model, where $\mathcal{O}_t = 1$ denotes that the action is optimal at time t, and $\mathcal{O}_t = 0$ denotes that the action is not optimal. The probability distribution of $\mathcal{O}$ is $p(\mathcal{O}_t = 1|s_t, \mathbf{u}_t) = \exp(r(s_t, \mathbf{u}_t))$[1], and the evidence lower bound (ELBO) is given by:

$$\begin{aligned} &\log p(\mathcal{O}_{1:T}) \\ &\geq \mathbb{E}_{(z,a_{1:T})\sim\pi(z,\mathbf{u}_{1:T})}\left[\sum_{t=1}^{T} r(s_t, \mathbf{u}_t) - \log\pi(\mathbf{u}_t|s_t)\right], \end{aligned} \tag{1}$$

where $\pi(\mathbf{u}|s)$ is the policy function. Standard reinforcement learning only needs to maximize the cumulative reward. However, the derived ELBO indicates that we also need to maximize an extra term, which is the policy entropy at each visited state. Based on this finding, [8] proposed the soft actor-critic (SAC) algorithm via adding an entropy term to the RL objective, which provides an implicit way to encourage exploration. In addition, PGM has shown great power in solving more complex control tasks [10], [17]. More applications of PGM in RL can be seen in Section II-A.

IV. Variational Multi-Agent Policy Diversification

A. Variational Lower Bound for Diverse Dec-POMDPs

Different from vanilla Dec-POMDPs, we inject a latent variable z for Dec-POMDPs in the PGM (shown in Fig. 1(b)) to form a diverse Dec-POMDP. In the diverse Dec-POMDPs, the latent variable z can discriminate or encode the diverse solutions. We need to derive the variational lower bound for diverse Dec-POMDPs, which will then be used to infer the latent variable z and do planning jointly. We apply the structured variational inference to optimize the evidence lower bound of the diverse Dec-POMDPs. In structured variational inference, different parts of the proposal distributions can be optimized separately, which means we can fix part of the approximate functions and then optimize other approximate functions. In this PGM, we will use three types of approximate functions–the actor networks $q_{\phi_i}(u_t^i|o_{1:t}^i, z)$, the global state discriminator $q_\theta(z|s_{1:t+1}, \mathbf{u}_{1:t})$ and the local observation discriminators $q_{\theta_{loc}^i}(z|o_{1:t+1}^i, u_{1:t}^i)$. When $q_\theta(\cdot)$ and $q_{\theta_{loc}^i}^i(\cdot)$ are fixed, the learning procedure is same as MARL, so that $q_{\phi_i}(\cdot)$ can be learned via a vanilla MARL algorithm (such as MAPPO). Conversely, when $q_{\phi_i}(\cdot)$ is fixed as the optimal policy, we can learn the inference functions $q_\theta(\cdot)$ and $q_{\theta_{loc}^i}^i(\cdot)$ for the hidden variables. To get optimal actions in diverse Dec-POMDPs, we derive the evidence lower bound(ELBO) as bellow:

$$\begin{aligned} &\log p(\mathcal{O}_{1:T}) = \log \mathbb{E}_{\tau\sim\mathcal{D}}\left[\frac{p(\mathcal{O}_{1:T}, \tau)}{q(\tau)}\right] \\ &\geq \mathbb{E}_{\tau\sim\mathcal{D}} \log\left[\frac{p(\mathcal{O}_{1:T}, \tau)}{q(\tau)}\right] \\ &\simeq \mathbb{E}_{\tau\sim\mathcal{D}} \sum_{t=1}^{T} r(s_t, \mathbf{u}_t) - \frac{1}{n}\sum_{i=1}^{n} \log q_{\phi_i}(u_t^i|o_{1:t}^i, z) \\ &+ \log q_\theta(z|s_{1:t+1}, \mathbf{u}_{1:t}) - \log p(z) + \log q_\theta(z|s_{1:t+1}, \mathbf{u}_{1:t}) \\ &- \frac{1}{n}\sum_{i=1}^{n} \log q_{\theta_{loc}^i}(z|o_{1:t+1}^i, u_{1:t}^i) \\ &= \mathbb{E}_{\tau\sim\mathcal{D}} \sum_{t=1}^{T} r(s_t, \mathbf{u}_t) + \underbrace{2\log q_\theta(z|s_{1:t+1}, \mathbf{u}_{1:t})}_{\text{diversity term}} \\ &\underbrace{- \frac{1}{n}\sum_{i=1}^{n} \log q_{\theta_{loc}^i}(z|o_{1:t+1}^i, u_{1:t}^i)}_{\text{information bottleneck term}} - \mathcal{B}, \end{aligned} \tag{2}$$

where the joint trajectory $\tau = \{\mathbf{u}_{1:T}, s_{1:T}, o_{1:T}, z\}$ is sampled from a trajectory dataset $\mathcal{D}$, and $\mathcal{B}$ represents a baseline in the ELBO. In this paper, the marginal distribution $p(z)$ is a categorical distribution and the number of categories is denoted as n_z.

As shown in the ELBO, there is a diversity term, which should be maximized to get diverse joint behaviors. And there is also an information bottleneck term [33], which enhances the diversity of the joint behavior and prevents the diversity from degenerating to the behavior of a single agent. Many recent deep RL algorithms, such soft actor-critic (SAC) [8] and MAPPO, have involved the entropy term of actions in the practical implementation, so we merge the entropy term into the baseline $\mathcal{B}$ and we will omit this term in the remaining sections for simplicity.

The simple combination of different parts of the ELBO may lead to poor performance. Recent work [8], [9], [12], [29] modified the ELBO via adding adjustable coefficients that balance different parts of the ELBO, which can stabilize the training process. In the following section, we will introduce a similar learning strategy via adding a reward balancing coefficient and borrow the principle of PI controller [2] to tune this coefficient dynamically.

B. Modified ELBO with Dynamic Lagrange Multiplier

The goal of diversity Dec-POMDPs is to maximize the diversity and simultaneously keep cumulative rewards to the target return $\mathcal{R}_{target}$. On this account, it can be formulated as

[1]Rewards should be negative. This assumption can be guaranteed simply by subtracting the maximum reward.

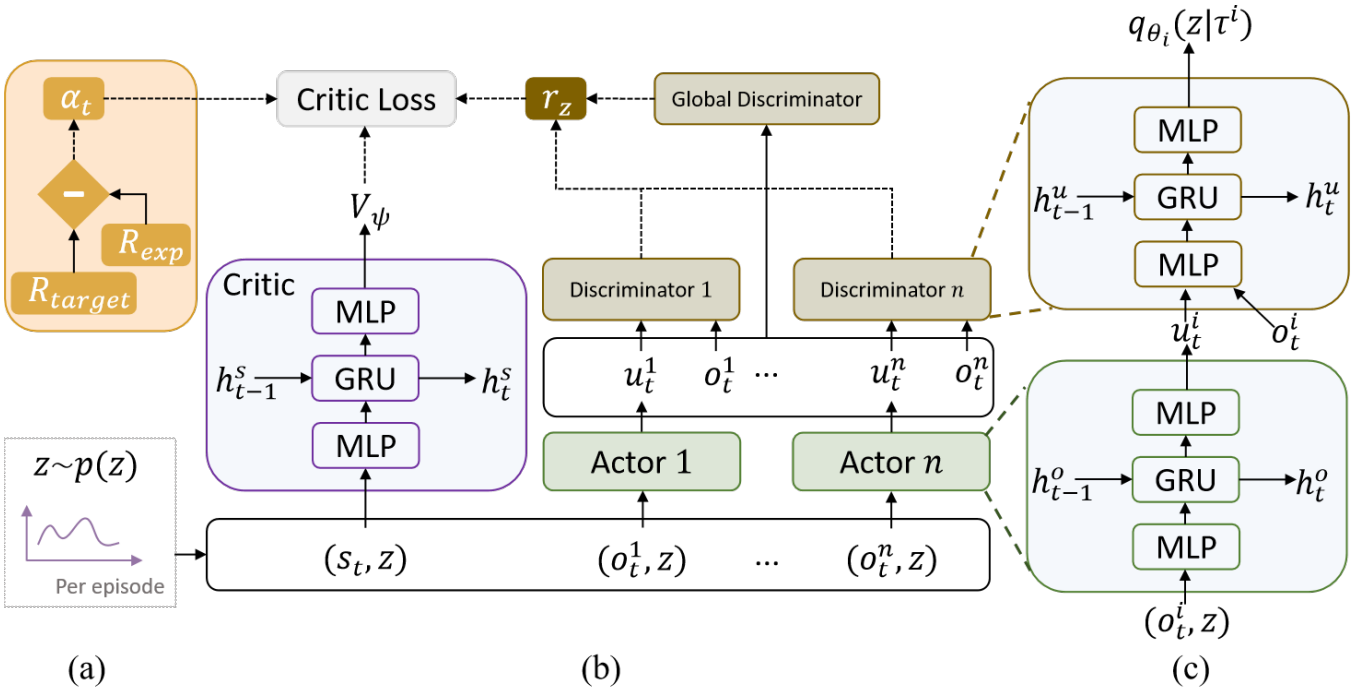


Fig. 2: The architecture of VMAPD. (a) The latent variable z is sampled from a categorical distribution $p(z)$ at the start of each episode. (b) The overall pipeline of VMAPD. (c) The network structures of actors and discriminators.

the following constrained optimization problem:

$$\max_{\pi} \ \mathbb{E}_{\tau\sim\mathcal{D}_\pi}\Big[2\log q_\theta(z|s_{1:t+1},\mathbf{u}_{1:t}) - \frac{1}{n}\sum_{i=1}^{n}\log q_{\theta^i_{loc}}(z|o^i_{1:t+1},u^i_{1:t})\Big], \quad \text{s.t. } \mathbb{E}_{\tau\sim\mathcal{D}_\pi}\sum_{t=1}^{T} r(s_t,\mathbf{u}_t) = \mathcal{R}_{target}, \tag{3}$$

where π is the joint policy over all the agents and $\mathcal{D}_\pi$ is the trajectory dataset collected by the joint policy. To stabilize cumulative rewards to the target return $\mathcal{R}_{target}$, we use the Lagrange multiplier method and introduce a Lagrange multiplier α_t in the following objective:

$$\mathcal{L}_\alpha = \mathbb{E}_{\tau\sim\mathcal{D}_\pi}\Big[\sum_{t=1}^{T}\alpha_t r(s_t,\mathbf{u}_t) + 2\log q_\theta(z|s_{1:t+1},\mathbf{u}_{1:t}) - \frac{1}{n}\sum_{i=1}^{n}\log q_{\theta^i_{loc}}(z|o^i_{1:t+1},u^i_{1:t})\Big]. \tag{4}$$

We need to evolve α_t during the training process to achieve decoupling between rewards maximization and policy diversification. To achieve this goal, α_t should dynamically change from a small value to a large one. Specifically, at the beginning of training, α_t should be small enough to disentangle the latent variable z. Then α_t should gradually increase to a large value to maximize the extrinsic rewards. In addition, α_t should not change too fast or oscillates too frequently. When α_t increases too fast or oscillates, it will be hard for the critic network to approximate the expected return and it will be hard for the policy network to be optimized along with the value function.

We borrow the principle of the non-linear PI controller from [29] to dynamically adjust the coefficient α_t, based on the expected return, $\mathcal{R}^t_{exp}$. We use the difference e_t between the expected return with the target return, $\mathcal{R}_{target}$, as the feedback to tune α_t (i.e., $e_t = \mathcal{R}^t_{exp} - \mathcal{R}_{target}$). The corresponding algorithm is denoted by:

$$\alpha_t = \Delta\alpha_t + \alpha_{t-1}, \tag{5}$$

where $\Delta\alpha_t = [\sigma(-e(t)) - \sigma(-e(t-1))] - e(t)$; $\alpha(0)$ is a small initial value; $\sigma(\cdot)$ is a sigmoid function. In the next section, we will present our final algorithm, denoted as VMAPD (i.e., "variational multi-agent policy diversification") and show how we optimize the ELBO and how to design a model to learn the coefficient in practice.

C. The VMAPD Algorithm

This section shows how VMAPD algorithm is implemented with a MARL algorithm. In this paper, we choose MAPPO as the backbone to optimize our proposed objective. MAPPO is an on-policy MARL algorithm, composed of n actors with policy network $q_{\phi_i}(u^i_t|o^i_{1:t})$ and a centralized critic network $V_\psi(s_{1:t})$. We develop a new variant of MAPPO via inserting a latent variable z into the input of the policy network (i.e., $q_{\phi_i}(u^i_t|o^i_{1:t}, z)$) and the critic network (i.e., $V_\psi(s_{1:t}, z)$). We also utilize a global discriminator $f_\theta(s_{1:t+1}, \mathbf{u}_{1:t}) = q_\theta(z|s_{1:t+1}, \mathbf{u}_{1:t})$, which takes the global states and joint

actions as inputs and then outputs the probability of the latent variable z. In addition, there are n local discriminators $f_{\theta^i_{loc}}(o^i_{1:t+1}, u^i_{1:t}) = q_{\theta^i_{loc}}(z|o^i_{1:t+1}, u^i_{1:t})$, which take local observations and individual actions as inputs and then outputs the probability of the latent variable z. The global discriminator and local discriminators are trained in a supervised manner. At each training step, we sample trajectories (with latent variable z) from the dataset (collected by the joint policy π), and optimize discriminators via the categorical cross entropy loss:

$$\begin{aligned}\mathcal{L}_\theta &= \mathbb{E}_{\{s_{1:t+1}, \mathbf{u}_{1:t}, z\}\sim\mathcal{D}_\pi} CE(f_\theta(s_{1:t+1}, \mathbf{u}_{1:t}), z),\\ \mathcal{L}_{\theta^i_{loc}} &= \mathbb{E}_{\{o^i_{1:t+1}, u^i_{1:t}, z\}\sim\mathcal{D}_\pi} CE(f_{\theta^i_{loc}}(o^i_{1:t+1}, u^i_{1:t}), z),\end{aligned} \tag{6}$$

where CE is the cross entropy loss, and we use the Adam optimizer [16] to optimize above losses. In the practical implementation, we let the global state be the concatenation of all the local observations, i.e., $s_t = [o^1_t, ..., o^n_t]$.

To optimize the objective in Eq. 4 with policy iteration, we construct a pseudo reward with trained discriminators as below:

$$\begin{aligned}r_z(s_t, \mathbf{u}_t) = 2\log q_\theta(z|s_{1:t+1}, \mathbf{u}_{1:t})\\ - \frac{1}{n}\sum_{i=1}^{n}\log q_{\theta_i}(z|o^i_{1:t+1}, u^i_{1:t}).\end{aligned} \tag{7}$$

And the modified total reward with balancing coefficient α_t for agents can be written as:

$$r_{total}(s_t, \mathbf{u}_t) = \alpha_t r(s_t, \mathbf{u}_t) + r_z(s_t, \mathbf{u}_t). \tag{8}$$

The modified reward $r_{total}(s_t, \mathbf{u}_t)$ will be stored in the replay memory and then be used to train the critic network V_ψ and policy network q_{ϕ_i} in a vanilla MAPPO training manner.

We have shown that the coefficient α_t can be updated via a PI controller in Eq. 5. In this part, we will derive a more practical learning algorithm to tune the coefficient. The PI controller has shown a deeper connection with stochastic optimization of deep networks [1] and the PI controller can be implemented as an Adam [16] optimizer. Thus we design the following objective to learn the coefficient:

$$\begin{aligned}J(\alpha_t) &= \mathbb{E}[\exp(\alpha_t)e(t)]\\ &= \mathbb{E}[\exp(\alpha_t)(\mathcal{R}^t_{exp} - \mathcal{R}_{target})],\end{aligned} \tag{9}$$

and the expected return $\mathcal{R}^t_{exp}$ is computed with the extrinsic rewards in the replay memory. We then update α_t with gradient descent:

$$\alpha_t \leftarrow \alpha_{t-1} - \beta\nabla_{\alpha_{t-1}} J(\alpha_{t-1}), \tag{10}$$

where β is the learning rate. Fig. 2 shows the overall framework of the VMAPD algorithm. In our framework, we use GRU cells [4] as the recurrent units.

V. Experimental Results

A. Experiment Setup

In this section, we conduct experiments on various multi-agent benchmarks, i.e., Matrix Game, the Multi-agent Particle-world Environment (MPE) [20] and the StarCraft II Micro-management Challenge (SMAC) [28]. And the baseline algorithms include a value-based MARL algorithm—QMIX [27], and a policy-based MARL algorithm—MAPPO [36].

We further implement two more baselines based on DIAYN [5] and MASD [11]. The main difference between DIAYN, MASD, and VMAPD is how to calculate the pseudo (or intrinsic) reward r_z. In the experiments, we combine r_z of DIYAN and MASD with extrinsic rewards via the same process as VMAPD and all the methods use the same hyper-parameters.

B. Matrix Game

a_1 \ a_2	$u_2^{(1)}$	$u_2^{(2)}$
$u_1^{(1)}$	-1	**1**
$u_1^{(2)}$	**1**	-1

TABLE I: Payoff matrix of the multi-optimal-solution game. Boldface means the optimal joint action selection from payoff matrix.

In this subsection, we consider a multi-optimal-solution matrix game as shown in Table I. Table I shows a two-agent cooperative task. In this task, two agents need to select their actions individually and payoffs will be given based on their joint actions. The first agent has two optional actions (i.e., $u_1^{(1)}$ and $u_1^{(2)}$) and the second agent also has two optional actions (i.e., $u_2^{(1)}$ and $u_2^{(2)}$). For the sake of simplicity, we use p_1, p_2, p_3 and p_4 to indicate the joint actions $(u_1^{(1)}, u_2^{(1)}), (u_1^{(1)}, u_2^{(2)}), (u_1^{(2)}, u_2^{(1)})$ and $(u_1^{(2)}, u_2^{(2)})$ respectively. There are two optimal solutions (i.e., p_2 and p_3) in this task, and the agents will receive miscoordination penalties when they choose p_1 and p_4. To investigate the impact of reward balancing coefficient α, we conduct experiments on various fixed α, which ranges from 0 to 100, and each scenario is trained in the VMAPD manner. We also include results of DIAYN, MASD and VMAPD with auto learned α. The latent variable z is sampled from a categorical distribution with the number of categories n_z. In this experiment, we set $n_z = 40$. We report the distribution of z corresponding to the joint actions.

Fig. 3 shows the final results when each method converges. When $\alpha = 0$, agents can not receive extrinsic rewards and can only optimize their policies via pseudo reward r_z, so that agents can only produce diverse joint actions and each z locates in different joint actions uniformly. When α is increased to 0.5 and 1, the probability of choosing two optimal joint actions (p_2 and p_3) also increases. When α is increased to 10 and 100, the agents can only find a certain optimal solution (p_2 or p_3) because of the domination of extrinsic rewards. This phenomenon indicates that a fixed α may lead to many non-optimal solutions or a reduction in optimal solutions. Results also show that DIAYN and MASD fail to find diverse optimal solutions. However, VMAPD can dynamically update the α and gets more diverse optimal solutions. This demonstrates the effectiveness of the adaptive coefficient α in our method.

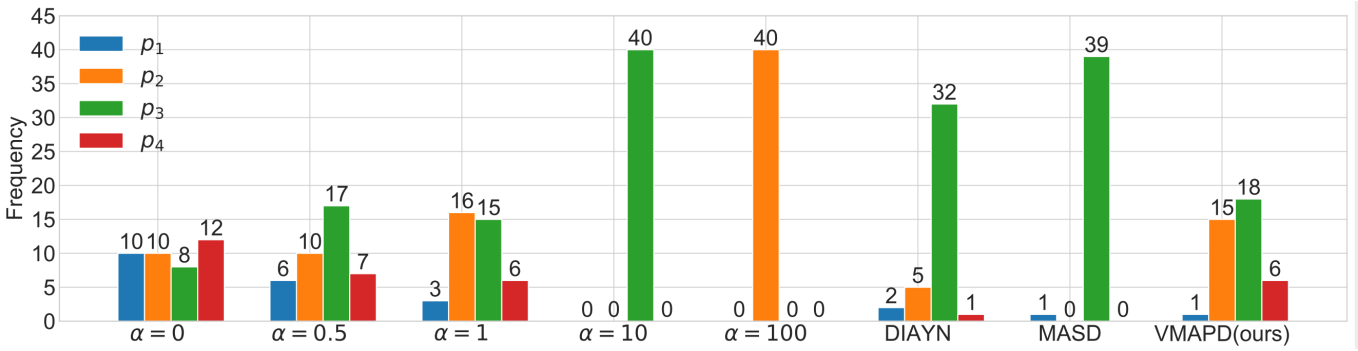


Fig. 3: Experimental results on the matrix game. We conduct experiments on various fixed α, ranging from 0 to 100. We also include results of DIAYN, MASD, and VMAPD with auto-learned α. The latent variable z is sampled from a categorical distribution with the number of categories $n_z = 40$. We report the distribution of z corresponding to the joint actions. VMAPD finds more diverse optimal solutions (i.e., p_2 and p_3) than other methods, which demonstrates the effectiveness of the adaptive coefficient α in our method.

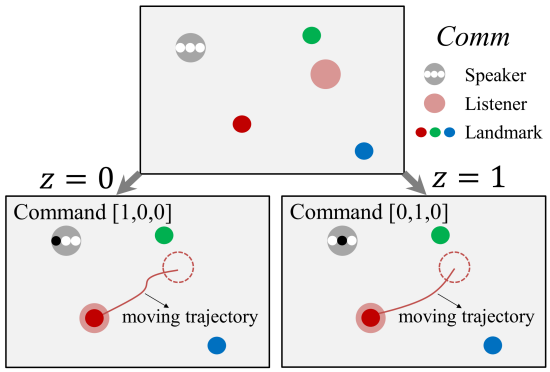


Fig. 4: The *Comm* task. The Speaker needs to give out a command to the Listener and the Listener needs to move to the red landmark based on the command. The Speaker gives out two different commands with different z and the Listener can still reach the target landmark with different commands. It indicates that the two agents have learned different strategies (or solutions) under different z.

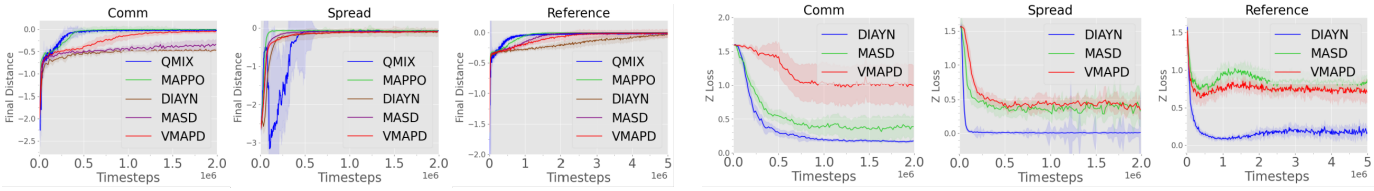
C. Multi-Agent Particle-World Environment

We experiment on three tasks originally used in [20], including *Spread*, *Reference* and *Comm*. And we compare VMAPD with QMIX, MAPPO, DIAYN, and MASD. All results are averaged over 10 seeds. In this experiment, we set $n_z = 5$. The performance of each algorithm at convergence is shown in Fig. 5(a). The loss curves of the discriminator $q_\theta(z|s_{0:t+1}, \mathbf{u}_{0:t})$ is also reported in as shown in Fig. 5(b). Experimental results show that VMAPD achieves competitive performances compared with previous algorithms (i.e., QMIX and MAPPO), while VMAPD can find more solutions with the same training steps. We can also notice that DIAYN and MASD fail to approach optimal solutions on the *Comm* task. The loss curves of the global discriminator indicate that the trained agents can produce distinguishing trajectories with different latent variables.

Taking *Comm* as an example, we show the diverse solutions obtained via the VMAPD algorithm. In *Comm*, there are two agents and three landmarks. At the beginning of each episode, positions of agents and landmarks will be randomly initialized. The two agents need to collaborate to reach a target landmark, which is randomly picked from the three landmarks. One of the agents, denoted as Speaker, has the information of the target landmark, and it can give a three-dimension binary command (e.g. [1,0,0] or [0,1,0]) to another agent (the Listener). The Listener has no information about the target landmark so that the two agents need to develop protocols for the target landmark and the command. Agents can adopt different protocols under different latent variables so long as they can execute the same protocol under the same z. In Fig. 4, the Speaker needs to give out a command to the Listener and the Listener needs to move to the red landmark based on the command. The Speaker gives out two different commands with different z and the Listener can still reach the target landmark with different commands. It indicates that the two agents have learned different solutions under different z. More examples of diverse solutions on MPE can be found in the supplementary material.

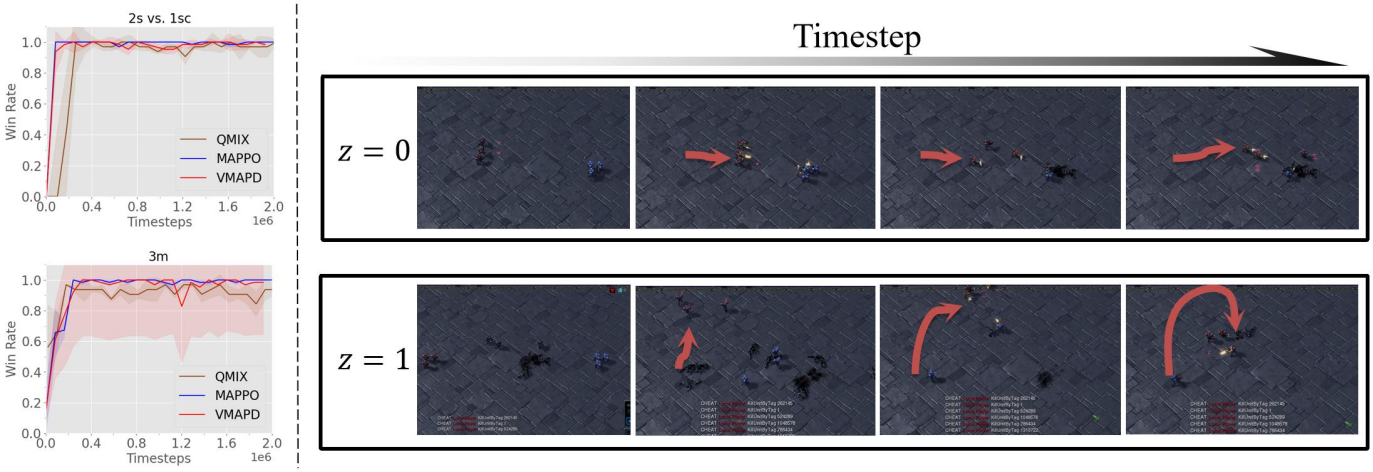
D. StarCraft II

We consider two StarCraft II maps (i.e., 2s vs. 1sc and 3m) from the SMAC benchmark [28]. We use the evaluation metric of [31] and report the median success rates over 6 seeds for all the algorithms. In this experiment, we set $n_z = 5$. Fig. 6(a) shows the training curves of different algorithms on the two tasks. VMAPD achieves competitive performances compared with QMIX and MAPPO, while VMAPD can find more solutions with the same training steps. Fig. 6(b) visualizes two different solutions obtained by the VMAPD algorithm on the 3m map. In this map, we need to control three agents (the red ones) to combat against three build-in agents (the blue ones). For the convenience of observation, we plot the moving trajectory of our side in a red arrow. When $z = 0$, the agents learned a brutal tactic, i.e., they move to the enemies



(a) (b)

Fig. 5: Training curves of different algorithms on three MPE tasks. (a) shows the performance of each algorithm. VMAPD achieves competitive performances compared with QMIX and MAPPO, while VMAPD can find more solutions with the same training steps. DIAYN and MASD fail to approach optimal solution on the *Comm* task. (b) shows the loss curves of the discriminator. The loss curves of the discriminator indicate that the trained agents can produce distinguishing trajectories with different latent variables.



(a) (b)

Fig. 6: (a) Training curves of different algorithms on two StarCraft II maps. VMAPD achieves competitive performances compared with QMIX and MAPPO, while VMAPD can find more solutions with the same training steps. (b) visualizes two different solutions obtained by our algorithm on the 3m map. In this map, we need to control three agents (the red ones) to combat against three build-in agents (the blue ones). When $z = 0$, the agents learned a brutal tactic, i.e., they just move to the enemies directly and start to attack enemies. When $z = 1$, the agents learned an outflanking tactic, i.e., they first move to the upside of the map, which makes only two of three enemies can track their locations. They beat the first two enemies which follow them and then move to the center of the map to eliminate the last enemy. However, the MAPPO algorithm can only produce a brutal tactic.

directly and start to attack enemies. When $z = 1$, the agents learned an outflanking tactic, i.e., they first move to the upside of the map, which makes only two of three enemies can track their locations. They beat the first two enemies which follow them and then move to the center of the map to eliminate the last enemy. However, the MAPPO algorithm can only produce a brutal tactic (i.e., attacking enemies directly). The solution obtained by the MAPPO algorithm and more examples of diverse solutions obtained by our method can be found in the supplementary material.

VI. Conclusion

In this paper, we have proposed a probabilistic graphical model for multi-agent reinforcement learning to learn coordinated diverse solutions under CTDE via adding elaborate pseudo reward. Our proposed algorithm, denoted as VMAPD, is implemented practically by using a latent variable and evidence lower bound(ELBO) and applying policy iteration to maximize the ELBO. Experimental results show that VMAPD achieves competitive performance while better diversity compared with recent algorithms. Furthermore, our method is flexible to be integrated into other MARL algorithms to obtain diverse solutions. Currently, our method is implemented with latent variables which are sampled from a categorical distribution. In the future, we will try to incorporate the VMAPD algorithm with continuous latent variables.

Acknowledgment

This work is supported by the National Key R&D Program of China (2021YFF1201303 and 2019YFB1404804), National

Natural Science Foundation of China (grants 61872218), Guoqiang Institute of Tsinghua University, Tsinghua University Initiative Scientific Research Program, and Beijing National Research Center for Information Science and Technology (BNRist). The funders had no roles in study design, data collection and analysis, the decision to publish, and preparation of the manuscript.

REFERENCES

[1] W. An, H. Wang, Q. Sun, J. Xu, Q. Dai, and L. Zhang. A pid controller approach for stochastic optimization of deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8522–8531, 2018.

[2] K. J. Åström, T. Hägglund, and K. J. Astrom. *Advanced PID control*, volume 461. ISA-The Instrumentation, Systems, and Automation Society Research Triangle Park, 2006.

[3] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[5] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

[6] T. Furmston and D. Barber. Variational methods for reinforcement learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 241–248. JMLR Workshop and Conference Proceedings, 2010.

[7] T. Haarnoja, K. Hartikainen, P. Abbeel, and S. Levine. Latent space policies for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pages 1851–1860. PMLR, 2018.

[8] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[9] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[10] K. Hausman, J. T. Springenberg, Z. Wang, N. Heess, and M. Riedmiller. Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations*, 2018.

[11] S. He, J. Shao, and X. Ji. Skill discovery of coordination in multi-agent reinforcement learning. *arXiv preprint arXiv:2006.04021*, 2020.

[12] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[13] S. Huang, H. Su, J. Zhu, and T. Chen. Svqn: Sequential variational soft q-learning networks. In *International Conference on Learning Representations*, 2019.

[14] M. Igl, L. Zintgraf, T. A. Le, F. Wood, and S. Whiteson. Deep variational reinforcement learning for pomdps. In *International Conference on Machine Learning*, pages 2117–2126. PMLR, 2018.

[15] E. Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift Für Physik*, 31(1):253–258, 1925.

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] A. X. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*, 2019.

[18] Y. Lee, J. Yang, and J. J. Lim. Learning to coordinate manipulation skills via skill behavior diversification. In *International Conference on Learning Representations*, 2019.

[19] S. Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

[20] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017.

[21] A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson. Maven: Multi-agent variational exploration. *arXiv preprint arXiv:1910.07483*, 2019.

[22] S. Mohamed and D. J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. *arXiv preprint arXiv:1509.08731*, 2015.

[23] J.-B. Mouret and S. Doncieux. Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity. In *2009 IEEE Congress on Evolutionary Computation*, pages 1161–1168. IEEE, 2009.

[24] F. A. Oliehoek and C. Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016.

[25] T. Osa, V. Tangkaratt, and M. Sugiyama. Discovering diverse solutions in deep reinforcement learning. *arXiv preprint arXiv:2103.07084*, 2021.

[26] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017.

[27] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 4295–4304. PMLR, 2018.

[28] M. Samvelyan, T. Rashid, C. S. De Witt, G. Farquhar, N. Nardelli, T. G. Rudner, C.-M. Hung, P. H. Torr, J. Foerster, and S. Whiteson. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 2019.

[29] H. Shao, H. Lin, Q. Yang, S. Yao, H. Zhao, and T. Abdelzaher. Dynamicvae: Decoupling reconstruction error and disentangled representation learning. *arXiv preprint arXiv:2009.06795*, 2020.

[30] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[31] T. Wang, T. Gupta, A. Mahajan, B. Peng, S. Whiteson, and C. Zhang. Rode: Learning roles to decompose multi-agent tasks. *arXiv preprint arXiv:2010.01523*, 2020.

[32] Y. Wen, Y. Yang, R. Luo, J. Wang, and W. Pan. Probabilistic recursive reasoning for multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2019.

[33] A. Wieczorek and V. Roth. On the difference between the information bottleneck and the deep information bottleneck. *Entropy*, 22(2):131, 2020.

[34] K. Xie, H. Bharadhwaj, D. Hafner, A. Garg, and F. Shkurti. Latent skill planning for exploration and transfer. In *International Conference on Learning Representations*, 2021.

[35] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang. Mean field multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 5571–5580. PMLR, 2018.

[36] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu. The surprising effectiveness of mappo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[37] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.