

---

# Sequential Multi-Agent Dynamic Algorithm Configuration

---

Anonymous Author(s)

Affiliation

Address

email

## Abstract

1       The performance of an algorithm often critically depends on its hyperparameter con-  
2       figuration. Dynamic Algorithm Configuration (DAC) is a recent trend in automated  
3       machine learning, which can dynamically adjust the algorithm’s configuration  
4       during the execution process and relieves users from tedious trial-and-error tuning  
5       tasks. Recently, Multi-Agent Reinforcement Learning (MARL) approaches have  
6       improved the configuration of multiple heterogeneous hyperparameters, making  
7       various parameter configurations for complex algorithms possible. However, many  
8       complex algorithms have inherent inter-dependencies among multiple parameters  
9       (e.g., determining the operator type first and then the operator’s parameter), which  
10      are, however, not considered in previous approaches, thus leading to sub-optimal  
11      results. In this paper, we propose the Sequential Multi-Agent DAC (Seq-MADAC)  
12      framework to address this issue by considering the inherent inter-dependencies of  
13      multiple parameters. Specifically, we propose a sequential advantage decompo-  
14      sition network, which can leverage action-order information through sequential  
15      advantage decomposition. Experiments from synthetic functions to the config-  
16      uration of multi-objective optimization algorithms demonstrate Seq-MADAC’s  
17      superior performance over state-of-the-art MARL methods and show strong gen-  
18      eralization across problem classes. Seq-MADAC establishes a new paradigm for  
19      the widespread dependency-aware automated algorithm configuration. Our code is  
20      available in the supplemental file.

## 21   1 Introduction

22   Identifying proper configurations of hyperparameters is critical for many learning and optimization  
23   algorithms [13, 24]. While automated methods, e.g., Algorithm Configuration (AC) [12, 22], have  
24   emerged to alleviate user’s burden of manual trial-and-error tuning, the static configuration policies  
25   obtained by AC may not achieve optimal performance, because algorithms may require different  
26   configurations at different execution stages [35].

27   Dynamic AC (DAC) [2, 1] is a prevalent paradigm that enables dynamic adaptation of the algorithm’s  
28   configuration during the execution process, which is more flexible compared to AC. Specifically,  
29   DAC formulates the configuration process as a contextual Markov Decision Process (cMDP) and  
30   then leverages Reinforcement Learning (RL) [30] to learn dynamic configuration policies. DAC has  
31   been shown to outperform static AC methods on many tasks, including learning rate tuning of deep  
32   neural networks [6], step-size control of evolution strategies [33], and heuristic selection mechanisms  
33   in AI planning systems [35]. Due to the increasing complexity of real-world problem modeling, the  
34   performance of an algorithm usually relies on multiple types of hyperparameters. Traditional DAC  
35   methods that tune one type of hyperparameter while fixing the others frozen may not yield promising  
36   results, since they cannot capture the inter-relationships and dependencies between different types of

hyperparameters. Recently, Multi-Agent RL (MARL) approaches have been proposed to enhance the configuration of multiple heterogeneous hyperparameters by modeling it as a Multi-Agent MDP (MMDP) [41], broadening DAC’s application scope and enabling various parameter configurations for complex algorithms.

In addition to multiple complex hyperparameters, another important barrier of real-world AC problems is that these hyperparameters are not completely decoupled, which may have inherent inter-dependencies. For example, when configuring for neural network tuning, once selecting Adam [14] as the optimizer, we can access the Adam-specific parameters, e.g.,  $\beta_1$  and  $\beta_2$ . Unfortunately, previous approaches do not consider this inherent property. Besides, current advanced MARL algorithms mainly focus on issues such as coordination mechanisms in dynamic environments and non-stationary environment adaptation. They are not specifically designed to handle inherent inter-dependencies and communication among agents.

In this paper, we propose a Sequential Multi-Agent DAC (Seq-MADAC) framework to address this issue by considering the inherent inter-dependencies of multiple parameters. Specifically, we formulate this task as a contextual sequential MMDP and propose a sequential advantage decomposition network (SADN), which can leverage action-order information through sequential advantage decomposition. We also theoretically show the rationality of SADN based on Individual Global Max (IGM) principles [34].

Empirically, we compare SADN with other RL methods that consider sequential actions including ACE [18] and SAQL [3], as well as a range of general advanced MARL algorithms, including VDN [37], QMIX [29], MAPPO [43], HAPPO [16], and HASAC [21]. On controllable synthetic functions, we demonstrate that in higher-dimensional problems and noisy scenarios, SADN shows a more pronounced advantage in optimization efficiency and robustness. On more complex and challenging multi-objective optimization problems, SADN shows superior performance over state-of-the-art MARL methods and demonstrates strong generalization across problem classes.

Our contributions include three perspectives:

1. Formulating a sequential decision-making framework to model parameter inter-dependencies as a contextual sequential MMDP, enabling sequential action ordering in dynamic hyperparameter configuration.
2. Introducing SADN with sequential advantage decomposition to exploit action-order information for improved coordination and differentiated credit assignment.
3. Conducting extensive experiments on both simple synthetic and complex multi-objective optimization problems to validate SADN’s effectiveness in capturing parameter dependencies, superior optimization performance, and robust generalization capability.

## 2 Background

### 2.1 Dynamic Algorithm Configuration

Compared to the static configuration scheme of AC, DAC aims at dynamically adjusting algorithm configuration hyperparameters throughout the optimization process, which can be formulated as a contextual MDP  $\mathcal{M}_{\mathcal{I}} := \mathcal{M}_{i \sim \mathcal{I}}$  [2], and be addressed by leverage RL techniques. Here,  $\mathcal{I}$  represents the problem instance space, and each  $\mathcal{M}_i := \langle \mathcal{S}, \mathcal{A}, \mathcal{T}_i, r_i \rangle$  corresponds to one target problem instance  $i \in \mathcal{I}$  [2, 8]. Such context notion  $\mathcal{I}$  enables for studying policy generalization [15]. For a target algorithm  $A$  with configuration hyperparameter space  $\Theta$ , a DAC policy  $\pi \in \Pi$  takes the state  $s \in \mathcal{S}$  as input and outputs an action  $a \in \mathcal{A}$ . Here, the state typically represents the historical performance changes of algorithm  $A$ , while the action corresponds to a hyperparameter configuration  $\theta \in \Theta$  of  $A$ . DAC aims at improving the performance of  $A$  on a set of instances (e.g., optimization functions). Specifically, given a probability distribution  $p$  over the instance space  $\mathcal{I}$ , DAC aims to identify an optimal policy  $\pi^*$ . That is,

$$\pi^* \in \arg \min_{\pi \in \Pi} \int_{i \in \mathcal{I}} p(i) c(\pi, i) di, \quad (1)$$

where  $i \in \mathcal{I}$  is an instance to be optimized, and  $c(\pi, i) \in \mathbb{R}$  is the cost function of the target algorithm with policy  $\pi$  on the instance  $i$ . Previous works have shown the superiority of DAC compared to the static policies in many scenarios, e.g., learning rate adaptation in SGD [6], step-size adaptation in

87 CMA-ES [33], and heuristic selection in planning [35]. However, both of traditional DAC methods  
 88 and these applications all only involve a single type of hyperparameter. Dynamic configurations of  
 89 complex algorithms with multiple types of hyperparameters have been found to be difficult [8, 1].

90 MADAC [41] aims to simultaneously configure multiple hyperparameters of different types, in  
 91 order to cope with the increasing complexity of algorithm structure and increasing number of  
 92 hyperparameters. To address this issue, MADAC models it as a cooperative multi-agent problem with  
 93 one agent configuring one parameter, to take the interactions of different parameters into account.  
 94 Another advantage of the cooperative multi-agent modeling framework is its capacity to address  
 95 the combinatorial explosion challenge associated with joint action spaces. However, in practice,  
 96 parameters may have inherent inter-dependencies to each other (e.g., determining the operator type  
 97 first and then the operator parameters [10]), which is not considered in MADAC. It is quite important  
 98 to take these inherent inter-dependencies into consideration, since we can explore the configuration  
 99 space more effectively without exploring illegal combinations, which can also bring better results.

100 CANDID DAC [3] considers the interaction effects among hyperparameters exhibiting different  
 101 levels of importance, which is termed as Coupled Action-Dimensions with Importance Differences  
 102 (CANDID) and assumes that important parameters should be executed first. However, in practice,  
 103 there exist complex inherent inter-dependencies between parameters, and the important parameters  
 104 may need to be adjusted after their prior parameters (e.g., the operator parameter is more important,  
 105 but better performance can be achieved by adjusting the operator parameter after determining the  
 106 operator type first). Moreover, CANDID DAC tries to address this issue by extending Independent  
 107 Q-Learning (IQL) [38] to Sequential Agent Q-Learning (SAQL), i.e., adding the actions made by  
 108 prior agents to the state of subsequent agents. However, this may suffer similar disadvantages to IQL.  
 109 The team reward is used in the individual agent’s training, and the lack of reward allocation cannot  
 110 reveal the interactions between the agents, which may lead to non-convergence [29], because each  
 111 agent’s learning is interfered by the learning process of others and the team reward cannot provide  
 112 the corresponding guidance.

## 113 2.2 MARL

114 A fully observable cooperative multi-agent system [42] can be formulated as an MMDP [5], defined  
 115 as  $\mathcal{M} := \langle \mathcal{N}, \mathcal{S}, \{\mathcal{A}_j\}_{j=1}^n, \mathcal{T}, r \rangle$ , where  $\mathcal{N}$  represents  $n$  agents,  $\mathcal{S}$  is the state space, and  $\mathcal{A}_j$  is  
 116 agent  $j$ ’s action space. At each time-step, agent  $j \in \mathcal{N}$  acquires  $s \in \mathcal{S}$  and then chooses an action  
 117  $a^{(j)} \in \mathcal{A}_j$ . The joint action  $\mathbf{a} = \langle a^{(1)}, \dots, a^{(n)} \rangle$  leads to next state  $s' \sim \mathcal{T}(\cdot | s, \mathbf{a})$  with a shared  
 118 reward  $r(s, \mathbf{a})$ . The goal of an MMDP is to find a joint policy that maps the states to probability  
 119 distributions over joint actions,  $\pi : \mathcal{S} \rightarrow \Delta(\mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_n)$ , where  $\Delta(\mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_n)$   
 120 stands for the distribution over joint actions, with the goal of maximizing the global value function:

$$Q^\pi(s, \mathbf{a}) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, \mathbf{a}_t) \mid s_0 = s, \mathbf{a}_0 = \mathbf{a} \right]. \quad (2)$$

121 Recently, many algorithms have been proposed to solve the cooperative MARL problem, which can  
 122 be basically divided into two categories: value-based methods and policy gradient-based methods.

123 Among the value-based methods, VDN [37] addresses the problem of multi-agent collaboration  
 124 through value function decomposition. It makes a main assumption that the team Q-value function  
 125 can be additively decomposed into the simple sum of the local Q-values of each agent. That is,  
 126  $Q(s, \mathbf{a}) \approx \sum_{i=1}^n Q_i(s, a_i)$ , where  $Q(s, \mathbf{a})$  is the joint action value function for the whole team,  $s$   
 127 is the shared state,  $\mathbf{a}$  is the joint action,  $n$  is the number of agents,  $Q_i(s, a_i)$  is the agent  $i$ ’s local  
 128 individual action value function, and  $a_i$  is the action taken by agent  $i$ . One critical principle the  
 129 value decomposition methods need to satisfy is the Individual Global Max (IGM) principle defined in  
 130 Definition 1.

131 **Definition 1** (IGM [34]). *For a joint action-value function  $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ , if there exist individual*  
 132 *action-value functions  $[Q_i : \mathcal{S} \times \mathcal{A}_i \rightarrow \mathbb{R}]_{i=1}^n$ , such that the following holds:*

$$\arg \max_{\mathbf{a}} Q(s, \mathbf{a}) = \begin{pmatrix} \arg \max_{a_1} Q_1(s, a_1) \\ \arg \max_{a_2} Q_2(s, a_2) \\ \vdots \\ \arg \max_{a_n} Q_n(s, a_n) \end{pmatrix} \quad (3)$$

133 then, we say that  $[Q_i]$  satisfy IGM for  $Q$ , which means the decomposition satisfies the IGM principle.

134 This critical principle guarantees the optimal efficient decentralized execution, as each agent only  
 135 needs to choose its own optimal action regardless of other agents' actions. Obviously, VDN's additive  
 136 decomposition satisfies the IGM principle, but is highly restricted to the additive form, which fails  
 137 to model the various complex situations in many cooperative MARL scenarios. QMIX [29] uses  
 138 a mixing network with non-negative weights to learn a monotonic combination of local individual  
 139 action value functions as the joint action value function for the whole team, which extends the forms  
 140 of factorization.

141 Among the policy gradient-based methods, MAPPO [43] extends the popular single-agent RL  
 142 algorithm Proximal Policy Optimization (PPO) [32] to the multi-agent RL setting and achieves  
 143 surprising effectiveness. However, MAPPO is mainly designed for homogeneous agents with shared  
 144 action space and policy parameters. HAPPO [16] further addresses heterogeneous agents with  
 145 theoretical property of monotonic improvement guarantee. HASAC [21] obtains the maximum  
 146 entropy objective for MARL from probabilistic graphical models to escape from converging to a  
 147 suboptimal Nash Equilibrium. A2PO [28] is an agent-by-agent sequential update algorithm with  
 148 theoretical guarantee of the monotonic policy improvement that accelerates optimization with a  
 149 semi-greedy agent selection rule. However, these recent works do not fit the need of DAC well. Their  
 150 execution is fully decentralized with each agent unaware of others' current actions, which ignores the  
 151 inherent inter-dependencies between the agents' actions.

152 ACE [18] models the multi-agent decision-making problem into a sequential decision-making problem  
 153 with the agents taking action one by one, formulating this as a Sequentially Expanded MDP (SE-  
 154 MDP). At each timestep, the agents make decisions based on the actions taken by their prior agents  
 155 at the current timestep, and the action value function of each individual agent is updated according to  
 156 the performance of their subsequent agents which make decisions based on the taken action. The  
 157 update scheme of action value functions is as follows:

$$Q_i(s, \mathbf{a}_{1:i-1}, a_i) \leftarrow \begin{cases} \max_{a_{i+1}} Q_{i+1}(s, \mathbf{a}_{1:i}, a_{i+1}), & \text{if } i < n \\ r + \gamma \max_{a'_1} Q_1(s', a'_1) & \text{if } i = n \end{cases} \quad (4)$$

158 The sequential modeling relieves the non-stationary training problem in MARL because the agents  
 159 are aware of their prior agents' actions, which also fits the need of DAC for being able to capture the  
 160 inherent inter-dependencies between the agents' actions. However, the update scheme of ACE can  
 161 suffer from fragile training due to its long sequence of action value function update. That is, if some  
 162 agents fail or fall into local optima within the long action value function update chain, the learning of  
 163 the whole chain will be affected and even damaged.

## 164 3 Method

### 165 3.1 Contextual Sequential Multi-Agent MDP

166 Previous works have paid attention to sequential modeling to cope with the high dimensional action  
 167 spaces (Sequential MDP [25]) and non-stationary training (SE-MDP [18]) in MARL. The main idea is  
 168 to break the global state transition into a sequence of intermediate decision making steps, and select the  
 169 current timestep's action one dimension by one dimension. That is, the original transition  $(s, \mathbf{a}, s')$  is  
 170 transformed into  $n$  intermediate decision making steps  $(s, a_1), (s, a_1, a_2), \dots, (s, a_1, a_2, \dots, a_n, s')$ ,  
 171 where  $n$  is the number of action dimension, and when selecting the  $i$ -th action  $a_i$  for dimension  
 172  $i$ , the previous selected actions  $\mathbf{a}_{1:i-1}$  for dimension 1 to  $i - 1$  are available for decision making.  
 173 Prior work shows that under the CANDID properties [3], sequential policies can coordinate action  
 174 selection between dimensions while avoiding combinatorial explosion of the action space. Their  
 175 results encourage an extended study of sequential policies for DAC.

176 However, the previous sequential modeling is typically a single-agent MDP, with one agent selecting  
 177 actions one dimension by one dimension, which fails to model the complex cooperation and com-  
 178 munication among heterogeneous agents. Moreover, the different contributions of different action  
 179 dimensions to the global reward are neglected in the sequential single-agent MDP modeling, which  
 180 harms the learning efficiency and may lead to sub-optimal results.

181 In the DAC task, there are often different heterogeneous parameters (i.e., they are of different types  
 182 and have different effects on the final performance of the target algorithm) with complex inherent



206 Following Definition 2, we give the multi-agent advantage decomposition lemma [16] in the sequential  
 207 form in Lemma 1. For the proof, please see Appendix A.3.

**Lemma 1** (Multi-agent Advantage Decomposition [16]). *In any cooperative Markov game, given a joint policy  $\pi$ , the global advantage function  $A^\pi(s, \mathbf{a})$ , and  $n$  agents in total, for any state  $s$ , the following equations hold:*

$$A^\pi(s, \mathbf{a}) = \sum_{i=1}^n A_i^\pi(s, \mathbf{a}_{1:i-1}, a_i).$$

208 Inspired by the action value function decomposition and the implicit reward assignment by using a  
 209 decomposition network proposed in [37], as well as the multi-agent advantage decomposition lemma  
 210 in [16], we decompose the advantage function sequentially. In specific, the  $i$ -th agent maintains  
 211 a network that models the  $i$ -th advantage function  $A_i^\pi(s, \mathbf{a}_{1:i-1}, a_i)$  and selects the action that  
 212 maximizes it, and we compute the global advantage function by the sum of all individual advantage  
 213 functions, using Lemma 1. The  $i$ -th advantage function is updated implicitly when the global  
 214 advantage function is updated by backpropagating gradients. To update the global advantage function,  
 215 we maintain a global value network, like the critic in the actor-critic framework, and compute the  
 216 target global advantage function by Generalized Advantage Estimation (GAE, [31]) with  $\lambda = 0$  (i.e.,  
 217 one-step temporal difference), which corresponds to the update scheme of action value function in  
 218 Q-learning (the detailed proof see Appendix A.1). The global advantage function update scheme is  
 219 as follows:

$$A(s, \mathbf{a}) \leftarrow A(s, \mathbf{a}) + \alpha \cdot [r + \gamma V(s') - V(s) - A(s, \mathbf{a})],$$

220 where  $A(s, \mathbf{a})$  is the global advantage function,  $s$  is the current state,  $\mathbf{a}$  is the joint action,  $s'$  is the  
 221 next state,  $\alpha$  is the steplength,  $\gamma$  is the discount factor, and  $V(s)$  is the global value function.

222 Our method also satisfies the IGM principle in the sequential setting, which is stated in Theorem 1.

223 **Theorem 1.** *Selecting the best joint action  $\mathbf{a}$  for state  $s$  to maximize the global action value function*  
 224 *is equivalent to sequentially selecting the best action  $a_i$  for state  $s$  to maximize the agent  $i$ 's advantage*  
 225 *function. That is,*

$$\arg \max_{\mathbf{a}} Q(s, \mathbf{a}) = \left( \begin{array}{c} \arg \max_{a_1} A_1(s, a_1) \\ \arg \max_{a_2} A_2(s, \arg \max_{a_1} A_1(s, a_1), a_2) \\ \vdots \\ \arg \max_{a_n} A_n(s, \arg \max_{a_1} A_1(s, a_1), \arg \max_{a_2} A_2(s, \arg \max_{a_1} A_1(s, a_1), a_2), \dots, a_n) \end{array} \right).$$

226 Detailed proofs are provided in Appendix A.2.

## 227 4 Experiment

### 228 4.1 Experimental Settings

229 To comprehensively evaluate the performance of SADN, we compare it with other RL methods  
 230 that consider sequential actions, including ACE [18] and SAQL [3], as well as a range of general  
 231 advanced MARL algorithms, including value-based methods like VDN [37] and QMIX [29], and  
 232 policy gradient-based methods like MAPPO [43], HAPPO [16] and HASAC [21]. We also compare it  
 233 with an extension of MAPPO, MAPPOar [9], which considers sequential actions by simply adding the  
 234 prior actions to the state of the subsequent agents. We also investigate HAPPO's ability of sequential  
 235 modeling by adjusting the permutation of updating agents.

236 We consider the following environments for comparing these methods:

237 **Sigmoid.** The Sigmoid task [2] is basically an approximation task with parameters changing along  
 238 the time. For a sampled instance  $i$  and the  $h$ -th parameter, the target sigmoid function to approximate  
 239 is defined as  $\text{sig}(t, s_{i,h}, p_{i,h}) = \frac{1}{1 + e^{-s_{i,h}(t - p_{i,h})}}$ , and the team reward at timestep  $t$  on instance  $i$  is  
 240 defined as  $r_t^i = \prod_{h=0}^{H-1} (1 - |\text{sig}(t, s_{i,h}, p_{i,h}) - a_{h,t}|)$ , where  $H$  is the number of parameters, and  $a_{h,t}$   
 241 is the configuration for the  $h$ -th parameter at timestep  $t$ .

**Seq-Sigmoid.** Despite considering multiple parameters, the original Sigmoid lacks parameters with inherent inter-dependencies, making it unable to reflect the actual situation and the behavior of complex algorithms. Therefore, we modify the original Sigmoid into Seq-Sigmoid to bring in the inherent inter-dependencies among parameters. The main modification is that at timestep  $t$ , the former parameter’s value will determine a scaling factor  $\alpha_{h-1,t}$ , which controls the latter parameter’s slope  $s_{i,h}$ . In this way, the former parameter’s configuration will have strong influence on the configuration of the latter parameter. Moreover, in order to avoid that having this dependency affects the configuration of the former parameter itself, we use the formula  $\min(|\text{sig}(t, \alpha_{h,t}s_{i,h}, p_{i,h}) - a_{h,t}|, |1 - \text{sig}(t, \alpha_{h,t}s_{i,h}, p_{i,h}) - a_{h,t}|)$  to measure the approximation, which returns the same value when two values of  $a_{h,t}$  are symmetrical about 0.5.

**Seq-Sigmoid-Mask.** This benchmark masks  $s_{i,h}$  and  $p_{i,h}$  (i.e.,  $\text{state}_t \in \{t\}$ ) and sets  $s_{i,h} = 1$  to avoid too much randomness in Seq-Sigmoid, which makes the task a single instance task (i.e., the approximation of  $\text{sig}(t, 1, p_h)$ ), but with randomness (i.e.,  $p_h \sim \mathcal{N}(T/2, T/4, H)$ ).

**Seq-Sigmoid-Robust ( $n$ ).** This benchmark gives  $n$  parameters random configurations no matter how the corresponding agents tune them, with other parameters configured properly. This is to simulate the ineffective learning of some agents in complex dynamic algorithm configuration scenarios.

**MOEA/D.** MOEA/D [41, 44] is a real-scenario DAC benchmark [41] based on the popular multi-objective evolutionary algorithm MOEA/D [44] with heterogeneous parameters, which is a strong stochastic benchmark due to the randomness of the evolutionary optimization process. The action space of the MOEA/D environment includes four parameters of MOEA/D: weights, neighborhood size, types of the reproduction operator, and the corresponding parameters of the reproduction operators. The state includes the features of the problem instance, the optimization process, and the evolution statistics of the current population. For the reward, we use the triangle-based reward function proposed in [41], which has been proved effective in the MOEA/D environment.

Details on the Seq-Sigmoid and the MOEA/D environments can be found in Appendix B.

## 4.2 RQ1: Is the sequential information useful?

On the original Sigmoid benchmark, we traditionally train the agents with a set of given instances (i.e., a set of  $s_{i,h}$  and  $p_{i,h}$ ) and test them on other sampled instances. In order to better reflect the generalization ability of the algorithms, we resample the instance (i.e., resample a new  $s_{i,h}$  and  $p_{i,h}$ ) every time when one episode is done and the environment is reset. Therefore, the agents are truly trained over the distribution of the problem instances, and every point on the training curve can reflect the generalization ability of the learned policy at the exact training phase.

Figure 2 demonstrates the smoothed training curves of the return values of our method and the compared methods on Seq-Sigmoid and Seq-Sigmoid-Mask with 5/10 dimensions (i.e., 5/10 parameters). As mentioned in the previous paragraph, we resample the instance every episode, and thereby the training curves reflect the policies’ performance over the instance distribution at certain training phase. It can be observed that our method SADN outperforms other methods on every benchmark, with faster convergence, less variance, and better final performance. Basically, the methods exploiting the sequential information (i.e., SADN, ACE, and SAQL) succeed to capture the inherent inter-dependencies between the parameters to some extent and achieve better performance, especially on the Seq-Sigmoid-Mask benchmark, where the traditional MARL methods fail to cope with the complex inter-dependencies and randomness. However, ACE shows apparent performance decay on the Seq-Sigmoid benchmark, which may be because the diversity of problem instances damages the long chain of action value function update scheme in ACE (i.e., some action value functions in the chain fail to adapt to new instances and harm the learning process of the whole chain).

The comparison of the sequential methods based on the correct order and the reverse order is given in Appendix C.2 due to space limitation. Basically, the reverse order underperforms the correct order and produces larger variance due to the wrong capture of the inherent inter-dependencies between the parameters, except for HAPPO, which only considers the update order of the agents rather than the order of taking actions. The comparison of using the correct and the reverse order provides evidence for the effectiveness of modeling the inherent inter-dependencies correctly.

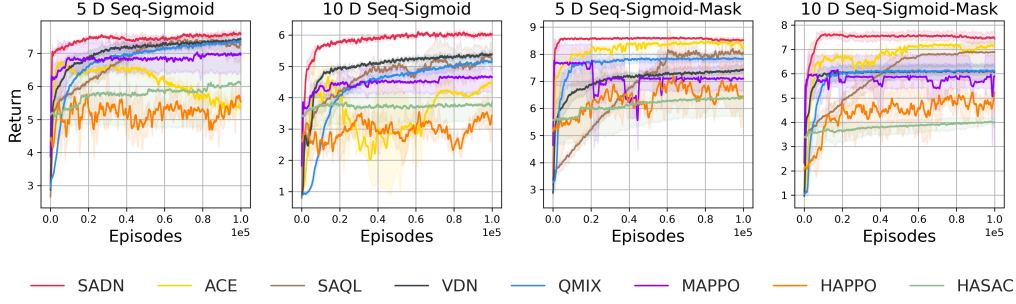


Figure 2: Training curves of return value obtained by the compared methods on four Seq-Sigmoid variant tasks, where the results are averaged over 3 runs.

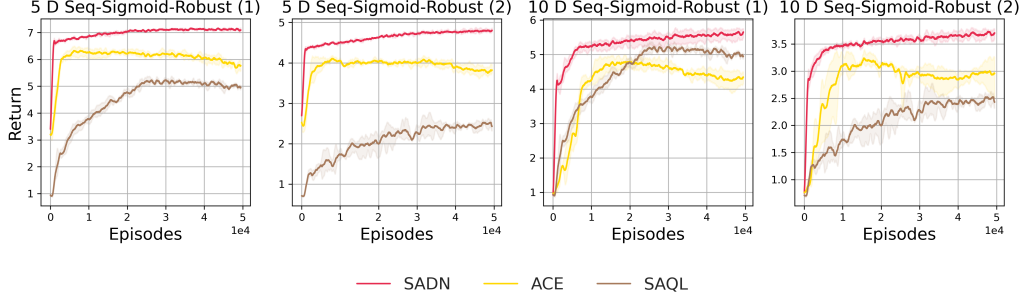


Figure 3: Training curves of return value obtained by the sequential methods on four Seq-Sigmoid-Robust tasks, where the results are averaged over 3 runs.

293 We also provide experiment results of our method and compared methods on the original Sigmoid in  
 294 Appendix C.1 due to space limitation, to show the effectiveness of our method even without strong  
 295 inter-dependencies between the parameters.

### 296 4.3 RQ2: How robust is SADN?

297 In complex dynamic parameter configuration problems, some agents often struggle to learn effectively,  
 298 e.g., the configuration range of one certain parameter is not suitable, causing the corresponding agent  
 299 to struggle to learn effectively. Especially in sequential scenarios, this issue can be more pronounced,  
 300 because those hindering agents can give misleading information through communication to other  
 301 agents. Therefore, we test the methods designed for sequential decision-making on Seq-Sigmoid-  
 302 Robust, to examine their ability to cope with this issue.

303 Figure 3 plots the training curves of SADN (ours), ACE and SAQL on Seq-Sigmoid-Robust with  
 304 5/10 dimensions and 1/2 random agents. SADN shows better performance and robustness when  
 305 coping with this complex situation, while ACE shows apparent performance decay because its fragile  
 306 long chain update scheme suffers from the instability of learning caused by the random agents, and  
 307 SAQL sticks in poor performance due to the interference of the random agents.

### 308 4.4 RQ3: How does SADN perform on tuning real complex algorithms MOEA/D?

309 In the MOEA/D environment, we set the proper parameter order as: 1) determining the neighbor-  
 310 hood size to decide whether to explore or exploit; 2) choosing the operator type; 3) choosing the  
 311 corresponding operator’s parameters; 4) deciding whether to update the weights to bring in new  
 312 subproblems. We compare our SADN with different SOTA MARL methods, as well as the RL



Table 1: IGD values obtained by SADN and the compared methods on different problems. Each result consists of the mean of 10 runs. The best mean value on each problem is highlighted in bold. The symbols ‘+’, ‘−’ and ‘≈’ indicate that the result is significantly superior to, inferior to, and almost equivalent to our method SADN, respectively, according to the Wilcoxon rank-sum test with significance level 0.05. The top three problems are the problems for training, and the bottom five problems are for testing.

Problem	Dim	MOEA/D	SADN	ACE	SAQL	VDN	QMIX	MAPPO	HAPPO	HASAC
DTLZ2	6	4.593e-02	3.809e-02	3.851e-02	3.950e-02	3.905e-02	3.906e-02	<b>3.79e-02</b>	3.838e-02	4.045e-02
	9	4.598e-02	<b>3.875e-02</b>	4.057e-02	4.337e-02	4.009e-02	4.120e-02	3.907e-02	4.095e-02	4.103e-02
	12	4.599e-02	<b>3.874e-02</b>	4.063e-02	4.698e-02	3.958e-02	4.161e-02	3.970e-02	4.075e-02	4.140e-02
WFG4	6	5.729e-02	<b>4.537e-02</b>	4.626e-02	4.817e-02	4.904e-02	4.827e-02	4.639e-02	4.956e-02	5.703e-02
	9	5.736e-02	<b>5.190e-02</b>	5.853e-02	5.365e-02	5.800e-02	5.541e-02	5.341e-02	6.138e-02	6.169e-02
	12	5.751e-02	<b>5.213e-02</b>	5.751e-02	6.241e-02	5.687e-02	5.835e-02	5.340e-02	6.071e-02	6.187e-02
WFG6	6	5.855e-02	<b>3.908e-02</b>	3.962e-02	3.964e-02	4.080e-02	4.214e-02	3.937e-02	4.022e-02	4.536e-02
	9	6.641e-02	4.884e-02	5.111e-02	<b>4.384e-02</b>	5.558e-02	7.416e-02	6.194e-02	5.921e-02	5.879e-02
	12	6.864e-02	<b>4.837e-02</b>	5.490e-02	5.676e-02	5.444e-02	7.999e-02	6.067e-02	5.218e-02	4.993e-02
Train:+/-≈		0/9/0		0/8/1	1/8/0	0/9/0	0/9/0	1/8/0	0/9/0	0/8/1
DTLZ4	6	5.455e-02	<b>3.895e-02</b>	4.221e-02	5.019e-02	3.986e-02	4.331e-02	3.909e-02	3.987e-02	4.410e-02
	9	6.229e-02	7.319e-02	4.492e-02	6.895e-02	<b>4.361e-02</b>	7.108e-02	6.168e-02	4.645e-02	4.678e-02
	12	6.790e-02	7.852e-02	4.650e-02	8.589e-02	<b>4.647e-02</b>	5.532e-02	5.794e-02	5.107e-02	4.825e-02
WFG5	6	6.303e-02	4.749e-02	4.752e-02	5.390e-02	4.761e-02	4.786e-02	<b>4.719e-02</b>	4.746e-02	4.744e-02
	9	6.351e-02	4.773e-02	4.792e-02	4.788e-02	4.784e-02	4.770e-02	<b>4.762e-02</b>	4.793e-02	4.780e-02
	12	6.381e-02	4.793e-02	4.791e-02	4.954e-02	4.780e-02	<b>4.778e-02</b>	4.789e-02	4.799e-02	4.785e-02
WFG7	6	5.803e-02	<b>3.893e-02</b>	3.942e-02	3.958e-02	4.051e-02	4.207e-02	3.918e-02	3.980e-02	4.372e-02
	9	5.812e-02	<b>4.053e-02</b>	4.470e-02	4.257e-02	4.566e-02	4.407e-02	4.119e-02	4.607e-02	4.632e-02
	12	5.814e-02	<b>4.060e-02</b>	4.447e-02	4.975e-02	4.432e-02	4.445e-02	4.132e-02	4.597e-02	4.689e-02
WFG8	6	<b>7.875e-02</b>	1.018e-01	1.029e-01	1.054e-01	1.169e-01	1.054e-01	1.037e-01	1.156e-01	1.252e-01
	9	9.680e-02	<b>7.853e-02</b>	9.787e-02	9.716e-02	8.720e-02	8.180e-02	7.880e-02	9.964e-02	9.991e-02
	12	8.710e-02	<b>7.891e-02</b>	8.387e-02	9.023e-02	8.552e-02	8.279e-02	7.923e-02	8.636e-02	8.734e-02
WFG9	6	5.600e-02	3.987e-02	4.012e-02	4.010e-02	4.156e-02	4.266e-02	<b>3.986e-02</b>	4.041e-02	4.395e-02
	9	5.748e-02	4.341e-02	4.552e-02	<b>4.281e-02</b>	7.965e-02	5.274e-02	4.573e-02	5.079e-02	5.429e-02
	12	5.827e-02	<b>4.249e-02</b>	4.477e-02	7.291e-02	8.023e-02	5.246e-02	4.495e-02	6.556e-02	8.189e-02
Test:+/-≈		1/12/2		2/9/4	0/11/4	2/12/1	2/11/2	4/5/6	2/11/2	2/10/3
average rank	6	8	<b>1.75</b>	3.625	5.5	5.75	6.25	2.	4.875	7.25
	9	7.375	<b>2.5</b>	4.75	4.375	4.75	5.375	3.125	6.25	6.5
	12	7.25	<b>2.5</b>	3.875	8.125	3.5	4.875	3.625	5.5	5.75

methods that consider sequential actions. We use the Inverted Generational Distance (IGD) [4] as the metric to measure the performance of the algorithms, which is the smaller the better.

As demonstrated in Table 1, our method SADN achieves significantly superior performance on most problem instances, and averagely ranks the best on all problem dimension settings. Moreover, our method shows strong generalization abilities across problem classes. The comparison results of using the correct and the reverse order, which are provided in Appendix C.3 due to space limitation, also demonstrate the effectiveness of correctly considering the parameter order on the real-scenario multi-objective evolutionary algorithm. Generally, the correct order produces better results than the reverse order within the same sequential methods.

## 5 Conclusion

This paper considers the inherent inter-dependencies among hyperparameters in DAC, which are natural and common in practice. The previous contextual MMDP modeling fails to capture these inherent inter-dependencies, while the previous cooperative MARL algorithms cannot utilize the information for better performance. Thus, we propose a new contextual sequential MMDP formulation to capture the inherent inter-dependencies among hyperparameters and also propose a sequential advantage decomposition network to solve it. Experiments from synthetic white-box tasks with apparent inter-dependencies to a complex real-scenario multi-objective evolution algorithm demonstrate the effectiveness of our proposed method, as well as its strong generalization ability. One limitation is that the parameter order requires to be given by the experts in advance, which may be difficult with a large number of parameters. In the future, one could apply large language models to set the proper order of the parameter configuration, as well as causal models to automatically learn the parameter’s causal structure from the data [23].

## References

- [1] S. Adriaensen, A. Biedenkapp, G. Shala, N. Awad, T. Eimer, M. Lindauer, and F. Hutter. Automated dynamic algorithm configuration. *arXiv preprint arXiv:2205.13881*, 2022.
- [2] A. Biedenkapp, H. F. Bozkurt, T. Eimer, F. Hutter, and M. Lindauer. Dynamic algorithm configuration: Foundation of a new meta-algorithmic framework. In *Proceedings of the 24th European Conference on Artificial Intelligence*, pages 427–434, Santiago de Compostela, Spain, 2020.
- [3] P. Bordne, M. A. Hasan, E. Bergman, N. Awad, and A. Biedenkapp. Candid dac: Leveraging coupled action dimensions with importance differences in DAC. *arXiv preprint arXiv:2407.05789*, 2024.
- [4] P. A. N. Bosman and D. Thierens. The balance between proximity and diversity in multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7(2):174–188, 2003.
- [5] C. Boutilier. Planning, learning and coordination in multiagent decision processes. In *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 195–210, De Zeeuwse Stromen, The Netherlands, 1996.
- [6] C. Daniel, J. Taylor, and S. Nowozin. Learning step size controllers for robust neural network training. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 1519–1525, Phoenix, AZ, 2016.
- [7] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 825–830, Honolulu, HI, 2002.
- [8] T. Eimer, A. Biedenkapp, M. Reimer, S. Adriaensen, F. Hutter, and M. Lindauer. DACBench: A benchmark library for dynamic algorithm configuration. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pages 1668–1674, Montreal, Canada, 2021.
- [9] W. Fu, C. Yu, Z. Xu, J. Yang, and Y. Wu. Revisiting some common practices in cooperative multi-agent reinforcement learning. In *Proceedings of the 39th International Conference on Machine Learning*, Baltimore, Maryland, USA, 2022.
- [10] H. Guo, Z. Ma, J. Chen, Y. Ma, Z. Cao, X. Zhang, and Y.-J. Gong. ConfigX: Modular configuration for evolutionary algorithms via multitask reinforcement learning. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*, number 25, pages 26982–26990, 2025.
- [11] S. Huband, P. Hingston, L. Barone, and L. While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006.
- [12] F. Hutter, H. H. Hoos, K. Leyton-Brown, and T. Stützle. Paramils: An automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36:267–306, 2009.
- [13] F. Hutter, L. Kotthoff, and J. Vanschoren. *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2019.
- [14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.
- [15] R. Kirk, A. Zhang, E. Grefenstette, and T. Rocktäschel. A survey of generalisation in deep reinforcement learning. *arXiv preprint arXiv:2111.09794*, 2021.
- [16] J. G. Kuba, R. Chen, M. Wen, Y. Wen, F. Sun, J. Wang, and Y. Yang. Trust region policy optimisation in multi-agent reinforcement learning. *arXiv preprint arXiv:2109.11251*, 2021.
- [17] S. Kukkonen and K. Deb. A fast and effective method for pruning of non-dominated solutions in many-objective problems. In *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature*, pages 553–562, Reykjavik, Iceland, 2006.

- [18] C. Li, J. Liu, Y. Zhang, Y. Wei, Y. Niu, Y. Yang, Y. Liu, and W. Ouyang. Ace: Cooperative multi-agent q-learning with bidirectional action-dependency. In *In Proceedings of the 37th AAAI conference on artificial intelligence*, pages 8536–8544, Washington, DC, 2023.
- [19] K. Li. Decomposition multi-objective evolutionary optimization: From state-of-the-art to future opportunities. *arXiv preprint arXiv:2108.09588*, 2021.
- [20] K. Li, A. Fialho, S. Kwong, and Q. Zhang. Adaptive operator selection with bandits for a multi-objective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 18(1):114–130, 2013.
- [21] J. Liu, Y. Zhong, S. Hu, H. Fu, Q. FU, X. Chang, and Y. Yang. Maximum entropy heterogeneous-agent reinforcement learning. In *The Twelfth International Conference on Learning Representations*, Vienna, Austria, 2024.
- [22] M. López-Ibáñez, J. Dubois-Lacoste, L. Pérez Cáceres, M. Birattari, and T. Stützle. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58, 2016.
- [23] Z. Ma, J. Chen, H. Guo, and Y.-J. Gong. Neural exploratory landscape analysis for meta-black-box-optimization. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [24] Z. Ma, H. Guo, Y.-J. Gong, J. Zhang, and K. C. Tan. Toward automated algorithm design: A survey and practical guide to meta-black-box-optimization. *IEEE Transactions on Evolutionary Computation*, 2025.
- [25] L. Metz, J. Ibarz, N. Jaitly, and J. Davidson. Discrete sequential prediction of continuous actions for deep RL. *arXiv preprint arXiv:1705.05035*, 2017.
- [26] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.
- [27] Y. Qi, X. Ma, F. Liu, L. Jiao, J. Sun, and J. Wu. MOEA/D with adaptive weight adjustment. *Evolutionary Computation*, 22(2):231–264, 2014.
- [28] Y. Qing, S. Liu, J. Cong, K. Chen, Y. Zhou, and M. Song. A2po: Towards effective offline reinforcement learning from an advantage-aware perspective. *Advances in Neural Information Processing Systems*, 37:29064–29090, 2024.
- [29] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4295–4304, Stockholm, Sweden, 2018.
- [30] S. R. S. and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.
- [31] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.
- [32] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [33] G. Shala, A. Biedenkapp, N. H. Awad, S. Adriaensen, M. Lindauer, and F. Hutter. Learning step-size adaptation in CMA-ES. In *Proceedings of the 16th International Conference on Parallel Problem Solving from Nature*, pages 691–706, Leiden, The Netherlands, 2020.
- [34] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, pages 5887–5896. PMLR, 2019.

- [35] D. Speck, A. Biedenkapp, F. Hutter, R. Mattmüller, and M. Lindauer. Learning heuristic selection with dynamic algorithm configuration. In *Proceedings of the 31th International Conference on Automated Planning and Scheduling*, pages 597–605, Guangzhou, China, 2021.
- [36] J. Sun, X. Liu, T. Bäck, and Z. Xu. Learning adaptive differential evolution algorithm from optimization experiences by policy gradient. *IEEE Transactions on Evolutionary Computation*, 25(4):666–680, 2021.
- [37] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. F. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. 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*, pages 2085–2087, Stockholm, Sweden, 2018.
- [38] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395, 2017.
- [39] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh. A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Transactions on Evolutionary Computation*, 21(3):440–462, 2017.
- [40] Z. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao. Adaptive replacement strategies for MOEA/D. *IEEE Transactions on Cybernetics*, 46(2):474–486, 2016.
- [41] K. Xue, J. Xu, L. Yuan, M. Li, C. Qian, Z. Zhang, and Y. Yu. Multi-agent dynamic algorithm configuration. In *Advances in Neural Information Processing Systems*, pages 20147–20161, New Orleans, LA, USA, 2022.
- [42] Y. Yang and J. Wang. An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*, 2020.
- [43] C. Yu, A. Velu, E. Vinitzky, J. Gao, Y. Wang, A. Bayen, and Y. Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, pages 24611–24624, 2022.
- [44] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [\[Yes\]](#)

Justification: The main claims are that sequential modeling is more effective in the DAC task and experiments in section 4 proves it.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: In section 5, we discuss the limitations of our work as requiring experts providing parameter order in advance.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: In Appendix A, we provide the proof of the advantage update scheme, our method's IGM principle satisfaction, and the lemma proposed in [16]

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

### 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: In Appendix B, we propose the detailed information about the environment setting parameters and the algorithms' hyperparameters.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

## 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: In the Abstract, we provide an anonymized URL to our source code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: In Appendix B, we propose the detailed information about the environment setting parameters and the algorithms' hyperparameters.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [\[Yes\]](#)

Justification: In section 4, our experiments are all conducted on various different seeds and use the mean value as the results. We use the Wilcoxon rank-sum test with significance level 0.05 to measure whether we get significantly better performance.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: All the experiments are easy to carry out, which has little requirement for computer resources, and the detailed information of the computer resources we use see Appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: This paper does not involve human subjects or participants, and only uses open-source benchmarks. This paper focuses on algorithm configuration, which has little societal impact and potential harmful consequences.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: This paper focuses on algorithm configuration, which has little societal impact and potential harmful consequences.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.



- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

## 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper does not involve released models and datasets.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the original paper of all the open-source code and benchmarks we use in our paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

## 13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We provide the documentation alongside our code in an anonymized URL.

Guidelines:

- The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

#### 14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper focuses on algorithm configuration, which does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

#### 15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper focuses on algorithm configuration, which does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

#### 16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [NA]

Justification: Every components of our method does not involve LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

## 789 A Proof

### 790 A.1 Deriving the global advantage update scheme from the Q-learning update scheme

In this appendix, we derive our global advantage update scheme (i.e., the GAE with  $\lambda = 0$ ):

$$A(s, \mathbf{a}) \leftarrow A(s, \mathbf{a}) + \alpha[r + \gamma V(s') - V(s) - A(s, \mathbf{a})]$$

from the Q-learning update scheme

$$Q(s, \mathbf{a}) \leftarrow Q(s, \mathbf{a}) + \alpha[r + \gamma \max_{\mathbf{a}'} Q(s', \mathbf{a}') - Q(s, \mathbf{a})]$$

791 as follows:

$$Q(s, \mathbf{a}) \leftarrow Q(s, \mathbf{a}) + \alpha[r + \gamma \max_{\mathbf{a}'} Q(s', \mathbf{a}') - Q(s, \mathbf{a})]$$

$$A(s, \mathbf{a}) + V(s) \leftarrow A(s, \mathbf{a}) + V(s) + \alpha[r + \gamma \max_{\mathbf{a}'} (A(s', \mathbf{a}') + V(s')) - A(s, \mathbf{a}) - V(s)]$$

$$A(s, \mathbf{a}) \leftarrow A(s, \mathbf{a}) + \alpha[r + \gamma \max_{\mathbf{a}'} A(s', \mathbf{a}') + \gamma V(s') - A(s, \mathbf{a}) - V(s)]$$

$$A(s, \mathbf{a}) \leftarrow A(s, \mathbf{a}) + \alpha[r + \gamma V(s') - V(s) - A(s, \mathbf{a})]$$

792 Note that in Q-learning, we choose the action greedily with respect to the learned Q-function by  
 793  $\pi(s) := \arg \max_{a \in \mathcal{A}} Q(s, a)$ , and  $V(s) = \mathbb{E}_{\pi}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s] = \max_{a \in \mathcal{A}} Q(s, a)$ . Therefore,  
 794  $\max_{\mathbf{a}'} A(s', \mathbf{a}') = \max_{\mathbf{a}'} Q(s', \mathbf{a}') - V(s') = 0$

### 795 A.2 Proof of Theorem 1

796 **Theorem 1.** *Selecting the best joint action  $\mathbf{a}$  for state  $s$  to maximize the global action value function*  
 797 *is equivalent to sequentially selecting the best action  $a_i$  for state  $s$  to maximize the agent  $i$ 's advantage*  
 798 *function. That is,*

$$\arg \max_{\mathbf{a}} Q(s, \mathbf{a}) = \left( \begin{array}{c} \arg \max_{a_1} A_1(s, a_1) \\ \arg \max_{a_2} A_2(s, \arg \max_{a_1} A_1(s, a_1), a_2) \\ \vdots \\ \arg \max_{a_n} A_n(s, \arg \max_{a_1} A_1(s, a_1), \arg \max_{a_2} A_2(s, \arg \max_{a_1} A_1(s, a_1), a_2), \dots, a_n) \end{array} \right).$$

799 *Proof.* Let  $\mathbf{a}^*$  denote the optimal action that maximizes the Q-function (that is,  $\mathbf{a}^* =$   
 800  $\arg \max_{\mathbf{a}} Q(s, \mathbf{a})$ ), and for the agent  $i$ , its policy based on the actions of its prior agents be de-  
 801 noted as  $\pi_i(s, \mathbf{a}_{1:i-1}) := \arg \max_{a_i} A_i(s, \mathbf{a}_{1:i-1}, a_i)$  and the joint policy  $\pi := (\pi_1, \pi_2, \dots, \pi_n)$ .

802 By induction, we prove that for the agent  $i$ , if its prior agents have selected the optimal actions  $\mathbf{a}_{1:i-1}^*$ ,  
 803 it can select its optimal action  $a_i^*$  by maximizing its advantage function  $A_i(s, \mathbf{a}_{1:i-1}^*, a_i)$ .

804 For agent  $n$ :

$$\begin{aligned} a_n^* &= \arg \max_{a_n} Q(s, \mathbf{a}_{1:n-1}^*, a_n) \\ &= \arg \max_{a_n} [Q(s, \mathbf{a}_{1:n-1}^*, a_n) - \mathbb{E}_{a_n \sim \pi_n} [Q(s, \mathbf{a}_{1:n-1}^*, a_n)]] \\ &= \arg \max_{a_n} A_n(s, \mathbf{a}_{1:n-1}^*, a_n) \end{aligned}$$

805 where the second equation holds because  $\mathbb{E}_{a_n \sim \pi_n} [Q(s, \mathbf{a}_{1:n-1}^*, a_n)]$  is a constant, and the third  
 806 equation refers to the definition of  $A_n(s, \mathbf{a}_{1:n-1}^*, a_n)$  in Definition 2.

807 We now assume that for the agent  $i + 1$  to  $n$ , this property holds, and for the agent  $i$ :

$$\begin{aligned} a_i^* &= \arg \max_{a_i} Q(s, \mathbf{a}_{1:i-1}^*, a_i, \arg \max_{\mathbf{a}_{i+1:n}} Q(s, \mathbf{a}_{1:i-1}^*, a_i, \mathbf{a}_{i+1:n})) \\ &= \arg \max_{a_i} \mathbb{E}_{\mathbf{a}_{i+1:n} \sim \pi_{i+1:n}} [Q(s, \mathbf{a}_{1:i-1}^*, a_i, \mathbf{a}_{i+1:n})] \\ &= \arg \max_{a_i} \{ \mathbb{E}_{\mathbf{a}_{i+1:n} \sim \pi_{i+1:n}} [Q(s, \mathbf{a}_{1:i-1}^*, a_i, \mathbf{a}_{i+1:n})] - \mathbb{E}_{\mathbf{a}_{i:n} \sim \pi_{i:n}} [Q(s, \mathbf{a}_{1:i-1}^*, \mathbf{a}_{i:n})] \} \\ &= \arg \max_{a_i} A_i(s, \mathbf{a}_{1:i-1}^*, a_i) \end{aligned}$$

where the second equation holds because the inductive hypothesis implies that if the  $i$ -th agent selects the optimal  $a_i^*$ ,  $\pi_{i+1:n}(s, \mathbf{a}_{1:i}^*) = \mathbf{a}_{i+1:n}^*$ , and thus we have

$$\begin{aligned} \forall a_i, \mathbf{a}_{i+1:n}, \mathbb{E}_{\mathbf{a}_{i+1:n} \sim \pi_{i+1:n}} [Q(s, \mathbf{a}_{1:i-1}^*, a_i^*, \mathbf{a}_{i+1:n})] &= Q(s, \mathbf{a}_{1:i-1}^*, a_i^*, \mathbf{a}_{i+1:n}^*) \\ &\geq Q(s, \mathbf{a}_{1:i-1}^*, a_i, \mathbf{a}_{i+1:n}) \end{aligned}$$

which means

$$\forall a_i, \mathbb{E}_{\mathbf{a}_{i+1:n} \sim \pi_{i+1:n}} [Q(s, \mathbf{a}_{1:i-1}^*, a_i^*, \mathbf{a}_{i+1:n})] \geq \mathbb{E}_{\mathbf{a}_{i+1:n} \sim \pi_{i+1:n}} [Q(s, \mathbf{a}_{1:i-1}^*, a_i, \mathbf{a}_{i+1:n})]$$

Therefore,  $a_i^* = \arg \max_{a_i} \mathbb{E}_{\mathbf{a}_{i+1:n} \sim \pi_{i+1:n}} [Q(s, \mathbf{a}_{1:i-1}^*, a_i, \mathbf{a}_{i+1:n})]$ , and the second equation holds. The third equation holds because  $\mathbb{E}_{\mathbf{a}_{i:n} \sim \pi_{i:n}} [Q(s, \mathbf{a}_{1:i-1}^*, \mathbf{a}_{i:n})]$  is a constant, and the forth equation refers to the definition of  $A_i(s, \mathbf{a}_{1:i-1}^*, a_i)$  in Definition 2.

Therefore, by induction, we conclude:

$$\forall i, a_i^* = \arg \max_{a_i} A_i(s, \mathbf{a}_{1:i-1}^*, a_i)$$

Notably, when  $i = 1$ ,  $a_1^* = \arg \max_{a_1} A_1(s, a_1)$ , and for subsequent agents, the agent 2 will select  $a_2^* = \arg \max_{a_2} A_2(s, a_1^*, a_2)$ , and so on, with the agent  $n$  selecting  $a_n^* = \arg \max_{a_n} A_n(s, \mathbf{a}_{1:n-1}^*, a_n)$ .  $\square$

Thus, we have completed the proof of Theorem 1

### A.3 Proof of Lemma 1

**Lemma 1** (Multi-agent Advantage Decomposition [16]). *In any cooperative Markov game, given a joint policy  $\pi$ , the global advantage function  $A^\pi(s, \mathbf{a})$ , and  $n$  agents in total, for any state  $s$ , the following equations hold:*

$$A^\pi(s, \mathbf{a}) = \sum_{i=1}^n A_i^\pi(s, \mathbf{a}_{1:i-1}, a_i)$$

*Proof.* Similar to the proof in [16], according to the Definition 2, we have

$$\begin{aligned} A^\pi(s, \mathbf{a}) &= Q^\pi(s, \mathbf{a}) - V^\pi(s) \\ &= \sum_{i=1}^n Q_{1:i}^\pi(s, \mathbf{a}_{1:i}) - Q_{1:i-1}^\pi(s, \mathbf{a}_{1:i-1}) \\ &= \sum_{i=1}^n A_i^\pi(s, \mathbf{a}_{1:i-1}, a_i) \end{aligned}$$

which finishes the proof.  $\square$

## B Detailed settings and information of the environments

In this section, we introduce our detailed experimental settings, including network architectures, experimental configurations, and hyperparameter settings.

### B.1 Seq-Sigmoid

In this subsection, we introduce the original Sigmoid benchmark, the Seq-Sigmoid benchmark, and its variants in detail.

**Sigmoid [2].** The pseudo-code of the Sigmoid benchmark is provided in Algorithm 1. The Sigmoid task is basically an approximation task with parameters changing along the time, with the approximation target as  $\text{sig}(t, s_{i,h}, p_{i,h})$ , the state provided in line 5, and the reward at  $t$  calculated in line 7.

---

**Algorithm 1** Benchmark Outline: Sigmoid

---

- 1: **Benchmark Parameters:** number of parameters  $H$ , number of parameter values  $C_h$ , episode length  $T$  (i.e., the environment is done when executing  $T$  steps);
  - 2:  $s_i \sim \mathcal{U}(-100, 100, H)$ ;
  - 3:  $p_i \sim \mathcal{N}(T/2, T/4, H)$ ;
  - 4: **for**  $t \in \{0, 1, \dots, T\}$  **do**
  - 5:   The state at step  $t$ :  $state_t \in s_i \cup p_i \cup \{t\}$ ;
  - 6:   Selecting actions:  $a_{h,t} \in \left\{ \frac{0}{C_h}, \frac{1}{C_h}, \dots, \frac{C_h-1}{C_h} \right\}$  for all  $0 \leq h < H$  and  $0 \leq t \leq T$ ;
  - 7:    $r_t^i \leftarrow \prod_{h=0}^{H-1} (1 - |\text{sig}(t, s_{i,h}, p_{i,h}) - a_{h,t}|)$ ;
  - 8: **end for**
- 

830 **Seq-Sigmoid.** The pseudo-code of the Seq-Sigmoid benchmark is provided in Algorithm 2. The  
831 main modification (i.e., in line 7 and 8) is that, at timestep  $t$ , the former parameter's value will  
832 determine a scaling factor  $\alpha_{h-1,t}$ , which controls the latter parameter's slope  $s_{i,h}$ . In this way, the  
833 former parameter configuration will have strong influence on the configuration of the latter parameter.  
834 Moreover, to avoid having this dependency affect the configuration of the former parameter itself, we  
835 use the formula  $\min(|\text{sig}(t, \alpha_{h,t} s_{i,h}, p_{i,h}) - a_{h,t}|, |1 - \text{sig}(t, \alpha_{h,t} s_{i,h}, p_{i,h}) - a_{h,t}|)$  to measure the  
836 approximation, which returns the same value when two  $a_{h,t}$  are symmetrical about 0.5.

---

**Algorithm 2** Benchmark Outline: Seq-Sigmoid

---

- 1: **Benchmark Parameters:** number of parameters  $H$ , number of parameter values  $C_h$ , episode length  $T$  (i.e., the environment is done when executing  $T$  steps);
  - 2:  $s_i \sim \mathcal{U}(-100, 100, H)$ ;
  - 3:  $p_i \sim \mathcal{N}(T/2, T/4, H)$ ;
  - 4: **for**  $t \in \{0, 1, \dots, T\}$  **do**
  - 5:   The state at step  $t$ :  $state_t \in s_i \cup p_i \cup \{t\}$ ;
  - 6:   Selecting actions:  $a_{h,t} \in \left\{ \frac{0}{C_h}, \frac{1}{C_h}, \dots, \frac{C_h-1}{C_h} \right\}$  for all  $0 \leq h < H$  and  $0 \leq t \leq T$ ;
  - 7:    $\alpha_{0,t} = 1, \alpha_{h,t} = \begin{cases} 10 & \text{if } a_{h-1,t} \geq 0.5 \\ 0.1 & \text{if } a_{h-1,t} < 0.5 \end{cases}, 0 < h < H$ ;
  - 8:    $r_t^i \leftarrow \prod_{h=0}^{H-1} (1 - \min(|\text{sig}(t, \alpha_{h,t} s_{i,h}, p_{i,h}) - a_{h,t}|, |1 - \text{sig}(t, \alpha_{h,t} s_{i,h}, p_{i,h}) - a_{h,t}|))$ ;
  - 9: **end for**
- 

837 **Seq-Sigmoid-Mask.** The pseudo-code of the Seq-Sigmoid-Mask benchmark is provided in Algo-  
838 rithm 3. The main modification compared to the Seq-Sigmoid benchmark is that we mask  $s_{i,h}$  and  
839  $p_{i,h}$  (i.e., the  $state_t \in \{t\}$  in line 4) and set  $s_{i,h} = 1$  (i.e., in line 7) to avoid too much randomness.

---

**Algorithm 3** Benchmark Outline: Seq-Sigmoid-Mask

---

- 1: **Benchmark Parameters:** number of parameters  $H$ , number of parameter values  $C_h$ , episode length  $T$  (i.e., the environment is done when executing  $T$  steps);
  - 2:  $p_i \sim \mathcal{N}(T/2, T/4, H)$ ;
  - 3: **for**  $t \in \{0, 1, \dots, T\}$  **do**
  - 4:   The state at step  $t$ :  $state_t \in \{t\}$ ;
  - 5:   Selecting actions:  $a_{h,t} \in \left\{ \frac{0}{C_h}, \frac{1}{C_h}, \dots, \frac{C_h-1}{C_h} \right\}$  for all  $0 \leq h < H$  and  $0 \leq t \leq T$ ;
  - 6:    $\alpha_{0,t} = 1, \alpha_{h,t} = \begin{cases} 10 & \text{if } a_{h-1,t} \geq 0.5 \\ 0.1 & \text{if } a_{h-1,t} < 0.5 \end{cases}, 0 < h < H$ ;
  - 7:    $r_t^i \leftarrow \prod_{h=0}^{H-1} (1 - \min(|\text{sig}(t, \alpha_{h,t} * 1, p_{i,h}) - a_{h,t}|, |1 - \text{sig}(t, \alpha_{h,t} * 1, p_{i,h}) - a_{h,t}|))$ ;
  - 8: **end for**
- 

840 **Seq-Sigmoid-Robust ( $n$ ).** The pseudo-code of the Seq-Sigmoid-Robust ( $n$ ) benchmark is provided  
841 in Algorithm 4. The benchmark gives random configurations for the  $n$  parameters no matter how  
842 the corresponding agents tune them in Seq-Sigmoid (i.e., in line 7), with other parameters config-  
843 ured properly, to simulate the ineffective learning of some agents in complex dynamic algorithm

844 configuration scenarios. In particular, we give random configurations for the  $\lfloor n/2 \rfloor$ -th parameter in  
 845 Seq-Sigmoid-Robust (1) and the  $\lfloor n/2 \rfloor$ -th and the  $\lfloor n/2 \rfloor + 1$ -th parameters in in Seq-Sigmoid-Robust  
 846 (2).

---

**Algorithm 4** Benchmark Outline: Seq-Sigmoid-Robust (n)

---

1: **Benchmark Parameters:** number of parameters  $H$ , number of parameter values  $C_h$ , episode  
 length  $T$  (i.e., the environment is done when executing  $T$  steps),  $n$  parameter indexes  
 $h_1, h_2, \dots, h_n$ ;  
 2:  $s_i \sim \mathcal{U}(-100, 100, H)$ ;  
 3:  $p_i \sim \mathcal{N}(T/2, T/4, H)$ ;  
 4: **for**  $t \in \{0, 1, \dots, T\}$  **do**  
 5:   The state at step  $t$ :  $state_t \in s_i \cup p_i \cup \{t\}$ ;  
 6:   Selecting actions:  $a_{h,t} \in \left\{ \frac{0}{C_h}, \frac{1}{C_h}, \dots, \frac{C_h-1}{C_h} \right\}$  for all  $0 \leq h < H$  and  $0 \leq t \leq T$ ;  
 7:   Reselect  $a_{i,t}, i \in \{h_1, h_2, \dots, h_n\}$  at random;  
 8:    $\alpha_{0,t} = 1, \alpha_{h,t} = \begin{cases} 10 & \text{if } a_{h-1,t} \geq 0.5 \\ 0.1 & \text{if } a_{h-1,t} < 0.5 \end{cases}, 0 < h < H$ ;  
 9:    $r_t^i \leftarrow \prod_{h=0}^{H-1} (1 - \min(|\text{sig}(t, \alpha_{h,t} s_{i,h}, p_{i,h}) - a_{h,t}|, |1 - \text{sig}(t, \alpha_{h,t} s_{i,h}, p_{i,h}) - a_{h,t}|))$ ;  
 10: **end for**

---

## 847 B.2 MOEA/D

848 MOEA/D is the proposed DAC benchmark [41] based on a real-scenario multi-objective evolutionary  
 849 algorithm MOEA/D [44] with heterogeneous parameters, which is a strong stochastic benchmark due  
 850 to the randomness of the evolution optimization process.

851 **MOEA/D algorithm** The multi-objective evolutionary algorithm MOEA/D is originally designed  
 852 to solve the multi-objective optimization problems (MOPs), which can be defined as

$$\min \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_m(\mathbf{x})) \quad \text{s.t.} \quad \mathbf{x} \in \Omega, \quad (5)$$

853 where  $\mathbf{x} = (x_1, \dots, x_D)$  is a solution,  $\mathbf{F} : \Omega \rightarrow \mathbb{R}^m$  constitutes  $m$  objective functions,  $\Omega =$   
 854  $[x_i^L, x_i^U]^D \subseteq \mathbb{R}^D$  is the solution space, and  $\mathbb{R}^m$  is the objective space. MOP aims to find a set of  
 855 solutions that represent the best possible trade-offs between competing objectives. These solutions  
 856 are collectively called the Pareto front (PF) defined as follows:

857 **Definition 3.** A solution  $\mathbf{x}^*$  is Pareto-optimal with respect to Eq. (5), if  $\nexists \mathbf{x} \in \Omega$  such that  $\forall i :$   
 858  $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*)$  and  $\exists i : f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$ . The set of all Pareto-optimal solutions is called Pareto-  
 859 optimal set (PS). The set of the corresponding objective vectors of PS, i.e.,  $\{\mathbf{F}(\mathbf{x}) \mid \mathbf{x} \in \text{PS}\}$ , is  
 860 called Pareto Front (PF).

861 MOEA/D consists of two main processes, *decomposition* and *collaboration* [44, 39, 19]. In decompo-  
 862 sition, MOEA/D solves a number of sub-problems through a number of weights and an aggregation  
 863 function to approximate the PF. Several aggregation functions have been proposed for MOEA/D.  
 864 Here, we introduce the common Tchebycheff approach (TCH) used in our paper as follows:

865 Given a weight vector  $\mathbf{w} = (w_1, \dots, w_m)$  where  $w_i \geq 0, \forall i \in \{1, \dots, m\}$  and  $\sum_{i=1}^m w_i = 1$ , the  
 866 sub-problem by TCH is formulated as

$$\min_{\mathbf{x} \in \Omega} g(\mathbf{x} \mid \mathbf{w}, \mathbf{z}^*) = \max_{1 \leq i \leq m} \{w_i \cdot |f_i(\mathbf{x}) - z_i^*|\}, \quad (6)$$

867 where  $\mathbf{z}^* = (z_1^*, \dots, z_m^*)$  is the ideal point consisting of the best objective values obtained so far.

868 The fundamental intuition behind collaboration is that adjacent sub-problems tend to share similar  
 869 properties. For instance, they may have similar objective functions and/or optimal solutions [19].  
 870 In particular, the neighborhood of a sub-problem is controlled by the Euclidean distance of its  
 871 corresponding weight vector with respect to the others, as well as the hyperparameter *neighborhood*  
 872 *size*: two sub-problems are neighbors of each other if the distance between them is smaller than the  
 873 neighborhood size. In the selection process for a sub-problem, parent solutions are randomly selected  
 874 from the corresponding neighborhood of each sub-problem. The sub-problem solutions within the  
 875 same neighborhood are then replaced with the newly generated offspring solution.

876 The pseudo-code of MOEA/D is shown in Algorithm 5.

---

**Algorithm 5** MOEA/D

---

**Input:** Population size  $N$ , number  $T$  of iterations

**Output:** A set of Pareto-optimal solutions

```
1: Initialize a population  $\{\mathbf{x}^{(i)}\}_{i=1}^N$  of solutions, and a corresponding set  $W = \{\mathbf{w}^{(i)}\}_{i=1}^N$  of weight vectors
2:  $t \leftarrow 0$ 
3: while  $t < T$  do
4:   for  $i = 1 : N$  do
5:     Randomly select parent solutions from the neighborhood of  $\mathbf{w}^{(i)}$ , denoted as  $\Theta^{\mathbf{w}^{(i)}}$ 
6:     Use crossover and mutation operators to generate an offspring solution  $\mathbf{x}'^{(i)}$ 
7:     Evaluate the offspring solution to obtain  $F(\mathbf{x}'^{(i)})$ 
8:     Update the ideal point  $\mathbf{z}^*$ :
9:     for  $j \in \{1, 2, \dots, m\}$  do
10:      if  $f_j(\mathbf{x}'^{(i)}) < \mathbf{z}_j^*$  then
11:         $\mathbf{z}_j^* \leftarrow f_j(\mathbf{x}'^{(i)})$ 
12:      end if
13:    end for
14:    Update the corresponding solution of each sub-problem within  $\Theta^{\mathbf{w}^{(i)}}$  by  $\mathbf{x}'^{(i)}$ :
15:    for  $\mathbf{w}^{(j)} \in \Theta^{\mathbf{w}^{(i)}}$  do
16:      if  $g(\mathbf{x}'^{(i)} \mid \mathbf{w}^{(j)}, \mathbf{z}^*) < g(\mathbf{x}^{(j)} \mid \mathbf{w}^{(j)}, \mathbf{z}^*)$  then
17:         $\mathbf{x}^{(j)} \leftarrow \mathbf{x}'^{(i)}$ 
18:      end if
19:    end for
20:  end for
21:   $t \leftarrow t + 1$ 
22: end while
```

---

877 **Action** The action space of MOEA/D is of four dimensions, corresponding to the four heterogeneous  
878 MOEA/D parameters:

879 1. **Weights.** In MOEA/D, weights are used to transform an MOP into multiple single-objective  
880 sub-problems. Inspired by MOEA/D-AWA [27], we set the action space for weights by adjusting (T)  
881 or not adjusting (N) the weights. If the action is T, the weights will be updated, otherwise the weights  
882 will remain the same. The weights adaptation mechanism is as follows.

883 The sparsity level of each solution is calculated  $\mathbf{x}^{(i)}$  based on vicinity distance [17]:

$$SL(\mathbf{x}^{(i)}, \{\mathbf{x}^{(p)}\}_{p=1}^N) = \prod_{j=1}^m l(\mathbf{x}^{(i)}, j), \quad (7)$$

884 where  $l(\mathbf{x}^{(i)}, j)$  is the Euclidean distance between  $\mathbf{x}^{(i)}$  and its  $j$ -th nearest neighbor in the population  
885  $\{\mathbf{x}^{(p)}\}_{p=1}^N$ . In the calculation, the  $m$  closest neighbors in the population are used, where  $m$  denotes  
886 the number of objectives. Sub-problems corresponding to the solutions with sparsity levels in the  
887 bottom 5%, namely the overcrowded solutions, are subsequently removed.

888 To ensure that the total number of the sub-problems is still  $N$ ,  $0.05N$  new sub-problems and their  
889 corresponding solutions should be added, which come from an elite population that keeps all historical  
890 non-dominated solutions, with a maximum capacity of  $1.5N$ . When the size of the elite population  
891 surpasses this capacity, the solutions with the lowest sparsity level are eliminated. For every solution  
892  $\mathbf{x}'$  in the elite population, its sparsity level is calculated with respect to the current population (i.e.,  
893  $SL(\mathbf{x}', \text{Pop})$ , where Pop represents the set of the remaining  $0.95N$  solutions). Subsequently, the  
894 solution with the highest sparsity level with respect to the current population is selected from the elite  
895 population and added to the current population. This selection and addition procedure is carried out  
896 for  $0.05N$  times. For each newly added solution, the corresponding sub-problem (i.e., weight vector)  
897 is generated in a specific way as the Algorithm 3 in [27].

898 2. **Neighborhood size.** The neighborhood size is used to control the maximum distance between  
899 solutions and their neighbors. A small size aims to exploit the local area, while a large size aims to



Table 2: State at step  $t$  in MOEA/D.

Index	Parts of state	Feature	Notes
0	1	$1/m$	$m$ : Number of objectives
1	1	$1/D$	$D$ : Number of variables
2	2	$t/T$	Computational budget that has been used
3	2	$N_{\text{stag}}/T$	Stagnant count ratio
4	3	$HV_t$	Hypervolume value
5	3	$NDRatio_t$	Ratio of non-dominated solutions
6	3	$Dist_t$	Average distance
7	3	$HV_t - HV_{t-1}$	Change of HV between steps $t$ and $t-1$
8	3	$NDRatio_t - NDRatio_{t-1}$	Change of NDRatio between steps $t$ and $t-1$
9	3	$Dist_t - Dist_{t-1}$	Change of Dist between steps $t$ and $t-1$
10	3	$\text{Mean}(\text{List}(HV, t, 5))$	Mean of HV in the last 5 steps
11	3	$\text{Mean}(\text{List}(NDRatio, t, 5))$	Mean of NDRatio in the last 5 steps
12	3	$\text{Mean}(\text{List}(Dist, t, 5))$	Mean of Dist in the last 5 steps
13	3	$\text{Std}(\text{List}(HV, t, 5))$	Standard deviation of HV in the last 5 steps
14	3	$\text{Std}(\text{List}(NDRatio, t, 5))$	Standard deviation of NDRatio in the last 5 steps
15	3	$\text{Std}(\text{List}(Dist, t, 5))$	Standard deviation of Dist in the last 5 steps
16	3	$\text{Mean}(\text{List}(HV, t, t))$	Mean of HV in all the steps so far
17	3	$\text{Mean}(\text{List}(NDRatio, t, t))$	Mean of NDRatio in all the steps so far
18	3	$\text{Mean}(\text{List}(Dist, t, t))$	Mean of Dist in all the steps so far
19	3	$\text{Std}(\text{List}(HV, t, t))$	Standard deviation of HV in all the steps so far
20	3	$\text{Std}(\text{List}(NDRatio, t, t))$	Standard deviation of NDRatio in all the steps so far
21	3	$\text{Std}(\text{List}(Dist, t, t))$	Standard deviation of Dist in all the steps so far

900 explore a wide objective space [40]. The action space is of four dimensions, that is, 15, 20, 25 and  
 901 30, where 20 is the default value.

902 3. Types of reproduction operators. We consider four types of DE operators with different search  
 903 abilities introduced in [20]. For reproducing an offspring solution for the  $i$ -th sub-problem, let  
 904  $\mathbf{x}^{(i)}$  and  $\mathbf{x}'^{(i)}$  denote its current solution and the generated offspring solution, respectively and the  
 905 equations for four types of DE operators are shown as follows:

- 906 • OP1:  $\mathbf{x}'^{(i)} = \mathbf{x}^{(i)} + F \times (\mathbf{x}^{(r_1)} - \mathbf{x}^{(r_2)})$ ,
- 907 • OP2:  $\mathbf{x}'^{(i)} = \mathbf{x}^{(i)} + F \times (\mathbf{x}^{(r_1)} - \mathbf{x}^{(r_2)}) + F \times (\mathbf{x}^{(r_3)} - \mathbf{x}^{(r_4)})$ ,
- 908 • OP3:  $\mathbf{x}'^{(i)} = \mathbf{x}^{(i)} + K \times (\mathbf{x}^{(i)} - \mathbf{x}^{(r_1)}) + F \times (\mathbf{x}^{(r_2)} - \mathbf{x}^{(r_3)}) + F \times (\mathbf{x}^{(r_4)} - \mathbf{x}^{(r_5)})$ ,
- 909 • OP4:  $\mathbf{x}'^{(i)} = \mathbf{x}^{(i)} + K \times (\mathbf{x}^{(i)} - \mathbf{x}^{(r_1)}) + F \times (\mathbf{x}^{(r_2)} - \mathbf{x}^{(r_3)})$ .

910 Here,  $\mathbf{x}^{(r_1)}, \mathbf{x}^{(r_2)}, \mathbf{x}^{(r_3)}, \mathbf{x}^{(r_4)}$ , and  $\mathbf{x}^{(r_5)}$  are different parent solutions randomly selected from the  
 911 neighborhood of  $\mathbf{x}^{(i)}$ . The scaling factor  $F > 0$  controls the impact of the vector differences on the  
 912 mutant vector, and  $K \in [0, 1]$  functions similarly to  $F$ .

913 4. Parameters of reproduction operators. The parameters (particularly the scaling factor) of the  
 914 reproduction operators in MOEA/D have a significant effect on the algorithm's performance [36].  
 915 We set the scaling factor  $K$  to a fixed value of 0.5 as recommended [20], and dynamically adjust  
 916 the scaling factor  $F$  with four discrete dimensions, i.e., 0.4, 0.5, 0.6 and 0.7, with 0.5 serving as the  
 917 default value.

918 **State** The state of MOEA/D includes three parts: 1. The feature of the problem instance (e.g., the  
 919 numbers of objectives and variables); 2. The feature of the optimization process (e.g., the budget  
 920 has been used); 3. Information about the evolution situation of the population (e.g., the hypervolume  
 921 value and the average distance of the current population). The detailed state feature is demonstrated  
 922 in Table 2.

923 **Transition** One step of transition takes place at one generation change in the evolutionary process  
 924 of MOEA/D.

**Reward** A positive reward of the MOEA/D environment is assigned to the agent team when a better solution is found compared to the current best solution. Since optimization becomes harder when the current best solution is more optimal with time, we should assign more reward to better found solutions in the latter stage. We use the triangle-based reward function proposed in [41] defined as follows, which has been proved effective in the MOEA/D environment.

At step  $t$ ,

$$r_t = \begin{cases} (1/2) \cdot (p_{t+1}^2 - p_t^2) & \text{if } f(s_{t+1}) < f_t^* \\ 0 & \text{otherwise} \end{cases}, \quad (8)$$

where

$$p_{t+1} = \begin{cases} \frac{f(s_0) - f(s_{t+1})}{f(s_0)} & \text{if } f(s_{t+1}) < f_t^* \\ p_t & \text{otherwise} \end{cases}, \quad (9)$$

and  $f_t^*$  is the minimum metric value found until step  $t$ .

**Problem instance** The problem instances of MOEA/D include the well-known multi-objective optimization problems (MOP) benchmarks DTLZ [7] and WFG [11] with variable number of objective and problem dimensions, which cover different difficulty levels of MOPs.

### B.3 Hyperparameters of the algorithms compared in the experiments

For all the compared MARL algorithms, we use their default suggested hyperparameter settings in EPyMARL<sup>1</sup> (i.e., VDN [37], QMIX [29], MAPPO [43]) or its official implementation (i.e., HASAC [21]). For a fair comparison, we use the same hyperparameters for algorithms of the same type (i.e., we use the same hyperparameters for the value-based methods: SADN, ACE [18], SAQL [3], VDN [37] and QMIX [29], as well as same hyperparameters for the policy gradient-based PPO methods: MAPPO [43] and HAPPO [16]). The detailed hyperparameters in the experiments are given in Table 3. For a fair comparison, we use one single 64-dimensional hidden layer for all the value networks and the policy networks.

Table 3: The hyperparameters of the compared algorithms in MOEA/D, and "-" means the certain algorithm does not have that hyperparameter.

Hyperparameter	SADN	ACE	SAQL	VDN	QMIX	MAPPO	HAPPO	HASAC
Hidden layer size	64	64	64	64	64	64	64	64
Learning rate	1e-4	1e-4	1e-4	1e-4	1e-4	3e-4	3e-4	5e-4
Batch size	32	32	32	32	32	10	10	10
Discount	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Target update interval	200	200	200	200	200	200	200	50
Number of steps to look ahead	1	-	-	-	-	5	5	20
Entropy coef	-	-	-	-	-	0.01	0.01	-
Grad norm clip	10	10	10	10	10	10	10	-

More detailed information can be found in our code available at supplemental files.

## C Additional results

### C.1 Results on the original Sigmoid benchmark

We compare our method with the famous decomposition value-based method VDN [37], as well as the widely used value-based single-agent RL algorithm DQN [26] on the original Sigmoid benchmark. As shown in Figure 4, our method can achieve competitive and even better results compared to

<sup>1</sup><https://github.com/uoee-agents/epymarl>

the effective decomposition-based VDN when there are no strong inter-dependencies among the parameters. Moreover, the single-agent RL algorithm DQN fails to learn effectively because of the combinatorial explosion of the action space on the 5D Sigmoid, and even fails to train on the 10D Sigmoid, which is also observed in [41], emphasizing the importance of the multi-agent modeling.

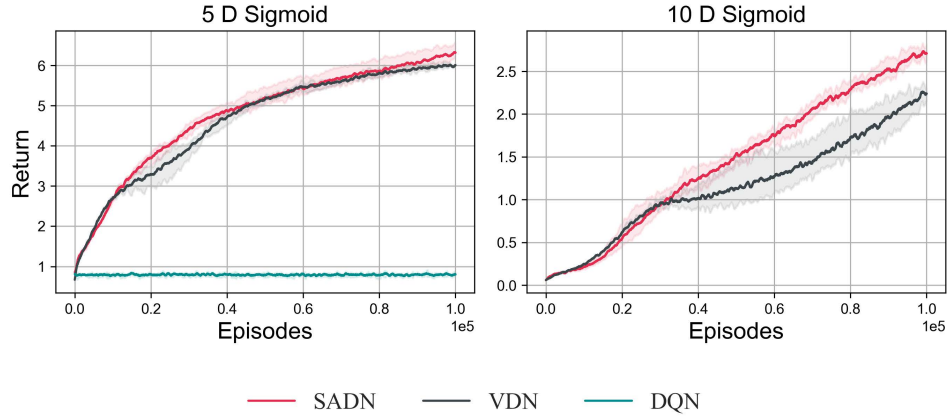


Figure 4: Training curves of return value obtained by the compared methods on the original Sigmoid benchmark, where the results are averaged over 3 runs

## C.2 Comparison of the correct order and the reserve order on Seq-Sigmoid and Seq-Sigmoid-Mask

The comparison of the sequential methods based on the correct order and the reverse order is shown in Figure 5. Basically, the reverse order produces lower performance and larger variance than the correct order and produces due to the wrong capture of the inherent inter-dependencies between the parameters, which misleads the learning of the whole team. There is an exception, HAPPO, which only considers the update order of the agents rather than the order of taking actions. HAPPO avoids suffering from the misleading of the reverse order, but fails to exploit the sequential information of the correct order as well. The comparison of using the correct and the reverse order provides evidence for the effectiveness of modeling the inherent inter-dependencies correctly, which also leads to more efficient training.

## C.3 Comparison of the correct order and the reserve order on MOEA/D

The results in Table 4 demonstrate the effectiveness of correctly considering the parameter order on the real-scenario multi-objective evolutionary algorithm. Generally, the correct order produces better results than the reverse order within the same sequential methods, which evidently shows that the sequential information of the correct order can boost better performance, and the improvement of the performance comes from the the correct order, rather than simply making the actions selected by the previous agents available to the subsequent agents.

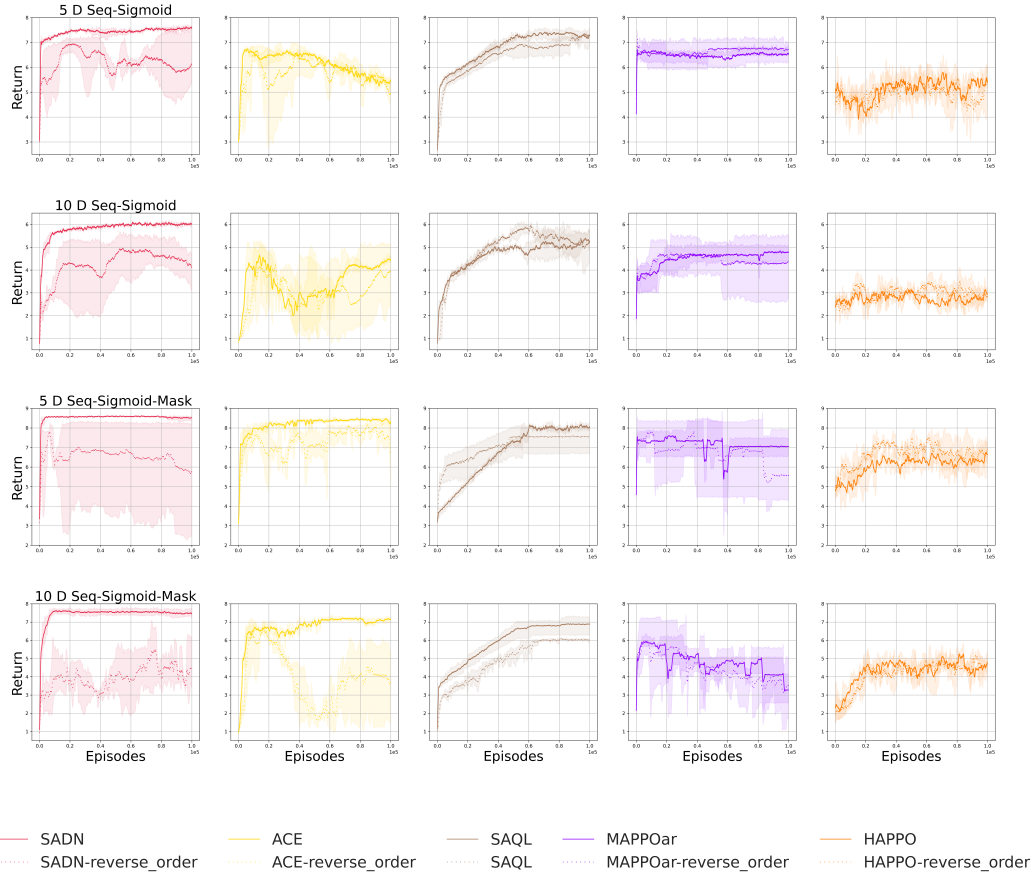


Figure 5: Training curves of return value obtained by correct order and reverse order on four Seq-Sigmoid variant tasks, where the results are averaged over 3 runs.

Table 4: IGD values of order and reverse order and compared methods on different problems. Each result consists of the mean of 10 runs. The best mean value on each problem is highlighted in bold. The symbols '+', '-' and '≈' indicate that the result is significantly superior to, inferior to, and almost equivalent to our method SADN, respectively, according to the Wilcoxon rank-sum test with significance level 0.05. The top three problems are the problems for training, and the bottom five problems are for testing.

Problem	Dim	MOEA/D	SADN	SADN-r	ACE	ACE-r	SAQL	SAQL-r	MAPPOar	MAPPOar-r
DTLZ2	6	4.593e-02	3.809e-02	3.819e-02	3.851e-02	4.615e-02	3.950e-02	4.061e-02	3.804e-02	<b>3.786e-02</b>
	9	4.598e-02	<b>3.875e-02</b>	4.158e-02	4.057e-02	4.619e-02	4.337e-02	4.354e-02	4.308e-02	4.167e-02
	12	4.599e-02	<b>3.874e-02</b>	4.176e-02	4.063e-02	4.634e-02	4.698e-02	4.734e-02	3.886e-02	4.284e-02
WFG4	6	5.729e-02	<b>4.537e-02</b>	4.847e-02	4.626e-02	6.282e-02	4.817e-02	4.747e-02	4.677e-02	4.576e-02
	9	5.736e-02	<b>5.190e-02</b>	6.026e-02	5.853e-02	6.725e-02	5.365e-02	5.375e-02	5.471e-02	5.356e-02
	12	5.751e-02	<b>5.213e-02</b>	6.021e-02	5.751e-02	6.983e-02	6.241e-02	5.958e-02	5.351e-02	5.797e-02
WFG6	6	5.855e-02	<b>3.908e-02</b>	4.027e-02	3.962e-02	5.811e-02	3.964e-02	3.995e-02	3.944e-02	3.921e-02
	9	6.641e-02	4.884e-02	7.757e-02	5.111e-02	5.992e-02	<b>4.384e-02</b>	4.432e-02	6.093e-02	5.949e-02
	12	6.864e-02	<b>4.837e-02</b>	7.674e-02	5.490e-02	7.908e-02	5.676e-02	5.755e-02	5.786e-02	6.710e-02
Train:+/-/≈		0/9/0		0/8/1	0/8/1	0/9/0	1/8/0	1/8/0	0/7/2	1/8/0
DTLZ4	6	5.455e-02	<b>3.895e-02</b>	3.980e-02	4.221e-02	5.598e-02	5.019e-02	4.856e-02	3.932e-02	4.008e-02
	9	6.229e-02	7.319e-02	5.366e-02	<b>4.492e-02</b>	5.765e-02	6.895e-02	5.833e-02	6.252e-02	5.847e-02
	12	6.790e-02	7.852e-02	5.196e-02	<b>4.650e-02</b>	6.667e-02	8.589e-02	8.688e-02	5.336e-02	6.211e-02
WFG5	6	6.303e-02	4.749e-02	4.748e-02	4.752e-02	6.145e-02	5.390e-02	5.539e-02	<b>4.713e-02</b>	4.719e-02
	9	6.351e-02	<b>4.773e-02</b>	4.778e-02	4.792e-02	6.156e-02	4.788e-02	4.787e-02	5.070e-02	4.798e-02
	12	6.381e-02	4.793e-02	4.759e-02	4.791e-02	6.169e-02	4.954e-02	4.983e-02	<b>4.751e-02</b>	4.980e-02
WFG7	6	5.803e-02	<b>3.893e-02</b>	3.994e-02	3.942e-02	5.819e-02	3.958e-02	3.980e-02	3.922e-02	3.907e-02
	9	5.812e-02	<b>4.053e-02</b>	4.954e-02	4.470e-02	5.878e-02	4.257e-02	4.235e-02	4.285e-02	4.154e-02
	12	5.814e-02	<b>4.060e-02</b>	4.950e-02	4.447e-02	5.936e-02	4.975e-02	4.986e-02	4.115e-02	4.437e-02
WFG8	6	<b>7.875e-02</b>	1.018e-01	1.077e-01	1.029e-01	1.177e-01	1.054e-01	1.031e-01	1.005e-01	1.026e-01
	9	9.680e-02	<b>7.853e-02</b>	9.191e-02	9.787e-02	1.053e-01	9.716e-02	9.299e-02	7.947e-02	7.911e-02
	12	8.710e-02	7.891e-02	9.195e-02	8.387e-02	9.330e-02	9.023e-02	8.947e-02	<b>7.873e-02</b>	8.164e-02
WFG9	6	5.600e-02	<b>3.987e-02</b>	4.053e-02	4.012e-02	5.495e-02	4.010e-02	4.219e-02	3.994e-02	3.975e-02
	9	5.748e-02	4.341e-02	5.831e-02	4.552e-02	5.562e-02	4.281e-02	<b>4.168e-02</b>	4.677e-02	4.446e-02
	12	5.827e-02	<b>4.249e-02</b>	5.839e-02	4.477e-02	7.265e-02	7.291e-02	6.463e-02	5.299e-02	4.986e-02
Test:+/-/≈		1/12/2		3/10/2	2/9/4	1/13/1	0/11/4	2/11/2	3/8/4	4/9/2
average rank	6	7.5	<b>2.</b>	5.625	4.5	8.625	5.75	6.25	2.5	2.25
	9	7.375	<b>2.5</b>	5.5	4.75	7.5	4.375	3.625	5.625	3.75
	12	6.25	<b>2.25</b>	5.25	2.75	8.	6.75	6.875	2.5	4.375

## 973 **D Compute resource used**

974 The experiments are conducted on 4 GPUs (each with 82.58 TFLOPS and 24 GB) and an AMD  
975 EPYC 7513 CPU (32 cores). For our method SADN, one training run of  $1e6$  steps on the 10 D  
976 Seq-Sigmoid benchmark takes approximately 4 hours, while one training run of  $4e5$  steps on the  
977 MOEA/D benchmark takes about 14 hours. Among all algorithms, our proposed SADN ranks second,  
978 slightly slower than the fastest method, VDN.