Conditionally Optimistic Exploration for Cooperative Deep Multi-Agent Reinforcement Learning (Supplementary Material)

Xutong Zhao[1,2] **Yangchen Pan**[3] **Chenjun Xiao**[4] **Sarath Chandar**[1,2,6] **Janarthanan Rajendran**[1,5]

[1] Mila - Quebec AI Institute
[2] École Polytechnique de Montréal
[3] University of Oxford
[4] University of Alberta
[5] Université de Montréal

A PSEUDO-COUNT FOR DEEP RL

Counting visitations in high-dimensional or continuous state space could be challenging. This section introduces how we approximate counts by applying the static hashing [Tang et al., 2017] method, a well-established count approximation approach in RL, adopted successfully in works such as Rashid et al. [2020].

In particular, the state $s \in \mathcal{S}$ is projected to a lower-dimensional feature space by $\phi(s) = sgn(Ag(s)) \in \{-1, 1\}^k$, where $g : \mathcal{S} \to \mathbb{R}^D$ is an optional pre-processing function, $A \in \mathbb{R}^{k \times D}$ is a projection matrix with entries drawn i.i.d. from a unit Gaussian distribution $\mathcal{N}(0, 1)$, and $sgn(\cdot)$ is the element-wise sign function. This method clusters similar states in $\mathcal{S}$ to one feature in a small, countable feature space, which enables us to count. The k value controls the granularity of state approximation: higher k leads to more distinguishable features yet less generalizability across similar states. We record the visitation count for the tuple of the state feature $\phi(s)$ and all agents' joint action $\mathbf{a}$, denoted by $N(s, \mathbf{a})$ for simplicity of notation. Note that for each agent i, the count up to its action a_i satisfies:

$$\begin{aligned} N(s, a_{<i}, a_i) &= \sum_{a_{i+1}} N(s, a_{<i}, a_i, a_{i+1}) \\ &= \sum_{a_{>i}} N(s, a_{<i}, a_i, a_{>i}), \end{aligned}$$

where $a_{<i}$ and $a_{>i}$ denote the joint actions taken by preceding and subsequent agents of i, respectively. This relationship shows that we can obtain any count up to a_i by summing up the counts of joint actions that overlap $a_{<i}$ at state s. This relationship is naturally aligned with the tree structure, where the total count of each node equals the number of action sequences going through that node. Thus we are able to perform optimistic exploration using conditional counts.

B ENVIRONMENT DETAILS

Multi-Agent Particle Environment Multi-Agent Particle Environment (MPE) [Lowe et al., 2017, Mordatch and Abbeel, 2018] is a suite of two-dimensional navigation tasks where the entities in the environment obey physics properties. We choose three tasks that do not involve agent-wide communication: *Sparse Spread*, *Sparse Tag*, and *Adversary*. In the first two tasks, reward signals are sparse and agents receive positive rewards only when they jointly complete the task. They are almost fully observable except each agent does not observe the velocity of other agents. *Adversary* is fully observable.

Level-Based Foraging Level-Based Foraging (LBF) [Albrecht and Ramamoorthy, 2015, Christianos et al., 2020, Papoudakis et al., 2020] is a set of food-collection tasks in a grid-world. Each agent or food item is assigned a level value, such that a group of agents can pick up a food item if the sum of agents' levels is greater than or equal to the item's level. Agents receive a positive reward only when a food item is picked up, hence LBF requires efficient coordinated exploration. We choose four tasks with different grid dimensions, number of agents, and number of food items. By default, they are all fully observable.

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

StarCraft Multi-Agent Challenge StarCraft Multi-Agent Challenge (SMAC) [Samvelyan et al., 2019] consists of battle tasks where a group of agents is learned to defeat another group. Each agent could only observe entities within a fixed-sized window. All tasks have dense rewards, and agents start engaging immediately after the game starts. As Mahajan et al. [2019] point out, SMAC tasks are not designed to evaluate cooperative exploration. In order to assess coordination in partially-observable and non-stationary settings, we choose one easy task *2s-vs-1sc* and one hard task *3s-vs-5z*.

C EVALUATION PROTOCOL

In each task we train all algorithms for four million timesteps. During training we perform 41 evaluations at constant timestep intervals, that is, 100k timestep intervals, and at each evaluation point we evaluate for 100 episodes. We train each algorithm with parameter sharing, where all agent networks share the same set of parameters, and the one-hot identity of each agent as additional network input helps the neural network to develop diverse behaviour.

We evaluate algorithms' performance in a task by two metrics: maximum returns and average returns. The maximum return refers to the highest mean evaluation return across five seeds achieved at one evaluation point during training. This metric evaluates algorithms' best-reached performance in a task The average return is the evaluation return averaged over all evaluation points during training. This metric reflects both sample efficiency and final performance.

D ADDITIONAL RESULTS

Table 1 summarizes the *maximum* returns for all eight algorithms (including the ablations) in all nine tasks, which also reports the maximum win-rates in SMAC tasks. Figure 1 presents learning curves of the evaluation returns achieved during training by ablations in all nine tasks. Sparse-reward tasks have bold titles.

Table 1: Maximum Returns and 95% Confidence Interval for All Eight Algorithms in All Nine Tasks, and Maximum Win-rates for SMAC Tasks.

		Tasks \Algs.	COE	COE-Cond-IQ	COE-Cond-CQ	UCB-Ind	UCB-Cen	EMC	MAVEN	QMIX
MPE		Adversary	22.68 ± 0.80	19.18 ± 1.70	24.14 ± 0.83	23.16 ± 1.28	23.02 ± 0.93	22.03 ± 2.12	23.52 ± 1.50	22.70 ± 1.61
		Sparse Tag	1.60 ± 0.41	0.16 ± 0.18	1.98 ± 0.77	1.28 ± 0.31	1.44 ± 0.05	1.23 ± 0.35	0.06 ± 0.03	1.16 ± 0.29
		Sparse Spread	2.11 ± 1.86	0.99 ± 0.85	1.46 ± 1.05	1.51 ± 1.06	1.80 ± 1.15	1.31 ± 0.92	0.43 ± 0.85	1.46 ± 0.28
LBF		10x10-3p-3f	0.99 ± 0.01	0.98 ± 0.02	0.98 ± 0.01	0.98 ± 0.02	0.99 ± 0.01	0.96 ± 0.04	0.37 ± 0.18	0.94 ± 0.03
		15x15-3p-5f	0.45 ± 0.10	0.36 ± 0.09	0.29 ± 0.15	0.37 ± 0.08	0.31 ± 0.14	0.24 ± 0.04	0.04 ± 0.01	0.20 ± 0.02
		15x15-4p-3f	0.93 ± 0.03	0.89 ± 0.02	0.63 ± 0.13	0.75 ± 0.11	0.48 ± 0.31	0.71 ± 0.13	0.06 ± 0.01	0.51 ± 0.09
		15x15-4p-5f	0.69 ± 0.08	0.38 ± 0.05	0.32 ± 0.07	0.52 ± 0.20	0.57 ± 0.15	0.50 ± 0.08	0.05 ± 0.01	0.33 ± 0.04
SMAC	ret	2s-vs-1sc	20.25 ± 0.01	19.57 ± 0.73	20.24 ± 0.00	15.88 ± 7.79	20.19 ± 0.07	20.22 ± 0.06	20.22 ± 0.04	20.16 ± 0.05
		3s-vs-5z	21.32 ± 0.75	21.16 ± 0.56	21.47 ± 0.59	16.93 ± 4.24	19.86 ± 5.03	14.84 ± 4.19	20.15 ± 1.43	18.57 ± 3.01
	win	2s-vs-1sc	1.00 ± 0.00	0.92 ± 0.09	1.00 ± 0.00	0.77 ± 0.38	0.99 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.00
		3s-vs-5z	0.97 ± 0.00	0.93 ± 0.05	0.98 ± 0.02	0.56 ± 0.45	0.61 ± 0.37	0.27 ± 0.37	0.87 ± 0.16	0.65 ± 0.30

E ABLATION DETAILS

In this section, we present in detail the ablation variants introduced in Section 5.3.

COE-Cond-IQ directly adopts the idea of UCT, without considering the partial observability issue of each agent. In order to enable decentralized execution, we simultaneously learn a Q-value function dependent on preceding agents' actions and its independent counterpart. Similar to the MACPF factorization [Wang et al., 2022], each agent i has an independent Q-network $Q_i^{idp}(\tau_i, a_i; \phi_i)$ parameterized by ϕ_i, and a dependency correction network $c_i^{dep}(\tau_i, a_i|a_{<i}; \psi_i)$ parameterized by ψ_i, whose sum constructs the dependent Q-network $Q_i^{dep}(\tau_i, a_i|a_{<i}; \phi_i, \psi_i) = Q_i^{idp}(\tau_i, a_i; \phi_i) + c_i^{dep}(\tau_i, a_i|a_{<i}; \psi_i)$.

Individual agent's action-value networks Q_i^{dep} and Q_i^{idp} are separately trained by minimizing the mean-squared TD error on each Q-network:

$$\mathcal{L}_i^{dep}(\psi_i) = \mathbb{E}_{\mathcal{D}}[(Q_i^{dep}(\tau_i, a_i|a_{<i}) - y_i^{dep})^2] \tag{1}$$

$$\mathcal{L}_i^{idp}(\phi_i) = \mathbb{E}_{\mathcal{D}}[(Q_i^{idp}(\tau_i, a_i) - y_i^{idp})^2] \tag{2}$$

where $y_i^{dep} = (r + \gamma \max_{a_i'}(Q_i^{dep}(\tau_i', a_i'|a_{<i})))$ and $y_i^{idp} = (r + \gamma \max_{a_i'}(Q_i^{idp}(\tau_i', a_i')))$ are the update targets, and $\mathcal{D}$ contains trajectory data collected by Q_i^{dep}'s. To ensure Q_i^{dep} and Q_i^{idp} achieve the same performance, they are constructed

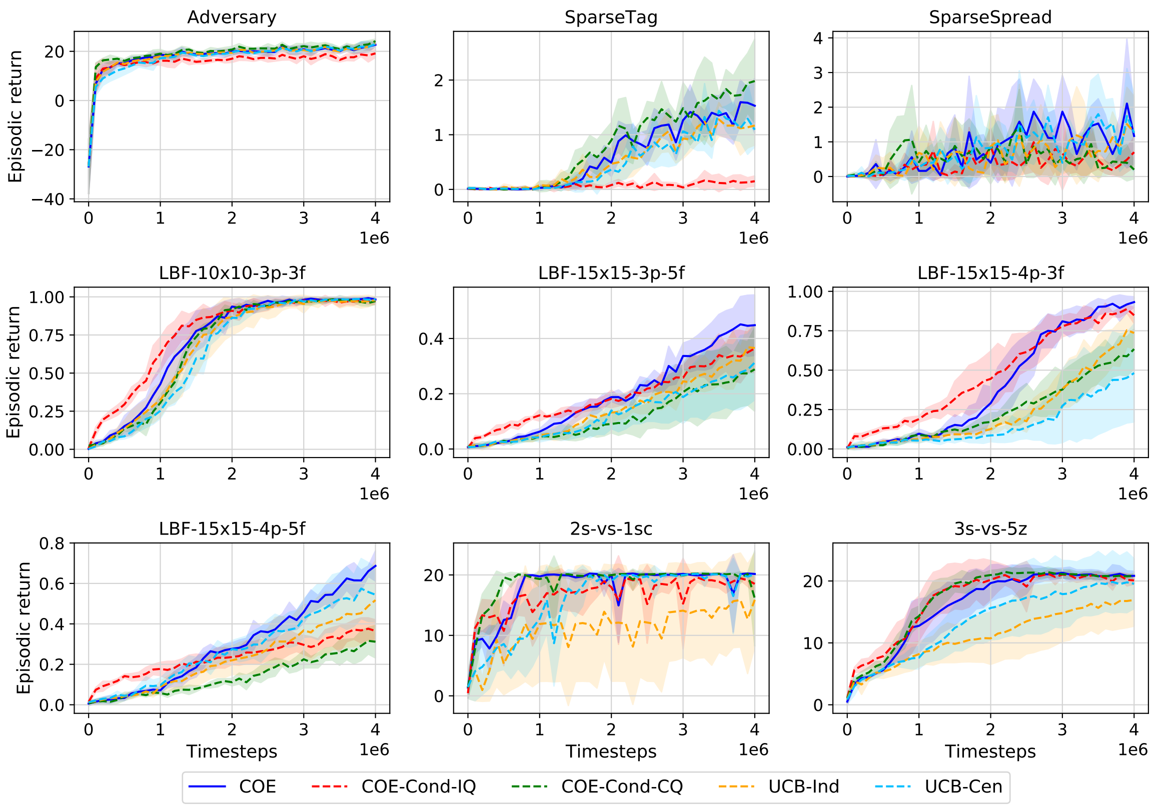

Figure 1: Episodic Returns and 95% Confidence Interval for All Ablations in All Tasks.

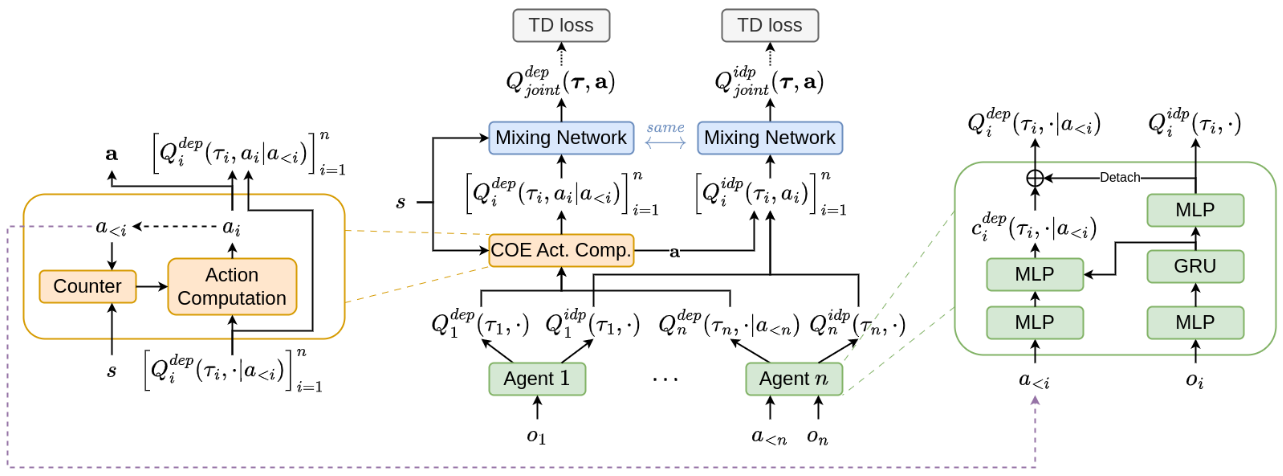

Figure 2: Learning Framework for COE-Cond-CQ.

and trained in a way that strengthens their coupling: Q_i^{dep} is the combination of Q_i^{idp} and a correction network; during training the same mini-batch of trajectory data sampled from $\mathcal{D}$ is used to compute both $\mathcal{L}_i^{dep}$ and $\mathcal{L}_i^{idp}$.

COE exploration is applied to this variant in a similar way as being applied to value decomposition methods. The optimistic bonus is added to Q_i^{dep} at action selection during training. Note that for each agent i the optimistic TD update target is applied to both Equation (1) and Equation (2):

$$y_i = \left(r(s, \mathbf{a}) + \frac{c_{\text{rew}}}{\sqrt{N(s, a_{<i}, a_i)}} \right) + \gamma \max_{a_i'} \left(Q_i(\tau_i', a_i') + \frac{c_{\text{boot}}}{\sqrt{N(s', a_{<i}', a_i')}} \right), \tag{3}$$

where $c_{\text{rew}}, c_{\text{boot}} \in \mathbb{R}_+$ are hyper-parameters controlling the scale of the optimistic bias in reward and bootstrapped target, respectively. During decentralized execution, agents take actions according to Q_i^{idp}'s only.

We name this variant COE-Cond-IQ as it could be considered as a direct adoption of UCT to IQL [Tan, 1993]. As opposed to the utility function that learns implicit dependency via centralized training in value decomposition methods, each agent learns a Q-value function, that explicitly captures the correlation among agents by conditioning on previous agents' actions. COE-Cond-IQ also complies with the CTDE paradigm. However, it ignores the partial observability of each individual agent. Each agent only has access to its own local trajectory history.

Another ablation we introduce is COE-Cond-CQ, which combines centralized training and COE-Cond-IQ. The learning framework of COE-Cond-CQ is illustrated in Figure 2. The same mixing network $\text{Mixer}(\cdot;\theta)$ we use in COE is used to compute both dependent and independent joint Q-values:

$$Q_{joint}^{dep}(\boldsymbol{\tau}, \mathbf{a}) = \text{Mixer}\left([Q_i^{dep}(\tau_i, a_i|a_{<i})]_{i=1}^N, s; \theta\right) \tag{4}$$

$$Q_{joint}^{idp}(\boldsymbol{\tau}, \mathbf{a}) = \text{Mixer}\left([Q_i^{idp}(\tau_i, a_i)]_{i=1}^N, s; \theta\right) \tag{5}$$

Similarly, centralized training also optimizes both dependent and independent mean-squared TD error:

$$\mathcal{L}^{dep}([\psi]_{i=1}^N, \theta) = \mathbb{E}_{\mathcal{D}}[(Q_{joint}^{dep}(\boldsymbol{\tau}, \mathbf{a}) - y^{dep})^2] \tag{6}$$

$$\mathcal{L}^{idp}([\phi]_{i=1}^N, \theta) = \mathbb{E}_{\mathcal{D}}[(Q_{joint}^{idp}(\boldsymbol{\tau}, \mathbf{a}) - y^{idp})^2] \tag{7}$$

where $y^{dep} = (r + \gamma \max_{\mathbf{a}'}(Q_{joint}^{dep}(\boldsymbol{\tau}', \mathbf{a}')))$ and $y^{idp} = (r + \gamma \max_{\mathbf{a}'}(Q_{joint}^{idp}(\boldsymbol{\tau}', \mathbf{a}')))$ are update targets for dependent and independent networks, respectively. Exploration is performed the same way as COE, and action selection is performed the same way as COE-Cond-IQ.

In the ablation UCB-Ind, each agent performs UCB-based exploration independently. It is straightforward to obtain UCB-Ind: we simply replace any conditional count terms in COE with independent counts, which do not rely on other agents' actions.

The ablation UCB-Cen augments the global reward with an intrinsic reward $\frac{c_{\text{rew}}}{\sqrt{N(s,\mathbf{a})}}$. Agents learn optimistic Q-values through centralized training.

F HYPERPARAMETER SETTINGS

To perform hyperparameter optimization we follow the same protocol presented by Papoudakis et al. [2020]. We select one task from each benchmark environment and optimize the hyperparameters of all algorithms in this task. In particular, we select *Sparse Tag* from MPE, *15x15-3p-5f* from LBF, and *3s-vs-5z* from SMAC. We perform a coarse grid search on hyperparameter settings and train each configuration with three seeds. We identify the best configuration according to the maximum evaluation returns. This best configuration on each task is then used for all tasks in the respective environment for the final experiments with five seeds.

For methods that use intrinsic reward — i.e. COE, EMC, and MAVEN — we only test constant intrinsic reward scales. For COE, the hyperparameter combination with $c_{\text{act}} = c_{\text{rew}} = c_{\text{boot}} = 0$ is ignored as this setting refers to the greedy-action QMIX. For MAVEN, we determine the hyperparameter settings according to the original paper and its accompanying codebase. In particular, we sweep the intrinsic scales only when "MI intrinsic" is True. The hyperparameters "MI intrinsic" and "RNN discriminator" cannot both be True. When MAVEN uses ε-greedy, the epsilon annealing time is 50k timesteps. Every epsilon annealing schedule — utilized by either MAVEN or QMIX — is linear with an initial value of 1.0 and a final value of 0.0.

References

Stefano V Albrecht and Subramanian Ramamoorthy. A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems. *arXiv preprint arXiv:1506.01170*, 2015.

Filippos Christianos, Lukas Schäfer, and Stefano Albrecht. Shared experience actor-critic for multi-agent reinforcement learning. *Advances in neural information processing systems*, 33:10707–10717, 2020.

Table 2: Common QMIX Hyperparameters for All algorithms across All Tasks.

Hyperparameter Name	Value
hidden dimension	128
reward standardization	True
network type	GRU
evaluation epsilon	0
target update	0.01 (soft)

Table 3: Hyperparameters for COE: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0001	0.0003	0.0005
feature dimension k	8/12/16	8	16	8
c_{act}	0/0.01/0.05	0.01	0.01	0
c_{boot}	0/0.01/0.05	0	0	0
c_{rew}	0/0.01/0.05	0.05	0	0.05

Table 4: Hyperparameters for COE-Cond-IQ: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0001	0.0001	0.0005
feature dimension k	8/12/16	8	8	16
c_{act}	0/0.01/0.05	0	0.05	0.01
c_{boot}	0/0.01/0.05	0	0	0
c_{rew}	0/0.01/0.05	0.05	0	0

Table 5: Hyperparameters for COE-Cond-CQ: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0001	0.0003	0.0005
feature dimension k	8/12/16	12	16	8
c_{act}	0/0.01/0.05	0.05	0.01	0.05
c_{boot}	0/0.01/0.05	0	0	0.01
c_{rew}	0/0.01/0.05	0.05	0.01	0.01

Table 6: Hyperparameters for UCB-Indep: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0001	0.0003	0.0005
feature dimension k	8/12/16	8	12	12
c_{act}	0/0.01/0.05	0.01	0.01	0
c_{boot}	0/0.01/0.05	0	0.01	0
c_{rew}	0/0.01/0.05	0	0.01	0.01

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multi-agent variational exploration. *Advances in Neural Information Processing Systems*, 32, 2019.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Proceedings*

Table 7: Hyperparameters for UCB-Central: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0001	0.0003	0.0005
feature dimension k	8/12/16	8	8	16
c_{rew}	0/0.01/0.05	0.05	0.01	0.05

Table 8: Hyperparameters for EMC: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0001	0.0003	0.0005
curiosity scale	0.001/0.005/0.01/0.05/0.1/0.5/1.0	0.01	0.001	0.001

Table 9: Hyperparameters for MAVEN: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0003	0.0001	0.0005
RNN discriminator	True/False	False	False	False
MI intrinsic	True/False	True	True	True
curiosity scale	0.001/0.005/0.01/0.05/0.1/0.5/1.0	0.01	0.05	0.005
noise bandit	True/False	True	False	True
epsilon start	0.0/1.0	1.0	1.0	1.0

Table 10: Hyperparameters for QMIX: Values Swept in Grid-search and Best Configuration for each Benchmark.

Hyperparameter Name	Swept values	MPE	LBF	SMAC
learning rate	0.0001/0.0003/0.0005	0.0001	0.0001	0.0005
epsilon anneal	50,000/200,000	50,000	50,000	50,000

of the AAAI conference on artificial intelligence, volume 32, 2018.

Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. *arXiv preprint arXiv:2006.07869*, 2020.

Tabish Rashid, Bei Peng, Wendelin Boehmer, and Shimon Whiteson. Optimistic exploration even with a pessimistic initialisation. *arXiv preprint arXiv:2002.12174*, 2020.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 2019.

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

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Jiangxing Wang, Deheng Ye, and Zongqing Lu. More centralized training, still decentralized execution: Multi-agent conditional policy factorization. *arXiv preprint arXiv:2209.12681*, 2022.