SPD: Synergy Pattern Diversifying Oriented Unsupervised Multi-agent Reinforcement Learning

Yuhang Jiang,* Jianzhun Shao,* Shuncheng He, Hongchang Zhang, Xiangyang Ji
Department of Automation
Tsinghua University, Beijing, China
{jiangyh19, sjz18, hesc16, hc-zhang19}@mails.tsinghua.edu.cn
xyji@tsinghua.edu.cn

Abstract

Reinforcement learning typically relies heavily on a well-designed reward signal, which gets more challenging in cooperative multi-agent reinforcement learning. Alternatively, unsupervised reinforcement learning (URL) has delivered on its promise in the recent past to learn useful skills and explore the environment without external supervised signals. These approaches mainly aimed for the single agent to reach distinguishable states, insufficient for multi-agent systems due to that each agent interacts with not only the environment, but also the other agents. We propose *Synergy Pattern Diversifying Oriented Unsupervised Multi-agent Reinforcement Learning* (SPD) to learn generic coordination policies for agents with no extrinsic reward. Specifically, we devise the *Synergy Pattern Graph* (SPG), a graph depicting the relationships of agents at each time step. Furthermore, we propose an episode-wise divergence measurement to approximate the *discrepancy of synergy patterns*. To overcome the challenge of sparse return, we decompose the discrepancy of synergy patterns to per-time-step pseudo-reward. Empirically, we show the capacity of SPD to acquire meaningful coordination policies, such as maintaining specific formations in Multi-Agent Particle Environment and pass-and-shoot in Google Research Football. Furthermore, we demonstrate that the same instructive pretrained policy's parameters can serve as a good initialization for a series of downstream tasks' policies, achieving higher data efficiency and outperforming state-of-the-art approaches in Google Research Football.

1 Introduction

Deep reinforcement learning (RL) has exhibited excellent capabilities in tackling complex tasks, including Go [42, 38], poker [28] and simulated robotic tasks [39, 13]. However, one of the key challenges in deploying RL algorithms is to devise an elaborate reward signal for eliciting desired behaviour of agents [14]. Due to the sharing of the team reward, the situation gets more complicated for cooperative multi-agent reinforcement learning (MARL), especially in real-world applications such as coordination of robot swarms [17] and autonomous vehicle teams [54], where verifying the effectiveness of the designed reward can be prohibitively costly.

As for the single agent, unsupervised learning has been incorporated into RL to acquire diverse *skills* for the agent without extrinsic reward from the environment, and this scenario is known as unsupervised reinforcement learning (URL). Recent advances in URL [10, 15, 21, 32, 40] have demonstrated the potential of the agent to explore the environment and learn useful skills with no supervision. Moreover, when endowed with the task-related extrinsic reward in downstream tasks, these skills exhibit significant generalizability, leading to efficient adaptation. An important

*Equal contribution.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

hypothesis explaining the phenomenon is made by DIAYN [10] that the higher the coverage of learned skills over the set of possible behaviour, the greater the probability of gaining effective skills.

Above mentioned URL approaches mostly depict the diversity of skills by the diversity of visited states, however, insufficient for MARL due to the neglect of interactions with other agents. Intuitively, this coordination "skill" in MARL corresponding to the skill in single-agent URL should be taken into consideration as well. We term the coordination "skill" as the *Synergy Pattern*, which is the perennial coordinated behaviour of agents induced by interactions of joint policies and the environment. In retrospect, the coordination of agents are usually modeled by *graphs*, such as coordination graphs [12, 4, 52] and graph neural networks [18, 22]. Nevertheless, these graph-based approaches are mainly tasked for communication and deploying the graphs directly to unsupervised MARL can be challenging.

We diversify the synergy patterns of agents by bridging the graphs and unsupervised MARL. Specifically, we devise the *Synergy Pattern Graph* (SPG), in which there is a vertex for each agent and the weight of the edge is derived from the function depicting the relations of agents, as the per-time-step instance of the synergy pattern. Our method *Synergy Pattern Diversifying Oriented Unsupervised Multi-agent Reinforcement Learning* (SPD) focuses on obtaining diverse synergy patterns while following the same hypothesis in URL. To this end, we approximate the *Discrepancy of Synergy Patterns* by quantifying the divergence between two SPG batches, which are sampled from the distributions of different synergy patterns respectively. The first step is to measure the discrepancy between two SPG elements. This problem is essentially to measure the similarity of graphs, which has been well-studied in the field of graph matching[41, 27, 33, 9], and one of the promising approaches is the Gromov-Wasserstein discrepancy [6, 53]. With the adoption of the Gromov-Wasserstein discrepancy as the pair-wise measurement, we then formulate the discrepancy of synergy patterns as an episode-wise measurement and deploy it as the pseudo episode return for unsupervised MARL.

Our contributions for SPD, a general framework for unsupervised MARL, are summarized as follows: i) we devise SPG with a general formulation as the instance to depict the synergy patterns of agents; ii) we propose an episode-wise divergence measurement to estimate the discrepancy of synergy patterns; iii) we derive the formulation for the decomposition of the pseudo episode return to alleviate the problem of the sparse return; iv) The empirical results demonstrate that SPD achieves better performance in MARL compared to conventional URL approaches and shows great potentials to learn synergy patterns with generalizability for downstream tasks.

2 Related works

Multi-agent reinforcement learning (MARL) has shown great potential in solving multi-agent cooperation tasks. A popular paradigm in MARL is the centralized training with decentralized execution (CTDE) that combines the advantage of independent Q-learning [46] and joint action learning [7]. Centralized training makes agents cooperate better while decentralized execution benefits the flexible deployment capability. We mainly focus on the value-based CTDE methods. Following the CTDE paradigm, a series of works concentrate on distributing the team reward to all agents by value function factorization [45, 35], and deriving the Individual-global-max principle for policy optimality analysis [44, 48, 36]. Another bunch of works introduce roles or skills to agents for better team cooperation [25, 50, 5]. However, these methods usually suffer from poor generalization ability and have considerable performance drop when transferred to unseen scenarios. Wang et al. [51] reports agents' generalization by splitting the action space, but the method relies heavily on the handcrafted hyper-parameter. The concept of agents' forming a graph is also widely used to represent agents' cooperation in MARL, including the coordination graph [4, 52] and graph neural networks [18, 22]. But these methods are usually designed for communication, which brings extra costs for decentralized execution. In contrast to these methods, our proposed method uses the synergy graph to train team skills without rewards from the environment, which brings the agents strong generalization ability in downstream tasks, and takes no extra cost in execution.

Unsupervised reinforcement learning (URL) has been particularly intriguing in the context of RL, consider the high cost of real-world rollouts with delicately designed reward signal. This paper is concerned about unsupervised behaviour learning for deep RL that requires learning with only the task-agnostic intrinsic reward. Early works [11, 1] mainly focuses on controlling the visited states via regulating a latent variable Z, where the mutual information is maximized with visited

states. DIAYN [10] demonstrates the exceptional performance of URL by maximizing $I(Z;S)$ while simultaneously minimizing $I(Z;A|S)$. The recent studies have also shown interests on adapting pre-trained skills on MDPs from aspects of the initial state [3] and the dynamics model [40]. Besides, WURL [15] adopts the idea of optimal transport and increases the discrepancy of learned skills by enlarging the distance of visited state distributions estimated by Wasserstein distance. These approaches aiming for achieving easily distinguishable states, however, they are deficient to solve the growing size of joint observation space in multi-agent systems. By contrast, our method devise a graph to depict the relative relations between agents which alleviate the growing size challenge. There are also works encouraging diversity for effective exploration [20, 24, 31, 55], while they mostly enhance single policy's diversity or the inter-agent diversity and do not get rid of the dependence of the task-related reward. We follow a similar hypothesis to that in Eysenbach et al. [10] to learn diverse synergy patterns for gaining useful coordination policies.

3 Preliminaries

3.1 Notation

We consider a Dec-POMDP [30] for the fully cooperative multi-agent task with n agents, defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{O}, \mathcal{I}, \Omega, n, \gamma \rangle$, where $s \in \mathcal{S}$ and $a^i \in \mathcal{A}$ denote the true state of the environment and the discrete action chosen by agent $i \in \mathcal{I} \equiv \{1, \ldots, n\}$, respectively. $\gamma \in [0,1)$ is the discount factor. The dynamics of the environment can be formulated as the state transition function $\mathcal{P}(s'|s, \boldsymbol{a}) : \mathcal{S} \times \boldsymbol{A} \times \mathcal{S} \to [0,1]$ and the reward function $\mathcal{R}(s, \boldsymbol{a}) : \mathcal{S} \times \boldsymbol{A} \to \mathbb{R}$ shared by all agents, where $\boldsymbol{a} \in \boldsymbol{A} \equiv \mathcal{A}^n$ denotes the joint action. Each agent i only has access to its own observation $o^i \in \mathcal{O}$ derived from the observation function $\Omega(s,i) : \mathcal{S} \times \mathcal{I} \to \mathcal{O}$, due to the *partial observability*.

At each time step t, agent i resolves to execute the action a^i according to a stochastic policy $\pi^i(a^i|\tau^i) : \mathcal{T} \times \mathcal{A} \to [0,1]$, where $\tau^i \in \mathcal{T} \equiv (\mathcal{O} \times \mathcal{A})^*$ denotes the action-observation history of the agent i. The goal of MARL is to find an optimal joint policy $\boldsymbol{\pi}^* \equiv \prod_{i=1}^n \pi^{i*}$ satisfying the condition: $Q^{\boldsymbol{\pi}^*}(s, \boldsymbol{a}) \geq Q^{\boldsymbol{\pi}}(s, \boldsymbol{a}), \forall \boldsymbol{\pi}$ and $(s, \boldsymbol{a}) \in \mathcal{S} \times \boldsymbol{A}$, where $Q^{\boldsymbol{\pi}}(s_t, \boldsymbol{a}_t) = \mathbb{E}_{s_{t+1:\infty}, \boldsymbol{a}_{t+1:\infty}} \left[\sum_{i=0}^{\infty} \gamma^i r_{t+i} | s_t, \boldsymbol{a}_t \right]$ is the joint action-value function induced by joint policy $\boldsymbol{\pi}$.

We adopt the paradigm of centralized training with decentralized execution (CTDE). During training the true state s and the joint action-observation history $\boldsymbol{\tau} \equiv \{\tau^i\}_{i=1}^n$ are accessible to the learning algorithm, while each agent i selects an action based on the policy conditioning only on its own action-observation history τ^i during testing (execution).

3.2 Optimal transport and Gromov-Wasserstein discrepancy

Optimal transport (OT) is a general problem of moving one distribution of mass to another as efficiently as possible. Likewise, the distinction of two given graphs can be regarded as an optimal transport problem to find the correspondence between their vertices and edges. Previous work [6] proposed *Gromov-Wasserstein Discrepancy*, which is extended from Gromov-Wasserstein distance [26] as a measure for the divergence of the graphs.

Definition 3.1. *[6, 53] (Gromov-Wasserstein discrepancy) Denote a measure graph as $G(\mathcal{V}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu})$, where $\mathcal{V} \equiv \{v_i\}_{i=1}^{|\mathcal{V}|}$ is the set of vertices, $\omega_{ij} \in \boldsymbol{\mathcal{W}}$ is the weight of the undirected edge $\{i,j\}$ between vertices $v_i, v_j \in \mathcal{V}$, and $\boldsymbol{\mu} = [\mu_i] \in \Sigma^{\mathcal{V}}$ is a Borel probability measure defined on $\mathcal{V}$. $\forall p \in [1, \infty]$ and each $G_\nu, G_\varsigma \in \mathcal{G}$, where $\mathcal{G}$ denotes the collection of measure graphs, the Gromov-Wasserstein discrepancy between G_ν and G_ς can be derived as*

$$d_{gw}\left(G_\nu, G_\varsigma\right) := \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_\nu, \boldsymbol{\mu}_\varsigma)} \left(\sum_{i,j \in \mathcal{V}_\nu} \sum_{i',j' \in \mathcal{V}_\varsigma} \left| \omega_{ij}^\nu - \omega_{i'j'}^\varsigma \right|^p T_{ii'} T_{jj'} \right)^{\frac{1}{p}}, \tag{1}$$

where $\Pi\left(\boldsymbol{\mu}_\nu, \boldsymbol{\mu}_\varsigma\right) = \left\{ \boldsymbol{T} \geq \boldsymbol{0} \mid \boldsymbol{T} \boldsymbol{1}_{|\mathcal{V}_\varsigma|} = \boldsymbol{\mu}_\nu, \boldsymbol{T}^\top \boldsymbol{1}_{|\mathcal{V}_\nu|} = \boldsymbol{\mu}_\varsigma \right\}$.

In general, the Gromov-Wasserstein discrepancy measures the divergence between two graphs by comparing the differences of the edges' relationship in one graph to that in the other. We refer the reader to Mémoli [26], Chowdhury & Mémoli [6], Peyré et al. [34] for mathematical foundations.

4 Methodology

This section is organized as the following: In Sec. 4.1, we delineate the concept of *Synergy Pattern* and propose a novel structure of graph as the instance of it, namely *Synergy Pattern Graph*. We then carefully derive the formulation for *Discrepancy of Synergy Patterns* and make an approximation of it for convenience in practice in Sec. 4.2. In Sec. 4.3, our unsupervised framework SPD is depicted in detail and the mechanism for the decomposition of the pseudo reward is explained. Sec. 4.4 provides the implementation details for solving the optimization problem.

4.1 Synergy pattern

With the awareness of interactions between agents' actions, we describe the perennial coordinated behaviour of agents by the *Synergy Pattern*. Previous methods [12, 4, 18, 22] utilize the structure of *graph* to depict it, whereas most of them focus on the local relationship between agent pairs towards a specific task. To get rid of the extrinsic reward signal, we propose *Synergy Pattern Graph* (SPG), denoted as $G^{sp}(\mathcal{V}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}, \zeta)$, where $v_i \in \mathcal{V}$ is the vertex for agent $i \in \mathcal{I}$ and the weight $\omega_{ij} \in \boldsymbol{\mathcal{W}}$ of edge $\{i, j\}$ is derived from a *synergy pattern function* which could depict agents' relative relations:

$$\zeta(\tau^i, \tau^j) : \mathcal{T} \times \mathcal{T} \to \mathbb{R}_0^+$$

The synergy pattern function follows a general formulation as any function with range $\mathbb{R}_0^+$, for instance, the information-theoretic influence [49]: $\zeta(\tau^i, \tau^j) := \mathrm{MI}^{\boldsymbol{\pi}}_{j|i}(o'^j; o^i, a^i | o^j, a^j)$ or the communication messages [4]. It can depend on more than merely the observations and joint actions as well, whereas we omit this for simplicity. At each time step, SPG varies depending on the history of the multi-agent system rather than being predefined before training. We will use G_t to denote $G^{sp}(\mathcal{V}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}, \zeta)$ at time step t in the following part of this paper for ease of notation.

Accordingly, the synergy pattern can be described by the distribution of SPG obtained by deploying a joint policy $\boldsymbol{\pi}$ in the environment: $\mathbb{P}(G^{\boldsymbol{\pi}} | \boldsymbol{\pi})$ with domain Λ, where Λ is the collection of all SPG.

4.2 Discrepancy of synergy patterns

We use the discrepancy between two distributions of SPG: $\mathbb{P}(G^{\boldsymbol{\pi}_1} | \boldsymbol{\pi_1})$ and $\mathbb{P}(G^{\boldsymbol{\pi}_2} | \boldsymbol{\pi_2})$ to denote the discrepancy between two synergy patterns. This process can be decomposed into two phases: i) measuring the distance between two samples from each distribution respectively; ii) extending the metric for a pair of samples to that for distributions. The former one is essentially to estimate the discrepancy between graphs. We adopt the Gromov-Wasserstein discrepancy of graphs, as mentioned in section 3.2, to measure this distance. Besides, we solve the latter one by applying optimal transport on SPG distributions.

This whole process leads to the following definition:

Definition 4.1. *(Discrepancy of Synergy Patterns) Let $p_1 = \mathbb{P}(G^{\boldsymbol{\pi}_1} | \boldsymbol{\pi_1})$ and $p_2 = \mathbb{P}(G^{\boldsymbol{\pi}_2} | \boldsymbol{\pi_2})$ be two distributions of SPG induced by joint polices $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$, respectively. Denote an joint distribution $\varrho \in \Gamma[p_1, p_2]$ as a bijective plan transporting probability mass from p_1 to p_2, where $\Gamma[p_1, p_2]$ is the set of all distributions on product space $\Lambda \times \Lambda$ with p_1 and p_2 as their marginal probabilities. Then the discrepancy of synergy patterns is defined as*

$$d_{sp}(p_1, p_2) = \inf_{\varrho \in \Gamma[p_1, p_2]} \int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2}) \, \mathrm{d}\varrho \tag{2}$$

where d_{gw} is the Gromov-Wasserstein discrepancy according to Eq. (1). Besides, we show that d_{sp} is a legitimate pseudometric on Λ and the proof is deferred to Appendix A.

However, solving Eq. (2) for the discrepancy of synergy patterns is almost intractable. In practice, we collect two SPG batches to approximate the discrepancy of synergy patterns. Here, we give the definition for this approximation:

Definition 4.2. *(Proximal Discrepancy of Synergy Patterns) Denote $\bar{\lambda}^{\boldsymbol{\pi}_1} = \{G_t^{\boldsymbol{\pi}_1}\}_{t=0}^{B_1} \sim \mathbb{P}(G^{\boldsymbol{\pi}_1} | \boldsymbol{\pi_1})$ and $\bar{\lambda}^{\boldsymbol{\pi}_2} = \{G_t^{\boldsymbol{\pi}_2}\}_{t=0}^{B_2} \sim \mathbb{P}(G^{\boldsymbol{\pi}_2} | \boldsymbol{\pi_2})$ sampled from the distributions of SPG, where $B_1, B_2 \in \mathbb{N}^+$. Assume $B_1 \leq B_2$ by symmetry. Let $D_{B_1 \times B_2} = \left[d_{gw}(G_{t_1}^{\boldsymbol{\pi}_1}, G_{t_2}^{\boldsymbol{\pi}_2})\right], \forall 0 \leq t_1 \leq B_1, 0 \leq t_2 \leq B_2$*

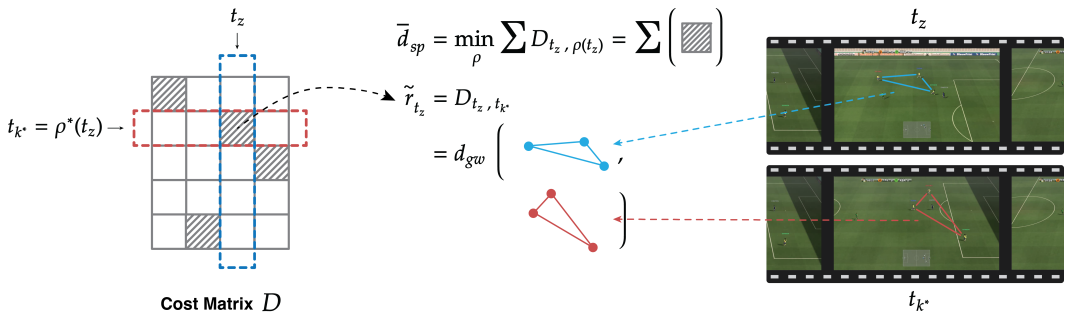

Figure 1: Illustration for the process of obtaining proximal discrepancy of synergy patterns $\bar{d}_{sp}$ (left) and the process of computing pseudo reward (right).

be the cost matrix. Suppose that an injective function $\rho : \{1, \ldots, B_1\} \rightarrow \{1, \ldots, B_2\}$ denotes an matching function. By minimizing the total cost, an approximation for the discrepancy of synergy patterns is:

$$\bar{d}_{sp}(\bar{\lambda}^{\boldsymbol{\pi}_1}, \bar{\lambda}^{\boldsymbol{\pi}_2}) = \min_{\rho} \sum_{t=0}^{B_1} D_{t,\rho(t)} \tag{3}$$

It is vital to point out that the SPG from $\bar{\lambda}^{\boldsymbol{\pi}_2}$ corresponding to $G_t^{\boldsymbol{\pi}_1}$ may not be $G_t^{\boldsymbol{\pi}_2}$ since the relations of agents may be analogous at different time steps. Hence, we search the whole $\bar{\lambda}^{\boldsymbol{\pi}_2}$ for $G_{\rho(t)}^{\boldsymbol{\pi}_2}$ that is relatively adjacent to $G_t^{\boldsymbol{\pi}_1}$ based on the Gromov-Wasserstein discrepancy.

4.3 Synergy pattern diversifying oriented unsupervised multi-agent reinforcement learning

Based on the discrepancy of synergy patterns, we can now delineate the framework for SPD. Suppose that $Z \in \mathbb{N}^+$ denotes the number of different synergy patterns. Our goal is to learn a set of joint policies $\{\boldsymbol{\pi}_z\}_{z=1}^{Z}$ with diverse synergy patterns to explore the environment sufficiently when there is no extrinsic reward. To increase the divergence among synergy patterns, we formulate the pseudo episode return for the episode data $\boldsymbol{\tau}_z$ produced by the joint policy $\boldsymbol{\pi}_z$:

$$\tilde{R}_z(\boldsymbol{\tau}_z, \{\boldsymbol{\tau}_k\}) = \min_{k \neq z, 1 \leq k \leq Z} \bar{d}_{sp}(\bar{\lambda}^{\boldsymbol{\pi}_z}, \bar{\lambda}^{\boldsymbol{\pi}_k}), \tag{4}$$

where $1 \leq z \leq Z$, and $\bar{\lambda}^{\boldsymbol{\pi}_z}$ is the SPG set derived from $\boldsymbol{\tau}_z$. Besides, we obtain $\bar{\lambda}^{\boldsymbol{\pi}_k}$ by sampling a batch of data from the replay buffer of $\boldsymbol{\pi}_k$ and ensure $\bar{\lambda}^{\boldsymbol{\pi}_k}$ have more elements: $|\bar{\lambda}^{\boldsymbol{\pi}_k}| \geq |\bar{\lambda}^{\boldsymbol{\pi}_z}|$.

Our method is to maximize the expected return: $\max_{\boldsymbol{\pi}_z} \mathbb{E}[\tilde{R}_z(\boldsymbol{\tau}_z, \{\boldsymbol{\tau}_k\})]$, resulting in the maximization of each discrepancy of one synergy pattern to its nearest counterpart. However, the pseudo episode return in Eq. (4) depending on the whole episode data $\boldsymbol{\tau}_z$ is sparse since it only derives a single final reward for one episode. This imposes the trouble for MARL algorithms to learn an accurate action-value function $Q^{\boldsymbol{\pi}_z}(s, \boldsymbol{a})$ and tends to impair the performance of value-based MARL algorithms. To bypass this predicament, intuitively, the pseudo episode return $\tilde{R}_z(\boldsymbol{\tau}_z, \{\boldsymbol{\tau}_k\})$ can be decomposed into pseudo rewards for every time step according to the optimal matching function ρ^*. Concretely, the pseudo reward at time step t can be formulated as:

$$\tilde{r}_t(\boldsymbol{\tau}_z, \{\boldsymbol{\tau}_k\}) = d_{gw}(G_t^{\boldsymbol{\pi}_z}, G_{\rho^*(t)}^{\boldsymbol{\pi}_{k^*}}), \quad G_t^{\boldsymbol{\pi}_z} \in \bar{\lambda}^{\boldsymbol{\pi}_z}, G_{\rho^*(t)}^{\boldsymbol{\pi}_{k^*}} \in \bar{\lambda}^{\boldsymbol{\pi}_{k^*}}, \tag{5}$$

where ρ^* is the optimal solution to $\bar{d}_{sp}(\bar{\lambda}^{\boldsymbol{\pi}_z}, \bar{\lambda}^{\boldsymbol{\pi}_k})$ by Eq. (3), and k^* denotes the id of the synergy pattern with minimal discrepancy to $\bar{\lambda}^{\boldsymbol{\pi}_z}$ according to Eq. (4). In other word, the pseudo reward is the Gromov-Wasserstein discrepancy of the SPG at current time step and the SPG from the counterpart of another synergy pattern.

We discriminate the concepts we proposed in Appendix D for a clearer understanding. The visualization for the pseudo reward derivation is shown in Fig. 1.

4.4 Implementation

We implement SPD with QMIX [35], learning a set of joint policies that maximize the discrepancy of synergy patterns to each other. QMIX is designed to have a mixing network structure, learning a global action-value function $Q := \{Q_{tot} | Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a}) = f_s\left(Q^1(\tau^1, a^1), \ldots, Q^n(\tau^n, a^n)\right), \frac{\partial f_s}{\partial Q^i} \geq 0, Q^i(\cdot,\cdot) \in \mathbb{R}\}$ which satisfies the IGM principle [44], to accomplish the cooperative task for agents.

During unsupervised learning, we select $1 \leq z \leq Z$ for the synergy pattern at the start of each episode randomly or in turn, and interact with the environment according to the joint policy $\boldsymbol{\pi}_z$ throughout the episode. In practice, the Euclidean distance of o^i and o^j is adopted as the weight ω_{ij} of edge $\{i, j\}$ to build the SPG, i.e., the formulation of synergy pattern function is $\zeta(\tau^i, \tau^j) = \|o^i - o^j\|_2$. We will show in Sec. 5 that, even with such a synergy pattern function to grab the simple information of relationships of agents, SPD can still learn useful synergy patterns such as pass-and-shoot in Google football Research and accelerate the learning procedure when endowed with the task-related reward.

Since solving for the Gromov-Wasserstein discrepancy between two measure graphs in Eq. (1) is a computationally costly process, we adopt [53]'s Regularized Proximal Gradient method to estimate it. We set $p = 2$ in Eq. (1) in our experiments. As for the probability measure, we set $\boldsymbol{\mu} = \frac{1}{n}\mathbf{1}_n$ since the agents are assumed to have equal contributions, where n is the number of agents or vertices in synergy patterns and $\mathbf{1}_n$ is a size n vector of ones. In the m-th iteration, the current optimal transport $\boldsymbol{T}^{(m)}$ can be updated by

$$\begin{aligned}\boldsymbol{T}^{(m+1)} &= \underset{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_\nu, \boldsymbol{\mu}_\varsigma)}{\operatorname{argmin}} \sum_{i,j \in \mathcal{V}_\nu} \sum_{i',j' \in \mathcal{V}_\varsigma} \left|\omega^\nu_{ij} - \omega^\varsigma_{i'j'}\right|^2 T^{(m)}_{ii'} T^{(m)}_{jj'} + \alpha \operatorname{KL}(\boldsymbol{T} \| \boldsymbol{T}^{(m)}) \\ &= \underset{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_\nu, \boldsymbol{\mu}_\varsigma)}{\operatorname{argmin}} \langle \boldsymbol{L}(\boldsymbol{\mathcal{W}}_\nu, \boldsymbol{\mathcal{W}}_\varsigma, \boldsymbol{T}^{(n)}), \boldsymbol{T} \rangle + \alpha \operatorname{KL}(\boldsymbol{T} \| \boldsymbol{T}^{(n)}),\end{aligned} \tag{6}$$

where $\boldsymbol{L}(\boldsymbol{\mathcal{W}}_\nu, \boldsymbol{\mathcal{W}}_\varsigma, \boldsymbol{T}^{(n)}) = \boldsymbol{\mathcal{W}}_\nu \boldsymbol{\mu}_\nu \mathbf{1}^\top_{|\mathcal{V}_\varsigma|} + \mathbf{1}_{|\mathcal{V}_\nu|} \boldsymbol{\mu}_\varsigma^\top \boldsymbol{\mathcal{W}}_\varsigma^\top - 2\boldsymbol{\mathcal{W}}_\nu \boldsymbol{T} \boldsymbol{\mathcal{W}}_\varsigma^\top$, and Kullback-Leibler (KL) divergence is added as the proximal term. $\langle \cdot, \cdot \rangle$ denotes the inner product of two matrices and α is a hyper-parameter to balance two items. The Eq. (6) can then be solved with nearly-linear convergence [2] by the Sinkhorn-Knopp algorithm [43, 8]. As soon as $\boldsymbol{T}^{(m)}$ converges to a stable optimal transport $\hat{\boldsymbol{T}}$, the estimation of the Gromov-Wasserstein discrepancy can be calculated by plugging $\hat{\boldsymbol{T}}$ into Eq. (1).

To solve Eq. (3), we utilize Kuhn–Munkres algorithm [29], which could solve such an assignment problem in polynomial time, to obtain the optimal matching funciton ρ^*. Therefore, the complete formula for pseudo reward at time step t is as below:

$$\tilde{r}_t(\boldsymbol{\tau}_z, \{\boldsymbol{\tau}_k\}) = d_{gw}(G^{\boldsymbol{\pi}_z}_t, G^{\boldsymbol{\pi}_{k^*}}_{\rho^*(t)}) = \left(\sum_{i,j \in \mathcal{V}^{\boldsymbol{\pi}_z}_t} \sum_{i',j' \in \mathcal{V}^{\boldsymbol{\pi}_{k^*}}_{\rho^*(t)}} \left|\omega^{\boldsymbol{\pi}_z}_{t,(ij)} - \omega^{\boldsymbol{\pi}_{k^*}}_{\rho^*(t),(i'j')}\right|^2 \hat{T}_{ii'} \hat{T}_{jj'} \right)^{\frac{1}{2}}, \tag{7}$$

where $G^{\boldsymbol{\pi}_z}_t \in \bar{\lambda}^{\boldsymbol{\pi}_z}, G^{\boldsymbol{\pi}_{k^*}}_{\rho^*(t)} \in \bar{\lambda}^{\boldsymbol{\pi}_{k^*}}$, and k^* denotes the id of the synergy pattern with minimal discrepancy to $\bar{\lambda}^{\boldsymbol{\pi}_z}$ according to Eq. (4). The whole framework is summarized in Algorithm 1.

5 Experiments

In this section, we first compare properties of SPD with that of other URL baselines on Multi-agent Particle Environment (MPE) [23] in Sec. 5.1. And in Sec. 5.2, we evaluate the generalization ability of our method on downstream tasks on Google Research Football (GRF) [19]. Our code is available at `https://github.com/thu-rllab/SPD`.

5.1 Synergy pattern diversifying

We demonstrate that our method SPD is capable of capturing more inherent properties of multi-agent systems such as the impacts from other agents, comparing to conventional URL algorithms. In order

Algorithm 1 SPD

Input: Z, t_{start}. Empty graph buffers $\{\mathcal{D}_k\}_{k=1}^{Z}$. Empty replay buffers $\{\mathcal{D}_k^{RL}\}_{k=1}^{Z}$. The synergy pattern function ζ as described in section 4.1. Maximal time steps for each episode t_{max}.
Parameter: joint policy networks $\{\boldsymbol{\pi}_{\theta_k}\}_{k=1}^{Z}$

Initialize $t_{\text{env}} = 0$, $\{\theta_k\}_{k=1}^{Z}$.
for *each episode* **do**
  Select z randomly or in turn. $\boldsymbol{\tau}_z \leftarrow \{\}$, $\bar{\lambda}^{\boldsymbol{\pi}_z} \leftarrow \{\}$;
  while not *done* **do**
    Interact with the environment by $\boldsymbol{\pi}_z$ to get the tuple $(s, \boldsymbol{a}, s')$;
    $\boldsymbol{\tau}_z \leftarrow \boldsymbol{\tau}_z \cup (s, \boldsymbol{a}, s')$.
    Build SPG G with $\zeta(\boldsymbol{\tau}_z)$;
    $\bar{\lambda}^{\boldsymbol{\pi}_z} \leftarrow \bar{\lambda}^{\boldsymbol{\pi}_z} \cup \{G\}, \mathcal{D}_z \leftarrow \mathcal{D}_z \cup \{G\}$;
    $t_{\text{env}} \leftarrow t_{\text{env}} + 1$;
  end while
  if $t_{\text{env}} \geq t_{\text{start}}$ **then**
    for *each* $k \neq z$ **do**
      $\bar{\lambda}^{\boldsymbol{\pi}_k} \sim \mathcal{D}_k$.
      # via Sinkhorn − Knopp algorithm
      Obtain $d_{gw}(G^{\boldsymbol{\pi}_z}, G^{\boldsymbol{\pi}_k})$ according to Eq. (6) as well as Eq. (1);
      # via Kuhn − Munkres algorithm
      Obtain $\bar{d}_{sp}^{k}(\bar{\lambda}^{\boldsymbol{\pi}_z}, \bar{\lambda}^{\boldsymbol{\pi}_k})$ by solving Eq. (3) for ρ^*;
    end for
    $k^* \leftarrow \text{argmin}_k \, \bar{d}_{sp}^{k}$;
    Calculate $\tilde{r}$ by plugging k^*, ρ^{k^*} into Eq. (7);
    $\mathcal{D}_z^{RL} \leftarrow \mathcal{D}_z^{RL} \cup \{\boldsymbol{\tau}_z, \{\tilde{r}\}_{t=1}^{t_{\text{max}}}\}$;
    Use an MARL algorithm to train θ_z with $\mathcal{D}_z^{RL}$.
  end if
end for

to verify this claim, we first evaluate the diversity of coordination policies learned by SPD and URL baselines in Multi-agent Particle Environment[2] [23, 47]. We customize the scenario *SimpleTag*, which is a predator-prey environment, for URL experiments by discarding the extrinsic reward from the environment. Instead, we are concerned about the diversity of agents' formations driven by learned coordination policies. The detailed description of the scenario please refer to Appendix B.1.

Baselines and evaluation We compare our method SPD with several conventional single-agent URL approaches. Here is the details about the implementation of URL algorithms, which we carried out to learn policies without external reward:

- SPD: we follow the narration of the implementation details in Sec. 4.4. The pseudo-reward is derived via Eq. (1)∼(7).
- Diversity Is All You Need (DIAYN) [10]: The joint policy $\boldsymbol{\pi}(\boldsymbol{a}|\boldsymbol{\tau}_z, z)$ is conditioned on a discrete latent variable z sampled from a pre-defined categorical distribution $p(z)$: $z \sim p(z)$. The pseudo-reward with formulation as $r_z := \log q_\phi(z|\{o^i\}_{i=1}^{n}) - \log p(z)$ is used for inspiring agents to visit states that can be distinguished easily, where $q_\phi(z|\{o^i\}_{i=1}^{n})$ is a learned discriminator. In our experiment, we replace the actor network $\boldsymbol{\pi}(\boldsymbol{a}|\boldsymbol{\tau}_z, z)$ shared by all coordination policies with Z individual networks $\{\boldsymbol{\pi}_z\}_{z=1}^{Z}$, considering the large observation space and action space.
- Wasserstein Unsupervised Reinforcement Learning (WURL) [15]: The pseudo-reward is designed to enhance the distance between different policies, and it is obtained by estimating the Wasserstein distance of the induced state distributions: $r_z = W(\boldsymbol{\tau}_z, \bar{\boldsymbol{\tau}}_k)$, where $\boldsymbol{\tau}_z$ is the trajectory driven by current joint policy $\boldsymbol{\pi}_z(\boldsymbol{a}|\boldsymbol{\tau}_z)$ and $\bar{\boldsymbol{\tau}}_k$ is a sampled batch from another joint policy $\boldsymbol{\pi}_k$.

[2] https://github.com/Farama-Foundation/PettingZoo/tree/master/pettingzoo/mpe

The URL algorithms are implemented with QMIX to learn diverse joint policies $\{\boldsymbol{\pi}_z\}_{z=1}^{Z}$, and we set $Z = 10$ in practice. The hyper-parameters are kept to be the same, and please refer to Appendix B.2 for details. For evaluation, each algorithm is carried out with 5 different random seeds and trained with 4×10^6 steps interacting with the environment. All the intrinsic pseudo-rewards are derived based on the positions of agents, which is a part of the original observation space.

Since DIAYN and WURL are designed for the single agent case, we deploy them by regarding all agents as a single agent. Concretely, we use observations from all agents to create the features that DIAYN and WURL need.

Method	DSR(%)	Normalized $\bar{d}_{sp}$
DIAYN	59.9 ± 0.9	$(4.0 \pm 2.7) \times 10^{-2}$
WURL	58.7 ± 1.5	$(3.9 \pm 3.6) \times 10^{-3}$
SPD (Ours)	58.6 ± 0.7	1.0 ± 0.1

Table 1: The discriminator success rate (DSR) and normalized proximal discrepancy of synergy patterns $\bar{d}_{sp}$ for URL methods DIAYN, WURL and SPD (Ours).

We evaluate these algorithms following the same protocol: (i) collect sample transitions except extrinsic reward using each joint policy $\boldsymbol{\pi}_z(\boldsymbol{a}|\boldsymbol{\tau}_z)$, (ii) calculate the pseudo-rewards $\{r_z\}$ for each time step and then assign them to the trajectory $\boldsymbol{\tau}_z$, (iii) employ learned joint policies once the training procedure is completed to collect $\{\boldsymbol{\tau}_z^{\text{final}}\}_{z=1}^{Z}$ for evaluation. We use two metrics to evaluate how diverse are the learned joint policies from different aspects: *the diversity of states* and *the discrepancy among relative relationships of agents* induced by $\{\boldsymbol{\pi}_z\}_{z=1}^{Z}$. As for the former one, we adopt the structure of the discriminator $q_\phi(z|\{o^i\}_{i=1}^{n})$ from DIAYN, which is a learned neural network to classify each policy $\boldsymbol{\pi}_z$ from state samples in $\{\boldsymbol{\tau}_z^{\text{final}}\}_{z=1}^{Z}$, and use the success rate of the discriminator to measure how diverse are the states, namely Discriminator Success Rate (DSR). As discussed in Sec. 4.2, the discrepancy of synergy patterns d_{sp} according to Eq. (2) is an effective measure for the latter one. Table 1 shows that the performance of all algorithms on DSR is close, whereby SPD significantly outperforms the others on the discrepancy of synergy patterns. Such results demonstrate that SPD can inspire agents to visit comparably identifiable states as the conventional URL approaches, while further motivating agents to explore with other types of coordination policies in the meanwhile.

Ablation study We examine the following components of our method in detail to show their effects: (i) Reward scale β_r: we multiply the pseudo-reward in Eq. (5) with a factor vary in $\{0.1, 1.0, 10.0\}$ to change the scale of the final returns received by agents, and 10.0 is the default value in the aforementioned experiments. (ii) Sparse return $\tilde{R}_z$: To evaluate the effect of the decomposition of the pseudo-return in Eq. (4), we directly use it as the pseudo-reward of the last step in the episode. (iii) On-policy $\rho(t) = t$: As discussed in Sec. 4.2 that the SPG from $\bar{\lambda}^{\boldsymbol{\pi}_2}$ corresponding to $G_t^{\boldsymbol{\pi}_1}$ may not be $G_t^{\boldsymbol{\pi}_2}$. To verify this claim, we set $\rho(t) = t$ directly and the pseudo-reward becomes to $\tilde{r}_t^{(\text{no match})}(\boldsymbol{\tau}_z, \{\boldsymbol{\tau}_k\}) = d_{gw}(G_t^{\boldsymbol{\pi}_z}, G_t^{\boldsymbol{\pi}_{k^*}})$, where $G_t^{\boldsymbol{\pi}_z} \in \bar{\lambda}^{\boldsymbol{\pi}_z}, G_t^{\boldsymbol{\pi}_{k^*}} \in \bar{\lambda}^{\boldsymbol{\pi}_{k^*}}$. (iv) Sub-optimal $\bar{\rho}$: The matching function ρ^* is obtained by Kuhn–Munkres algorithm [29]. Therefore, the precision is influenced by the number of iterations, which defaults to $\mathcal{N}_{\text{KM}}^{\text{iter}} = 100$ in the experiments. A lower number of iterations $\mathcal{N}_{\text{KM}}^{\text{iter}} = 25$ is used to show the influence from the precision of the matching function.

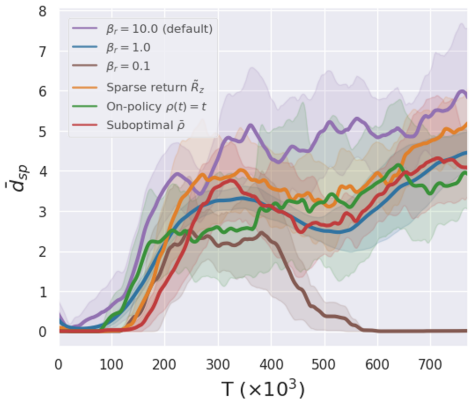

Figure 2: Ablation study. We change the scaling factor β_r multiplying the pseudo-reward from [0.1, 1.0, 10.0]. "Sparse return $\tilde{R}_z$" means using Eq. (4) as final the reward directly. "On-policy $\rho(t) = t$" means we apply no optimal transport and set $\rho(t) = t$ in Eq. (3) to suit the on-policy manner computing pseudo-reward on-the-fly. And we reduce the precision of the solution for Eq. (6) in "Sub-optimal $\bar{\rho}$".

We use $\bar{d}_{sp}$ as the evaluating indicator and the results is shown in Fig. 2. The curves start from the time step that the exploration rate epsilon has stopped annealing. We can summarize that the reward scale plays an important role in learning good synergy patterns. Canceling the optimal transport for

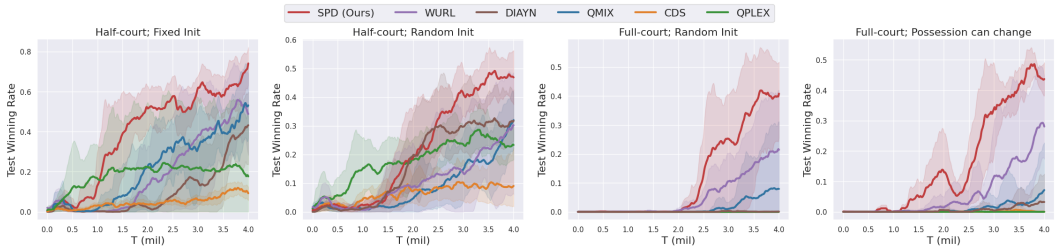

Figure 3: Compare the performances of using one model trained by URL algorithms as the parameter initialization for four different tasks with learning by traditional MARL algorithms directly.

matching or decreasing its accuracy both harm the performance. And the "Sparse return $\tilde{R}_z$" curve shows that amortizing the episode return to each time step simplifies the training.

5.2 Generalizability of learned synergy patterns

In this part we demonstrate the generalization ability of the models trained by SPD. Specifically, we first train SPD on the complicated MARL environment: Google Research Football [19] without environment reward. Then we use the trained model as the initialization for different tasks with specific rewards, and use common MARL algorithms to train it.

Task description We choose the map "academy_3_vs_1_with_keeper" with the setting from CDS [5], where three players need to cooperate to shoot and get a score under the defense of an opposing player and a goalkeeper. To simplify the training, the original environment ends an episode if the ball or the player return to the left half-court. Players always start at the same place. Losing the control of the ball also cause the failure. The team is rewarded 100 at the end of an episode when there is a goal, otherwise -1. On the basis of this scenario which we call "Half-court; Fixed Init", we design a series of tasks with more difficulty. (i) We initialize the agents randomly in a circle with 5% court width's radius, which increases the difficulty by introducing more randomness. (ii) We remove the restrictions on the left-half court, which greatly increases the exploration space. (iii) The possession change will not end the episode, which may increase the percent of the bad samples in the replay buffer, since at the beginning of training the ball is easily robbed by the preset AI opponent.

Utilizing SPD model We first train the models by SPD with $Z = 20$ on the "Half-court; Possession can change" scenario without the environment reward for 10^6 time steps, as well as WURL [15] and DIAYN [10]. Then we use the same trained parameters as the initialization for QMIX [35] on four different downstream scenarios. In contrast to the manual selection of the synergy patterns, we test them on the downstream task before MARL training, choosing the parameters of the model with the highest average return as the model initialization. We show the results in Fig. 3, together with the SOTA on GRF including QMIX [35], QPLEX [48] and CDS [5]. The results show that using the model learned by conventional single-agent URL methods (WURL and DIAYN) as the initialization mostly performs similarly to the baseline QMIX, while WURL slightly outperforms QMIX on the "Full-court" maps. In contrast, SPD learns faster and finally gets higher winning rate on all scenarios, demonstrating the effectiveness of d_{sp} as an evaluation metric for the difference of agents' relationship and showing that SPD embeds the model with stronger generalization ability. Actually, one difference between SPD and these URL methods is that SPD encourages the diversity of the relationship among agents while these diversity methods encourages the diversity of visited states. Therefore, the models trained by WURL may reach diverse states on the 'Full-court' maps which need more exploration and we believe this accounts for the slightly better performance.

Visualization of the learned synergy patterns To comprehensively evaluate SPD, we visualize[3] the trajectories of all the skills and show three of them in Fig. 4. The trajectories of other skills can be found in Appendix C. We find that agents can perform some meaningful behavior from a human perspective, even without the supervision of the environment reward. Agents can dribble for a long

[3]The videos for all synergy patterns can be found at: `https://sites.google.com/view/spd-umarl`

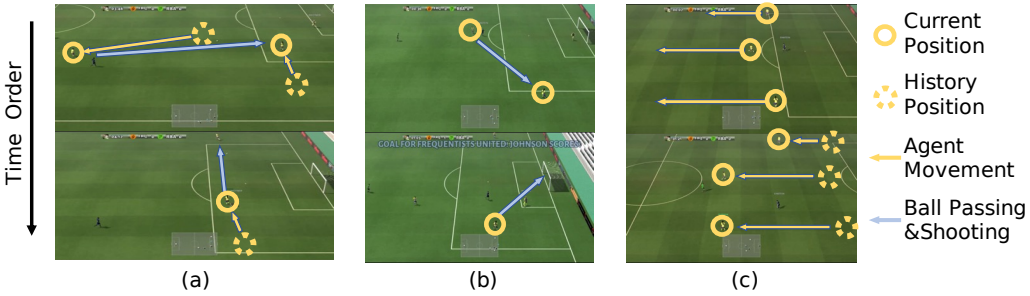

Figure 4: Visualization of the learned synergy patterns without environment reward. (a) Dribbling and long passing. (b) Passing and shooting. (c) Running back to play defense.

distance, pass between teammates, shoot, and run back together in different skills. To our concern that the different performance of different joint policies learned by SPD are only due to the randomness of the GRF environment itself but not the inter-policy diversity we want, we deploy each learned joint policy for 10 episodes and visualize them. The results show that each joint policy exhibits the similar coordination behavior across episodes. The visualization shows that making the team coordinate more diverse drives agents to learn meaningful behaviour, which is favorable for downstream tasks.

6 Discussion

6.1 Limitations

There are still some limitations or difficulties with further applications. Though the discriminator success rate (DSR) used in Sec. 5.1 is not a good measure for relationships of agents, it does evaluate the diversity of visited states in some ways. The results on DSR show that all of the URL approaches, including our method SPD, are far away from satisfying, implying that there is still space for improvement. Furthermore, a synergy pattern function ζ that embeds more information than the one used in this paper is tend to require higher computational cost. Moreover, incorporating diversity-based exploration technique with SPD may improve the intra-agent diversity as well as inter-policy diversity. These limitations may suggest further research directions.

6.2 Negative societal impact

Due to the domain gap between virtual environments and real-world scenarios, applying our method directly to some real-world applications, such as traffic management, is risky and may result in damage to the system. To avoid harmful operations, human supervision is required to restrict the learned models.

7 Conclusion

In this work, we propose a general unsupervised framework *Synergy Pattern Diversifying Oriented Unsupervised Multi-agent Reinforcement Learning* (SPD) to learn useful synergy patterns for agents. Synergy pattern graph (SPG) is devised to depict the relationship among all agents. Furthermore, we approximate the discrepancy of synergy patterns by applying optimal transport on the batches sampled from the distributions of SPG. In our experiments, we show that SPD surpasses conventional URL approaches on capturing the relative relations of agents and learns useful synergy patterns with no task-related reward. We also empirically demonstrate that the learned policies can accelerate the downstream training procedure and outperform state-of-the-art algorithms in GRF.

8 Acknowledgement

This work was supported by the National Key R&D Program of China under Grant 2018AAA0102801, National Natural Science Foundation of China under Grant 61620106005.

References

[1] Achiam, J., Edwards, H., Amodei, D., and Abbeel, P. Variational option discovery algorithms. *arXiv preprint arXiv:1807.10299*, 2018.

[2] Altschuler, J., Niles-Weed, J., and Rigollet, P. Near-linear time approximation algorithms for optimal transport via sinkhorn iteration. *Advances in neural information processing systems*, 30, 2017.

[3] Baumli, K., Warde-Farley, D., Hansen, S., and Mnih, V. Relative variational intrinsic control. *arXiv preprint arXiv:2012.07827*, 2020.

[4] Böhmer, W., Kurin, V., and Whiteson, S. Deep coordination graphs. In *International Conference on Machine Learning*, pp. 980–991. PMLR, 2020.

[5] Chenghao, L., Wang, T., Wu, C., Zhao, Q., Yang, J., and Zhang, C. Celebrating diversity in shared multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[6] Chowdhury, S. and Mémoli, F. The gromov-wasserstein distance between networks and stable network invariants. *CoRR*, abs/1808.04337, 2018. URL `http://arxiv.org/abs/1808.04337`.

[7] Claus, C. and Boutilier, C. The dynamics of reinforcement learning in cooperative multiagent systems. In *AAAI Conference on Artificial Intelligence (AAAI)*, 1998.

[8] Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[9] Dong, Y. and Sawin, W. Copt: Coordinated optimal transport on graphs. *Advances in Neural Information Processing Systems*, 33:19327–19338, 2020.

[10] Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=SJx63jRqFm`.

[11] Gregor, K., Rezende, D. J., and Wierstra, D. Variational intrinsic control. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

[12] Guestrin, C., Lagoudakis, M. G., and Parr, R. Coordinated reinforcement learning. In Sammut, C. and Hoffmann, A. G. (eds.), *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002*, pp. 227–234. Morgan Kaufmann, 2002.

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

[14] Hadfield-Menell, D., Milli, S., Abbeel, P., Russell, S. J., and Dragan, A. Inverse reward design. *Advances in neural information processing systems*, 30, 2017.

[15] He, S., Jiang, Y., Zhang, H., Shao, J., and Ji, X. Wasserstein unsupervised reinforcement learning. *arXiv preprint arXiv:2110.07940*, 2021.

[16] Hu, J., Jiang, S., Harding, S. A., Wu, H., and wei Liao, S. Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning. 2021.

[17] Hüttenrauch, M., Šošić, A., and Neumann, G. Guided deep reinforcement learning for swarm systems. *arXiv preprint arXiv:1709.06011*, 2017.

[18] Jiang, J., Dun, C., Huang, T., and Lu, Z. Graph convolutional reinforcement learning. *arXiv preprint arXiv:1810.09202*, 2018.

[19] Kurach, K., Raichuk, A., Stańczyk, P., Zając, M., Bachem, O., Espeholt, L., Riquelme, C., Vincent, D., Michalski, M., Bousquet, O., et al. Google research football: A novel reinforcement learning environment. *arXiv preprint arXiv:1907.11180*, 2019.

[20] Lee, L., Eysenbach, B., Parisotto, E., Xing, E., Levine, S., and Salakhutdinov, R. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*, 2019.

[21] Liu, H. and Abbeel, P. Behavior from the void: Unsupervised active pre-training. *Advances in Neural Information Processing Systems*, 34, 2021.

[22] Liu, Y., Wang, W., Hu, Y., Hao, J., Chen, X., and Gao, Y. Multi-agent game abstraction via graph attention neural network. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[23] Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Abbeel, O. P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems (NeurIPS)*, 2017.

[24] Lupu, A., Cui, B., Hu, H., and Foerster, J. Trajectory diversity for zero-shot coordination. In *International Conference on Machine Learning*, pp. 7204–7213. PMLR, 2021.

[25] Mahajan, A., Rashid, T., Samvelyan, M., and Whiteson, S. Maven: Multi-agent variational exploration. *Advances in Neural Information Processing Systems*, 32, 2019.

[26] Mémoli, F. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011.

[27] Mena, G., Belanger, D., Linderman, S., and Snoek, J. Learning latent permutations with gumbel-sinkhorn networks. *arXiv preprint arXiv:1802.08665*, 2018.

[28] Moravčík, M., Schmid, M., Burch, N., Lisỳ, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

[29] Munkres, J. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

[30] Oliehoek, F. A., Amato, C., et al. *A concise introduction to decentralized POMDPs*. Springer, 2016.

[31] Parker-Holder, J., Pacchiano, A., Choromanski, K. M., and Roberts, S. J. Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems*, 33:18050–18062, 2020.

[32] Pathak, D., Gandhi, D., and Gupta, A. Self-supervised exploration via disagreement. In *International conference on machine learning*, pp. 5062–5071. PMLR, 2019.

[33] Petric Maretic, H., El Gheche, M., Chierchia, G., and Frossard, P. Got: an optimal transport framework for graph comparison. *Advances in Neural Information Processing Systems*, 32, 2019.

[34] Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[35] Rashid, T., Samvelyan, M., De Witt, C. S., Farquhar, G., Foerster, J., and Whiteson, S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. *arXiv preprint arXiv:1803.11485*, 2018.

[36] Rashid, T., Farquhar, G., Peng, B., and Whiteson, S. Weighted qmix: Expanding monotonic value function factorisation. *arXiv e-prints*, pp. arXiv–2006, 2020.

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

[38] Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

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

[40] Sharma, A., Gu, S., Levine, S., Kumar, V., and Hausman, K. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

[41] Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., and Borgwardt, K. Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics*, pp. 488–495. PMLR, 2009.

[42] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[43] Sinkhorn, R. and Knopp, P. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

[44] Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2019.

[45] Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V. F., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2018.

[46] Tan, M. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *International Conference on Machine Learning (ICML)*, 1993.

[47] Terry, J., Black, B., Grammel, N., Jayakumar, M., Hari, A., Sullivan, R., Santos, L. S., Dieffendahl, C., Horsch, C., Perez-Vicente, R., et al. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:15032–15043, 2021.

[48] Wang, J., Ren, Z., Liu, T., Yu, Y., and Zhang, C. Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062*, 2020.

[49] Wang, T., Wang, J., Wu, Y., and Zhang, C. Influence-based multi-agent exploration. *arXiv preprint arXiv:1910.05512*, 2019.

[50] Wang, T., Dong, H., Lesser, V., and Zhang, C. Roma: Multi-agent reinforcement learning with emergent roles. In *International Conference on Machine Learning (ICML)*, 2020.

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

[52] Wang, T., Zeng, L., Dong, W., Yang, Q., Yu, Y., and Zhang, C. Context-aware sparse deep coordination graphs. *arXiv preprint arXiv:2106.02886*, 2021.

[53] Xu, H., Luo, D., and Carin, L. Scalable gromov-wasserstein learning for graph partitioning and matching. *Advances in neural information processing systems*, 32, 2019.

[54] Xu, Z., Lyu, Y., Pan, Q., Hu, J., Zhao, C., and Liu, S. Multi-vehicle flocking control with deep deterministic policy gradient method. In *International Conference on Control and Automation (ICCA)*, 2018.

[55] Zhou, Z., Fu, W., Zhang, B., and Wu, Y. Continuously discovering novel strategies via reward-switching policy optimization. *arXiv preprint arXiv:2204.02246*, 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] .
   (b) Did you describe the limitations of your work? [Yes] See Sec. 6.1.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Sec. 6.2.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] .
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] .
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We have opened the source code.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] .
   (b) Did you mention the license of the assets? [N/A] .
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A] .
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] .
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] .
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] .
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] .
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] .

A Properties of the discrepancy of synergy patterns as a legitimate pseudometric

Proposition A.1. *By the definition 4.1, d_{sp} is a legitimate pseudometric on Λ which can be proved by examining following properties:*

i) $d_{sp}(p_1, p_2) \geq 0$,

Proof. As Chowdhury & Mémoli [6] show, the Gromov-Wasserstein discrepancy is a pseudometric on $\mathcal{G}$, where $\mathcal{G}$ denotes the collection of measure graphs, leading to $d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2}) \geq 0$. Following the definition in Eq. (2), proving $d_{sp}(p_1, p_2) \geq 0$ is trivial since all elements are non-negative. □

ii) $d_{sp}(p, p) = 0$,

Proof. Similarly, based on $d_{gw}(G^{\boldsymbol{\pi}}, G^{\boldsymbol{\pi}}) = 0$, we can choose ϱ carefully as ϱ_0 with $\mathbb{P}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2} = G) = \delta(G)$ and $\mathbb{P}(G^{\boldsymbol{\pi}_1} = G, G^{\boldsymbol{\pi}_2}) = \delta(G)$ everywhere, where δ is the Dirac distribution. Hence we have:

$$d_{sp}(p, p) = \left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2})\, \mathrm{d}\varrho\right]_{\varrho=\varrho_0} = \int 0 \cdot \mathrm{d}\varrho = 0$$

□

iii) $d_{sp}(p_1, p_2) = d_{sp}(p_2, p_1)$,

Proof. To avoid unnecessary confusion, we notate the joint distribution of the L.H.S as ϱ_l and that of the R.H.S. as ϱ_r, respectively. Denote $\varrho_l = \varrho_{12}$ as one of the joint distributions when the infimum of the L.H.S. is reached:

$$d_{sp}(p_1, p_2) = \left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2})\, \mathrm{d}\varrho_l\right]_{\varrho_l=\varrho_{12}}$$

We show that the infimum of the R.H.S. is reached when $\varrho_r = \varrho_{21}$ by contradictions, where ϱ_{21} can be obtained by exchanging the probabilities of $G^{\boldsymbol{\pi}_1}$ and $G^{\boldsymbol{\pi}_2}$ in ϱ_{12}. Suppose there exists $\varrho_r = \varrho'_{21}$ such that:

$$\left[\int d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_1})\, \mathrm{d}\varrho_r\right]_{\varrho_r=\varrho'_{21}} < \left[\int d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_1})\, \mathrm{d}\varrho_r\right]_{\varrho_r=\varrho_{21}}$$

Then we can update the joint distribution for the L.H.S. with $\varrho_l = \varrho'_{12}$ by exchanging the probabilities of $G^{\boldsymbol{\pi}_1}$ and $G^{\boldsymbol{\pi}_2}$ in ϱ'_{21} similarly. Based on the property of the Gromov-Wasserstein discrepancy: $d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2}) = d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_1})$, we have:

$$\begin{aligned}
\left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2})\, \mathrm{d}\varrho_l\right]_{\varrho_l=\varrho'_{12}} &= \left[\int d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_1})\, \mathrm{d}\varrho_r\right]_{\varrho_r=\varrho'_{21}} \\
&< \left[\int d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_1})\, \mathrm{d}\varrho_r\right]_{\varrho_r=\varrho_{21}} \\
&= \left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2})\, \mathrm{d}\varrho_l\right]_{\varrho_l=\varrho_{12}}
\end{aligned}$$

This contradicts the claim that ϱ_l is one of the joint distributions when the infimum of the L.H.S. is reached. Hence, we can derive:

$$\begin{aligned}
d_{sp}(p_2, p_1) &= \inf_{\varrho_r \in \Gamma[p_2, p_1]} \int d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_1})\, \mathrm{d}\varrho_r \\
&= \left[\int d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_1})\, \mathrm{d}\varrho_r\right]_{\varrho_r=\varrho_{21}} \\
&= \left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2})\, \mathrm{d}\varrho_l\right]_{\varrho_l=\varrho_{12}} \\
&= d_{sp}(p_1, p_2)
\end{aligned}$$

Name	Description	Value
γ	Discounted factor	0.99
ε anneal time	Time-steps for ε to anneal from ε_{s} to ε_{f}, where ε is the probability for agents choosing random actions.	100000
ε_{s}	Initial ε at start	1
ε_{f}	Final ε	0.05
$\mathcal{N}_{\text{env}}$	The number of parallel environments	1
$\lvert\mathcal{D}^{RL}\rvert$	The size of the replay buffer for MARL learning	5000
$\mathcal{N}_{\text{rnn}}$	Dimension of RNN cells	256
lr	Learning rate	0.001
$\mathcal{N}_{\text{batch}}$	Batch size	128
t_{target}	Time interval for updating the target network	200
G_{max}	Clipping value for all gradients	10
$\lvert\mathcal{D}\rvert$	The size of the graph buffer for SPD	10000
t_{start}	The start steps for employing SPD to obtain pseudo-reward	5000
α	The factor of the regularized term in Eq. (6)	0
B_1, B_2	The size of the SPG batches	50
$\mathcal{N}^{\text{iter}}_{\text{Sinkhorn}}$	The number of the iterations for Sinkhorn-Knopp algorithm	50
$\mathcal{N}^{\text{iter}}_{\text{KM}}$	The number of the iterations for Kuhn–Munkres algorithm	100

Table 2: Hyper-parameters.

which finishes the proof. □

iv) $d_{sp}(p_1, p_\chi) + d_{sp}(p_2, p_\chi) \geq d_{sp}(p_1, p_2)$.

Proof. We prove the triangle inequality by contradictions similar to iii). Suppose that $\exists p_\chi$, s.t. $d_{sp}(p_1, p_\chi) + d_{sp}(p_2, p_\chi) < d_{sp}(p_1, p_2)$. Denote the joint distributions for the infimum of the discrepancy of synergy patterns as $\varrho_{1\chi}$, $\varrho_{2\chi}$ and ϱ_{12} respectively. Let $\varrho_{\chi 2}$ be the joint distribution by exchanging the probability of $G^{\boldsymbol{\pi}_2}$ and $G^{\boldsymbol{\pi}_\chi}$. Then we can use $\varrho_{1\chi}$ and $\varrho_{\chi 2}$ to find the probabilities of $G^{\boldsymbol{\pi}_1}$ and $G^{\boldsymbol{\pi}_2}$ corresponding to the same $G^{\boldsymbol{\pi}_\chi}$, obtaining the new joint distribution ϱ'_{12}. By the triangle inequality of the Gromov-Wasserstein discrepancy, we can write:

$$
\begin{aligned}
\left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_2})\,\mathrm{d}\varrho\right]_{\varrho=\varrho'_{12}} &\leq \left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_\chi})\,\mathrm{d}\varrho\right]_{\varrho=\varrho_{1\chi}} + \left[\int d_{gw}(G^{\boldsymbol{\pi}_\chi}, G^{\boldsymbol{\pi}_2})\,\mathrm{d}\varrho\right]_{\varrho=\varrho_{\chi 2}} \\
&= \left[\int d_{gw}(G^{\boldsymbol{\pi}_1}, G^{\boldsymbol{\pi}_\chi})\,\mathrm{d}\varrho\right]_{\varrho=\varrho_{1\chi}} + \left[\int d_{gw}(G^{\boldsymbol{\pi}_2}, G^{\boldsymbol{\pi}_\chi})\,\mathrm{d}\varrho\right]_{\varrho=\varrho_{2\chi}} \\
&= \quad d_{sp}(p_1, p_\chi) + d_{sp}(p_2, p_\chi) \\
&< \quad d_{sp}(p_1, p_2)
\end{aligned}
$$

which contradicts the claim that $d_{sp}(p_1, p_2)$ is the infimum. Hence we have $\forall p_\chi, d_{sp}(p_1, p_\chi) + d_{sp}(p_2, p_\chi) \geq d_{sp}(p_1, p_2)$. □

B Details for experiments and reproducibility

B.1 Description for the MPE environment in Sec. 5.1

Multi-agent Particle Environment (MPE) [23, 47] consists of n agents and l landmarks in a 2D world. Each agent has to resolve to select the action from its discrete action space to move around. In practice, the experiments in Sec. 5.1 are carried out in the customed scenario *SimpleTag*, which is a predator-prey environment with $n = 4, l = 1$. In the original scenario, there are 1 good agent that can move faster to keep itself away from the others and 3 adversaries trying to hit the good agent. Besides, there is a obstacle blocking the way denoted by 1 landmark. To customize the environment for URL experiments, we discard the extrinsic reward from the environment. Instead, we are concerned about the diversity of agents' formations driven by learned coordination policies.

B.2 Details about hardware and reproducibility

The hardware An AMD Ryzen 3975WX CPU with 32-Cores and three RTX-3090-11G GPUs are employed to run all the experiments with five random seeds. As for MPE, it takes around 10 hours for

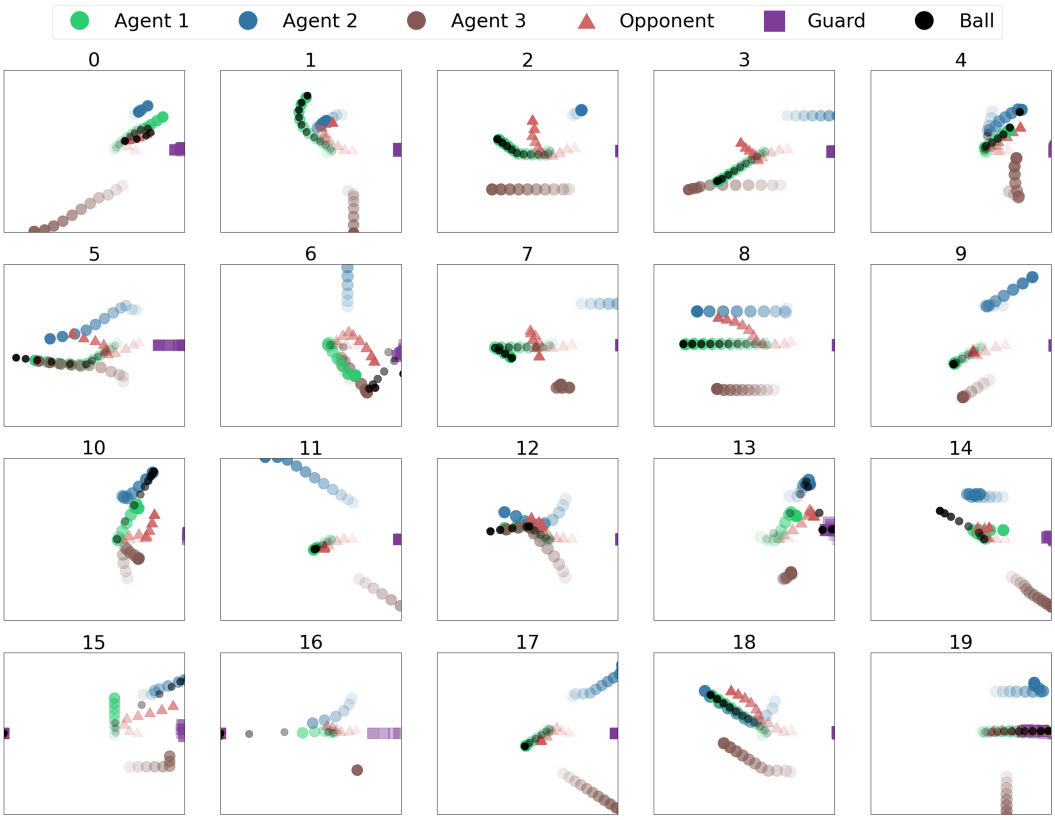

Figure 5: Position trajectories of $Z = 20$ synergy patterns, and the number of each sub-figure is the id z of the synergy pattern. The transparency of the circle denotes the time in the episode and the more transparent circle is at the smaller time step.

10^6 time steps to train 10 joint policies with SPD. Besides, SPD takes around 36 hours for 4×10^6 time steps in GRF to train 20 joint policies.

Code and reproducibility Our entire code [4] is opened for reproducing all the experiments. To reproduce the results, please refer to the instruction file "README.md".

Training details We adopt QMIX [35] as our MARL algorithm to learn synergy patterns. Recurrent Neural Network (RNN) is used in the policy to alleviate the partial observability. The mixing network has one hyper-layer as described in QMIX with 64 units. The optimizer to optimize the neural networks is "Adam". Each URL algorithm is deployed to learn different joint policies ($Z = 10$ for MPE and $Z = 20$ for GRF) and mixing networks every time. We summarize most of the hyper-parameters for the two experiments in Sec. 5 in Table. 2.

C Visualization for learned synergy patterns in GRF

To comprehensively illustrate the diversity of synergy patterns learned by SPD, we directly visualize the $Z = 20$ learned joint policies in GRF. During this procedure, The agents $1, 2, 3$ are controlled by the learned policies and the opponents are built-in AI. Fig. 5 shows the position trajectories of the agents and the opponents by different learned joint policies. The results demonstrate that the agents can learn useful synergy patterns, such as dribbling ($z = 1, 2, 3, 7, \dots$), collaborating to maintain the formations ($z = 4, 8, 18$), and passing-and-shooting ($z = 6$), with only pseudo-reward from SPD.

[4] We custom the code from `https://github.com/hijkzzz/pymarl2` [16] to carry out the experiments in this paper.

Name	Description
Synergy Pattern	The perennial coordinated behaviour of agents.
Synergy Pattern Graph (SPG) $G^{sp}(\mathcal{V}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}, \zeta)$, a.k.a. SPG element	A graph, where $v_i \in \mathcal{V}$ is the vertex for agent $i \in \mathcal{I}$ and the weight $\omega_{ij} \in \boldsymbol{\mathcal{W}}$ of edge $\{i, j\}$ depicts agents' relative relations.
Synergy Pattern Function ζ	A general function which could depict agents' relative relations.
SPG batch $\bar{\lambda}$	A batch of SPG sampled from the distribution of SPG.
Discrepancy of Synergy Patterns d_{sp}	The discrepancy between two distributions of SPG.
Proximal Discrepancy of Synergy Patterns $\bar{d}_{sp}$	The sum of the d_{gw} between two SPG batches.

Table 3: Concepts in SPD.

D Discrimination of proposed concepts

We summarize the concepts mentioned in this work in Table 3 for a clearer understanding.

E Evaluation on SMAC

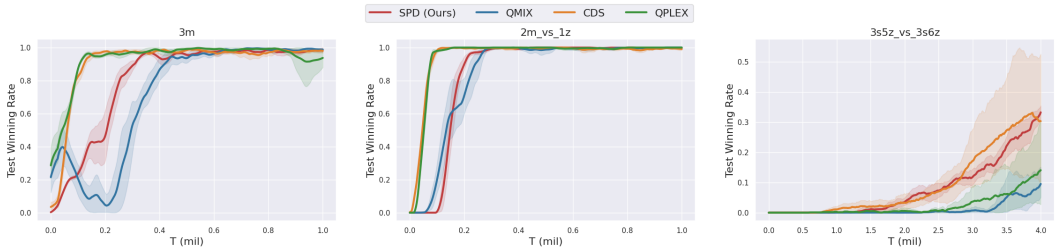

Figure 6: Comparison of our approach SPD against baseline algorithms on three SMAC maps.

We also assess SPD on SMAC [37] to demonstrate the efficacy of the learnt joint policies and offer a more thorough perspective of the effect of the URL training. As for our method SPD, we follow the same URL training procedure in Sec. 5.2 and use QMIX as our training method as well. The $Z = 20$ pre-trained policies are learned by QMIX on the same maps as the downstream tasks without external reward but only pseudo-reward provided by SPD for 10^6 time steps, and are tested before MARL training on the downstream tasks to choose parameters as the initialization. During the MARL training, each algorithm is carried out with 5 random seeds.

We evaluate each algorithm on three maps: `3m`, `2m_vs_1z` and `3s5z_vs_3s6z` (super-hard). The results are shown in Fig. 6. On the map `2m_vs_1z` with normal difficulty, SPD performs similarly to the baseline QMIX, while it learns more efficiently compared to QMIX on map `3m`. We believe the fact that these two maps can be explored by ϵ-greedy strategy sufficiently and do not need well-performed team behaviour may account for this. Since we use QMIX as our training algorithm, the performance is limited by it and is surpassed by CDS and QPLEX. However, SPD significantly outperforms the baseline QMIX and reach similar performance as CDS on the super-hard map `3s5z_vs_3s6z`, which demonstrates that SPD do learn the relationship of agents and encourage team behaviour. Such results suggest that the role of SPD is more pronounced in tasks where there is a greater need for cooperation. In addition, combining exploration-based algorithms, such as CDS, with SPD may achieve better performance and foreshadow future research directions.