

000
001
002
003
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053

AGENT-CHAINED POLICY OPTIMIZATION

Anonymous authors

Paper under double-blind review

ABSTRACT

We study Cooperative Multi-Agent Reinforcement Learning (MARL), where the aim is to train decentralized policies that maximize a shared return. Existing methods typically employ either iterative best-response updates, which converge only to Nash Equilibria (NE) that may be far from the global optimum, or simultaneous learning with centralized critics, which lack convergence guarantees to the optimal joint policy without strong assumptions on decomposable value functions. We introduce the Agent-Chained Belief MDP (AC-BMDP), which reformulates MARL as a serialized decision process where agents act sequentially while maintaining beliefs over actions taken by preceding agents. This enables the definition of agent-specific value functions that are naturally chained together. Building on this framework, we propose Agent-Chained Policy Iteration (ACPI) and prove that it converges to the globally optimal joint policy. We further develop this framework into a practical actor-critic algorithm, Agent-Chained Policy Optimization (ACPO). On standard benchmarks, ACPO consistently surpasses state-of-the-art baselines, with the performance advantage growing significantly as the number of agents increases.

1 INTRODUCTION

Cooperative multi-agent systems are increasingly employed to tackle complex tasks ranging from autonomous vehicle fleets to distributed sensor networks, where decentralized coordination is required (Dafoe et al., 2020). The goal of Cooperative Multi-Agent Reinforcement Learning (MARL) is to learn a set of policies that enable multiple agents to collectively maximize a shared return. Despite substantial progress, achieving decentralized (fully factorized) policies that provably converge to the global optimum remains an open challenge.

Existing MARL methods largely fall into one of two categories: iterative best-response updates or simultaneous policy updates. First, iterative best-response updates bypass the problem of joint policy optimization by instead solving for a game-theoretic equilibrium. For instance, Multi-Agent Policy Iteration (MA-PI) (Zhong et al., 2024) adopts the iterative best-response approach which is designed to converge to a Nash equilibrium (NE), and provides the foundation for practical algorithms such as HAPPO (Kuba et al., 2022) and HATRPO (Zhong et al., 2024). However, NEs are a fixed point where policies cannot be unilaterally improved, and this can be arbitrarily far from the optimal joint policy in fully cooperative settings (see Table 1 for a simple illustration). Furthermore, they require each agent to be updated in turn while holding the other agents fixed. When these methods are extended to high-dimensional settings, this prolongs training time and prohibits scaling to a larger number of agents in practice.

The second approach is simultaneous policy updates, which considers the multi-agent problem as a single-agent MDP defined over the joint action space. A wide range of algorithms follow this paradigm, which includes, but not limited to, MAPPO (Yu et al., 2022) and MADDPG (Lowe et al., 2017). These methods are natural extensions of single-agent RL, where all agents are trained in parallel, making them simple to implement and scalable in practice. However, they typically rely on a centralized value function defined over the joint action space, while policies must remain decentralized during execution. This mismatch creates a gap between training and deployment, and convergence to an optimal policy often requires additional assumptions such as value decomposition (Rashid et al., 2018; Zhang et al., 2021).

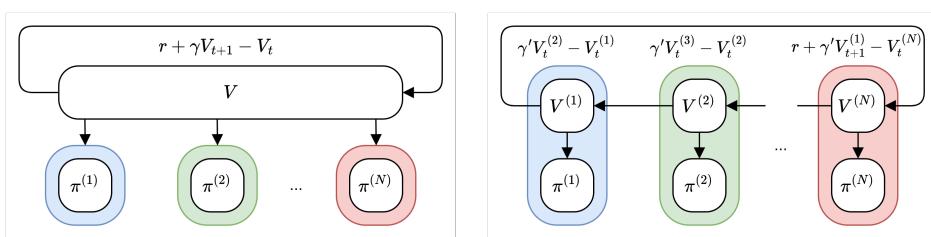


Figure 1: Centralized Critics (left) commonly used in prior work and Decentralized Critics (right) with agent chaining used in ACPO.

In summary, state-of-the-art MARL methods can largely be categorized into iterative best-response updates or simultaneous updates. Iterative best-response methods offer convergence guarantees, but only to an NE, and their sequential updates lead to prohibitive training costs as the number of agents increases. Simultaneous update methods avoid this bottleneck but lack convergence guarantees unless strong assumptions are imposed. Ultimately, neither class is fundamentally designed to ensure convergence to the globally optimal joint policy that maximizes return.

In this work, we leverage a key property of MARL: Any simultaneous problem can be recast as a sequential one where the agents take actions sequentially. However, this serialized version of the problem cannot be solved directly since each agent cannot observe the actions executed by other agents. We formalize this by introducing the Agent-Chained Belief MDP (AC-BMDP), which is an MDP with the belief over the actions by preceding agents. While the formulation chains agents sequentially, each agent still executes its own policy independently, consistent with the CTDE paradigm. The combination of serialization and the notion of a belief creates well-defined agent-specific value functions which are naturally *chained* to the value function of the next agent.

Our main theoretical contribution is Agent-Chained Policy Iteration (ACPI), which we prove convergence to the globally optimal joint policy in the underlying Multi-Agent MDP. Building on ACPI, we introduce Agent-Chained Policy Optimization (ACPO), a practical actor-critic algorithm under the Centralized Training with Decentralized Execution (CTDE) setting. Unlike iterative best-response updates, our formulation aims for optimal cooperative policies and facilitates parallel policy updates. Also, unlike simultaneous policy updates where the centralized value functions and fully factorized policies are not aligned without proper assumptions, our agent-chained formulation ensures that value functions and policies are inherently aligned (see Figure 1).

Empirically, we evaluate our approach on a suite of standard MARL benchmarks, including Multi-Robot Warehouse, SMACv2 and MA-MuJoCo. We show that ACPO consistently outperforms strong baselines on all tasks, where the gap widens as the number of agents increases. To the best of our knowledge, ACPO is the first algorithm under CTDE that directly targets convergence to the globally optimal policy, instead of a Nash equilibrium.

2 BACKGROUND

2.1 MULTI-AGENT MDP

We consider a cooperative multi-agent environment with $\mathcal{N} = \{1, \dots, N\}$ agents, formally defined as a Multi-Agent Markov Decision Process (MMDP) (Boutilier, 1996). At time step t , each agent $i \in \mathcal{N}$ takes action $a_t^{(i)} \in A^{(i)}$ sampled from policy $\pi^{(i)}(a_t^{(i)} | s_t)$ where $s_t \in S$ is the state. The state transition is Markovian, i.e. the next state s_{t+1} is given by transition function $T(s_{t+1} | s_t, \vec{a}_t)$ where \vec{a}_t is the joint action $\vec{a}_t = [a_t^{(1)}, \dots, a_t^{(N)}]$. Each agent receives shared reward r_t generated by the common reward function $R(s_t, \vec{a}_t)$.

The goal of the multi-agent RL task is to find a set of agent policies $\vec{\pi} = [\pi^{(1)}, \dots, \pi^{(N)}]$ that maximize the total expected return $J = \mathbb{E}_{\tau \sim \text{Pr}(\tau | \vec{\pi})} [\sum_t \gamma^t r_t]$ where γ is the discount factor. As in previous work on cooperative MARL, we assume that the policies $\vec{\pi}$ are allowed to be trained in a centralized manner, but require decentralized execution, known as the Centralized Training with Decentralized Execution (CTDE) setting.

108
109

2.2 PREVIOUS APPROACHES IN COOPERATIVE MARL

110

Iterative best-response update (Iterative BR) One way to solve MMDPs is to define a reduced MDP for each agent and learn the best response policy in order to converge to a Nash equilibrium (NE) (Bertsekas, 2020). Formally, each agent i solves its own reduced MDP $\langle S, A^{(i)}, T^{(i)}, R^{(i)} \rangle$, where $T^{(i)}(s_{t+1}|s_t, a_t^{(i)}) := \mathbb{E}_{\vec{\pi}^{-i}}[T(s_{t+1}|s_t, a_t^{(i)}, \vec{a}^{-i})]$, and \vec{a}^{-i} is the joint action excluding the action of agent i and is distributed according to the most recent policies $\vec{\pi}^{-i}$. The reward function is marginalized similarly as $R_t^{(i)}(s_t, a_t^{(i)}) = \mathbb{E}_{\vec{\pi}^{-i}}[R(s_t, a_t^{(i)}, \vec{a}^{-i})]$. Recent methods such as HATRPO (Kuba et al., 2022) and HAPPO (Zhong et al., 2024) leverage this insight and use single-agent RL methods such as TRPO (Schulman et al., 2015) and PPO (Schulman et al., 2017) to train the best response policy for each agent, respectively. However, this only approximates to an NE where each agent cannot unilaterally improve its policy, or Quantal Response Equilibrium (QRE) (Liu et al., 2024) which is an NE defined over the entropy-augmented reward function. Crucially, there is no guarantee that this NE or QRE will coincide with a globally optimal policy in the underlying MMDP. Furthermore, Iterative BR does not scale well in practice. Since each agent is trained sequentially while holding the other agent fixed, training time significantly increases as the number of agents increase.

124

125

Simultaneous policy update An alternative way to solve MMDPs is to simply view the problem as a single-agent MDP defined over the joint action space. Previous methods such as MAPPO (Yu et al., 2022) and MADDPG (Lowe et al., 2017) train a centralized critic over this MDP. MAPPO, for instance, trains each policy via an objective of the form, $J(\pi^{(i)}) = \mathbb{E}_{\pi^{(i)}, \vec{\pi}^{-i}}[A(s, a^{(i)}, \vec{a}^{-i})]$ where $A(s, \vec{a})$ is the advantage computed via Generalized Advantage Estimation (GAE) (Schulman et al., 2016). At first glance, it may seem that this approach is maximizing return over factorized policies, since each policy $\pi^{(i)}$ is trained simultaneously. However, a more careful look reveals that each agent updates its policy to maximize $A^{(i)}(s, a^{(i)})$ where \vec{a}^{-i} is marginalized out. This does not guarantee that the joint policy will be improved unless we impose strong assumptions such as value decomposition (Rashid et al., 2018; Zhang et al., 2021). In general, the convergence guarantees of these methods are not well-understood. In comparison to Iterative BR approaches, the parallel nature of the policy updates mitigates the bottleneck during training time.

137

138

2.3 LIMITATION OF PREVIOUS APPROACHES

139

140

Consider the simple Matrix Game in Table 1 with 2 agents and 3 actions, which was considered in Liu et al. (2024). The three NEs are (A, A) , (B, B) , (C, C) , since there is no incentive for either agent to change its action at each of those NEs. (C, C) is a global optimum of the game since it achieves the highest return. As shown in Table 1, Iterative BR and simultaneous policy update, which are the foundations for HAPPO and MAPPO, respectively, cannot escape suboptimal NEs. The only way it can converge to (C, C) is for the policy to be initialized with high probability towards (C, C) . However, it is not desirable to rely on the initialization of the policy for finding good NEs, since it implies that extensive hyperparameter tuning is required. Similarly, Heterogenous-Agent Soft Policy Iteration (HASPI) (Liu et al., 2024) can only find the global optimum for specific combinations of the entropy parameter α and the initialization of the policy (e.g. $\alpha = 5$ and $\pi^{(1)}(A) = \pi^{(2)}(A) = 0.6$), if it happens to coincide with the QRE. Similar to NE, there is no guarantee that the QRE will coincide with the global optimum, and thus generally requires extensive hyperparameter tuning to find the right balance between reward and entropy maximization.

151

152

Our goal in the rest of the paper is to derive a principled algorithm which directly targets the globally optimal policy in the MMDP. In Section 5, we present Agent-Chained Policy Iteration (ACPI) which converges to the global optimum regardless of the policy initialization. Detailed analysis on how ACPI is able to solve the Matrix Game introduced in Table 1 is provided in Appendix G.

156

157

3 SERIALIZATION

158

159

160

161

To address the challenges in simultaneous decision-making by multiple agents, we adopt a key property of MMDPs, namely that we can solve an equivalent serialized version of the problem, where each agent takes actions sequentially. This serialization approach, while explored in Bertsekas (2019), Kovařík et al. (2023) and Peralez et al. (2025) has not been successfully utilized in the Deep

		<i>A</i>	<i>B</i>	<i>C</i>	
		<i>A</i>	5	-20	-20
		<i>B</i>	-20	10	-20
		<i>C</i>	-20	-20	20
3x3 Matrix Game					
<i>A</i>	<i>A</i>	1	0	0	
	<i>B</i>	0	0	0	
	<i>C</i>	0	0	0	
Simul. Update					
<i>A</i>	<i>A</i>	1	0	0	
	<i>B</i>	0	0	0	
	<i>C</i>	0	0	0	
Iterative BR					
<i>A</i>	<i>A</i>	1	0	0	
	<i>B</i>	0	0	0	
	<i>C</i>	0	0	0	
HASPI ($\alpha=1$)					
<i>A</i>	<i>A</i>	0	0	0	
	<i>B</i>	0	0	0	
	<i>C</i>	0	0	1	
HASPI ($\alpha=5$)					
<i>A</i>	<i>A</i>	0	0	0	
	<i>B</i>	0	0	0	
	<i>C</i>	0	0	1	
ACPI					

Table 1: A 3x3 Matrix Game and converged policy values when initialized to $\pi^{(1)}(A) = \pi^{(2)}(A) = 0.6$. ACPI (Ours) converges to the global optimum (C, C) regardless of the initialization.

MARL setting. Specifically, each time step t is decomposed into N micro-steps. Thus, there are now two timescales at each decision point: the time step denoted by t and micro-steps denoted by t' , where $\lfloor t'/N \rfloor = t$. Each micro-step t' corresponds to an agent i committing to individually executing action $a^{(i)}$. Under this framework, the MMDP state is now augmented to include the actions taken by previous agents, i.e. $[s_t, \vec{a}_t^{<i}]$, where $\vec{a}_t^{<i} = [a_t^{(1)}, \dots, a_t^{(i-1)}]$. The state transitions within micro-steps are deterministic. A full state transition by T only occurs after all N agents have committed their actions:

$$T([s_t, \vec{a}_t^{<i+1}] \mid [s_t, \vec{a}_t^{<i}], a_t^{(i)}) = \mathbb{I}\{[\vec{a}_t^{<i}, a_t^{(i)}] = \vec{a}_t^{<i+1}\} \text{ if } i \in \{1, \dots, N-1\}$$

$$T([s_{t+1}, \emptyset] \mid [s_t, \vec{a}_t^{<N}], a_t^{(N)}) = T(s_{t+1} \mid s_t, \vec{a}_t)$$

Similarly, rewards are only generated by MMDP reward function R once all the agents have committed their actions.

$$R([s_t, \vec{a}_t^{<i}], a_t^{(i)}) = \begin{cases} R(s_t, \vec{a}_t) & \text{if } i = N \\ 0 & \text{otherwise} \end{cases}$$

Finally, the discount factor is denoted as γ' where $\gamma' = \gamma^{1/N}$.

The optimal policy for this serialized problem is in fact optimal for the original MMDP as well.

Theorem 3.1. (Peralez et al., 2025) For every MMDP, there exists a serialized multi-agent problem, of which its optimal policy is also optimal for the underlying MMDP.

Serialization has a number of practical advantages. First, by defining the action space on individual actions $|A^{(i)}|$, the action space of each timestep is reduced. It effectively mitigates the exponential complexity associated with the joint action space $|A| = |A^{(i)}|^N$ for search. Second, serialization inherently facilitates credit assignment across individual actions as all of the components of the MDP are defined with respect to individual actions. In contrast, the reward function R in the original problem is a function of the joint action \vec{a} , and thus requires a separate credit assignment mechanism (e.g. Foerster et al. (2018); Wang et al. (2022)) to attribute contributions of individual actions $a^{(i)}$ to overall rewards and returns. Finally, serialization changes the perspective of the problem into a single-agent MDP, thus making it easier to apply single-agent techniques and algorithms.

4 AGENT-CHAINED BELIEF MDP (AC-BMDP)

While serialization transforms the multi-agent problem into a sequential single-agent MDP over micro-steps and individual actions, a critical challenge remains: the actions taken by other agents are not observable. *This effectively renders the serialized problem as a partially observable one, even if the underlying problem is fully observable.* As a consequence, this necessitates the definition of a belief MDP, where the belief state represents a distribution over the actions taken by other agents. To this end, we propose the Agent-Chained Belief MDP (AC-BMDP), which is designed such that (1) the optimal policy coincides with that of the underlying MMDP and (2) incorporates the notion of a belief over preceding actions.

Motivated by (Nayyar et al., 2013), The action space in an AC-BMDP is no longer the actions $a^{(i)}$, but rather an action distribution $\phi^{(i)} \in \Delta(A^{(i)})$. The belief $b_t^{(i)}(\vec{a}_t^{< i}) \triangleq \Pr(\vec{a}_t^{< i} \mid s_t, \vec{\phi}_t^{< i})$ for agents $i \in \{2, \dots, N\}$ is defined as the distribution over the previous agent actions within the current time

step t . For agent 1, the belief is empty as there are no preceding agents. The state space $[s_t, b_t^{(i)}]$ now augments the state s_t with agent i 's current belief. The reward function and transition function are defined as follows:

$$R\left([s_t, b_t^{(i)}], \phi_t^{(i)}\right) = \sum_{\vec{a}_t^{< i}} b_t^{(i)}(\vec{a}_t^{< i}) \sum_{a_t^{(i)}} \phi_t^{(i)}(a_t^{(i)}) R\left([s_t, \vec{a}_t^{< i}], a_t^{(i)}\right).$$

$$T \left([s_t, b_t^{(i+1)}] \mid [s_t, b_t^{(i)}], \phi_t^{(i)} \right) = \begin{cases} 1 & \text{if } b_t^{(i+1)} = \tau \left([s_t, b_t^{(i)}], \phi_t^{(i)} \right) \\ 0 & \text{otherwise} \end{cases} \quad \text{if } i \in \{1, \dots, N-1\}$$

$$T([s_{t+1}, \emptyset] \mid [s_t, b_t^{(N)}], \phi_t^{(N)}) = \sum_{\vec{a}^{< N}} b_t^{(N)}(\vec{a}_t^{< N}) \sum_{a_t^{(N)}} \phi_t^{(N)}(a_t^{(N)}) T(s_{t+1} \mid s_t, \vec{a}_t^{< N}, a_t^{(N)})$$

where τ is the belief update rule provided in Appendix A.

Our formulation as an AC-BMDP highlights the fact that, by the nature of simultaneous action selection, each agent must infer the previous agents' actions in order to make optimal decisions. It is worth noting that opponent modelling and the prediction of other agents' actions have a rich history in MARL (Albrecht & Stone, 2018). However, while previous work focused on opponent modelling as an additional algorithmic module, we derive this formally as something necessary for solving the underlying MMDP and finding the globally optimal policy.

5 AGENT-CHAINED POLICY ITERATION

In this section, we present a policy iteration procedure called *Agent-Chained Policy Iteration* (ACPI) which is defined on the AC-BMDP, and formally prove that the fixed point of this procedure is also optimal in the underlying MMDP. Unlike Iterative BR (Zhong et al., 2024) where the fixed point of policy iteration is a NE, we show that ACPI is guaranteed to converge to the global optimum of the MMDP.

We start by defining the Bellman operator under the AC-BMDP.

Definition 5.1. (Agent-Chained Bellman Operators)

$$(\mathcal{T}^{\vec{\pi}} Q^{(1)})([s, b^{(N)}], \phi^{(N)}) := R([s, b^{(N)}], \phi^{(N)}) + \gamma' \mathbb{E}_{\substack{s' \sim T(\cdot | [s, b^{(N)}], \phi^{(N)}) \\ \phi^{(1)} \sim \pi^{(1)}(\cdot | s')}} \left[Q^{(1)}([s', \phi^{(1)}]) \right]$$

$$(\mathcal{T}^{\vec{\pi}} Q^{(i+1)})([s, b^{(i)}], \phi^{(i)}) := \gamma' \mathbb{E}_{\substack{b^{(i+1)} = T([s, b^{(i)}], \phi^{(i)}) \\ \phi^{(i+1)} \sim \pi^{(i+1)}(\cdot | s, b^{(i+1)})}} \left[Q^{(i+1)}([s, b^{(i+1)}], \phi^{(i+1)}) \right]$$

$$\quad \quad \quad \text{if } i \in \{1, \dots, N-1\},$$

where we used the fact that $R = 0$ for the micro-steps when agents $i \in \{1, \dots, N-1\}$ take actions. Note that we have defined the Bellman operators separately for agent N since its actions will affect the actual transitions and rewards.

We immediately see the benefit of serialization, where we now have a well-defined set of *decentralized value functions* for each agent. This is in sharp contrast to previous approaches which required restrictions on the environment to decompose the value function into individual utility functions (Peng et al., 2021; Rashid et al., 2018; Zhang et al., 2021).

The Q-values also have the intended meaning, which is the expected return given that agent i is in state $[s, b^{(i)}]$ and takes action distribution $\phi^{(i)}$ (which reduces to action $a^{(i)}$ for deterministic $\phi^{(i)}$). This now provides us with a natural way to assign credit for each agent. Moreover, we have now *chained* the agents by their Q-values, as the target for $Q^{(i)}$ is $Q^{(i+1)}$. This provides an intuitive interpretation for policy evaluation, where $Q^{(i)}([s, b^{(i)}], \phi^{(i)})$ considers how taking an action $\phi^{(i)}$ at state $[s, b^{(i)}]$ will affect the next agent's Q-values, $Q^{(i+1)}$.

By repeatedly applying $\mathcal{T}^{\vec{\pi}}$, we can obtain the Q -values for a given joint policy $\vec{\pi}$:

Lemma 5.2. (Agent-Chained Policy Evaluation) *The Agent-Chained Bellman Operators in Definition 5.1 are a contraction mapping under the infinity norm. Thus, starting with any $\vec{Q} = \langle Q^{(1)}, \dots, Q^{(N)} \rangle$ and a joint policy $\vec{\pi} = \langle \pi^{(1)}, \dots, \pi^{(N)} \rangle$, the repeated application of $\mathcal{T}^{\vec{\pi}}$ will return a set of Q -values for each agent $\langle Q^{(1, \vec{\pi})}, \dots, Q^{(N, \vec{\pi})} \rangle$ in the limit.*

270 *Proof.* See Appendix C.1. □

272 During policy improvement, each agent’s policy $\pi^{(i)}$ will be updated to select the greedy action with
 273 respect to their own $Q^{(i)}$,

$$275 \quad \forall i, b^{(i)}, s, \pi_{new}^{(i)}([s, b^{(i)}]) \leftarrow \arg \max_{\phi^{(i)}} Q^{(i, \vec{\pi})}([s, b^{(i)}], \phi^{(i)}). \quad (1)$$

277 **Lemma 5.3.** (Agent-Chained Policy Improvement) *Given a deterministic policy $\vec{\pi} = \langle \pi^{(1)}, \dots, \pi^{(N)} \rangle$, let $Q^{(i, \vec{\pi})}$ denote the i -th agent’s value function for a joint policy $\vec{\pi}$. If we update the new policy $\vec{\pi}_{new} = \langle \pi_{new}^{(1)}, \dots, \pi_{new}^{(N)} \rangle$ by Eq. 1, then*

$$281 \quad Q^{(i, \vec{\pi}_{new})}(s, b^{(i)}, \phi^{(i)}) \geq Q^{(i, \vec{\pi})}([s, b^{(i)}], \phi^{(i)})$$

283 *Proof.* See Appendix C.2. □

285 The policy evaluation step using Definition 5.1 and policy improvement with Eq. 1, also highlights
 286 an important property of agent-chaining: the policy $\pi^{(i)}(\phi^{(i)} \mid s, b^{(i)})$ and Q-values
 287 $Q^{(i)}([s, b^{(i)}], \phi^{(i)})$ are both defined on the same domain $[s, b^{(i)}]$ and individual actions $\phi^{(i)}$. While
 288 this fact is taken for granted in single-agent RL, this is unlike previous methods in MARL, where
 289 using a centralized value function results in a mismatch: e.g. $Q(s, \vec{a})$ is used to update $\pi^{(i)}(a^{(i)} \mid s)$
 290 for simultaneous policy update methods (Lowe et al., 2017; Yu et al., 2022).

291 We now provide our main theoretical result, that ACPI converges to the optimal policy in both the
 292 AC-BMDP as well as the underlying MMDP.

294 **Theorem 5.4.** (Agent-Chained Policy Iteration) *Starting from any deterministic policy $\vec{\pi} \in \Pi$, the
 295 sequence of value functions $\vec{Q}^{\vec{\pi}_n}$ and the improved policies $\vec{\pi}_{n+1}$ converges to the optimal value
 296 functions and the policy of the AC-BMDP.*

297 *i.e.* $Q^{(i, *)}([s, b^{(i)}], \phi^{(i)}) = \lim_{n \rightarrow \infty} Q^{(i, \vec{\pi}_n)}([s, b^{(i)}], \phi^{(i)}) \geq Q^{(i, \vec{\pi})}([s, b^{(i)}], \phi^{(i)})$ for any
 298 $\vec{\pi}, i, s, b^{(i)}, \phi^{(i)}$. Furthermore the optimal policy of the AC-BMDP is also optimal in the underlying
 299 MMDP.

301 *Proof.* See Appendix C.4. □

303 The proof for Theorem 5.4 makes use of the fact that there is no loss of generality when considering
 304 the space of deterministic action distributions (Corollary C.2). When $\phi^{(i)}$ is deterministic for all
 305 agents, the belief $b^{(i)}$ is also deterministic, and the AC-BMDP will reduce to a serialized version of
 306 the MMDP.

307 The full pseudocode for policy iteration is provided in Algorithm 1 in Appendix E. To the best of
 308 our knowledge, ACPI is the first policy iteration procedure which converges to the globally optimal
 309 policy and naturally extends to practical algorithms in the CTDE setting.

311 Finally, we note one subtle but important difference when comparing ACPI to MA-PI (Algorithm
 312 1 in Zhong et al. (2024)), which serves as the basis for practical Iterative BR approaches, such
 313 as HAPPO and HATRPO. During policy improvement, both MA-PI and ACPI enumerate over all
 314 agents $i \in \{1, \dots, N\}$. However, for MA-PI, the policy needs to be improved in sequence, where $\pi^{(i)}$
 315 can only be updated after $\vec{\pi}^{<i}$ has been updated. On the other hand, the result of policy evaluation
 316 for ACPI is a set of Q-values $\vec{Q} = \langle Q^{(1, \vec{\pi})}, \dots, Q^{(N, \vec{\pi})} \rangle$ for each agent, and policy improvement for
 317 each agent $\pi^{(i)}$ only requires its own $Q^{(i, \vec{\pi})}$. Thus, all agents under ACPI can update their policies
 318 in parallel. This difference will prove to be crucial for developing a practical algorithm (ACPO) in
 319 the following section.

320 6 AGENT-CHAINED POLICY OPTIMIZATION

321 Building on the ACPI derived from an AC-BMDP, we introduce Agent-Chained Policy Optimization
 322 (ACPO), a practical algorithm designed to approximate the optimal policy which maximizes return.

324 There are several ways to approximate the policy iteration procedure and derive a practical al-
 325 gorithm applicable to high-dimensional domains. One such method is to use Proximal Pol-
 326 icy Optimization (PPO) (Schulman et al., 2017) combined with Generalized Advantage Estima-
 327 tion (GAE) (Schulman et al., 2016), which is a popular choice in single-agent domains.

328 Following the Bellman operators in Definition 5.1, the Temporal Difference (TD) residual can be
 329 written as follows:
 330

$$\begin{aligned}\zeta_t^{(i)} &= \gamma' V^{(i+1)}([s_t, b_t^{(i+1)}]) - V^{(i)}([s_t, b_t^{(i)}]), \forall i = \{1, \dots, N-1\} \\ \zeta_t^{(N)} &= R([s_t, b_t^{(N)}], \phi_t^{(N)}) + \gamma' V^{(1)}(s_{t+1}) - V^{(N)}([s_t, b_t^{(N)}]) \\ &\approx r_t + \gamma' V^{(1)}(s_{t+1}) - V^{(N)}([s_t, b_t^{(N)}])\end{aligned}$$

337 where again we have used the fact that the reward R is 0 for any agent $i \in \{1, \dots, N-1\}$.
 338

339 The advantage is defined as the exponentially weighted sum over the TD residuals,

$$A_t^{(i)} = \sum_{j=i}^N (\gamma' \lambda')^{j-i} \zeta_t^{(j)} + \sum_{k=1}^{\infty} \sum_{j=1}^N (\gamma' \lambda')^{kN+j-i} \zeta_{t+k}^{(j)},$$

343 where the detailed derivation is provided in Appendix D.
 344

345 Using the advantage estimates, the PPO objective can be written as a variant of policy gradient with
 346 a clipped probability ratio:
 347

$$\mathcal{L}^{(i)}(\theta) = \mathbb{E}_{\phi_t^{(i)} \sim \pi_{\theta_{old}}^{(i)}(\cdot | s_t, b_t^{(i)})} [\min(w^{(i)}(s_t, b_t^{(i)}, \phi_t^{(i)}) A_t^{(i)}, \text{clip}(w^{(i)}(s_t, b_t^{(i)}, \phi_t^{(i)}), 1 \pm \epsilon) A_t^{(i)})] \quad (2)$$

350 where $w^{(i)}(s_t, b_t^{(i)}, \phi_t^{(i)}) := \pi_{\theta}^{(i)}(\phi_t^{(i)} | s_t, b_t^{(i)}) / \pi_{\theta_{old}}^{(i)}(\phi_t^{(i)} | s_t, b_t^{(i)})$.
 351

352 The objective in Eq. 2 can be further simplified and defined for a policy that produces actions $a^{(i)}$
 353 rather than action distribution $\phi^{(i)}$.
 354

$$\mathcal{L}^{(i)}(\theta) = \mathbb{E}_{a_t^{(i)} \sim \pi_{\theta_{old}}^{(i)}(\cdot | s_t, b_t^{(i)})} [\min(w^{(i)}(s_t, b_t^{(i)}, a_t^{(i)}) A_t^{(i)}, \text{clip}(w^{(i)}(s_t, b_t^{(i)}, a_t^{(i)}), 1 \pm \epsilon) A_t^{(i)})] \quad (3)$$

357 where $w^{(i)}(s_t, b_t^{(i)}, a_t^{(i)}) := \pi_{\theta}^{(i)}(a_t^{(i)} | s_t, b_t^{(i)}) / \pi_{\theta_{old}}^{(i)}(a_t^{(i)} | s_t, b_t^{(i)})$. We provide details on the equiva-
 358 lence between Eq. 2 and Eq. 3 in Appendix F.
 359

360 Overall, we now have a principled PPO-objective derived from an AC-BMDP, with the probability
 361 ratio $w^{(i)}$ defined for each individual policy $\pi^{(i)}$. This is unlike previous work, which contains a
 362 probability ratio that is the product of N policies (HAPPO) or ignores the product, resulting in a
 363 biased objective (MAPPO). For instance, HAPPO and HATRPO contains an importance sampling
 364 ratio defined in the form, $w_{BR}^{(i)}(s, \vec{a}) := \prod_{j=1}^N \pi_{\theta}^{(j)}(a^{(j)} | s) / \prod_{j=1}^N \pi_{\theta_{old}}^{(j)}(a^{(j)} | s)$. However, $w_{BR}^{(i)}$ is
 365 problematic as the importance sampling ratio of a product of policies has variance which scales
 366 exponentially with the number of agents (Wang et al., 2021b). For MAPPO, the correct importance
 367 sampling ratio is also $w_{BR}^{(i)}$. However, MAPPO simply ignores the product, and thus results in a
 368 biased PPO-objective.
 369

370 It is also worth noting that with PPO as the particular choice, ACPO results in a final objective
 371 similar to MAPPO with a few important modifications. Our advantage computation, which uses
 372 agent-chaining, is the most crucial change. In the next section, we show that this modification leads
 373 to a substantial increase in empirical performance compared to MAPPO, especially in complex
 374 domains with many agents.
 375

7 EXPERIMENTAL RESULTS

376 **Environments** We focus our empirical evaluation on Multi-Robot Warehouse (RWARE) (Pa-
 377 pouidakis et al., 2021) which simulates a real-world warehouse environment consisting of multiple

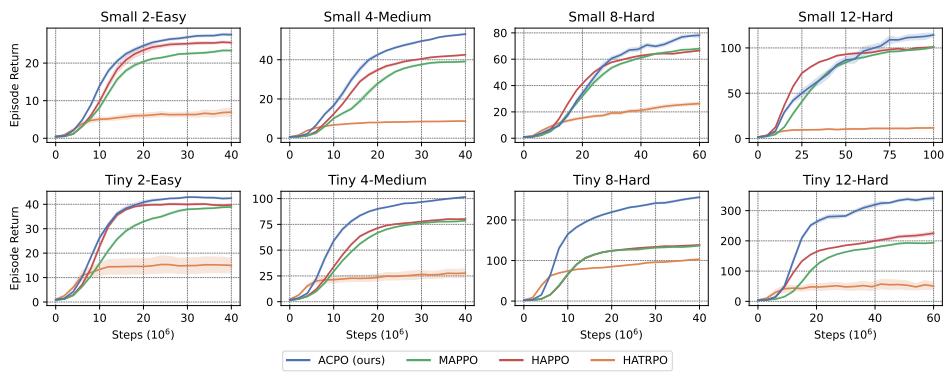


Figure 2: Return for Multi-Robot Warehouse (RWARE) where return is the number of items collected and delivered successfully. The mean and standard error over 10 seeds are reported for all tasks and algorithms except HATRPO and HAPPO on Small 12-Hard (5 seeds) and Tiny 12-Hard (8 seeds).

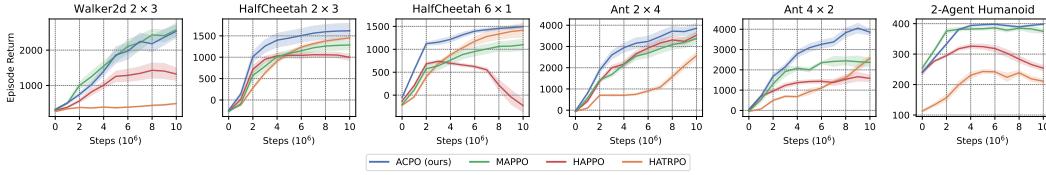


Figure 3: Mean Return and Standard Error over 10 seeds for MA-MuJoCo (Gymnasium).

robots picking up requested shelves and returning them to a designated location. The main challenge in RWARE is *coordination* where the agents must avoid collisions and maximize the number of shelves successfully delivered. We also evaluate our approach on StarCraft Multi-Agent Challenge v2 (SMACv2) (Ellis et al., 2023) and Multi-Agent MuJoCo (Peng et al., 2021), which are popular benchmarks in cooperative MARL with discrete and continuous action spaces, respectively. For SMACv2 and MA-MuJoCo, we closely follow the experimental setup in Zhong et al. (2024).

Baselines Our main baselines for ACPO are MAPPO which represents Simultaneous policy update methods and HAPPO/HATRPO which represent Iterative BR methods.¹ MAPPO and HAPPO are the current state-of-the-art on-policy methods in the three domains we consider. Following the baselines considered in Zhong et al. (2024), we also compare against QMIX (Rashid et al., 2018) an off-policy value-based method for discrete action spaces which shows strong performance in SMACv2. QMIX is a value decomposition method which aims to learn individual utility functions which can be aggregated to represent the underlying Q-values.

For a fair comparison, we use the code² for all baselines provided in MARLLib (Hu et al., 2023), with the same PPO backbone. For all baselines, we use the reported hyperparameters from Papoudakis et al. (2021) for RWARE, Ellis et al. (2023) for SMACv2 and Zhong et al. (2024) for MA-MuJoCo, and only tune appropriate values when it failed to reproduce the reported performance. For MAPPO, we found the reported hyperparameters to be sufficient for reproducing the results. For ACPO, the same hyperparameters as MAPPO are used for all experiments in order to isolate the effect of agent-chaining, and do not conduct any additional hyperparameter tuning specific to ACPO. We provide details on the hyperparameters in Appendix I.

Comparative Evaluation Our main results in Figure 2 show that ACPO outperforms all baselines³ on all tasks in RWARE, despite having the same backbone PPO and the same hyperparameters

¹We do not compare HATRPO on SMACv2 as it exceeds our computational budget for this work. Moreover, the results from Zhong et al. (2024) showed HATRPO had weaker performance in SMACv2 in comparison to HAPPO, MAPPO and QMIX. Details are provided in Appendix J.

²Our anonymous code is available at <https://anonymous.4open.science/r/anonymous-acpo-BD51>.

³We also compared with QMIX (Rashid et al., 2018) on RWARE. However, as QMIX failed to learn any meaningful behavior, we do not report their full results. This is consistent with the failure of QMIX on RWARE reported in Papoudakis et al. (2021).

Table 2: Mean return and standard error over 10 seeds on SMACv2.

	VDN	QMIX	HAPPO	MAPPO	ACPO (Ours)
protoss_5_vs_5	16.20 ± 0.49	16.53 ± 0.55	15.93 ± 0.54	17.03 ± 0.92	18.21 ± 0.46
zerg_5_vs_5	11.77 ± 0.41	14.33 ± 0.63	11.81 ± 0.63	11.84 ± 0.80	15.16 ± 0.98
protoss_10_vs_11	14.74 ± 0.50	14.53 ± 1.06	13.39 ± 0.50	14.57 ± 0.33	15.06 ± 0.63
terran_10_vs_11	12.19 ± 0.60	13.50 ± 0.73	10.57 ± 0.58	12.03 ± 0.50	13.35 ± 0.69
zerg_10_vs_11	13.38 ± 0.63	14.61 ± 0.66	10.77 ± 0.35	12.48 ± 0.52	13.20 ± 0.33

as MAPPO. We also see that the gap widens substantially as the number of agents increases, where the widest gap is seen in 8-agent and 12-agent domains. This provides evidence that ACPO performs substantially better when the environment requires higher levels of coordination. Intuitively, the 12-agent maps require the most coordination among agents since it is the scenario with the most agents crowded in a tight space. Thus, the performance gap jumps even further for the tiny map.

In the results for MA-MuJoCo (Gymnasium) in Figure 3 and SMACv2 in Table 2, ACPO is on par with or outperforms all baselines on all tasks. Notably, ACPO outperforms MAPPO on all tasks with the same hyperparameters, which demonstrates the benefit of agent-chaining. For SMACv2, ACPO outperforms all on-policy baselines, MAPPO and HAPPO. ACPO is also the only on-policy algorithm competitive with QMIX.

Ablation Results We ablate the core component of ACPO, which is the advantage computation based on agent chaining. As shown in Figure 4, the variant ACPO without agent chaining can be interpreted as MAPPO augmented with belief states as additional policy inputs. The performance of this variant remains close to MAPPO, indicating that the observed gains of ACPO are not attributable to the extra input, but rather to the agent-chained advantage computation itself.

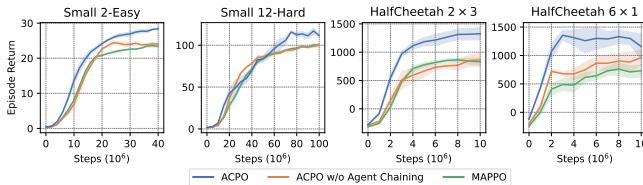


Figure 4: Ablation

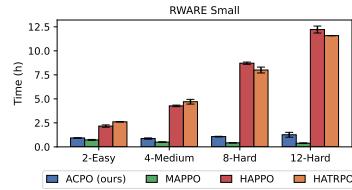


Figure 5: Runtime

Runtime Statistics In Figure 5, we show the wall-clock training time of running MAPPO, HAPPO, HATRPO and ACPO on RWARE for 5M timesteps. With the number of agents increasing from 2 to 12, the runtime of ACPO remains comparable to MAPPO, and is substantially faster than Iterative BR methods such as HAPPO and HATRPO, which require iteratively updating one agent at a time. Overall, ACPO significantly outperforms baselines in terms of return with only minimal additional computational overhead compared to MAPPO.

8 CONCLUSION

In this work, we introduced ACPI, a policy iteration procedure which returns the globally optimal policy for the underlying Multi-Agent MDP (MMDP), as well as ACPO, its practical approximation for Deep MARL. ACPO uses a novel architecture with decentralized critics chained together during centralized training. To the best of our knowledge it is the first algorithm to address the long-standing challenge of going beyond NEs and directly targeting convergence to the globally optimal policy under CTDE. Theoretical insights are reflected in experimental results, where we showed substantial increase in performance, especially in complex domains with many agents.

In single-agent RL, the development of principled yet practical algorithms such as PPO and SAC have led to the wide application of RL in areas such as self-driving cars (Kiran et al., 2022) and Large Language Models (LLMs) (Ouyang et al., 2022), among many others. Similarly, we hope that ACPO forms the basis for better algorithms as well as a wider adoption of MARL to various domains, including Multi-Agent LLMs (Wu et al., 2024; Liu et al., 2025).

486 THE USE OF LARGE LANGUAGE MODELS (LLMs)
487488 In this project, LLMs were used solely as an assist tool for improving the readability of this
489 manuscript. All ideas, proofs, and analyses are by the authors. The authors have verified and edited
490 all content generated by LLMs.
491492 REFERENCES
493

494 Alekh Agarwal, Nan Jiang, and Sham M. Kakade. Reinforcement learning: Theory and algorithms.
495 2019. URL <https://api.semanticscholar.org/CorpusID:148567317>.

496 Stefano V. Albrecht and Peter Stone. Autonomous agents modelling other agents: A comprehensive
497 survey and open problems. *Artificial Intelligence*, 258:66–95, 2018. ISSN 0004-3702. doi:
498 <https://doi.org/10.1016/j.artint.2018.01.002>. URL <https://www.sciencedirect.com/science/article/pii/S0004370218300249>.

500 Christopher Amato. A first introduction to cooperative multi-agent reinforcement learning, 2024.
501 URL <https://arxiv.org/abs/2405.06161>.

503 Dimitri Bertsekas. Multiagent value iteration algorithms in dynamic programming and reinforce-
504 ment learning, 2020. URL <https://arxiv.org/abs/2005.01627>.

505 Dimitri P. Bertsekas. Multiagent rollout algorithms and reinforcement learning. *ArXiv*,
506 abs/1910.00120, 2019. URL <https://api.semanticscholar.org/CorpusID:203610631>.

509 Craig Boutilier. Planning, learning and coordination in multiagent decision processes. In *Proceed-
510 ings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK '96, pp.
511 195–210, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc. ISBN 1558604179.

512 Tianshu Chu, Jie Wang, Lara Codella, and Zhaojian Li. Multi-agent deep reinforcement learning for
513 large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):
514 1086–1095, 2020. doi: 10.1109/TITS.2019.2901791.

515 Allan Dafoe, Edward Hughes, Yoram Bachrach, Tantum Collins, Kevin R. McKee, Joel Z. Leibo,
516 Kate Larson, and Thore Graepel. Open problems in cooperative ai, 2020. URL <https://arxiv.org/abs/2012.08630>.

519 Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan,
520 Jakob Foerster, and Shimon Whiteson. SMACv2: An improved benchmark for coop-
521 erative multi-agent reinforcement learning. In *Thirty-seventh Conference on Neural Information
522 Processing Systems Datasets and Benchmarks Track*, 2023. URL <https://openreview.net/forum?id=50jLGiJW3u>.

524 Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson.
525 Counterfactual multi-agent policy gradients. *Proceedings of the AAAI Conference on Artificial
526 Intelligence*, 32(1), Apr. 2018. doi: 10.1609/aaai.v32i1.11794. URL <https://ojs.aaai.org/index.php/AAAI/article/view/11794>.

528 Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in
529 actor-critic methods. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th Inter-
530 national Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden,
531 July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1582–1591.
532 PMLR, 2018. URL <http://proceedings.mlr.press/v80/fujimoto18a.html>.

533 Carlos Guestrin, Daphne Koller, and Ronald Parr. Multiagent planning with factored mdps. In
534 *Proceedings of the 15th International Conference on Neural Information Processing Systems:
535 Natural and Synthetic*, NIPS'01, pp. 1523–1530, Cambridge, MA, USA, 2001. MIT Press.

537 Siyi Hu, Yifan Zhong, Minquan Gao, Weixun Wang, Hao Dong, Xiaodan Liang, Zhihui Li, Xiaojun
538 Chang, and Yaodong Yang. Marllib: A scalable and efficient multi-agent reinforcement learning
539 library. *Journal of Machine Learning Research*, 24(315):1–23, 2023. URL <http://jmlr.org/papers/v24/23-0378.html>.

540 B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil
 541 Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A sur-
 542 vey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2022. doi:
 543 10.1109/TITS.2021.3054625.

544 Vojtěch Kovařík, Martin Schmid, Neil Burch, Michael Bowling, and Vilim Lisý. Rethink-
 545 ing formal models of partially observable multiagent decision making (extended abstract). In
 546 *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJ-
 547 CAI '23*, 2023. ISBN 978-1-956792-03-4. doi: 10.24963/ijcai.2023/783. URL <https://doi.org/10.24963/ijcai.2023/783>.

548 Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong
 549 Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *International
 550 Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=EcGGFkNTxdJ>.

551 Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. Efficient large-scale fleet management via
 552 multi-agent deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International
 553 Conference on Knowledge Discovery & Data Mining*, KDD '18, pp. 1774–1783, New York,
 554 NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355520. doi: 10.1145/
 555 3219819.3219993. URL <https://doi.org/10.1145/3219819.3219993>.

556 Jiarong Liu, Yifan Zhong, Siyi Hu, Haobo Fu, QIANG FU, Xiaojun Chang, and Yaodong Yang.
 557 Maximum entropy heterogeneous-agent reinforcement learning. In *The Twelfth International
 558 Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=tmqOhBC4a5>.

559 Shuo Liu, Zeyu Liang, Xueguang Lyu, and Christopher Amato. Llm collaboration with multi-agent
 560 reinforcement learning, 2025. URL <https://arxiv.org/abs/2508.04652>.

561 Ryan Lowe, YI WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mor-
 562 datch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Ad-
 563 vances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.,
 564 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/68a9750337a418a86fe06c1991a1d64c-Paper.pdf.

565 Ashutosh Nayyar, Aditya Mahajan, and Demosthenis Teneketzis. Decentralized stochastic control
 566 with partial history sharing: A common information approach. *IEEE Transactions on Automatic
 567 Control*, 58(7):1644–1658, 2013. doi: 10.1109/TAC.2013.2239000.

568 Tianwei Ni, Benjamin Eysenbach, Sergey Levine, and Ruslan Salakhutdinov. Recurrent model-
 569 free RL is a strong baseline for many POMDPs, 2022. URL <https://openreview.net/forum?id=E0zOKxQsZhN>.

570 Tianwei Ni, Michel Ma, Benjamin Eysenbach, and Pierre-Luc Bacon. When do transformers shine
 571 in RL? decoupling memory from credit assignment. In *Thirty-seventh Conference on Neural
 572 Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=APGXBNkt6h>.

573 Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong
 574 Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kel-
 575 ton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike,
 576 and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.
 577 URL <https://arxiv.org/abs/2203.02155>.

578 Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. Benchmarking
 579 multi-agent deep reinforcement learning algorithms in cooperative tasks. In *Thirty-fifth Confer-
 580 ence on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.
 581 URL <https://openreview.net/forum?id=cIrPX-Sn5n>.

582 Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr,
 583 Wendelin Boehmer, and Shimon Whiteson. FACMAC: Factored multi-agent centralised policy

594 gradients. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in*
 595 *Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=wZYWwJvkneF>.

596

597

598 Johan Peralez, Aurélien Delage, Jacopo Castellini, Rafael F. Cunha, and Jilles Steeve Dibangoye.
 599 Optimally solving simultaneous-move dec-pomdps: The sequential central planning approach. In
 600 *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February*
 601 *25 - March 4, 2025, Philadelphia, PA, USA*, pp. 23276–23285. AAAI Press, 2025. doi: 10.1609/
 602 *AAAI.V39I22.34494*. URL <https://doi.org/10.1609/aaai.v39i22.34494>.

603

604 Aswin Raghavan, Saket Joshi, Alan Fern, Prasad Tadepallia, and Roni Khardonb. Planning in fac-
 605 tored action spaces with symbolic dynamic programming. In *Proceedings of the Twenty-Sixth*
 606 *AAAI Conference on Artificial Intelligence*, AAAI’12, pp. 1802–1808. AAAI Press, 2012.

607

608 Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and
 609 Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent rein-
 610 forcement learning. In *Proceedings of the 35th International Conference on Machine Learning*,
 611 volume 80 of *Proceedings of Machine Learning Research*, pp. 4295–4304. PMLR, 10–15 Jul
 612 2018. URL <https://proceedings.mlr.press/v80/rashid18a.html>.

613

614 John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region
 615 policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*,
 616 volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–
 617 09 Jul 2015. PMLR. URL <https://proceedings.mlr.press/v37/schulman15.html>.

618

619 John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-
 620 dimensional continuous control using generalized advantage estimation. In Yoshua Bengio and
 621 Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016,*
 622 *San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL <http://arxiv.org/abs/1506.02438>.

623

624 John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy
 625 optimization algorithms, 2017. URL <https://arxiv.org/abs/1707.06347>.

626

627 Jiangxing Wang, Deheng Ye, and Zongqing Lu. More centralized training, still decentralized ex-
 628 ecution: Multi-agent conditional policy factorization. In *The Eleventh International Confer-
 629 ence on Learning Representations*, 2023. URL <https://openreview.net/forum?id=znLlSgN-4S0>.

630

631 Jianhong Wang, Wangkun Xu, Yunjie Gu, Wenbin Song, and Tim C. Green. Multi-agent reinfor-
 632 cements learning for active voltage control on power distribution networks. In *Proceedings of the*
 633 *35th International Conference on Neural Information Processing Systems*, NIPS ’21, Red Hook,
 634 NY, USA, 2021a. Curran Associates Inc. ISBN 9781713845393.

635

636 Li Wang, Yupeng Zhang, Yujing Hu, Weixun Wang, Chongjie Zhang, Yang Gao, Jianye Hao,
 637 Tangjie Lv, and Changjie Fan. Individual reward assisted multi-agent reinforcement learning.
 638 In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Pro-
 639 ceedings of Machine Learning Research*, pp. 23417–23432. PMLR, 17–23 Jul 2022. URL
 640 <https://proceedings.mlr.press/v162/wang22ao.html>.

641

642 Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. {DOP}: Off-policy
 643 multi-agent decomposed policy gradients. In *International Conference on Learning Representa-
 644 tions*, 2021b. URL <https://openreview.net/forum?id=6FqKiVAdI3Y>.

645

646 Muning Wen, Jakub Grudzien Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong
 647 Yang. Multi-agent reinforcement learning is a sequence modeling problem. In *Proceedings of the*
 648 *36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook,
 649 NY, USA, 2022. ISBN 9781713871088.

650

651 Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun
 652 Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and

648 Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First*
649 *Conference on Language Modeling*, 2024. URL <https://openreview.net/forum?id=BAakY1hNKS>.
650

651 Jianing Ye, Chenghao Li, Jianhao Wang, and Chongjie Zhang. Towards global optimality in
652 cooperative marl with the transformation and distillation framework, 2023. URL <https://arxiv.org/abs/2207.11143>.
653

654 Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The
655 surprising effectiveness of PPO in cooperative multi-agent games. In *Thirty-sixth Conference on*
656 *Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL <https://openreview.net/forum?id=YVXaxB6L2Pl>.
657

658 Tianhao Zhang, Yueheng Li, Chen Wang, Guangming Xie, and Zongqing Lu. Fop: Factorizing
659 optimal joint policy of maximum-entropy multi-agent reinforcement learning. In *Proceedings*
660 *of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings*
661 *of Machine Learning Research*, pp. 12491–12500. PMLR, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/zhang21m.html>.
662

663 Yifan Zhong, Jakub Grudzien Kuba, Xidong Feng, Siyi Hu, Jiaming Ji, and Yaodong Yang.
664 Heterogeneous-agent reinforcement learning. *Journal of Machine Learning Research*, 25(32):
665 1–67, 2024. URL <http://jmlr.org/papers/v25/23-0488.html>.
666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702 **A BELIEF UPDATE**
 703

704 $\tau([s_t, b_t^{(i)}], \phi_t^{(i)}) (\vec{a}_t^{<i+1}) = \frac{1}{\eta([s_t, b_t^{(i)}], \phi_t^{(i)})} \sum_{\vec{a}_t^{<i}} b_t^{(i)}(\vec{a}_t^{<i}) T([s_t, \vec{a}_t^{<i+1}] | [s_t, \vec{a}_t^{<i}], \phi_t^{(i)})$
 705
 706
 707 $= \frac{1}{\eta([s_t, b_t^{(i)}], \phi_t^{(i)})} \sum_{\vec{a}_t^{<i}} b_t^{(i)}(\vec{a}_t^{<i}) \sum_{a_t^{(i)}} \phi_t^{(i)}(a_t^{(i)}) T([s_t, \vec{a}_t^{<i+1}] | [s_t, \vec{a}_t^{<i}], a_t^{(i)})$
 708
 709

710 Finally, η is the normalization factor defined as

711 $\eta([s_t, b_t^{(i)}], \phi_t^{(i)}) = \sum_{\vec{a}_t^{<i+1}} \sum_{\vec{a}_t^{<i}} b_t^{(i)}(\vec{a}_t^{<i}) \sum_{a_t^{(i)}} \phi_t^{(i)}(a_t^{(i)}) T([s_t, \vec{a}_t^{<i+1}] | [s_t, \vec{a}_t^{<i}], a_t^{(i)})$
 712
 713

714 **B PROOF FOR THEOREM 3.1**
 715

716 The serialized multi-agent problem can be described as follows:
 717

718

- 719 • Action space $A^{(i)}$ of individual actions for some $i \in \mathcal{N}$
- 720 • States $[s_t, \vec{a}_t^{<i}]$ which augments the original state space with actions selected by previous
 721 agents $\vec{a}_t^{<i}$.
- 722 • Reward function

723 $R([s_t, \vec{a}_t^{<i}], a_t^{(i)}) = \begin{cases} R(s_t, \vec{a}_t) & \text{if } i = N \\ 0 & \text{otherwise} \end{cases}$
 724
 725

726 • Transition function

727 $T([s_{t+1}, \emptyset] | [s_t, \vec{a}_t^{<i}], a_t^{(i)}) = T(s_{t+1} | s_t, \vec{a}_t)$ if $i = N$
 728
 729 $T([s_t, \vec{a}_t^{<i+1}] | [s_t, \vec{a}_t^{<i}], a_t^{(i)})$
 730 $= \mathbb{I}\{[\vec{a}_t^{<i}, a_t^{(i)}] = \vec{a}_t^{<i+1}\}$ if $i \in \{1, \dots, N-1\}$
 731

732 • Discount factor γ' where $\gamma' = \gamma^{1/N}$

733 **Theorem 3.1.** (Peralez et al., 2025) For every MMDP, there exists a serialized multi-agent problem,
 734 of which its optimal policy is also optimal for the underlying MMDP.
 735

736 *Proof.* Let $\vec{\pi}^* = \langle \pi^{(1,*)}, \dots, \pi^{(N,*)} \rangle$ be the optimal policy over the serialized MMDP.
 737

738 $V^{(i,*)}([s, \vec{a}^{<i}])$
 739

740 $= \mathbb{E} \left[\sum_{j=i}^N (\gamma')^{j-i} R(s_0, \vec{a}_0^{<j}, a_0^{(j)}) \mid s_0 = s, \vec{a}_0^{<i} = \vec{a}^{<i}, \vec{\pi}^* \right] + \mathbb{E} \left[\sum_{t=1}^N \sum_{j=1}^N (\gamma')^{tN+j-i} R(s_t, \vec{a}_t^{<j}, a_t^{(j)}) \mid \vec{\pi}^* \right]$
 741
 742
 743 $= \mathbb{E} \left[(\gamma')^{N-i} R(s_0, \vec{a}_0^{<N}, a_0^{(N)}) \mid s_0 = s, \vec{a}_0^{<i} = \vec{a}^{<i}, \vec{\pi}^* \right] + \mathbb{E} \left[\sum_{t=1}^N (\gamma')^{tN+N-i} R(s_t, \vec{a}_t^{<N}, a_t^{(N)}) \mid \vec{\pi}^* \right]$
 744
 745
 746 $= \mathbb{E} \left[(\gamma')^{N-i} R(s_0, \vec{a}_0^{<N}, a_0^{(N)}) \mid s_0 = s, \vec{a}_0^{<i} = \vec{a}^{<i}, \vec{\pi}^* \right] + \mathbb{E} \left[\sum_{t=1}^N (\gamma')^{N-i} \gamma^t R(s_t, \vec{a}_t^{<N}, a_t^{(N)}) \mid \vec{\pi}^* \right]$
 747
 748
 749 $= \mathbb{E} \left[\sum_{t=0}^N (\gamma')^{N-i} \gamma^t R(s_t, \vec{a}_t^{<N}, a_t^{(N)}) \mid s_0 = s, \vec{a}_0^{<i} = \vec{a}^{<i}, \vec{\pi}^* \right]$
 750
 751
 752 $= (\gamma')^{N-i} \mathbb{E} \left[\sum_{t=0}^N \gamma^t R(s_t, \vec{a}_t^{<N}, a_t^{(N)}) \mid s_0 = s, \vec{a}_0^{<i} = \vec{a}^{<i}, \vec{\pi}^* \right]$
 753
 754

(4)

755 $V^{(1,*)}(s) = (\gamma')^{N-1} \mathbb{E} \left[\sum_{t=0}^N \gamma^t R(s_t, \vec{a}_t^{<N}, a_t^{(N)}) \mid \vec{\pi}^* \right]$ for agent 1.

756 $V^{(N,*)}([s, \vec{a}^{<N}]) = \mathbb{E} \left[\sum_{t=0} \gamma^t R \left(s_t, \vec{a}_t^{<N}, a_t^{(N)} \right) \mid s_0 = s, \vec{a}_0^{<N} = \vec{a}^{<N}, \vec{\pi}^* \right]$ for agent N .
 757

758 Thus, we have established that the optimal policy $\vec{\pi}^*$ in the serialized problem will obtain the same
 759 expected value in the MMDP (times a constant factor). The same holds for any policy $\vec{\pi}$, and thus
 760 there is a 1-1 mapping between serialized and simultaneous policies which yield the same value.
 761 \square
 762

763 **C PROOFS FOR POLICY ITERATION CONVERGENCE**
 764

765 **C.1 PROOF FOR LEMMA 5.2**
 766

767 **Lemma 5.2.** *(Agent-Chained Policy Evaluation) The Agent-Chained Bellman Operators in Def-
 768 ition 5.1 are a contraction mapping under the infinity norm. Thus, starting with any $\vec{Q} =$
 769 $\langle Q^{(1)}, \dots, Q^{(N)} \rangle$ and a joint policy $\vec{\pi} = \langle \pi^{(1)}, \dots, \pi^{(N)} \rangle$, the repeated application of $\mathcal{T}^{\vec{\pi}}$ will return
 770 a set of Q -values for each agent $\langle Q^{(1, \vec{\pi})}, \dots, Q^{(N, \vec{\pi})} \rangle$ in the limit.*

772 *Proof.* First, note that we can view $\langle Q^{(1)}, \dots, Q^{(N)} \rangle$ as a single Q-function with the state space
 773 further augmented by agent ID. Under this perspective, we now have a single policy denoted as π
 774 and a corresponding value function Q^π , defined on the AC-BMDP.
 775

776 Since the AC-BMDP is a single-agent Belief MDP, the rest follows standard convergence results of
 777 policy evaluation (Agarwal et al., 2019), which we include for completeness.
 778

779 For any agent $i \in \{1, \dots, N-1\}$, state $[s, b^{(i)}, i]$, action $\phi^{(i)}$ and arbitrary Q-values Q_1, Q_2 ,

$$\begin{aligned} & \left| \mathcal{T}^\pi Q_1([s, b^{(i)}, i], \phi^{(i)}) - \mathcal{T}^\pi Q_2([s, b^{(i)}, i], \phi^{(i)}) \right| \\ &= \left| \mathbb{E}_{\substack{[s, b^{(i+1)}, i+1] = T([s, b^{(i)}, i], \phi^{(i)}) \\ \phi^{(i+1)} \sim \pi(\cdot | s, b^{(i+1)}, i+1)}} \left[\gamma' Q_1([s, b^{(i+1)}, i+1], \phi^{(i+1)}) - \gamma' Q_2([s, b^{(i+1)}, i+1], \phi^{(i+1)}) \right] \right| \\ &\leq \gamma' \max_{\phi^{(i+1)}} \left| Q_1([s, b^{(i+1)}, i+1], \phi^{(i+1)}) - Q_2([s, b^{(i+1)}, i+1], \phi^{(i+1)}) \right| \\ &\leq \gamma' \max_{\substack{\phi^{(i+1)}, b^{(i+1)}, j \in \{1, \dots, N\}}} \left| Q_1([s, b^{(i+1)}, j], \phi^{(i+1)}) - Q_2([s, b^{(i+1)}, j], \phi^{(i+1)}) \right| \end{aligned}$$

789 For agent N ,

$$\begin{aligned} & \left| \mathcal{T}^\pi Q_1([s, b^{(N)}, N], \phi^{(N)}) - \mathcal{T}^\pi Q_2([s, b^{(N)}, N], \phi^{(N)}) \right| \\ &= \left| \mathbb{E}_{\substack{[s', 1] \sim T(\cdot | [s, b^{(N)}, N], \phi^{(N)}) \\ \phi^{(1)} \sim \pi(\cdot | s', 1)}} \left[\gamma' Q_1([s', 1], \phi^{(1)}) - \gamma' Q_2([s', 1], \phi^{(1)}) \right] \right| \\ &\leq \gamma' \max_{s', \phi^{(1)}} \left| Q_1([s', 1], \phi^{(1)}) - Q_2([s', 1], \phi^{(1)}) \right| \\ &\leq \gamma' \max_{\substack{s', \phi^{(1)}, j \in \{1, \dots, N\}}} \left| Q_1([s', j], \phi^{(1)}) - Q_2([s', j], \phi^{(1)}) \right| \end{aligned}$$

800 Thus, \mathcal{T}^π is a contraction mapping under the infinity norm, i.e. there exists $\gamma' \in [0, 1)$ such that
 801

802 $\|\mathcal{T}^\pi Q_1 - \mathcal{T}^\pi Q_2\|_\infty \leq \gamma' \|Q_1 - Q_2\|_\infty$
 803

804 Since \mathcal{T}^π is a contraction mapping, we have the following:
 805

$$\begin{aligned} \|Q_k - Q^\pi\|_\infty &= \|\mathcal{T}^\pi Q_{k-1} - \mathcal{T}^\pi Q^\pi\|_\infty \\ &\leq \gamma' \|Q_{k-1} - Q^\pi\|_\infty \\ &\vdots \\ &\leq (\gamma')^k \|Q_0 - Q^\pi\|_\infty \end{aligned}$$

If we let $k \rightarrow \infty$, $\|Q_k - Q^\pi\|_\infty = 0$, and $\lim_{k \rightarrow \infty} Q_k = Q^\pi$. By the Banach fixed-point theorem, this solution is unique.

□

C.2 PROOF FOR LEMMA 5.3

Lemma 5.3. *(Agent-Chained Policy Improvement) Given a deterministic policy $\vec{\pi} = \langle \pi^{(1)}, \dots, \pi^{(N)} \rangle$, let $Q^{(i, \vec{\pi})}$ denote the i -th agent's value function for a joint policy $\vec{\pi}$. If we update the new policy $\vec{\pi}_{new} = \langle \pi_{new}^{(1)}, \dots, \pi_{new}^{(N)} \rangle$ by Eq. 1, then*

$$Q^{(i, \vec{\pi}_{new})}(s, b^{(i)}, \phi^{(i)}) \geq Q^{(i, \vec{\pi})}([s, b^{(i)}], \phi^{(i)})$$

Proof. As in the proof for Lemma 5.2, we consider $\vec{\pi}$ to be a single policy π which is augmented by agent ID in the state space.

For any $i \in \{1, \dots, N-1\}$, $s, b^{(i)}, \phi^{(i)}$,

$$\begin{aligned} Q^\pi([s, b^{(i)}, i], \phi^{(i)}) &= \gamma' \mathbb{E}_{\substack{[s, b^{(i+1)}, i+1] = T([s, b^{(i)}, i], \phi^{(i)}) \\ \phi^{(i+1)} \sim \pi(\cdot | [s, b^{(i+1)}, i+1])}} \left[Q^\pi \left([s, b^{(i+1)}, i+1], \phi^{(i+1)} \right) \right] \\ &\leq \gamma' \mathbb{E}_{\substack{[s, b^{(i+1)}, i+1] = T([s, b^{(i)}, i], \phi^{(i)}) \\ \phi^{(i+1)} \sim \pi_{new}(\cdot | [s, b^{(i+1)}, i+1])}} \left[\max_{\phi^{(i+1)}} Q^\pi \left([s, b^{(i+1)}, i+1], \phi^{(i+1)} \right) \right] \\ &= \gamma' \mathbb{E}_{\substack{[s, b^{(i+1)}, i+1] = T([s, b^{(i)}, i], \phi^{(i)}) \\ \phi^{(i+1)} \sim \pi_{new}(\cdot | [s, b^{(i+1)}, i+1])}} \left[Q^\pi \left([s, b^{(i+1)}, i+1], \phi^{(i+1)} \right) \right] \end{aligned}$$

For $i = N$ and any $s, b^{(i)}, \phi^{(i)}$,

$$\begin{aligned} Q^\pi([s, b^{(N)}, N], \phi^{(N)}) &= R([s, b^{(N)}, N], \phi^{(N)}) + \gamma' \mathbb{E}_{\substack{[s', 1] = T([s, b^{(N)}, N], \phi^{(N)}) \\ \phi^{(1)} \sim \pi(\cdot | [s', 1])}} \left[Q^\pi \left([s', 1], \phi^{(1)} \right) \right] \\ &\leq R([s, b^{(N)}, N], \phi^{(N)}) + \gamma' \mathbb{E}_{\substack{[s', 1] = T([s, b^{(N)}, N], \phi^{(N)}) \\ \phi^{(1)} \sim \pi_{new}(\cdot | [s', 1])}} \left[\max_{\phi^{(1)}} Q^\pi \left([s', 1], \phi^{(1)} \right) \right] \\ &= R([s, b^{(N)}, N], \phi^{(N)}) + \gamma' \mathbb{E}_{\substack{[s', 1] = T([s, b^{(N)}, N], \phi^{(N)}) \\ \phi^{(1)} \sim \pi_{new}(\cdot | [s', 1])}} \left[Q^\pi \left([s', 1], \phi^{(1)} \right) \right] \end{aligned}$$

Thus, for any $i \in \{1, \dots, N\}$, $s, b^{(i)}, \phi^{(i)}$,

$$\begin{aligned} Q^\pi([s, b^{(i)}, i], \phi^{(i)}) &\leq R([s, b^{(i)}, i], \phi^{(i)}) + \gamma' \mathbb{E}_{\substack{[s, b^{(i+1)}, i+1] = T([s, b^{(i)}, i], \phi^{(i)}) \\ \phi^{(i+1)} \sim \pi_{new}(\cdot | [s, b^{(i+1)}, i+1])}} \left[Q^\pi \left([s, b^{(i+1)}, i+1], \phi^{(i+1)} \right) \right] \\ &\vdots \\ &\leq Q^{\pi_{new}}([s, b^{(i)}, i], \phi^{(i)}) \end{aligned}$$

□

C.3 CHARACTERIZATION OF $Q^{(i,*)}$

Before our main result, we provide a useful property of the optimal Q-values of the AC-BMDP, which has an affine structure.

Theorem C.1. *Let $Q^{(i,*)}$ denote the i -th agent's Q-values for the optimal policy under an AC-BMDP. Then, $Q^{(i,*)}$ is an affine function of $\phi^{(i)}$. i.e. for all $i \in \{1, \dots, N\}$, $s, b^{(i)}, \phi^{(i)}$,*

$$Q^{(i,*)}([s, b^{(i)}], \phi^{(i)}) = \sum_{a^{(i)}} \phi^{(i)}(a^{(i)}) Q^{(i,*)}([s, b^{(i)}], \delta_{a^{(i)}})$$

where $\delta_{a^{(i)}}$ denotes a particular action distribution which deterministically selects $a^{(i)}$.

864 *Proof.* We prove the claim by induction.

865 For any $s, b^{(N)}, \phi^{(N)}$ at terminal timesteps,

$$867 \quad 868 \quad Q^{(N)}([s, b^{(N)}], \delta_{a^{(N)}}) = R([s, b^{(N)}], a^{(N)})$$

$$870 \quad 871 \quad Q^{(N)}([s, b^{(N)}], \phi^{(N)}) = R([s, b^{(N)}], \phi^{(N)}) \\ 872 \quad = \sum_{a^{(N)}} \phi^{(N)}(a^{(N)}) R([s, b^{(N)}], a^{(N)}) \\ 873 \quad = \sum_{a^{(N)}} \phi^{(N)}(a^{(N)}) Q^{(N)}([s, b^{(N)}], \delta_{a^{(N)}})$$

$$877 \quad Q^{(N)}([s, b^{(N)} = \vec{a}^{<N}], \phi^{(N)}) = R([s, \vec{a}^{<N}], \phi^{(N)})$$

$$879 \quad 880 \quad Q^{(N)}([s, b^{(N)}], \phi^{(N)}) = R([s, b^{(N)}], \phi^{(N)}) \\ 881 \quad = \sum_{\vec{a}^{<N}} b^{(N)}(\vec{a}^{<N}) R([s, \vec{a}^{<N}], \phi^{(N)}) \\ 882 \quad = \sum_{\vec{a}^{<N}} b^{(N)}(\vec{a}^{<N}) Q^{(N)}([s, \vec{a}^{<N}], \phi^{(N)})$$

886 where $\vec{a}^{<N}$ in $Q^{(N)}([s, \vec{a}^{<N}], \phi^{(N)})$ denotes the belief that assigns probability 1 to $\vec{a}^{<N}$.

887 For agents $i \in \{1, \dots, N-1\}$,

$$890 \quad Q^*([s, b^{(i)}], \delta_{a^{(i)}}) = \gamma' \max_{\phi^{(i+1)}} Q^*([s, b^{(i+1)}], \phi^{(i+1)}) \\ 891 \quad = \gamma' \max_{\phi^{(i+1)}} \sum_{\vec{a}^{<i+1}} b^{(i+1)}(\vec{a}^{<i+1}) Q^*([s, \vec{a}^{<i+1}], \phi^{(i+1)}) \text{ (by induction)} \\ 892 \quad = \gamma' \max_{\phi^{(i+1)}} \frac{1}{\eta([s, b^{(i)}], \phi^{(i)})} \sum_{\vec{a}^{<i}} b^{(i)}(\vec{a}^{<i}) Q^*([s, \vec{a}^{<i}, a^{(i)}], \phi^{(i+1)})$$

893 where $[s, b^{(i+1)}] = T([s, b^{(i)}], \delta_{a^{(i)}})$.

$$894 \quad Q^*([s, b^{(i)}], \phi^{(i)}) = \gamma' \max_{\phi^{(i+1)}} Q^*([s, b^{(i+1)}], \phi^{(i+1)}) \\ 895 \quad = \gamma' \max_{\phi^{(i+1)}} \sum_{\vec{a}^{<i+1}} b^{(i+1)}(\vec{a}^{<i+1}) Q^*([s, \vec{a}^{<i+1}], \phi^{(i+1)}) \\ 896 \quad = \gamma' \max_{\phi^{(i+1)}} \frac{1}{\eta([s, b^{(i)}], \phi^{(i)})} \sum_{\vec{a}^{<i}} b^{(i)}(\vec{a}^{<i}) \sum_{a^{(i)}} \phi^{(i)}(a^{(i)}) Q^*([s, \vec{a}^{<i}, a^{(i)}], \phi^{(i+1)}) \\ 897 \quad = \gamma' \max_{\phi^{(i+1)}} \frac{1}{\eta([s, b^{(i)}], \phi^{(i)})} \sum_{\vec{a}^{<i}} b^{(i)}(\vec{a}^{<i}) Q^*([s, \vec{a}^{<i}, a^{(i)}], \phi^{(i+1)}) \\ 898 \quad = \sum_{a^{(i)}} \phi^{(i)}(a^{(i)}) Q^*([s, b^{(i)}], \delta_{a^{(i)}})$$

899 where $[s, b^{(i+1)}] = T([s, b^{(i)}], \phi^{(i)})$. □

900 A direct corollary of Theorem C.1 is that only the actions $a^{(i)}$ need to be enumerated rather than the full space of 1-step policies $\phi^{(i)}$.

901 **Corollary C.2.** For all $i \in \{1, \dots, N\}$, $s, b^{(i)}$,

$$902 \quad \max_{\phi^{(i)}} Q^{(i,*)} \left([s, b^{(i)}], \phi^{(i)} \right) = \max_{a^{(i)}} Q^{(i,*)} \left([s, b^{(i)}], \delta_{a^{(i)}} \right)$$

918 C.4 PROOF FOR THEOREM 5.4
919920 **Theorem 5.4.** *(Agent-Chained Policy Iteration) Starting from any deterministic policy $\vec{\pi} \in \Pi$, the*
921 *sequence of value functions $\vec{Q}^{\vec{\pi}_n}$ and the improved policies $\vec{\pi}_{n+1}$ converges to the optimal value*
922 *functions and the policy of the AC-BMDP.*923 *i.e. $Q^{(i,*)}([s, b^{(i)}], \phi^{(i)}) = \lim_{n \rightarrow \infty} Q^{(i, \vec{\pi}_n)}([s, b^{(i)}], \phi^{(i)}) \geq Q^{(i, \vec{\pi})}([s, b^{(i)}], \phi^{(i)})$ for any*
924 *$\vec{\pi}, i, s, b^{(i)}, \phi^{(i)}$. Furthermore the optimal policy of the AC-BMDP is also optimal in the underlying*
925 *MMDP.*927 *Proof.* By the monotonic improvement property in Lemma 5.3, we know that for any $i \in$
928 $\{1, \dots, N\}, b^{(i)}, s, \phi^{(i)}$,

930
$$Q^{\vec{\pi}_{n+1}}([s, b^{(i)}], \phi^{(i)}) \geq Q^{\vec{\pi}_n}([s, b^{(i)}], \phi^{(i)})$$

931
932 .

933 If there is no improvement,

934
$$\begin{aligned} Q^{\vec{\pi}_n}([s, b^{(i)}], \phi^{(i)}) &= Q^{\vec{\pi}_{n+1}}([s, b^{(i)}], \phi^{(i)}) \\ &= \gamma' Q^{\vec{\pi}_{n+1}}([s, b^{(i+1)}], \pi_{n+1}^{(i)}([s, b^{(i+1)}])) \\ &= \gamma' Q^{\vec{\pi}_n}([s, b^{(i+1)}], \pi_{n+1}^{(i)}([s, b^{(i+1)}])) \\ &= \gamma' \max_{\phi^{(i+1)}} Q^{\vec{\pi}_n}([s, b^{(i+1)}], \phi^{(i+1)}) \end{aligned}$$

935
936
937
938
939
940

941 where $[s, b^{(i+1)}] = T([s, b^{(i)}], \phi^{(i)})$. Thus, at the limit $\lim_{n \rightarrow \infty} Q^{\vec{\pi}_n}([s, b^{(i)}], \phi^{(i)})$, the Bellman
942 optimality equations are satisfied.
943944 Due to Corollary C.2, it is sufficient to consider the following policy improvement procedure con-
945 sidering only the $\delta_{a^{(i)}}$, which is the space of deterministic $\phi^{(i)}$:

946
$$\forall i, b^{(i)}, s, \pi_{new}^{(i)}([s, b^{(i)}]) \leftarrow \arg \max_{a^{(i)}} Q^{\vec{\pi}}([s, b^{(i)}], \delta_{a^{(i)}}) \quad (5)$$

947
948

949 Note that if we restrict ourselves to an AC-BMDP defined over the space of deterministic 1-step
950 policies $\delta_{a^{(i)}}$, all of the components in the AC-BMDP are equivalent to that of the serialized version
951 of the MMDP. \square
952953 D DETAILS ON ADVANTAGE COMPUTATION
954955 The advantage can be written as an exponentially-weighted sum over the TD residuals,
956

957
$$A_t^{(1)} = \sum_{j=1}^N (\gamma' \lambda')^{j-1} \zeta_t^{(j)} + \sum_{k=1}^{\infty} \sum_{j=1}^N (\gamma' \lambda')^{kN+j-1} \zeta_{t+k}^{(j)}$$

958
959
960

961
$$A_t^{(2)} = \sum_{j=2}^N (\gamma' \lambda')^{j-2} \zeta_t^{(j)} + \sum_{k=1}^{\infty} \sum_{j=1}^N (\gamma' \lambda')^{kN+j-2} \zeta_{t+k}^{(j)}$$

962
963
964

965 \vdots
966

967
$$A_t^{(N)} = \zeta_t^{(N)} + \sum_{k=1}^{\infty} \sum_{j=1}^N (\gamma' \lambda')^{kN+j-N} \zeta_{t+k}^{(j)}$$

968
969

970
$$\therefore A_t^{(i)} = \sum_{j=i}^N (\gamma' \lambda')^{j-i} \zeta_t^{(j)} + \sum_{k=1}^{\infty} \sum_{j=1}^N (\gamma' \lambda')^{kN+j-i} \zeta_{t+k}^{(j)}.$$

971

972 E PSEUDOCODES
973974 We first present the pseudocode for ACPI which is defined directly on the action space consisting of
975 1-step policies. $\phi^{(i)}$. This is a straightforward policy iteration procedure defined on the AC-BMDP.
976977 **Algorithm 1** Agent-Chained Policy Iteration
978

```

979 1: Randomly initialize  $\vec{\pi} = (\pi^{(1)}, \dots, \pi^{(N)})$  and  $\vec{Q} = (Q^{(1)}, \dots, Q^{(N)})$ .
980 2: while  $\vec{\pi}$  not converged do
981 3:   while  $\vec{Q}$  not converged do
982 4:     # Policy Evaluation
983 5:        $\forall i \in \{1, \dots, N-1\}, s, b^{(i)}, \phi^{(i)},$ 
984 6:          $Q^{(i)}([s, b^{(i)}], \phi^{(i)}) \leftarrow \gamma' \mathbb{E}_{\substack{b^{(i+1)}=T([s, b^{(i)}], \phi^{(i)}) \\ \phi^{(i+1)} \sim \pi^{(i+1)}(\cdot | s, b^{(i+1)})}} \left[ Q^{(i+1)}([s, b^{(i+1)}], \phi^{(i+1)}) \right]$ 
985 7:   end while
986 8:   # Policy Improvement
987 9:      $\forall s, b^{(N)}, \phi^{(N)},$ 
988 10:     $Q^{(N)}([s, b^{(N)}], \phi^{(N)})$ 
989 11:     $\leftarrow R([s, b^{(N)}], \phi^{(N)}) + \gamma' \mathbb{E}_{\substack{s' \sim T(\cdot | [s, b^{(N)}], \phi^{(N)}) \\ \phi^{(1)} \sim \pi^{(1)}(\cdot | s')}} \left[ Q^{(1)}(s', \phi^{(1)}) \right]$ 
990 12: 10: end while
991 13: 11: end while
992 14: 12: end while
993 15: 13: end while
994 16: 14: end while
995 17: 15: end while
996 18: 16: end while
997 19: 17: end while
998 20: 18: end while
999 21: 19: end while
1000 22: 20: end while

```

1001 As we showed in Corollary C.2, the AC-BMDP has a special structure which ensures that it is
1002 sufficient to consider the space of deterministic $\phi^{(i)}$ for finding an optimal policy. Thus, we can also
1003 define an equivalent policy iteration procedure over the action space $\delta_{a^{(i)}}$ (Algorithm 2).1004 Finally, we present ACPO which is a practical algorithm that aims to approximate ACPI via the PPO
1005 objective.

1006 We reiterate below the definition of the TD residual.

1007
1008

$$\zeta_t^{(i)} = \gamma' V^{(i+1)}([s_t, b_t^{(i+1)}]) - V^{(i)}([s_t, b_t^{(i)}]), \forall i = \{1, \dots, N-1\}$$
1009
1010
1011

1012
1013

$$\zeta_t^{(N)} = R([s_t, b_t^{(N)}], \phi_t^{(N)}) + \gamma' V^{(1)}(s_{t+1}) - V^{(N)}([s_t, b_t^{(N)}])$$
1014
1015
1016

1017
1018
1019
1020
1021
1022
1023
1024
1025

1026 **Algorithm 2** Agent-Chained Policy Iteration (Deterministic action distribution $\phi^{(i)}$)
1027

1028 1: Randomly initialize $\vec{\pi} = (\pi^{(1)}, \dots, \pi^{(N)})$ and $\vec{Q} = (Q^{(1)}, \dots, Q^{(N)})$.
1029 2: **while** $\vec{\pi}$ not converged **do**
1030 3: **while** \vec{Q} not converged **do**
1031 4: # Policy Evaluation
1032 5:
1033 $\forall i \in \{1, \dots, N-1\}, s, \vec{a}^{<i}, a^{(i)},$
1034
$$Q^{(i)}([s, b^{(i)} = \vec{a}^{<i}], \delta_{a^{(i)}}) \leftarrow \gamma' \mathbb{E}_{b^{(i+1)} = T([s, b^{(i)} = \vec{a}^{<i}], \delta_{a^{(i)}})} \left[Q^{(i+1)}([s, b^{(i+1)}], \delta_{a^{(i+1)}}) \right]$$

1035
$$\delta_{a^{(i+1)}} \sim \pi^{(i+1)}(\cdot | s, b^{(i+1)})$$

1036 6:
1037 $\forall s, \vec{a}^{<N}, a^{(N)},$
1038
$$Q^{(N)}([s, b^{(N)} = \vec{a}^{<N}], \delta_{a^{(N)}})$$

1039
$$\leftarrow R([s, \vec{a}^{<N}], a^{(N)}) + \gamma' \mathbb{E}_{s' \sim T(\cdot | [s, b^{(N)} = \vec{a}^{<N}], \delta_{a^{(N)}})} \left[Q^{(1)}(s', \delta_{a^{(1)}}) \right]$$

1040
$$\delta_{a^{(1)}} \sim \pi^{(1)}(\cdot | s')$$

1041 7: **end while**
1042 8: # Policy Improvement
1043 9:
1044 $\forall i \in \{1, \dots, N\}, s, \vec{a}^{<i},$
1045
$$\pi^{(i)}(s, \vec{a}^{<i}) \leftarrow \arg \max_{a^{(i)}} Q^{(i)}([s, \vec{a}^{<i}], \delta_{a^{(i)}})$$

1046 10: **end while**

1052
1053 **Algorithm 3** Agent-Chained Policy Optimization (PPO-based)
1054

1055 1: **Initialize:** Actor networks $\theta_0 = [\theta_0^{(1)}, \dots, \theta_0^{(N)}]$, Critic networks $\vec{\psi}_0 = [\psi_0^{(1)}, \dots, \psi_0^{(N)}]$.
1056 2: **while** $t \leq t_{max}$ **do**
1057 3: Collect transitions $(s_t, \{a_t^{(i)}\}_{i=1}^N, r_t, s_{t+1})$ by running the joint policy $\vec{\pi}_{\theta_k}$.
1058 4: Compute advantages after each episode: $\forall i = 1, \dots, N$,
1059
$$A_t^{(i)} = \sum_{j=i}^N (\gamma' \lambda')^{j-i} \zeta_t^{(j)} + \sum_{k=1}^{\infty} \sum_{j=1}^N (\gamma' \lambda')^{kN+j-i} \zeta_{t+k}^{(j)}$$

1060 5: Update actors with the PPO-Clip objective: $\forall i = 1, \dots, N$,
1061
1062
$$\mathbb{E}_{a_t^{(i)} \sim \pi_{\theta_{old}}^{(i)}(\cdot | s_t, b_t^{(i)})} [\min(w^{(i)}(s_t, b_t^{(i)}, a_t^{(i)}) A_t^{(i)}, \text{clip}(w^{(i)}(s_t, b_t^{(i)}, a_t^{(i)}), 1 - \epsilon, 1 + \epsilon) A_t^{(i)})]$$

1063 where $w^{(i)}(s_t, b_t^{(i)}, a_t^{(i)}) := \frac{\pi_{\theta}^{(i)}(a_t^{(i)} | s_t, b_t^{(i)})}{\pi_{\theta_{old}}^{(i)}(a_t^{(i)} | s_t, b_t^{(i)})}$.
1064 6: Update Decentralized Critics: $\forall i = 1, \dots, N$,
1065
$$\psi_{k+1}^{(i)} = \arg \min_{\psi^{(i)}} \mathbb{E} \left[\left(V_{\phi}^{(i)}(s_t) - \hat{R}_t^{(i)} \right)^2 \right]$$

1066 7: **end while**

1073 F FINAL PPO OBJECTIVE
1074
1075

1076 To simplify notation, we provide the derivation for agent 1 which does not have belief $b^{(i)}$ in its state
1077 space, and assume that the parameters θ are shared across agents.
1078

$$\begin{aligned}
1080 \quad & \nabla_{\theta} J(\theta) = \mathbb{E}_{\phi^{(1)} \sim \pi_{\theta}^{(1)}(\cdot|s)} \left[\nabla_{\theta} \log \pi_{\theta}^{(1)}(\phi^{(1)}|s) Q^{(1)}(s, \phi^{(1)}) \right] \\
1081 \quad & = \int_{\phi^{(1)}} \pi_{\theta}^{(1)}(\phi^{(1)}|s) \nabla_{\theta} \log \pi_{\theta}^{(1)}(\phi^{(1)}|s) Q^{(1)}(s, \phi^{(1)}) d\phi^{(1)} \\
1082 \quad & = \int_{\phi^{(1)}} \pi_{\theta}^{(1)}(\phi^{(1)}|s) \nabla_{\theta} \log \pi_{\theta}^{(1)}(\phi^{(1)}|s) \int_{a^{(1)}} \phi^{(1)}(a^{(1)}) Q^{(1)}(s, a^{(1)}) da^{(1)} d\phi^{(1)} \\
1083 \quad & = \int_{a^{(1)}} \int_{\phi^{(1)}} \pi_{\theta}^{(1)}(\phi^{(1)}|s) \nabla_{\theta} \log \pi_{\theta}^{(1)}(\phi^{(1)}|s) \phi^{(1)}(a^{(1)}) d\phi^{(1)} Q^{(1)}(s, a^{(1)}) da^{(1)} \\
1084 \quad & = \int_{a^{(1)}} \int_{\phi^{(1)}} \nabla_{\theta} \pi_{\theta}^{(1)}(\phi^{(1)}|s) \phi^{(1)}(a) d\phi^{(1)} Q^{(1)}(s, a^{(1)}) da^{(1)} \\
1085 \quad & = \int_{a^{(1)}} \nabla_{\theta} \left(\underbrace{\int_{\phi^{(1)}} \pi_{\theta}^{(1)}(\phi^{(1)}|s) \phi^{(1)}(a^{(1)}) d\phi^{(1)}}_{= \pi_{\theta}^{(1)}(a^{(1)}|s)} \right) Q^{(1)}(s, a^{(1)}) da^{(1)} \\
1086 \quad & = \int_{a^{(1)}} \nabla_{\theta} \pi_{\theta}^{(1)}(a^{(1)}|s) Q(s, a^{(1)}) da^{(1)} \\
1087 \quad & = \int_{a^{(1)}} \pi_{\theta}^{(1)}(a^{(1)}|s) \nabla_{\theta} \log \pi_{\theta}^{(1)}(a^{(1)}|s) Q(s, a^{(1)}) da^{(1)} \\
1088 \quad & = \mathbb{E}_{a^{(1)} \sim \pi_{\theta}^{(1)}(a^{(1)}|s)} \left[\nabla_{\theta} \log \pi_{\theta}^{(1)}(a^{(1)}|s) Q(s, a^{(1)}) \right]
\end{aligned}$$

We can obtain the same conclusion for other agents through a similar derivation.

G EXACT CALCULATION OF POLICIES IN THE MATRIX GAME

	<i>A</i>	<i>B</i>	<i>C</i>
<i>A</i>	5	-20	-20
<i>B</i>	-20	10	-20
<i>C</i>	-20	-20	20

Figure 6: 3x3 Matrix Game

Here we provide details on how ACPO can solve the Matrix Game provided in Table 1 and repeated above in Figure 6.

Due to serialization, ACPO considers this as a 2-step game even though the underlying game is a 1-step game. Since we are in a simple toy setting which can be solved by policy iteration, we only consider deterministic ϕ .

Policy evaluation for agent 1 is conducted as follows.

$$Q^{(1)}(\phi^{(1)}) = \gamma' \mathbb{E}_{\substack{b^{(2)} = \phi^{(1)} \\ \phi^{(2)} \sim \pi^{(2)}(\cdot|b^{(2)})}} \left[Q^{(2)}(b^{(2)}, \phi^{(2)}) \right]$$

In general, $\phi^{(1)} \in \Delta(A^{(1)})$ is continuous. However, this can be simplified if we only consider deterministic ϕ :

$$\begin{aligned}
1126 \quad & Q^{(1)}(\delta_A^{(1)}) = \gamma' \mathbb{E}_{a^{(2)} \sim \pi^{(2)}(\cdot|b^{(2)}=A)} \left[Q^{(2)}(b^{(2)} = A, a^{(2)}) \right] \\
1127 \quad & Q^{(1)}(\delta_B^{(1)}) = \gamma' \mathbb{E}_{a^{(2)} \sim \pi^{(2)}(\cdot|b^{(2)}=B)} \left[Q^{(2)}(b^{(2)} = B, a^{(2)}) \right] \\
1128 \quad & Q^{(1)}(\delta_C^{(1)}) = \gamma' \mathbb{E}_{a^{(2)} \sim \pi^{(2)}(\cdot|b^{(2)}=C)} \left[Q^{(2)}(b^{(2)} = C, a^{(2)}) \right]
\end{aligned}$$

where we have used $b^{(2)} = A$ to denote the fact that agent 2 knows with probability 1 that it is in state A since it knows that $\pi^{(1)}$ chooses A deterministically. Also, we used the notation $\delta_A^{(1)}$ to denote a particular action distribution ϕ which deterministically selects A .

1134 For agent 2, policy evaluation is simply the reward function given in the Matrix game:
 1135

$$Q^{(2)}(b^{(2)} = A, A) = R(A, A) = 5$$

1137 \vdots

$$Q^{(2)}(b^{(2)} = B, B) = R(B, B) = 10$$

1140 \vdots

$$Q^{(2)}(b^{(2)} = C, C) = R(C, C) = 20$$

1143

1144 For policy improvement,

$$\pi^{(2)}(b^{(2)} = A) \leftarrow \arg \max_{a^{(2)} \in \{A, B, C\}} Q^{(2)}(b^{(2)} = A, a^{(2)})$$

$$\pi^{(2)}(b^{(2)} = B) \leftarrow \arg \max_{a^{(2)} \in \{A, B, C\}} Q^{(2)}(b^{(2)} = B, a^{(2)})$$

$$\pi^{(2)}(b^{(2)} = C) \leftarrow \arg \max_{a^{(2)} \in \{A, B, C\}} Q^{(2)}(b^{(2)} = C, a^{(2)})$$

1151

1152 Thus, $\pi^{(2)}$ will select A, B, C given agent 1 deterministically selects A, B, C , respectively.
 1153

1154 For agent 1,

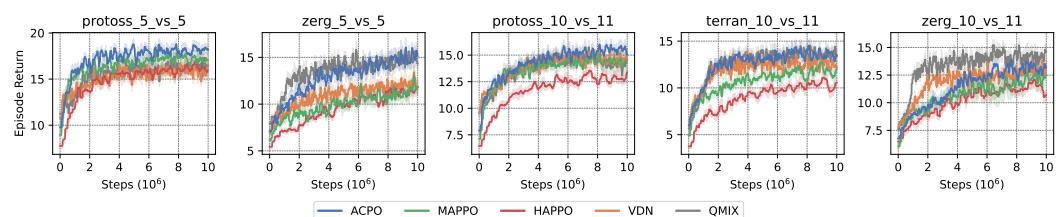
$$\pi^{(1)} \leftarrow \arg \max_{a^{(1)} \in \{A, B, C\}} \gamma' Q^{(2)}(b^{(2)} = a^{(1)}, \pi^{(2)}(b^{(2)} = a^{(1)}))$$

1155

1156 Now, let's say we are in an adversarial starting point where the policy is initialized to deterministically select (A, A) .
 1157

1158 Initially, policy improvement for $\pi^{(1)}$ leads agent 1 to continue selecting A since $\pi^{(2)}$ deterministically selects A and $R(A, A) = 5$ is better than $R(B, A) = R(C, A) = -20$. However, after the first iteration, agent 2 will update its policy to select A, B, C for $b^{(2)} = A, b^{(2)} = B, b^{(2)} = C$, respectively. Thus, agent 1 in iteration 2 will select C since $C = \arg \max_{a^{(1)}} \gamma' Q^{(2)}(b^{(2)} = a^{(1)}, \pi^{(2)})$, where $\pi^{(2)}$ is now updated.
 1164

1165 H LEARNING CURVE FOR SMACv2



1175 Figure 7: Return for SMACv2 with mean and standard error over 10 seeds.
 1176

1177 I HYPERPARAMETER DETAILS

1178

1179 For a fair comparison, we set the network type (MLP or GRU) and hidden layer size to be consistent
 1180 across all algorithms. The design choices follow the experimental setups of Zhong et al. (2024) and
 1181 Papoudakis et al. (2021). The discount factor γ is fixed, as it is inherent to the MMDP rather than a
 1182 tunable hyperparameter. In contrast ACPO employs serialization, where the advantage is computed
 1183 using $\gamma' = \gamma^{1/N}$.
 1184

1185

1186 For all baselines, we use the reported hyperparameters from Papoudakis et al. (2021) for RWARE,
 1187 Ellis et al. (2023) for SMACv2 and Zhong et al. (2024) for MA-MuJoCo. The full set of hyper-
 1188 parameters are provided in our anonymous code at the following link: <https://anonymous.4open.science/r/anonymous-acpo-BD51>.
 1189

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

Table 3: Common Parameters for All Algorithms

Parameter	RWARE	MA-MuJoCo	SMACv2
Network	MLP	MLP	GRU
Hidden Sizes	[128, 256]	[128, 128, 128]	[64]
γ	0.99	0.99	0.99

J COMPUTATIONAL RESOURCES

For RWARE experiments, we utilized a single NVIDIA GeForce RTX 3090 graphics processing unit (GPU). For the 2,4,8, and 12-agent environments, ACPO took 7H, 6H, 8H and 9H, respectively. The corresponding times were MAPPO (6H, 4H, 4H, 3H), HAPPO (19H, 37H, 71H, 107H) and HATRPO (21H, 38H, 70H, 105H).

In SMACv2 experiments, the hardware configuration depended on the scenario: a single NVIDIA GeForce RTX 3090 was used for 5-agent scenarios, whereas a single NVIDIA L40S GPU was used for 10-agent scenarios. For the 5-agent and 10-agent scenarios, ACPO took 16H and 20H, respectively. QMIX completed training in 24H and 34H, HAPPO required 33H and 51H, and MAPPO’s training times were 12H and 15H for the respective scenarios.

Due to significant computational requirements, experiments for HATRPO on SMACv2 were not conducted. We estimated that reaching 10M timesteps would take approximately 14 days for the 5-agent scenarios and 40 days for the 10-agent scenarios, which was deemed computationally prohibitive for our resources. Moreover, Zhong et al. (2024) reported weaker performance in SMACv2 compared to HAPPO, MAPPO, and QMIX.

K DISCUSSION ON CENTRALIZED TRAINING WITH DECENTRALIZED EXECUTION

In our work, we consider the standard MARL paradigm of Centralized Training Decentralized Execution (CTDE), which allows multiple policies to be jointly trained but must be executed in a decentralized fashion.

Here we introduce related work in three different settings that are often considered in MARL: Centralized Training Centralized Execution (CTCE), CTDE with Centralized Rollouts (CTDE-CR), and CTDE with Decentralized Rollouts (CTDE-DR).

Centralized Training Centralized Execution (CTCE) Centralized Training Centralized Execution (CTCE) methods such as Multi-Agent Transformer (MAT) (Wen et al., 2022) use a joint policy of the form $\pi(a^{(1)}, \dots, a^{(N)}|s)$ during both training rollouts and execution. MAT is a centralized Transformer model defined on the joint action space, and uses a joint observation encoder and joint policies during both training and execution. While a decentralized policy version is also considered, MAT requires a joint observation encoder during both training and execution. In MMDPs, the CTCE setting reduces to a Factored-Action MDP (Guestrin et al., 2001; Raghavan et al., 2012), which is a single-agent MDP with factored action spaces. In this case, single-agent techniques such as policy iteration and value iteration can be applied directly.

Generally, centralized control (CTCE) is not applicable to many real-world multi-agent systems such as power grids Wang et al. (2021a), traffic signal control Chu et al. (2020), and large-scale fleet management Lin et al. (2018) due to the large joint action space and prohibitive communication costs.

Centralized Training Decentralized Execution with Centralized Rollouts (CTDE-CR) There are also some work within CTDE but with additional assumptions to make training simpler. (Ye et al., 2023; Wang et al., 2023) aim to decentralize centralized solutions Amato (2024), by assuming that a single-agent joint policy can be used for training rollout. This joint policy is used during centralized training and distilled to decentralized policies before execution. As a single-agent problem,

this assumption makes convergence to optimal policies straightforward as in the CTCE case. We can view this setting as a special case of our work where we assume access to a joint policy during training rollouts (CTDE-CR). With this additional assumption, we can solve the serialized problem introduced in Section 3 without considering beliefs. However, this line of research inherits similar weaknesses of CTCE, and cannot be applied to many real-world multi-agent systems with a massive action space or prohibitive communication costs.

Centralized Training Decentralized Execution with Decentralized Rollouts (CTDE-DR) In CTDE-DR, the policy must be decentralized (fully factorized) during both training and execution, with the policy form $\vec{\pi} = \langle \pi^{(1)}, \dots, \pi^{(N)} \rangle$ where $\pi^{(i)} : S \rightarrow A^{(i)}$. This is the natural MARL paradigm we consider in our work. Algorithms for simultaneous policy update methods Lowe et al. (2017); Yu et al. (2022), iterative best response methods Kuba et al. (2022); Zhong et al. (2024); Liu et al. (2024) as well as value decomposition methods Rashid et al. (2018); Zhang et al. (2021) all fall under CTDE-DR.

L BELIEF APPROXIMATION FOR PRACTICAL IMPLEMENTATIONS

In ACPO (PPO-based), the clipped policy gradient objective in Eq. 3 is similar to MAPPO with an additional input to the policy, the belief $b^{(i)}$. In high-dimensional domains, the belief update for a POMDP is intractable to compute exactly and is often approximated with RNNs or Transformers (Ni et al., 2022; 2023). For AC-BMDPs, the belief is defined as a distribution over the unobservable $\vec{a}^{<i}$, which we approximate by having each agent predict the previous agents’ actions for the current time step.

Below we show our overall opponent modeling procedure using state inputs for simplicity⁴.

First, Agent 1 has no notion of belief, so it simply outputs $a^{(1)} \sim \pi^{(1)}(\cdot | s)$. Next, agent 2 predicts the action that agent 1 has taken, i.e. $\tilde{a}^{(1)} \sim \tilde{\pi}^{(1)}(\cdot | s)$, where $\tilde{\pi}^{(1)}$ is agent 2’s own policy with an additional agent ID of 1 as input. Now, agent 2 will output its action, i.e. $a^{(2)} \sim \pi^{(2)}(\cdot | s, \tilde{a}^{(1)})$ using the predicted action for agent 1.

M RETURN VS RUNTIME COMPARISON ON RWARE

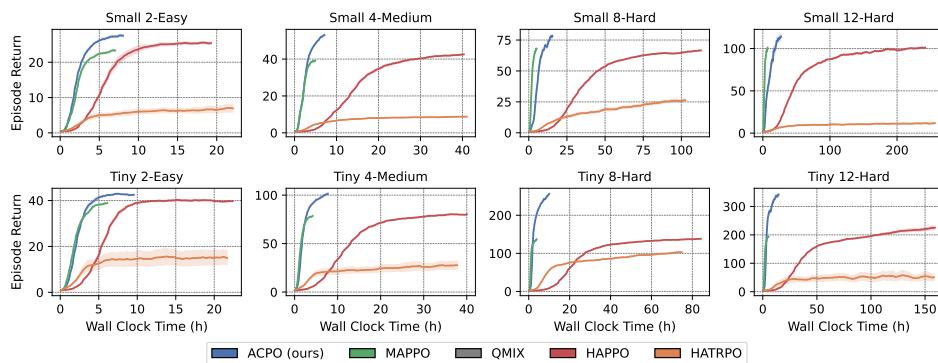


Figure 8: Episode return vs training runtime comparison on Multi-Robot Warehouse (RWARE). The mean and standard error over 10 seeds are reported for all tasks and algorithms except HATRPO and HAPPO on Small 12-Hard (5 seeds) and Tiny 12-Hard (8 seeds).

Figure 8 compares the episode returns as a function of the training runtime. The experiments were terminated once the maximum number of timesteps was reached, as depicted in Figure 2. Notably,

⁴For partially observable environments, we replace the state s_t with either its own partial observation $o_t^{(i)}$ for MLPs or the hidden state of an RNN $h_t^{(i)}$ which takes the action-observation history $\langle \vec{o}_{<t}^{(i)}, \vec{a}_{<t}^{(i)} \rangle$ as input.

1296 ACPO and MAPPO demonstrate substantially better runtime efficiency compared to HAPPO and
 1297 HATRPO, primarily due to their simultaneous rather than sequential policy updates. Furthermore,
 1298 ACPO consistently outperforms the other methods across all settings while maintaining a runtime
 1299 comparable to MAPPO. The slight increase in ACPO’s runtime is attributed to the additional cost of
 1300 belief computation.

1304 N AGENT-CHAINED TWIN DELAYED DETERMINISTIC POLICY 1305 GRADIENT (AC-TD3)

1307 Following the Bellman operators in Definition 5.1, the critic loss based on TD3 (Fujimoto et al.,
 1308 2018) can be written as follows:

1309 for $i = 1, \dots, N - 1$,

$$1311 J_Q^{(i)}(\psi) = \mathbb{E}_{\substack{(s, b^{(i)}, \phi^{(i)}) \sim \mathcal{D} \\ \phi^{(i+1)} \sim \pi_\theta^{(i+1)}(\cdot | s, b^{(i)}, \phi^{(i)})}} \left[\left(Q_\psi^{(i)}([s, b^{(i)}], \phi^{(i)}) - y^{(i)} \right)^2 \right]$$

$$1314 \text{s.t. } y^{(i)} = \gamma' Q_\psi^{(i+1)}([s, b^{(i)}], \phi^{(i+1)})$$

$$1317 J_Q^{(N)}(\psi) = \mathbb{E}_{\substack{(s, b^{(N)}, \phi^{(N)}, r, s') \sim \mathcal{D} \\ \phi^{(1)'} \sim \pi_\theta^{(1)}(\cdot | s')}} \left[\left(Q_\psi^{(N)}([s, b^{(N)}], \phi^{(N)}) - y^{(N)} \right)^2 \right]$$

$$1320 \text{s.t. } y^{(N)} = r + \gamma' Q_\psi^{(1)}([s', \phi^{(1)'}])$$

1322 We simplify this objective by considering deterministic $\phi^{(i)}$,

1324 for $i = 1, \dots, N - 1$,

$$1326 J_Q^{(i)}(\psi) = \mathbb{E}_{\substack{(s, \vec{a}^{<i}, a^{(i)}) \sim \mathcal{D} \\ a^{(i+1)} \sim \pi_\theta^{(i+1)}(\cdot | s, \vec{a}^{<i}, a^{(i)})}} \left[\left(Q_\psi^{(i)}([s, \vec{a}^{<i}], a^{(i)}) - y^{(i)} \right)^2 \right]$$

$$1329 \text{s.t. } y^{(i)} = \gamma' Q_\psi^{(i+1)}([s, \vec{a}^{<i}, a^{(i)}], a^{(i+1)})$$

$$1332 J_Q^{(N)}(\psi) = \mathbb{E}_{\substack{(s, \vec{a}^{<N}, a^{(N)}, r, s') \sim \mathcal{D} \\ \vec{a}' \sim \vec{\pi}_\theta(\cdot | s')}} \left[\left(Q_\psi^{(N)}([s, \vec{a}^{<N}], a^{(N)}) - y^{(N)} \right)^2 \right]$$

$$1335 \text{s.t. } y^{(N)} = r + \gamma' Q_\psi^{(1)}([s', a^{(1)'}])$$

1336 For practical implementations, it is often useful to consider k -step returns.

$$1339 J_Q^{(i)}(\psi) = \mathbb{E}_{\substack{(s_t, \vec{a}_t^{<i}, a_t^{(i)}, \{r_{t+j}\}_{j=0}^k, s_{t+k+1}) \sim \mathcal{D} \\ \vec{a}_{t+k+1} \sim \vec{\pi}_\theta(\cdot | s_{t+k+1})}} \left[\left(Q_\psi^{(i)}([s_t, \vec{a}_t^{<i}], a_t^{(i)}) - (\gamma')^{N-i} y_t^{(i)} \right)^2 \right]$$

$$1342 \text{s.t. } y_t^{(i)} = r_t + \gamma r_{t+1} + \dots + \gamma^k r_{t+k} + \gamma^{k+1} Q_\psi^{(i+1)}([s_{t+k+1}, \vec{a}_{t+k+1}^{<i}], a_{t+k+1}^{(i+1)})$$

1344 We find that using k -step returns in this way works better in practice as each agent now has a dense
 1345 reward signal in the targets (rather than only the last agent). We note that γ denotes the discount
 1346 factor in the original MMDP and $\gamma' = \gamma^{1/N}$. The $(\gamma')^{N-i}$ discount is to adjust the micro step
 1347 to match with the last agent. For example, for agent 1, the reward given at the current timestep is
 1348 $(\gamma')^{N-1} r_t$.

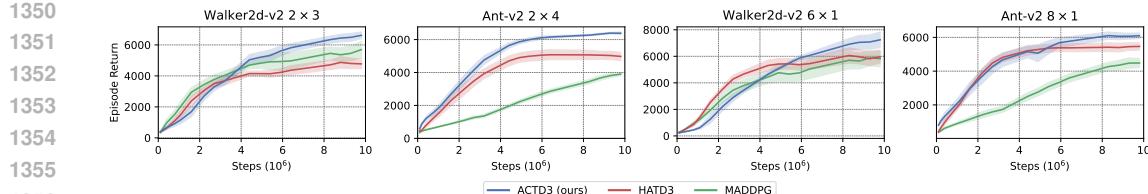


Figure 9: Mean Return and Standard Error over 5 seeds for MA-MuJoCo (Gym).

O OFF-POLICY COMPARISON

We evaluate an off-policy variant of ACPO, termed AC-TD3, by incorporating agent chaining into the TD3 (Fujimoto et al., 2018) algorithm, as described in Appendix J. The performance of ACTD3 is compared against off-policy baselines, HATD3 (Zhong et al., 2024) and MADDPG (Lowe et al., 2017). Experiments are conducted on the Ma-MuJoCo (Gym) environment using five random seeds. As illustrated in Figure 9, AC-TD3 consistently outperforms all baseline methods.

1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403