Efficient Reinforcement Learning for Zero-Shot Coordination in Evolving Games

Bingyu Hui, Lebin Yu, Quanming Yao, Yunpeng Qu, Xudong Zhang, Jian Wang[*]

Department of Electronic Engineering, Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China

huiby23@mails.tsinghua.edu.cn, paladinee15@gmail.com, qyaoaa@tsinghua.edu.cn, qyp21@mails.tsinghua.edu.cn, zhangxd@tsinghua.edu.cn, jian-wang@tsinghua.edu.cn

Abstract

Zero-shot coordination(ZSC), a key challenge in multi-agent game theory, has become a hot topic in reinforcement learning (RL) research recently, especially in complex evolving games. It focuses on the generalization ability of agents, requiring them to coordinate well with collaborators from a diverse, potentially evolving, pool of partners that are not seen before without any fine-tuning. Population-based training, which approximates such an evolving partner pool, has been proven to provide good zero-shot coordination performance; nevertheless, existing methods are limited by computational resources, mainly focusing on optimizing diversity in small populations while neglecting the potential performance gains from scaling population size. To address this issue, this paper proposes the Scalable Population Training (ScaPT), an efficient RL training framework comprising two key components: a meta-agent that efficiently realizes a population by selectively sharing parameters across agents, and a mutual information regularizer that guarantees population diversity. To empirically validate the effectiveness of ScaPT, this paper evaluates it along with representational frameworks in Hanabi cooperative game and confirms its superiority.

Introduction

As Multi-Agent Reinforcement Learning (MARL) continues to achieve remarkable success in a wide range of complex games and domains, its scalability is increasingly hindered by the greedy growth of computational resource demands, in the scenarios like agent MOBA games (Gao et al. 2023), robotic soccer (Zhu et al. 2024), and intelligent manufacturing systems (Do, Jang, and Kim 2025) owing to the demand for training a diverse and evolving set of agents. These challenges highlight the need for more scalable and resource-efficient MARL frameworks.

We address this problem in the context of Zero-Shot Coordination (ZSC)—a setting where agents must collaborate with unseen partners. This scenario is central to what we term "evolving games": environments where the pool of potential partners is not fixed, but rather represents a continuously shifting or evolving set of strategies. While self-play (Lowe et al. 2017) is widely used for training cooperation, it
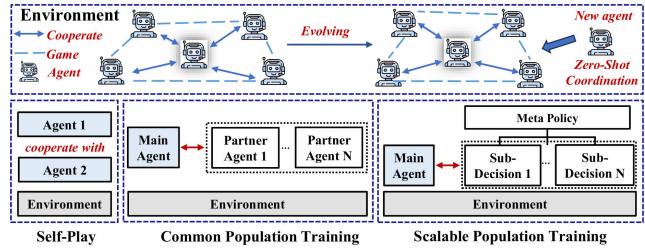
Figure 1: The diagram of different training paradigms in evolving games.

often leads to overfitting, limiting agents' ability to generalize. To overcome this, Hu et al. (2020) introduced the ZSC problem, where agents are evaluated on their ability to coordinate without prior exposure to their partners. Population-based training (Jaderberg et al. 2017) offers a promising solution by exposing agents to a diverse set of behaviors during training, thus encouraging more generalizable coordination strategies (Charakorn, Manoonpong, and Dilokthanakul 2023). In principle, if the training population covers all potential partner strategies, ZSC would be effectively solved. However, approaching this ideal requires training a large and diverse population of agents, which is itself constrained by limited computational resources, hindering the development of efficient and scalable RL methods.

Recent methods focus on improving Zero-Shot Coordination (ZSC) performance by enhancing population diversity, which serves as a key proxy for strategic variation in these evolving games. For example, Trajectory Diversity for Zero-Shot Coordination (Lupu et al. 2021) maximizes trajectory-level diversity to foster generalizable strategies. Similarly, Maximum Entropy Population-Based Training (Zhao et al. 2023) uses entropy maximization to broaden behavioral coverage and improve human-AI coordination. Heterogeneous Multi-Agent Zero-Shot Coordination by Coevolution (Xue et al. 2024) tackles real-world heterogeneity by co-evolving two distinct populations through iterative pairing and selection, enabling robust cooperation with diverse partners.

However, constrained by limited computational resources, existing studies typically maintain relatively small population sizes and attempt to improve the zero-shot coordination performance of trained agents by optimizing

objective functions to encourage policy "diversity" within these small populations. Nevertheless, focusing solely on individual-level behavioral diversity while neglecting the potential impact of population scale may limit the overall effectiveness of population-based methods in truly capturing the breadth of an evolving game's strategy space. In fact, the constrained population size could be one of the critical bottlenecks restricting current ZSC performance.

To address this challenge, we propose an efficient RL with scalable population-based training framework: ScaPT. Figure 1 illustrates the differences among self-play, common population-based training, and our proposed framework in evolving games. Our goal is to enable ZSC training to scale more efficiently to larger populations, mitigating the constraints imposed by computational resources. To achieve this, our framework is designed to meet two key criteria: (1) alleviate the coupling between population size and computational and parameter overhead, thereby mitigating the linear growth of resource consumption with increasing population size; (2) maintain comparable or even superior ZSC performance relative to conventional population-based methods. (2) maintain comparable or even superior ZSC performance relative to conventional population-based methods.

To satisfy these requirements, ScaPT employs a hierarchical meta-agent architecture that simulates arbitrarily large populations within a single network, thereby avoiding redundant training across individuals and effectively suppressing the linear growth of computational costs with population size. To promote behavioral diversity, ScaPT maximizes the conditional mutual information between the actions generated by sub-decision modules under given observations and the modules themselves.

In experiments conducted on two standard ZSC benchmark environments, matrix games and Hanabi, ScaPT achieves significantly better performance than existing algorithms, primarily due to its scalable population design that allows it to leverage larger agent pools under the same memory resources constraints. Furthermore, when comparing different algorithms under the same scalable population framework, ScaPT demonstrates outstanding zero-shot coordination game capabilities.

Our contributions are summarized below:

- We introduce ScaPT, a scalable population training framework for zero-shot coordination that surpasses the population size limits of conventional methods.

- ScaPT designs the scalable population using a hierarchical meta-agent architecture to simulate large populations, achieving a high degree of decoupling between population size and computational resources, while employing conditional mutual information to ensure diversity within the scalable population.

- Under constrained memory budgets, ScaPT outperforms prior methods by leveraging significantly larger agent populations, achieving superior zero-shot coordination performance with the same or lower computational cost.

Related Work

Self-play (Yu et al. 2022) is widely used in MARL, and can efficiently train cooperative agents, but it typically fails under zero-shot coordination. To address this issue, recent works introduce additional reasoning abilities, such as break symmetries of the task to keep agents from learning specified strategies (Hu et al. 2020; Treutlein et al. 2021; Muglich et al. 2022), conduct multi-level reasoning for higher-level consensus (Cui et al. 2021; Hu et al. 2021) and predict partners' actions (Lucas and Allen 2022; Yan et al. 2024). However, these approaches often lead to algorithm-level consensus or overfitting to a limited set of partners.

Population-based training enhances generalization by exposing a main agent to diverse partners. Common approaches include reducing the collaboration scores of different partner agents within the population to make them behave differently (Charakorn, Manoonpong, and Dilokthanakul 2023; Rahman et al. 2023), improving trajectory diversity of agents (Lupu et al. 2021) and increasing policy entropy of the population (Zhao et al. 2023). However, most approaches train distinct networks per agent or require differentiable action distributions (Lupu et al. 2021; Zhao et al. 2023), limiting efficiency and applicability; In contrast, ScaPT employs a meta-agent for efficient population training, and designs a generic mutual information term to guarantee population divergence. This differs from MI-based skill-discovery methods such as DIAYN (Eysenbach et al. 2018) and DADS (Sharma et al. 2019), which increase intra-agent skill diversity rather than inter-agent policy diversity.

Notably, the meta-agent in ScaPT is different from the agents in meta-RL (Nagabandi et al. 2018; Gupta et al. 2018). Traditional meta-RL aims to train agents that can quickly adapt to various tasks, the diversity of which is innate and invariable. In contrast, ScaPT's meta-agent is designed to exhibit various policies, the diversity of which is variable and what we seek to augment.

Scalable Population Training framework for Zero-Shot Coordination

We present ScaPT, a scalable population-training framework that achieves diverse zero-shot coordination.

Hierarchical Meta-Agent For Scalable Population

A major drawback of population training is its high resource demand. Inspired by meta-RL and multi-task learning (Ma et al. 2018; Sagirova, Kuratov, and Burtsev 2025), task-related parameters can be shared, while behavior-specific submodules are optimized independently.

Based on the aforementioned idea, we design a hierarchical meta-agent—a simple yet effective architecture—to realize a scalable population. Fig. 2 show the details of the proposed architecture. Here, h and c denote the hidden and cell states of the LSTM, respectively, and o^t is the observation input from the environment. The variable v^t represents the estimated state value. The submodules $P_1, P_2, ..., P_K$ are used to simulate different individuals in the population, Each
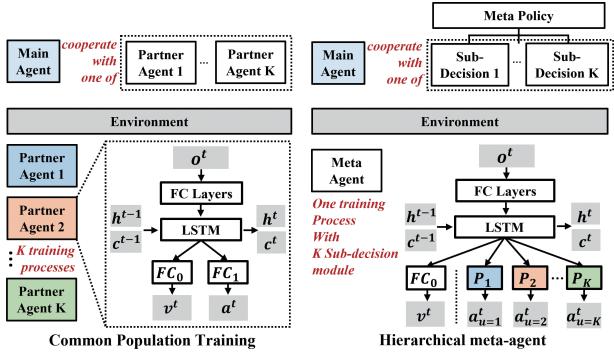
Figure 2: Comparison of common population and hierarchical meta-agent population

of P_i's output $a^t_{u=i}$ serves as the policy output for the i-th agent in the population, where K denotes the population size.

As can be seen, the hierarchical architecture of the meta-agent greatly reduces the parameters that need to be optimized (K complete sets $\rightarrow$ 1 complete set + K subsets). In this way, one can simulate populations of arbitrary size simply by adjusting the number of subsets, without the need to independently train each agent from scratch. In our implementation, we simply define the P_i as a two-layer fully connected network; however, this choice is not mandatory. More complex architectures could potentially yield further performance improvements, though this comes with higher computational cost.

Conditional Mutual Information Maximization

If all the agents in a population have similar policies, the best-response agent can only learn to cooperate with partners of one kind of policy, and in this way the advantage of population training disappears (See Ablation study section for details). Therefore, increasing the diversity of population (i.e. different agents in the population behave variously) has always been a key concern in this research area. It is noteworthy that a diverse population means different agents in the population act differently, and this can be achieved by making the meta-agent output distinct actions with different sub-decision modules given an observation and history trajectory. This operation can be formulated as maximizing the conditional mutual information:

$$I(A;U|H) = \int \int \int p(a,u,h) \log \frac{p(h)p(a,u,h)}{p(u,h)p(a,h)} du\,ds\,da \tag{1}$$

where H represents observation input (containing current observation and historical trajectory used for decision), U represents the index of sub-decision module that outputs action A, and $p(a,u,h)$ is the joint probability density function. Considering that reinforcement learning frameworks commonly estimate integrals using the Monte Carlo method, that is, sampling transitions from replay buffer to calculate, the unbiased estimation of mutual information $\hat{I}(A;U|H)$ can be written as:

$$\hat{I}(A;U|H) = \frac{1}{N} \sum_{j=1}^{N} \log \frac{p(a_j|u_j,h_j)}{p(a_j|h_j)}$$

$$= \frac{1}{N} \sum_{j=1}^{N} \left[\log p(a_j|u_j,h_j) \right. \tag{2}$$

$$\left. - \log \sum_{i=1}^{K} p(u_i|h_j) \cdot p(a_j|u_i,h_j) \right]$$

where K is the total number of sub-decision modules, N is the number of transitions, a_j, u_j, h_j are action, sub-decision module index and observation input of the j-th transition, and $p(a_j|u_j,h_j)$ is the conditional probability that an agent takes action a_j given u_j, h_j. For brevity, use I_j to denote the j-th term in $\hat{I}(A;U|H)$. The equation is: $I_j := \log p(a_j|u_j,h_j) - \log \sum_{i=1}^{K} p(u_i|h_j)p(a_j|u_i,h_j)$.

Notably, directly maximizing $\hat{I}(A;U|H)$ with gradient-based methods is not a preferable choice for two reasons. Firstly, The posterior probability $p(u_i|h_j)$ is hard to calculate. Secondly, the gradient of $p(a_j|u_i,h_j)$ is almost always equal to zero for many RL policies. For example, value-based policies commonly output actions that maximize the action-value function $Q(a,u,h)$ (or use ϵ-greedy for exploration). In this way, the value of $p(a_j|u_i,h_j)$ is determined only by whether $a_j = \arg\max_a Q(a,u_i,h_j)$. Consequently, the derivatives of $p(a_j|u_i,h_j)$ with respect to the neural network parameters are equal to zero.

We propose to optimize an alternative objective $\bar{I}(A;U|H)$, which have the following two properties:

1. $\bar{I}(A;U|H)$ provides gradients that can be used for neural network training, whether the meta-agent makes decisions by outputting action distributions or maximizing Q-functions.

2. Increasing $\bar{I}(A;U|H)$ also increases $\hat{I}(A;U|H)$.

The definition of $\bar{I}(A;U|H)$ is given below:

$$\bar{I}(A;U|H) = -\frac{1}{N} \sum_{j=1}^{N} \sum_{i=1,i \neq j}^{K} F(u_i,h_j,a_j) \tag{3}$$

where $F(u_i,h_j,a_j)$ represents the favor of the meta-agent for a_j given u_i, h_j and is required to be the direct output of a neural network so that its gradients can be used for gradient-based training. Below are several possible forms of $F(u_i,h_j,a_j)$:

1. If the neural network used for decision directly outputs action distribution (common for policy-gradient-based methods such as PPO (Schulman et al. 2017)), then $F(u_i,h_j,a_j)$ represents the probability of choosing a_j given u_i, h_j, which is $p(a_j|u_i,h_j)$;

2. If the neural network used for decision outputs advantage functions (or Q-values, common for value-based methods such as Dueling-DQN (Wang et al. 2016)), then $F(u_i,h_j,a_j) = A(u_i,h_j,a_j)$ (or $Q(u_i,h_j,a_j)$).

Moreover, we provide certain theoretical guarantee for maximizing $\bar{I}(A;U|H)$.

Theorem 1. *Given $F(u_j, h_j, a_j)$, if F is update to F' such that:*

$$\exists v \; s.t. \quad \arg\max_a F(u_v, h_j, a) = a_j \; \wedge$$
$$F'(u_v, h_j, a_j) < F(u_v, h_j, a_j) \; \wedge \qquad (4)$$
$$\forall i \neq v, \; F'(u_i, h_j, a_j) = F(u_i, h_j, a_j)$$

then the corresponding term I_j in $\hat{I}(A;U|H)$ is updated to I'_j and satisfies $I'_j \geq I_j$.

The proof is presented in the Appendix.

Computing $I(A;U|H)$ requires obtaining all agents' actions under the same observation, which is efficiently achieved via the meta-agent with a single forward pass—unlike standard population training that needs N slow LSTM passes. Notably, ScaPT's two components are complementary: the meta-agent enables efficient mutual information computation and training, while the conditional mutual information term guides the learning of a diverse population.

Instantiation

ScaPT only specifies how to build an efficient and diverse population, and is compatible with multiple base RL frameworks that optimize agent policies. Our instantiation is based on an value-based approach because it is confirmed that this kind of method is suitable for our experimental task Hanabi (Hu and Foerster 2019; Bard et al. 2020). The main agent only needs to cooperate well with all the agents in the population (which are realized using the partner meta-agent), and thus it is required to minimize the base TD-error (Van Hasselt, Guez, and Silver 2016):

$$L_m = \frac{1}{N} \sum_{j=1}^{N} r_j + \gamma \max_a Q_{\theta'_m}(h'_j, u_j, a)$$
$$- Q_{\theta_m}(h_j, u_j, a_j) \qquad (5)$$

where θ_m and θ'_m represent the parameters in the online Q-net and target Q-net of the main agent respectively. Notably, the main agent has the same neural network architecture as a normal agent shown in Fig. 2. In comparison, the meta-agent not only needs to learn coordination, but also needs to become diverse by maximizing $\bar{I}(A;U|H)$. Consequently, $\bar{I}(A;U|H)$ is added to the base TD-loss with a weight α, which controls the balance between cooperation ability and population diversity:

$$L_p = \frac{1}{N} \sum_{j=1}^{N} \Bigg[r_j + \gamma \max_a Q_{\theta'_p}(u_j, h'_j, a)$$
$$- Q_{\theta_p}(u_j, h_j, a_j) + \alpha \sum_{\substack{i=1 \\ i \neq j}}^{K} Q_{\theta_p}(u_i, h_j, a_j) \Bigg] \qquad (6)$$

where θ_p represent the parameters in the online Q-net of the partner meta-agent. To accelerate convergence, prioritized replay (Schaul et al. 2015) and dueling-net (Wang et al. 2016) is also utilized.

Another key component of instantiation is the training mode, which specifies the pairs of agents interacting with the environment and the corresponding transitions for training. Table 1 summarizes six feasible modes, with details in the Appendix. Take Mode-III as an example: it has act groups MP, PP, meaning agents generate transitions in two groups: [main agent, partner agent] and [partner agent, partner agent]. The training objectives require the main agent to cooperate well with the partner, while the partner only optimizes self-play scores without adapting. Each mode emphasizes different aspects, and we investigate their performance experimentally.

Experiments

Matrix game: Task Complexity vs. Population Size

To investigate how population size influences the ZSC performance of the main agent, we conducted comparative evaluations using collaborative single-step matrix games (Lupu et al. 2021). In each game, player 1 selects a row and player 2 selects a column independently. After both players have made their choices, the selected actions are revealed, and the agents receive the reward corresponding to the intersection of the chosen row and column. Specifically, we compared our approach against representative baselines, including **ET3** (Yan et al. 2024), **MEP** (Zhao et al. 2023), and **TrajeDi** (Lupu et al. 2021), **Individual agents**, where agents are trained independently, and **baseline** (Lupu et al. 2021). These methods cover both population-based and non-population-based paradigms, providing both a comparison across different algorithmic architectures and an in-depth view of how population size impacts the ZSC performance of population-based methods.

Fig. 3a presents the experimental results for various baseline algorithms under a matrix dimension of 50 and a fixed population size of 2. It can be observed that while all methods achieved strong performance in the self-play setting, their scores dropped significantly in the cross-play evaluation, indicating a lack of generalization between agents. Notably, ScaPT retained a non-negligible degree of ZSC capability, outperforming other baselines in this regard.

As illustrated in Fig. 3b, when the population size was increased, ScaPT's performance in cross-play improved markedly, demonstrating a strong resurgence in ZSC ability. This suggests that scaling the population size can enhance the diversity and representational capacity of the training process, thereby equipping the main agent with more generalized coordination strategies. The appendix includes examples of matrices with varying dimensions as well as the remaining experimental results.

ScaPT in Hanabi

We conducted experiments on Hanabi (Bard et al. 2020), a cooperative card game in which players must play cards in order by color and rank. Players can only see their teammates' cards, not their own, and must infer intentions and communicate through limited actions. As a result, self-play agents perform well due to shared training, but collaboration with unfamiliar agents is challenging, making Hanabi a

Index	Act Group	Objective for π_m	Objective for π_{pi}
I	MP	$\sum_{i=1}^{N} J(\pi_m, \pi_{pi})$	$J(\pi_m, \pi_{pi})$
II	MM, MP	$J(\pi_m, \pi_m) + \sum_{i=1}^{N} J(\pi_m, \pi_{pi})$	$J(\pi_m, \pi_{pi})$
III	MP, PP	$\sum_{i=1}^{N} J(\pi_m, \pi_{pi})$	$J(\pi_{pi}, \pi_{pi})$
IV	MM, MP, PP	$J(\pi_m, \pi_m) + \sum_{i=1}^{N} J(\pi_m, \pi_{pi})$	$J(\pi_{pi}, \pi_{pi})$
V	MP, PP	$\sum_{i=1}^{N} J(\pi_m, \pi_{pi})$	$J(\pi_{pi}, \pi_{pi}) + J(\pi_m, \pi_{pi})$
VI	MM, MP, PP	$J(\pi_m, \pi_m) + \sum_{i=1}^{N} J(\pi_m, \pi_{pi})$	$J(\pi_{pi}, \pi_{pi}) + J(\pi_m, \pi_{pi})$

Table 1: Feasible population training modes

(a) Matrix dimension = 50, population size = 2

(b) Matrix dimension = 50, population size = 30
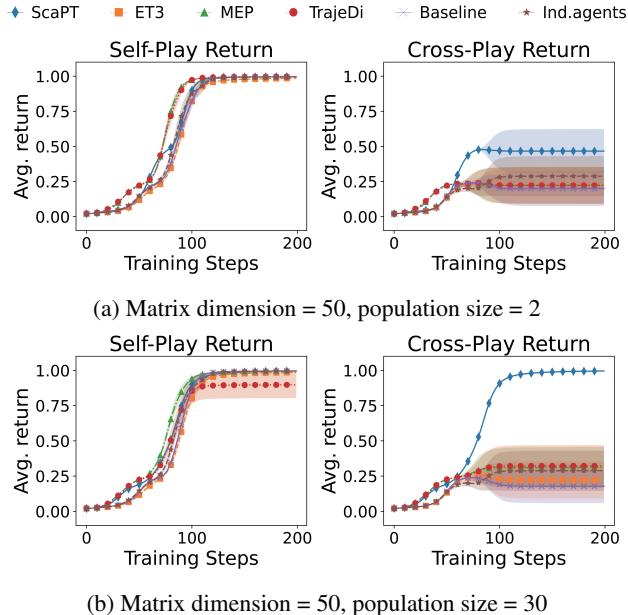
Figure 3: Results for matrix dimension = 50 with different population sizes

standard benchmark for zero-shot coordination.

Compare methods and Evaluation Criteria

We compare our method with five representative methods, including one self-play methods: **SP** — self-play training with parameter sharing, serving as a baseline; two non-population-based algorithms designed for ZSC: **OP** (Hu et al. 2020), **OBL** (Hu et al. 2021); and two population-based ZSC algorithms, **TrajeDi** (Lupu et al. 2021) and **MEP** (Zhao et al. 2023).

Zero-shot coordination requires agents to cooperate well with collaborators that have not been seen before, that is, "strangers". The corresponding evaluation criteria are slightly different from those of normal MARL. Below we give a formulaic representation in two-agent scenarios.

Use $J(\pi_1, \pi_2)$ to denote the expected cumulative discounted return obtained by the collaboration of π_1 and π_2. Use π_i^M to denote the policy obtained with training framework M and random seed i. The earliest metric to evaluate zero-shot cooperation performance is intra-algorithm cross-play (abbreviated as Intra-XP) score (Hu et al. 2020):

$$S_{intra-XP}(M) = \mathbb{E}[J(\pi_i^M, \pi_j^M)|i \neq j] \qquad (7)$$

Cross-play is a commonly used metric that evaluates agent's ZSC performance with partner agents trained using the same algorithm but different random seeds. However, since all agents share the same architecture, the test agent may retain implicit prior knowledge about its partners. Thus, relying solely on cross-play does not provide a rigorous assessment of zero-shot coordination.

In order to address the deficiencies of the aforementioned metric, Lucas and Allen (2022) propose one-sided zero-shot coordination (abbreviated as 1ZSC-XP) score:

$$S_{1ZSC-XP}(M) = \mathbb{E}[J(\pi^M, \pi^{M_t})] \qquad (8)$$

where M_t refers to a set of algorithms that are not specially designed for zero-shot coordination. The shortcoming of this criterion is that π^{M_t} still cannot represent all feasible "strangers", and the results may be biased.

Neither of the metrics is perfect, hence the following sections display the above two metrics for more comprehensive evaluation.

To enhance the credibility of the evaluation, we train models with several fixed random seeds(time consumption: 0-4 (2-player), 1-4 (5-player)) under each experiments, and test Intra-XP and 1ZSC-XP scores. Specifically, 1ZSC-XP scores are obtained by pairing the tested zero-shot coordination agents with 40 non-ZSC agents obtained with four kinds of self-play frameworks: IQL (Tan 1993), VDN (Sunehag et al. 2018), SAD and SAD+AUX (Hu and Foerster 2019).

Experiments with different framework under Resource Constraints

Firstly, in Hanabi, we evaluate each algorithm within its native framework under a strict resource constraints (128GB RAM and 48GB GPU memory), reflecting typical small-scale research servers.

We compare our method only with population methods designed for ZSC: TrajeDi and MEP on 5-player game. Table 2 shows the performance upper bounds of different methods under constrained RAM and GPU memory. Specifically, under these limits, TrajeDi and MEP can only support a population size of 1 due to the need to allocate separate replay buffers for each agent. In contrast, ScaPT achieves the best performance for both metrics by flexibly scaling the population size to 5 and 8.

Metrics	TrajeDi	MEP	ScaPT
Intra-XP	4.17±1.34	5.13±2.11	**11.07±1.65**
1ZSC-XP	4.22±1.47	6.69±1.62	**11.39±1.60**

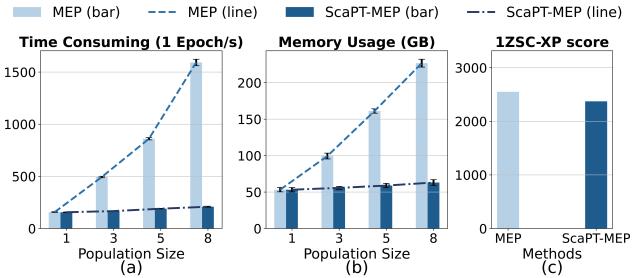Table 2: Best performance under Resource Constraints in 5-player hanabi game

Figure 4: Comparison of common population and meta-agent population: (a) training time consuming with population size increase; (b) resource consumption with population size increase; (c) sum of 1ZSC-XP scores over 40 experiments with different architectures using the MEP method.

To validate our archictecture and ensure fair comparison, we evaluate MEP under common and scalable population architectures in computational cost, training time, and ZSC performance on 2-player Hanabi. As shown in Fig. 4, the scalable population scales more efficiently with slower growth in resource use and training time, while the common framework grows approximately linearly. Both approaches achieve comparable ZSC performance, demonstrating the scalability and effectiveness of our method for complex tasks requiring large population agents.

Experiments with scalable population framework

We conduct a 2-player Hanabi experiment with population size 5 for all population-based algorithms. Table 3 reports mean and standard error of evaluation metrics. ScaPT achieves the highest 1ZSC-XP, demonstrating strong coordination with 40 unseen partners and ranks top on Intra-XP except for OBL. Although OBL scores high on Intra-XP, its 1ZSC-XP score is only 3.8, suggesting that OBL lacks genuine zero-shot coordination capability since it performs well mainly when cooperating with agents that are trained using the same algorithm as itself. Considering both metrics together, all population-based methods demonstrate solid performance. Among them, ScaPT delivers the most outstanding results, further confirming its superiority.

Besides, Fig. 5 visualizes the detailed pair-wise 1ZSC-XP cooperation scores of Table 3 using a heat-map. The heat-maps of three compared population-based methods are presented along for reference, and the heat-maps of other frameworks are presented in the Appendix. In each sub-figure, each row represents the coordination scores of a testing main agent pairing with all non-ZSC agents, and deeper colors represent higher scores. Different rows in a sub-figure
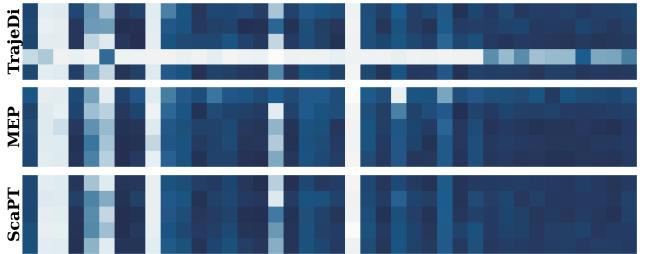

Figure 5: Detailed pair-wise 1ZSC-XP scores of TrajeDi, MEP and CMIMP. Deeper colors represent higher scores and each row represents the coordination scores of testing a main agent pairing with 40 non-ZSC agent, thus forming a 5×40 heat-map.

correspond to agents trained with different random number seeds under the same framework. There are two phenomena worth noting: Firstly, the differences between columns are consistent for these two frameworks. The reason is that some non-ZSC agents are relatively easy to cooperate with (e.g. column 7,8), while some others are not (e.g. column 2,3). Secondly, the performance stability of ScaPT is good: 1ZSC-XP scores of ScaPT models with different random seeds vary little, while TrajeDi and MEP demonstrate certain instability.

To validate the relationship between population size and ZSC performance, we conducted experiments on the Hanabi benchmark with all the population-based algorithms. By increasing the number of players from 2 to 5, we raised task difficulty to evaluate algorithm performance, while also varying population size to assess its effect on ZSC outcomes.

As shown in Table 4, increasing the number of players from 2 to 5, which raises task complexity, causes all algorithms to experience performance changes. TrajeDi suffers the largest drop, while MEP and ScaPT maintain relatively better performance. Overall, MEP performs strongest, with ScaPT showing comparable but slightly lower scores and more stable results across random seeds. Increasing the population size from 2 to 5 improves performance for all methods considerably—specifically, the 1ZSC-XP metric improves by 3.83 for TrajeDi, 1.38 for MEP, and 3.19 for ScaPT. As the population size increases from 5 to 8, the overall performance of all methods continues to improve, though with reduced marginal gains. However, a further increase to 15 results in performance degradation, indicating the presence of algorithm-specific optimal population sizes—an aspect that merits deeper exploration. ScaPT achieves the best overall results on both metrics, demonstrating strong generalization and underscoring the critical importance of population size for ZSC.

Ablation Study

As stated in equation 6, the training objective includes a weighted mutual information term that encourages behavioral differences among sub-modules. To examine its impact, Fig. 7 plots four key metrics across training epochs in 2-player hanabi environment under different training modes.

	SP	OP	OBL	TrajeDi*	MEP*	ScaPT
Intra-XP	3.96±0.49	14.94±0.67	**23.80±0.03**	15.95±2.47	20.09±0.69	20.77±0.40
1ZSC-XP	7.68±0.39	13.48±0.19	3.80±0.07	14.13±3.20	15.64±3.02	**15.95±3.05**

Table 3: Zero-shot coordination performance of different algorithms on the 2-player Hanabi game (population-based methods: TrajeDi, MEP, ScaPT, with population size 5). Algorithms marked * are implemented under the proposed scalable framework.

ps	Metrics	TrajeDi*	MEP*	ScaPT
ps=2	Intra-XP	6.08±1.86	8.13±2.11	7.88±1.72
	1ZSC-XP	3.71±0.60	7.08±1.62	6.18±1.53
ps=5	Intra-XP	9.91±1.48	9.51±1.64	**11.07±1.65**
	1ZSC-XP	8.38±0.75	8.84±0.93	11.35±0.58
ps=8	Intra-XP	9.04±0.44	10.98±0.33	10.61±0.93
	1ZSC-XP	9.86±1.73	10.48±1.60	**11.39±1.60**
ps=15	Intra-XP	8.98±0.87	6.78±0.37	8.40±0.83
	1ZSC-XP	9.01±1.65	8.25±1.09	9.10±1.81

Table 4: ZSC performance of population-based algorithms on 5-player Hanabi, under the scalable framework with different population sizes (PS).

	I	II	III	IV	V	VI
Intra-XP	1.16	**20.77**	8.93	12.43	12.07	19.22
1ZSC-XP	5.63	**15.95**	10.93	12.59	12.41	15.09

Table 5: Zero-shot coordination performance under different training modes (mean values only).

MM Score: The self-play score of the main agent;

MP Score: The cooperation score of the main agent and the partner agent, one of the key objectives of population training;

PP Score: The self-play score of the partner agents;

Diff Prob: The probability that different agents select the same action given the same observation—lower values indicate higher population diversity.

As is shown in Fig. 6, when the mutual information term $\bar{I}(A; U|H)$ is ignored (i.e. $\alpha = 0$), **Diff Prob** quickly rises to 1, meaning that the partner agents in the population act similarly. As a result, the generalization performance of the main agent is reduced (low XP scores during testing) despite that the training process goes smoothly (high MM/MP/PP scores during training). When α is set to a proper value ($\alpha = 1$), **Diff Prob** maintains relatively low, indicating a diverse population. A larger α ($\alpha = 10$) does not further increase diversity and reduces partner self-play performance (PP score), suggesting that excessive weight can harm rationality and coordination. To sum up, $\bar{I}(A; U|H)$ can help build a diverse population and thereby improve the zero-shot coordination, while assigning it a too large weight might have negative effect.
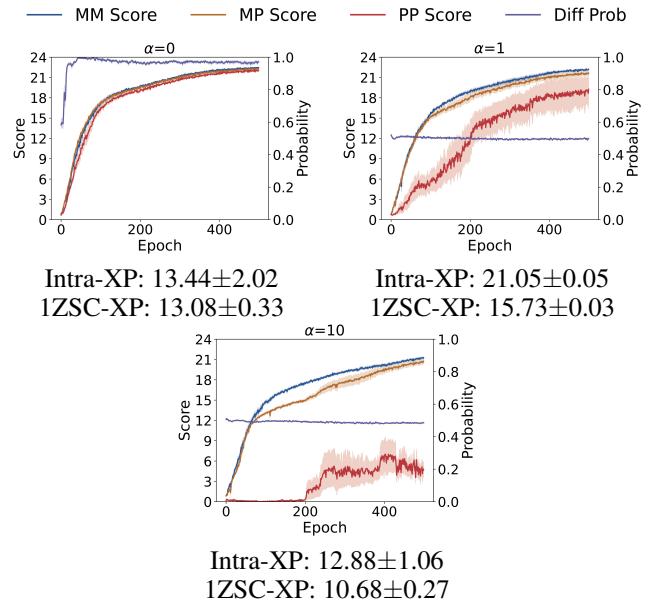

Figure 6: Training curves and testing scores of ScaPT with different α.

Comparison of Different Training Modes

Table 1 introduces six feasible training modes for population-based training. Then which mode is the best? Tab. 5 confirms that zero-shot coordination performance of different training modes varies a lot, and such differences are brought by the settings of secondary objectives. Mode-I is the worst, indicating that only optimizing the primary objective ($J(\pi_m, \pi_p)$) is not enough. Mode-II is the best, confirming the necessity of adding the self-play objective for the main agent. Notably, Mode-IV and Mode-VI additionally require increasing self-play scores for partner agents on the basis of Mode-II, and this operation is of no benefit judging from the results. The training curves for the different modes are provided in the appendix for reference.

Conclusion

In this paper, we analyzed limitations of population-based RL methods for ZSC, which struggle to scale efficiently enough to represent the diversity of evolving games, and proposed a population-size-free RL training framework. Experiments demonstrated its effectiveness and revealed a strong correlation between population size and ZSC performance, further validating the significance of our study.

Appendix

A. Proof of Theorem 1

Here, we restate our **Theorem 1**, where u_j, h_j and a_j denote the $j-th$ sub-decision module's sub-decision module index, observation and action, respectively.

Theorem 2. *Given $F(u_j, h_j, a_j)$, if F is update to F' such that:*

$$\exists v \; s.t. \quad \arg\max_a F(u_v, h_j, a) = a_j \; \wedge$$
$$F'(u_v, h_j, a_j) < F(u_v, h_j, a_j) \; \wedge \qquad \text{(A.1)}$$
$$\forall i \neq v, \; F'(u_i, h_j, a_j) = F(u_i, h_j, a_j)$$

then the corresponding term I_j in $\hat{I}(A; U|H)$ is updated to I'_j and satisfies $I'_j \geq I_j$.

Proof. In consideration of the relationship between $F(u, h, a)$ and $p(u, h, a)$, there are two cases to be addressed.

Case 1 $F(u, h, a) = p(a|u, h)$

With the condition stated in (A.1), only $p(a_j|u_v, h_j)$ will be changed among all the terms in I_j. Consequently,

$$
\begin{aligned}
I'_j - I_j = \; & \log \Big[p(u_v|h_j) \, p(a_j|u_v, h_j) \\
& + \sum_{\substack{i=1 \\ i \neq v}}^{K} p(u_i|h_j) \, p(a_j|u_i, h_j) \Big] \\
& - \log \Big[p(u_v|h_j) \, p'(a_j|u_v, h_j) \\
& + \sum_{\substack{i=1 \\ i \neq v}}^{K} p(u_i|h_j) \, p(a_j|u_i, h_j) \Big]
\end{aligned}
\qquad \text{(A.2)}
$$

Since $p'(a_j|u_v, h_j) < p(a_j|u_v, h_j)$ and $p(u_v|h_j) \geq 0$, $I'_j \geq I_j$.

Case 2 $F(u, h, a) = A(u, h, a)$

In this case, $p(a|u, h) = 1$ if $a = \arg\max Q(u, h, a)$ where $Q(u, h, a) = A(u, h, a) + V(u, h)$, else $p(a|u, h) = 0$ [1].

According to (A.1), only $Q(u_v, h_j, a_j)$ changes, and this leads to three possible outcomes:

1. $a_j \neq \arg\max Q(u_v, h_j, a)$ and
$a_j \neq \arg\max Q'(u_v, h_j, a)$.
In this situation, $p(a_j|u_v, h_j)$ remains the same, and $I'_j = I_j$.
2. $a_j = \arg\max Q(u_v, h_j, a)$ and
$a_j = \arg\max Q'(u_v, h_j, a)$.
Similarly, $I'_j = I_j$.

---

[1] If the meta-agent uses an ϵ-greedy strategy for exploration, then the corresponding value is $1 - \epsilon$ and $\epsilon/(|A| - 1)$. This difference has no impact on the proof.

3. $a_j = \arg\max Q(u_v, h_j, a)$ and
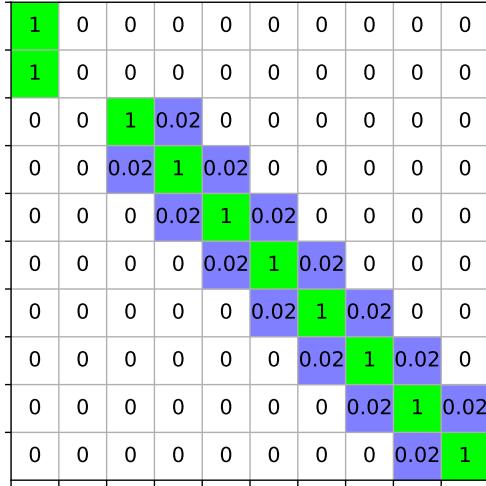$a_j \neq \arg\max Q'(u_v, h_j, a)$.
In this situation, $p(a_j|u_v, h_j) = 1$ and $p'(a_j|u_v, h_j) = 0$. As is proved before, $p'(a_j|u_v, h_j) < p(a_j|u_v, h_j)$ leads to $I'_j \geq I_j$.

$\square$

B. Implementation Details

Hardware and software settings We experiment on a server with 2 x RTX 3090 and a Intel Xeon Platium CPU (12 cores), and training one scalable population models takes around 15 hours. The experimental codes are modified based on the open source codes of OBL.

Neural network hyper parameters Fig.2 in the main text shows the architecture of meta agent with K sub-decision modules, which implies that it possesses an equivalent population size of K, and below introduces the hyper parameters of each module. FC layer is a linear transform with output size 512. LSTM has two layers with hidden dim 512. P_i is a two-layer fully connected network.

Training hyper parameters All the models are training 500 epochs with replay buffer size 35000 and batch size 128. Parameters are updated via Adam optimizer with learning rate 6.25e-5. Discount factor γ is set to 0.999.

C. Detailed results of Matrix game

This section present all all experiments on Matrix game with different dimensions. To evaluate the performance of various algorithms as task complexity increases, we raise the game difficulty by constructing matrices of varying dimensions, each based on a base 10×10 matrix template. The pseudocode for matrix generation is given below.

Algorithm 1: Generate Shifted Block Matrix

Require: Block count n_b, block size $d = 10$
Ensure: Matrix $M \in \mathbb{R}^{(n_b d) \times (n_b d)}$
1: Initialize $M \leftarrow 0$
2: **for** $k = 0$ to $n_b - 1$ **do**
3:     $B \leftarrow I_d$, $B[1][1] \leftarrow 0$, $B[1][0] \leftarrow 1$
4:     **for** $i = 2$ to $d-1$ **do**
5:         **for** $j \in \{i-1, \, i+1\}$ **do**
6:             **if** $1 < j < d$ **then**
7:                 $B[i][j] \leftarrow \epsilon$
8:             **end if**
9:         **end for**
10:     **end for**
11:     Flatten B to b, compute $s = (\text{arange}(d^2) + k) \bmod d^2$
12:     Reshape $b[s]$ to $B' \in \mathbb{R}^{d \times d}$
13:     Insert B' into $M[kd:(k+1)d, \; kd:(k+1)d]$
14: **end for**
15: **return** M

Matrix dimension = 10

matrix dimension = 30

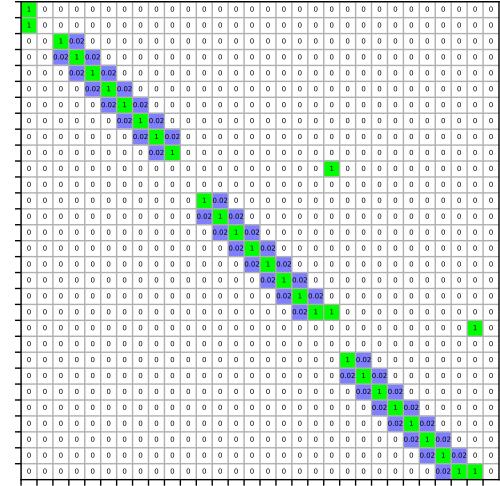Figure 7: Matrix game, dimension = 10
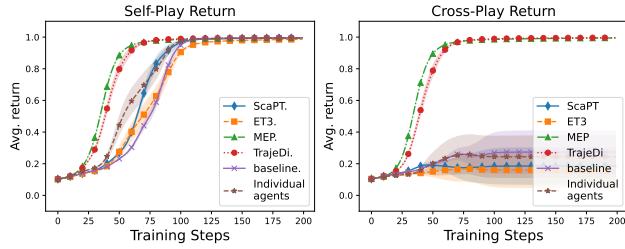

Figure 11: Matrix game, dimension = 10

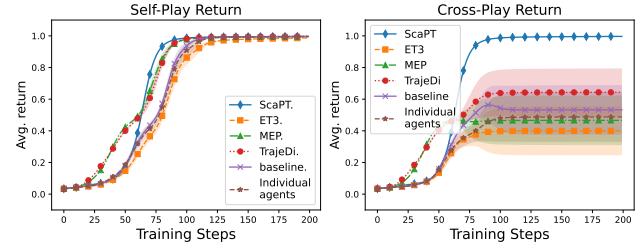Figure 8: Matrix dimension = 10, population size = 2

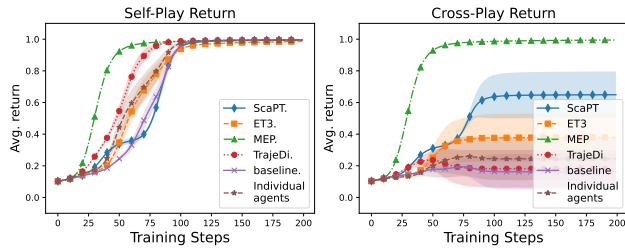Figure 12: Matrix dimension = 30, population size = 2

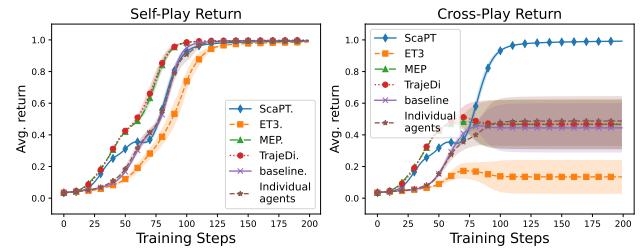Figure 9: Matrix dimension = 10, population size = 5

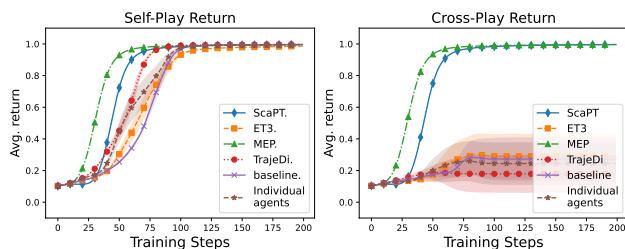Figure 13: Matrix dimension = 30, population size = 10
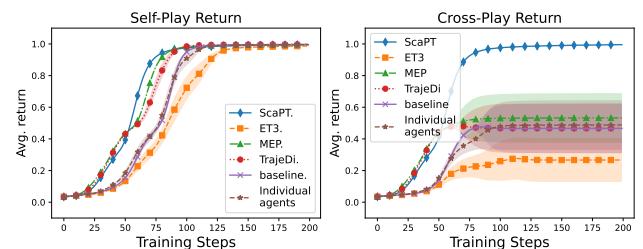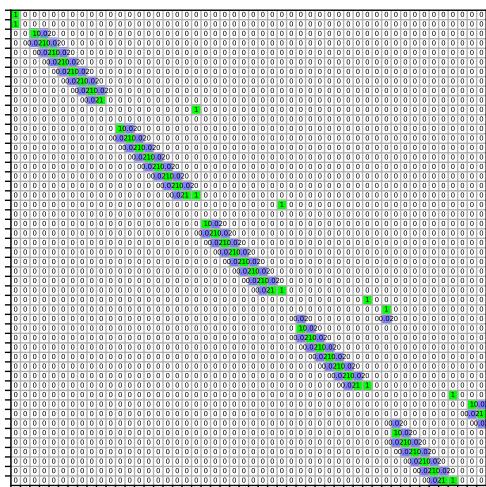

Figure 10: Matrix dimension = 10, population size = 10
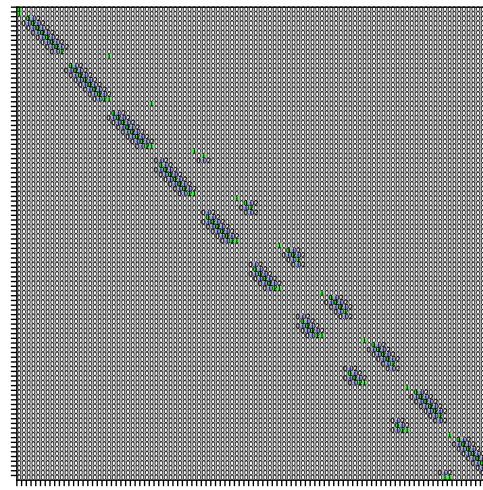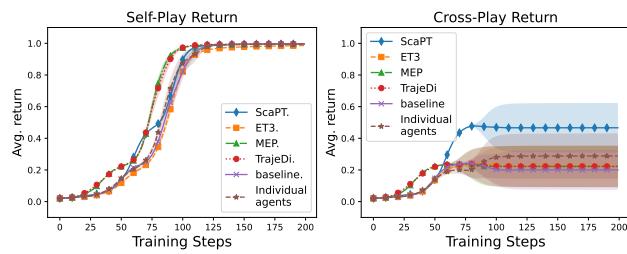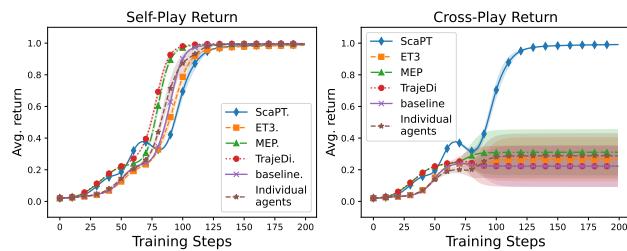

Figure 14: Matrix dimension = 30, population size = 20

matrix dimension = 50

Figure 15: Matrix game, dimension = 50

Figure 16: Matrix dimension = 50, population size = 2
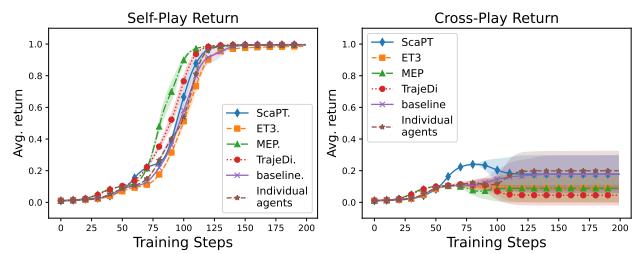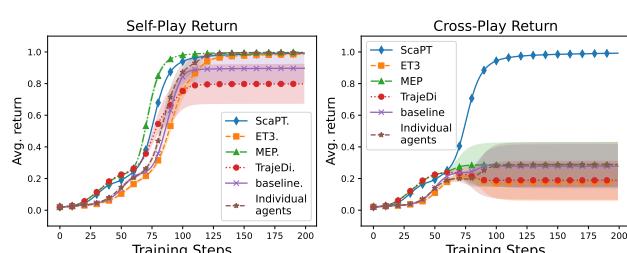

Figure 17: Matrix dimension = 50, population size = 10

Figure 18: Matrix dimension = 50, population size = 20

Figure 19: Matrix dimension = 50, population size = 30

matrix dimension = 100

Figure 20: Matrix game, dimension = 100

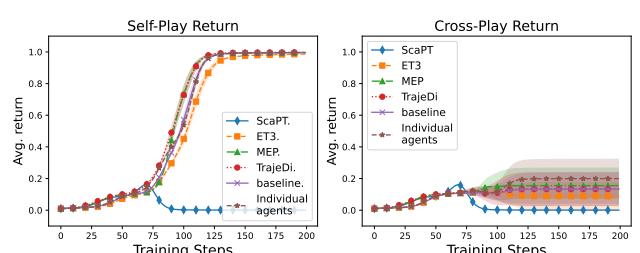Figure 21: Matrix dimension = 100, population size = 2

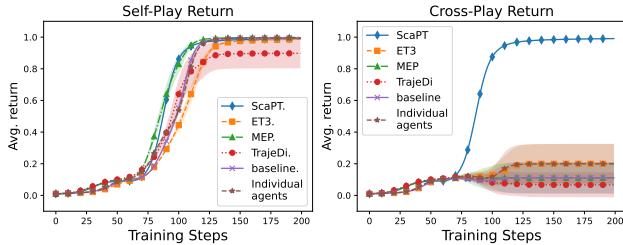Figure 22: Matrix dimension = 100, population size = 10

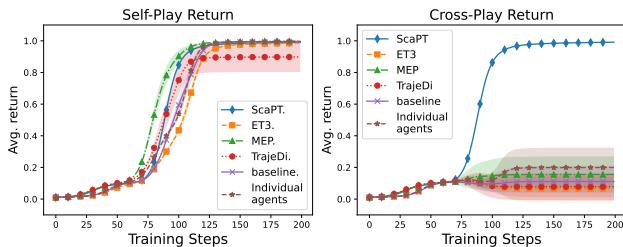Figure 23: Matrix dimension = 100, population size = 20

Figure 24: Matrix dimension = 100, population size = 30

Based on the aforementioned experiments, we summarize the following findings:

- As the dimension of the matrix game increases, corresponding to higher task difficulty, the performance of all algorithms degrades or even fails under fixed population sizes.

- Among the baselines, population-based methods cannot mitigate this performance degradation by simply increasing population size.

- In contrast, our proposed method successfully leverages larger populations to achieve superior performance on more challenging tasks, significantly outperforming other algorithms.

D. Detailed results of hanabi game

In this section, we present the detailed performance of various algorithms on Hanabi games with different numbers of players. As described in the main text, deeper colors represent higher scores, and each row indicates the coordination scores obtained by testing a main agent paired with 40 non-ZSC agents, resulting in a 5×40 heatmap. Due to the significantly higher training cost of the 5-player Hanabi setting, we report results based on only 4 random seeds for each algorithm in this setting.
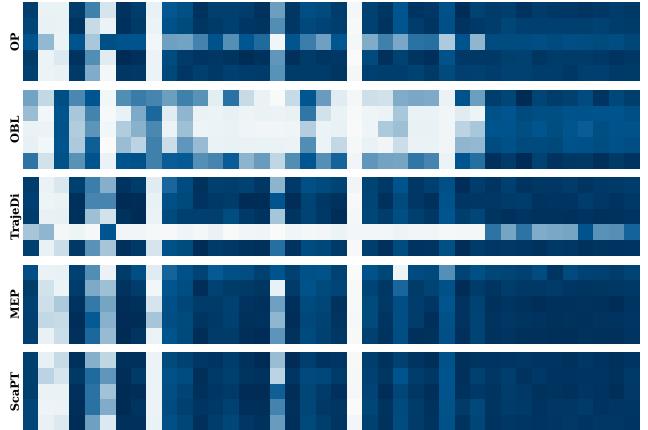
2-player hanabi game with population size(ps)=5

Figure 25: Detailed pair-wise 1ZSC-XP scores of all the testing frameworks(Hanabi, player=2, ps=5).

Detailed results of 5-player hanabi game with different population-size(ps)

Figure 26: Detailed pair-wise 1ZSC-XP scores of all the testing frameworks(Hanabi, player=5, ps=2).

Figure 27: Detailed pair-wise 1ZSC-XP scores of all the testing frameworks(Hanabi, player=5, ps=5).
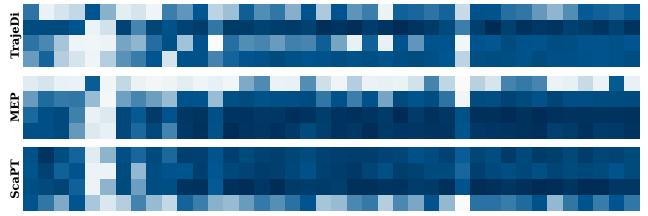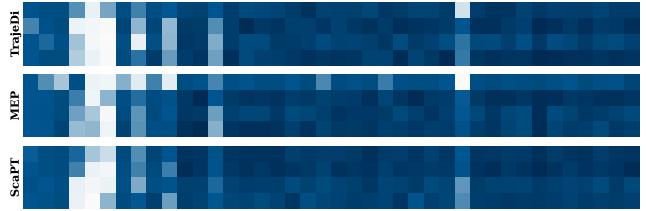
Figure 28: Detailed pair-wise 1ZSC-XP scores of all the testing frameworks(Hanabi, player=5, ps=8).

Figure 29: Detailed pair-wise 1ZSC-XP scores of all the testing frameworks(Hanabi, player=5, ps=15).

E. Detailed introduction of different training modes

Table.1 in the main text presents several population training modes, and the following content takes Mode IV as an example to introduce the meaning of each column:

- **Act Group: MM, MP, PP.**: There are three kinds of act groups will be used for interacting with the environment and generating transitions: [Main agent, Main agent], [Main agent, Partner agent] and [Partner agent, Partner agent]. Consequently,

- **Optimization Objective for** π_m**:** $J(\pi_m, \pi_m)$ + $\sum_{i=1}^{N} J(\pi_m, \pi_{pi})$: The main agent is required to cooperate well with itself and partner agents. This influences the transitions used for training main agent: it has two kinds of transitions, which are playing records with another main agent and playing records with a partner agent, and both of them are used for calculating main agent loss defined in Equation (6).

- **Optimization Objective for** π_p**:** $J(\pi_{pi}, \pi_{pi})$: The partner agents are only required to cooperate well with itself. Notably, it has two kinds of transitions, which are playing records with another partner agent and playing records with a main agent, and only the first will be used for calculating partner agent loss defined in Equation (7). In contrast, in Mode VI, two kinds of transitions are both used for training due to the different optimization objective for π_p.
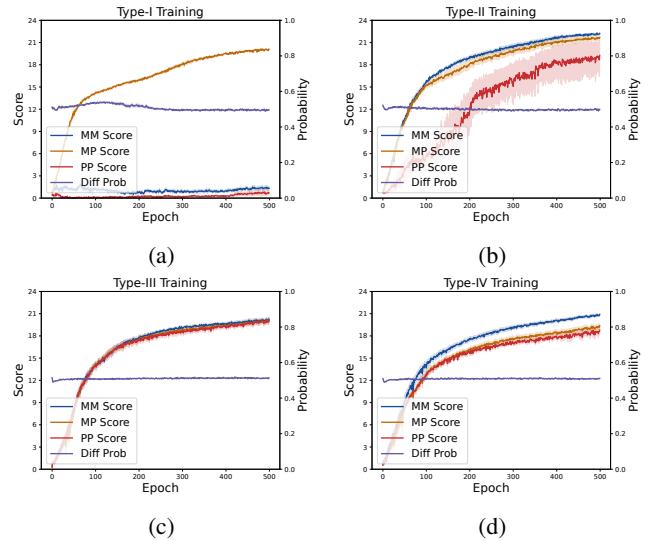

Algorithm 2: Training process of ScaPT with Mode IV

INPUT: Mutual information term weight α, batch size N_b, replay buffers A, B;

1: Initialize $\theta \leftarrow$ random, $\theta_p \leftarrow$ random;
2: Define action groups:
3:     $G_1 = $ [Main agent, Main agent]
4:     $G_2 = $ [Main agent, Partner agent]
5:     $G_3 = $ [Partner agent, Partner agent]
6: **while** not reached maximum iterations **do**
7:     **for** $G \in \{G_1, G_2, G_3\}$ **do**
8:         Reset environment if necessary;
9:         $o_1^t, o_2^t \leftarrow$ Observe(G);
10:        $h_1^t, h_2^t \leftarrow$ Update_hidden_states(o_1^t, o_2^t);
11:        $a_1^t \leftarrow \pi_\theta(h_1^t), a_2^t \leftarrow \pi_\theta(h_2^t)$;
12:        $r_1^t, r_2^t \leftarrow$ Environment_rewards(a_1^t, a_2^t);
13:        **if** $G = G_1$ **then**
14:            Store $\quad (o_1^t, h_1^t, a_1^t, r_1^t, o_1^{t+1}) \quad$ and $(o_2^t, h_2^t, a_2^t, r_2^t, o_2^{t+1}) \in A$;
15:        **end if**
16:        **if** $G = G_2$ **then**
17:            Store $(o_1^t, h_1^t, a_1^t, r_1^t, o_1^{t+1}) \in A$;
18:        **end if**
19:        **if** $G = G_3$ **then**
20:            Store $\quad (o_1^t, h_1^t, a_1^t, r_1^t, o_1^{t+1}) \quad$ and $(o_2^t, h_2^t, a_2^t, r_2^t, o_2^{t+1}) \in B$;
21:        **end if**
22:     **end for**
23:     Update networks:
24:     Sample N_b transitions from A, update θ using loss from Equation (6);
25:     Sample N_b transitions from B, update θ_p using loss from Equation (7);
26: **end while**

Algorithm 2 introduces the training process.
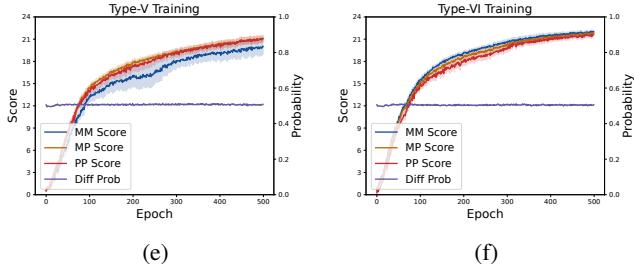

(a)

(b)

(c)

(d)

Figure 30: Training curves of different training modes.

Fig.24 shows the training curves of the different training modes. It can be seen that **Diff Prob** and **MP Score** exhibit consistent trends across all training modes, indicating that different modes are consistent in optimizing the primary objective ($J(\pi_m, \pi_p)$) and enhancing diversity.

Acknowledgments

References

Bard, N.; Foerster, J. N.; Chandar, S.; Burch, N.; Lanctot, M.; Song, H. F.; Parisotto, E.; Dumoulin, V.; Moitra, S.; Hughes, E.; et al. 2020. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280: 103216.

Charakorn, R.; Manoonpong, P.; and Dilokthanakul, N. 2023. Generating diverse cooperative agents by learning incompatible policies. In *The Eleventh International Conference on Learning Representations*.

Cui, B.; Hu, H.; Pineda, L.; and Foerster, J. 2021. K-level Reasoning for Zero-Shot Coordination in Hanabi. *Advances in Neural Information Processing Systems*, 34: 8215–8228.

Do, H.; Jang, J.; and Kim, J. 2025. Heterogeneous multi-robot system mission planning with cooperative replenishment through data-driven rendezvous point selection. *Intelligent Service Robotics*, 18(1): 61–73.

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

Gao, Y.; Liu, F.; Wang, L.; Lian, Z.; Wang, W.; Li, S.; Wang, X.; Zeng, X.; Wang, R.; Wang, J.; et al. 2023. Towards effective and interpretable human-agent collaboration in moba games: A communication perspective. *arXiv preprint arXiv:2304.11632*.

Gupta, A.; Mendonca, R.; Liu, Y.; Abbeel, P.; and Levine, S. 2018. Meta-reinforcement learning of structured exploration strategies. *Advances in neural information processing systems*, 31.

Hu, H.; and Foerster, J. N. 2019. Simplified Action Decoder for Deep Multi-Agent Reinforcement Learning. In *International Conference on Learning Representations*.

Hu, H.; Lerer, A.; Cui, B.; Pineda, L.; Brown, N.; and Foerster, J. 2021. Off-belief learning. In *International Conference on Machine Learning*, 4369–4379. PMLR.

Hu, H.; Lerer, A.; Peysakhovich, A.; and Foerster, J. 2020. "Other-Play" for Zero-Shot Coordination. In *International Conference on Machine Learning*, 4399–4410. PMLR.

Jaderberg, M.; Dalibard, V.; Osindero, S.; Czarnecki, W. M.; Donahue, J.; Razavi, A.; Vinyals, O.; Green, T.; Dunning, I.; Simonyan, K.; et al. 2017. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*.

Lowe, R.; Wu, Y. I.; Tamar, A.; Harb, J.; Pieter Abbeel, O.; and Mordatch, I. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30.

Lucas, K.; and Allen, R. E. 2022. Any-Play: An Intrinsic Augmentation for Zero-Shot Coordination. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 853–861.

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

Ma, J.; Zhao, Z.; Yi, X.; Chen, J.; Hong, L.; and Chi, E. H. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 1930–1939.

Muglich, D.; Schroeder de Witt, C.; van der Pol, E.; Whiteson, S.; and Foerster, J. 2022. Equivariant networks for zero-shot coordination. *Advances in Neural Information Processing Systems*, 35: 6410–6423.

Nagabandi, A.; Clavera, I.; Liu, S.; Fearing, R. S.; Abbeel, P.; Levine, S.; and Finn, C. 2018. Learning to Adapt in Dynamic, Real-World Environments through Meta-Reinforcement Learning. In *International Conference on Learning Representations*.

Rahman, A.; Fosong, E.; Carlucho, I.; and Albrecht, S. V. 2023. Generating Teammates for Training Robust Ad Hoc Teamwork Agents via Best-Response Diversity. *Transactions on Machine Learning Research*.

Sagirova, A.; Kuratov, Y.; and Burtsev, M. 2025. SRMT: shared memory for multi-agent lifelong pathfinding. *arXiv preprint arXiv:2501.13200*.

Schaul, T.; Quan, J.; Antonoglou, I.; and Silver, D. 2015. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

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

Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; et al. 2018. Value-Decomposition Networks

For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2085–2087.

Tan, M. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, 330–337.

Treutlein, J.; Dennis, M.; Oesterheld, C.; and Foerster, J. 2021. A new formalism, method and open issues for zero-shot coordination. In *International Conference on Machine Learning*, 10413–10423. PMLR.

Van Hasselt, H.; Guez, A.; and Silver, D. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Wang, Z.; Schaul, T.; Hessel, M.; Hasselt, H.; Lanctot, M.; and Freitas, N. 2016. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, 1995–2003. PMLR.

Xue, K.; Wang, Y.; Guan, C.; Yuan, L.; Fu, H.; Fu, Q.; Qian, C.; and Yu, Y. 2024. Heterogeneous multi-agent zero-shot coordination by coevolution. *IEEE Transactions on Evolutionary Computation*.

Yan, X.; Guo, J.; Lou, X.; Wang, J.; Zhang, H.; and Du, Y. 2024. An Efficient End-to-End Training Approach for Zero-Shot Human-AI Coordination. *Advances in Neural Information Processing Systems*, 36.

Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; and Wu, Y. 2022. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35: 24611–24624.

Zhao, R.; Song, J.; Yuan, Y.; Hu, H.; Gao, Y.; Wu, Y.; Sun, Z.; and Yang, W. 2023. Maximum entropy population-based training for zero-shot human-ai coordination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 6145–6153.

Zhu, H.; Zhong, H.; Yan, B.; and Li, X. 2024. Dynamic Formation Planning and Control for Robot Soccer Game with Multi-Agent Reinforcement Learning and Behavioral Model. In *2024 International Conference on Advanced Robotics and Mechatronics (ICARM)*, 837–842. IEEE.