# Coordination in Ad Hoc Teams with Generalized Policy Improvement

Rupal Nigam, Niket Parikh, Mikihisa Yuasa, Huy T. Tran
*Aerospace Engineering*
*University of Illinois at Urbana-Champaign*
United States
{rupaln2, niketnp2, myuasa2, huytran1}@illinois.edu

***Abstract*—Multi-agent reinforcement learning (MARL) is a recently popular approach for solving complex tasks with multiple agents present. Ad hoc teaming (AHT) is a more specific problem formulation in which an agent must successfully coordinate with other previously unseen teammates. We model our problem within the framework of a multi-agent Markov decision process (MMDP) and use successor features and generalized policy improvement for efficient and effective knowledge transfer between different teams. Theoretical optimality bounds are derived and an algorithm is proposed for zero-shot coordination with an ad hoc team. We empirically demonstrate that our method successfully transfers to new teams in a collaborative object-collecting game environment.**

***Index Terms*—multi-agent reinforcement learning, ad hoc teaming, coordination**

## I. Introduction

In recent years, ad hoc teaming (AHT) has been proposed as a challenge for multi-agent autonomous systems, in which an agent must be able to successfully coordinate with other unknown agents [1]. Classic multi-agent reinforcement learning (MARL) algorithms jointly train all agents in the team together, whereas in AHT, only a single agent (the learner) is controlled [2], [3]. Generalizing to teammates the learner has never seen before is a crucial open problem for AHT [4].

Prominent methods for AHT involve pre-training the learner with a set of teammates to generate a set of policies from which the best candidate is picked using inference during test time [5], [6]. Our approach also generates a set of policies using pre-training, but instead of choosing one to use, performs generalized policy improvement to leverage the whole set. Generalized policy improvement has been used for single-agent transfer learning where the reward function differed [7]. However, transfer learning within AHT presents a unique challenge because both the rewards and the dynamics change.

The contributions of this paper are as follows:

1) formalization of an ad hoc teaming MDP framework for coordination with new teammates,
2) theoretical bounds on the performance of the controlled agent when coordinating with an ad hoc team,
3) empirically demonstrating the performance of an algorithm in a collaborative game environment for zero-shot coordination with an ad hoc team.

This work was supported in part by ONR N00014-20-1-2249 and a NASA grant awarded to the Illinois/NASA Space Grant Consortium.

## II. Background and Related Work

### A. Multi-agent Reinforcement Learning

We model our problem within the framework of a multi-agent Markov decision process (MMDP) defined by a tuple $(\mathcal{S}, \mathcal{N}, \{\mathcal{A}^i\}_{i\in\mathcal{N}}, p, r, \gamma)$. Here, $\mathcal{S}$ is the state space, $\mathcal{N}$ is the set of agents, $\mathcal{A}^i$ is the action space of agent $i$, and $\gamma \in [0, 1)$ is the discount factor. Let $\mathcal{A} \coloneqq \times_{i\in\mathcal{N}}\mathcal{A}^i$ be the joint action space. Then $p(s'|s, \boldsymbol{a}) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ is the state transition function and $r(s, \boldsymbol{a}, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is the team reward function, for states $s, s' \in \mathcal{S}$ and joint action $\boldsymbol{a} \in \mathcal{A}$ composed of individual actions $a^i \in \mathcal{A}^i$.

At time step $t$, each agent $i \in \mathcal{N}$ executes an action $a^i_t$ given the current state $s_t$, after which the system transitions to state $s_{t+1}$ and the team receives reward $r(s_t, \boldsymbol{a}_t, s_{t+1})$. Let $\pi^i(a^i|s) : \mathcal{S} \times \mathcal{A}^i \mapsto [0, 1]$ be an individual policy for agent $i$ and $\pi(\boldsymbol{a}|s) \coloneqq \prod_{i\in\mathcal{N}} \pi^i(a^i|s)$ be the resulting joint policy of all agents. The performance of a joint policy $\pi$ can be described by its action-value function,

$$Q^\pi(s, \boldsymbol{a}) \coloneqq \mathbb{E}_{p,\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \boldsymbol{a}_t, s_{t+1}) \mid s_0 = s, \boldsymbol{a}_0 = \boldsymbol{a}\right]. \tag{1}$$

### B. Generalized Policy Updates with SFs

Generalized policy updates generalize two key operations in RL, policy evaluation and policy improvement, to operate over sets of tasks and policies, respectively [8]. Let $\mathcal{R}$ be a set of tasks defined by different reward functions. Generalized policy evaluation (GPE) computes the value function, $Q^\pi_r(s, \boldsymbol{a})$, of a policy $\pi$ for each task $r \in \mathcal{R}$. Generalized policy improvement (GPI) defines a policy $\pi'$ that improves over a set of policies $\Pi$ for a task $r$. Efficient implementations of GPE and GPI can be used for fast transfer in RL as follows.

Assume that each task in $\mathcal{R}$ can be defined as a linear combination of features,

$$r(s, \boldsymbol{a}, s') = \phi(s, \boldsymbol{a}, s')^\intercal \mathbf{w}, \tag{2}$$

where $\phi(s, \boldsymbol{a}, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}^d$ a function mapping to $d$ features and $\mathbf{w} \in \mathbb{R}^d$ is a weight vector specifying preferences

over features. Following [7], define the SFs of policy $\pi$ as,

$$\psi^{\pi}(s, \boldsymbol{a}) = \mathbb{E}_{p,\pi}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, \boldsymbol{a}_t, s_{t+1}) \mid s_0 = s, \boldsymbol{a}_0 = \boldsymbol{a}\right]. \tag{3}$$

SFs represent the expected discounted sum of features when starting in state $s$, taking action $\boldsymbol{a}$, and following policy $\pi$. The action-value function of $\pi$ on task $r$, $Q_r^{\pi}(s, \boldsymbol{a})$, can then be represented as,

$$Q_r^{\pi}(s, \boldsymbol{a}) = \psi^{\pi}(s, \boldsymbol{a})^{\intercal}\mathbf{w}. \tag{4}$$

Given the SFs of policy $\pi$, $\psi^{\pi}(s, \boldsymbol{a})$, we can now quickly perform GPE over tasks in $\mathcal{R}$ by computing Equation (4) using the weights $\mathbf{w}$ associated with each task. Assume that we are given a set of policies $\Pi = \{\pi_i\}_{i=1}^n$ and their corresponding action-value functions $\{Q_r^{\pi_i}\}_{i=1}^n$ for a task $r$. Following [7], GPI can be efficiently performed on task $r$ through a policy $\pi'$ defined as,

$$\pi'(s) \in \operatorname*{argmax}_{a \in \mathcal{A}} \max_{\pi \in \Pi} Q_r^{\pi}(s, a). \tag{5}$$

We refer to $\pi'$ as the GPI policy.

We can use these implementations of GPE and GPI for transfer learning by assuming the agent has pre-trained on a set of source tasks $\mathcal{R} = \{r_i(s, \boldsymbol{a}, s') = \phi(s, \boldsymbol{a}, s')^{\intercal}\mathbf{w}_i\}_{i=1}^n$, resulting in a set of optimal policies $\Pi = \{\pi_i^*\}_{i=1}^n$, where $\pi_i^*$ is an optimal policy for task $r_i$. We also compute the set of SFs associated with each policy, $\Psi = \{\psi^{\pi_i^*}\}_{i=1}^n$. Given a new target task $r_{n+1} = \phi(s, \boldsymbol{a}, s')^{\intercal}\mathbf{w}_{n+1}$, we can now use Equation (5) to define a policy $\pi'$ that is no worse than any policy in $\Pi$ on this task. Furthermore, if we compute the set $\{Q_{r_{n+1}}^{\pi_i^*}\}_{i=1}^n$ using Equation (4), we can implement $\pi'$ without additional learning on the target task. If additional learning is allowed, we can use $\pi'$ to optimize a policy for $r_{n+1}$, for example using SFQL (Algorithm 3 in [8]). We use generalized policy updates as a framework for fast transfer in an ad hoc team setting.

## III. Problem Formulation

### A. Ad Hoc MMDPs

We modify the general formulation of MMDPs for ad hoc teaming as follows. Let $\mathrm{a} \in \mathcal{N}$ be the *learner* (i.e., the agent whose policy we aim to optimize) and $\mathcal{N}_u = \mathcal{N} \setminus \{\mathrm{a}\}$ be the complementary set of all teammates (i.e., uncontrolled agents). We assume that each teammate follows a fixed policy, which is unknown to the learner. Teammate policies may be suboptimal with respect to the team reward $r$ and the ad hoc team considered due to, e.g., the teammates being trained for a different task or with different teammates, or being humans and having inherent biases towards different goals. We formally define this problem as an ad hoc MMDP.

**Definition 1** (Ad Hoc MMDP)**.** An ad hoc MMDP is defined by a tuple $M := (\mathcal{S}, \mathcal{N}, \mathrm{a}, \{\mathcal{A}^i\}_{i \in \mathcal{N}}, p, r, \{\pi^i\}_{i \in \mathcal{N}_u}, \gamma)$, where $\mathrm{a} \in \mathcal{N}$ is the learner, $\mathcal{N}_u = \mathcal{N} \setminus \{\mathrm{a}\}$ is the complementary set of teammates, and $\pi^i(s) : \mathcal{S} \mapsto \mathcal{A}^i$ is the fixed deterministic policy of teammate $i$.

We refer to an ad hoc MMDP $M$ as an ad hoc team. The performance of a learner policy $\pi^{\mathrm{a}}$ in ad hoc team $M$ can be described by its action-value function,

$$Q^{\pi^{\mathrm{a}}, \pi^{-\mathrm{a}}}(s, a^{\mathrm{a}}) := \mathbb{E}_{p, \pi^{\mathrm{a}}, \pi^{-\mathrm{a}}}\Bigg[\sum_{t=0}^{\infty} \gamma^t r(s_t, \boldsymbol{a}_t, s_{t+1}) \mid s_0 = s, a_0^{\mathrm{a}} = a^{\mathrm{a}}\Bigg], \tag{6}$$

where $\pi^{-\mathrm{a}}$ is the joint policy of all teammates induced by $\{\pi^i\}_{i \in \mathcal{N}_u}$. Our objective is to compute an optimal learner policy, $\pi^{\mathrm{a}^*}$, which satisfies,

$$Q^{\pi^{\mathrm{a}^*}, \pi^{-\mathrm{a}}}(s, a^{\mathrm{a}}) := \max_{\pi^{\mathrm{a}}} Q^{\pi^{\mathrm{a}}, \pi^{-\mathrm{a}}}(s, a^{\mathrm{a}}), \tag{7}$$

for all $s \in \mathcal{S}$ and $a^{\mathrm{a}} \in \mathcal{A}^i$. Note that optimizing a learner policy for an ad hoc team $M$ is equivalent to solving a single-agent Markov decision process (MDP) with a transition function $\tilde{p}$ that captures the impact of teammate policies $\pi^{-\mathrm{a}}$.

### B. Transfer Learning in Ad Hoc MMDPs

We assume that the learner can leverage prior training with a set of different teams to quickly adapt to a new ad hoc team. To formalize this transfer problem, we define the set of possible ad hoc teams $\mathcal{M}$ as,

$$\begin{aligned}\mathcal{M}(\mathcal{S}, \mathcal{D}, N, \mathrm{a}, \{\mathcal{A}^i\}_{i \in \mathcal{D}}, p, r, \{\pi^i\}_{i \in \mathcal{D} \setminus \{\mathrm{a}\}}, \gamma) := \\ \{M(\mathcal{S}, \mathcal{N}, \mathrm{a}, \{\mathcal{A}^i\}_{i \in \mathcal{N}}, p, r, \{\pi^i\}_{i \in \mathcal{N}_u}, \gamma)\},\end{aligned} \tag{8}$$

where $\mathcal{D}$ is the set of all possible teammates, $N \in \mathbb{Z}$ is the number of agents in each team, and $\mathcal{N} \subseteq \mathcal{D}$ is the set of agents in an ad hoc team (where $|\mathcal{N}| = N$). Let $\mathcal{M}_0 = \{M_i\}_{i=1}^n \subseteq \mathcal{M}$ be the set of source ad hoc teams with which the learner pre-trains. Our objective is then to quickly learn an optimal learner policy $\pi_{n+1}^{\mathrm{a}^*}$ for a new ad hoc team $M_{n+1} \in \mathcal{M} \setminus \mathcal{M}_0$, leveraging information from pre-training on $\mathcal{M}_0$.

## IV. Our Approach

### A. Generalized Policy Improvement for Ad Hoc Teaming

We propose a new approach, named GSAT (GPI with SFs for Ad Hoc Teaming), to address the transfer problem in ad hoc MMDPs. To use GPI for transfer learning, we need to define four components: features $\phi$, a set of policies $\Pi$, a new task, and a GPI policy to act in that new task. For this work, we assume the features $\phi$ are fixed and heuristically defined such that the team reward function $r$ can be modeled using Equation (2), which produces a weight vector $\mathbf{w}$. Alternatively, various methods have been proposed to learn useful features for SFs and the corresponding weights $\mathbf{w}$ [9], [10].

We define the set of policies $\Pi$ based on pre-training the learner with the set of source ad hoc teams $\mathcal{M}_0$. That is, we optimize a set of learner policies $\Pi = \{\pi_i^{\mathrm{a}^*}\}_{i=1}^n$, where $\pi_i^{\mathrm{a}^*}$ is the optimal learner policy for ad hoc team $M_i \in \mathcal{M}_0$. We assume this process also generates a corresponding set of learner SFs $\Psi = \{\psi^{\pi_i^{\mathrm{a}^*}, \pi_i^{-\mathrm{a}}}\}_{i=1}^n$. Here, we define the learner

SFs, $\psi^{\pi^{\mathrm{a}},\pi_i^{-\mathrm{a}}}$, as the SFs of learner policy $\pi^{\mathrm{a}}$ in ad hoc team $M_i \in \mathcal{M}$. That is,

$$\psi^{\pi^{\mathrm{a}},\pi_i^{-\mathrm{a}}}(s,a^{\mathrm{a}}) = \mathbb{E}_{p,\pi^{\mathrm{a}},\pi_i^{-\mathrm{a}}}\Bigg[\sum_{t=0}^{\infty}\gamma^t\phi(s_t,\boldsymbol{a}_t,s_{t+1}) \mid s_0=s, a_0^{\mathrm{a}}=a^{\mathrm{a}}\Bigg], \tag{9}$$

where $\pi_i^{-\mathrm{a}}$ is the joint policy of all teammates in ad hoc team $M_i$. The set $\Psi$ can be learned, for example, by using Algorithm 1 in the SI appendix of [9]. We use these SFs to model the action-value function of learner policy $\pi^{\mathrm{a}}$ in ad hoc team $M_i$ as,

$$Q^{\pi^{\mathrm{a}},\pi_i^{-\mathrm{a}}}(s,a^{\mathrm{a}}) = \psi^{\pi^{\mathrm{a}},\pi_i^{-\mathrm{a}}}(s,a^{\mathrm{a}})^{\intercal}\mathbf{w}. \tag{10}$$

We refer to $\psi^{\pi^{\mathrm{a}},\pi_i^{-\mathrm{a}}}$ as $\psi_i^{\pi^{\mathrm{a}}}$ and $Q^{\pi^{\mathrm{a}},\pi_i^{-\mathrm{a}}}$ as $Q_i^{\pi^{\mathrm{a}}}$ hereafter to simplify notation.

Existing work defines a new task through a new reward function $r_{n+1}$ [9]. Instead, we define a new task as a new ad hoc team $M_{n+1}$, which defines a new joint policy for teammates $\pi_{n+1}^{-\mathrm{a}}$ but uses the same team reward function $r$ used by all other ad hoc teams. That is, our tasks change through the dynamics induced by new teammate policies, rather than through the weights of a linear reward function. However, these new dynamics mean that our set of pre-trained learner SFs $\Psi$ is no longer valid in the new task. Therefore, we include a learning period where we update the SFs in $\Psi$ to account for the new task, and use these updated SFs to directly implement a GPI policy of the form in Equation (5).

### *B. Theoretical Bounds for GPI in Ad Hoc MMDPs*

Here we provide performance bounds for transferring policies in ad hoc MMDPs. The proofs can be found at https://tinyurl.com/gsat-proof. Given a set of learner policies $\Pi$ and their corresponding learner SFs in $M_{n+1}$, we define a GPI policy $\pi'_{\mathbf{w}}$ to act in $M_{n+1}$ as,

$$\pi'_{\mathbf{w}}(s) \in \operatorname*{argmax}_{a^{\mathrm{a}}\in\mathcal{A}^{\mathrm{a}}} \max_{\pi^{\mathrm{a}}\in\Pi} \psi_{n+1}^{\pi^{\mathrm{a}}}(s,a^{\mathrm{a}})^{\intercal}\mathbf{w}. \tag{11}$$

*1) GPI Performance Bound:* Here we extend the idea and proof of GPI to our setting. We follow the proof structure of Theorem 1 from [8], adjusted for our problem setting and notation. We assume deterministic policies for notational simplicity and remind the reader that a (fixed) joint policy $\pi_j^{-\mathrm{a}}$ is defined for a given ad hoc team $M_j \in \mathcal{M}$. Here the GPI policy is obtained from policies learned in the same MDP the GPI policy is executed in; therefore the subscript $j$ is dropped from terms in this theorem.

**Theorem 1.** Let $\pi_1^{\mathrm{a}},\ldots,\pi_n^{\mathrm{a}}$ be $n$ learner policies for ad hoc team $M \in \mathcal{M}$. We can approximate their action-value functions to obtain the set $\{\tilde{Q}^1,\tilde{Q}^2,\ldots,\tilde{Q}^n\}$, such that,

$$\mid Q^i(s,a^{\mathrm{a}}) - \tilde{Q}^i(s,a^{\mathrm{a}}) \mid \leq \epsilon,$$

for all $s\in\mathcal{S}$, $a^{\mathrm{a}}\in\mathcal{A}$, and $i\in\{1,\ldots,n\}$, where $\epsilon$ is the approximation error. We additionally denote $\boldsymbol{a}=(a^{\mathrm{a}},\boldsymbol{a}^{-\mathrm{a}})=(a^{\mathrm{a}},\pi^{-\mathrm{a}}(s))$ for any $s\in\mathcal{S}$. We now define a GPI learner policy $\pi^{\mathrm{a}}(s)$ as,

$$\pi^{\mathrm{a}}(s) \coloneqq \operatorname*{argmax}_{a^{\mathrm{a}}} \max_i \tilde{Q}^i(s,a^{\mathrm{a}}),$$

with its corresponding action-value function denoted by $Q^{\pi^{\mathrm{a}}}$. We can bound its performance as,

$$\mid Q^{\pi^{\mathrm{a}}}(s,a^{\mathrm{a}}) - \max_i Q^i(s,a^{\mathrm{a}}) \mid \leq \frac{2}{1-\gamma}\epsilon \tag{12}$$

In other words, the performance of the greedily-defined policy $\pi^{\mathrm{a}}(s)$ with any team in $\mathcal{M}$ is no worse than each of the policies in the set of agent policies $\pi_1^{\mathrm{a}},\ldots,\pi_n^{\mathrm{a}}$ learned earlier.

*2) Policy Transfer in Ad Hoc MMDPs:* Here we derive a performance bound for transferring a policy that was optimized in an ad hoc MMDP different from the one in which it is executed. Specifically, we bound the difference between the action-value function of a deterministic policy for the learner optimized for $M_k$ and executed in $M_k$ and the action-value function of a deterministic policy for the learner optimized for $M_j$ but executed in $M_k$.

**Lemma 1.**

$$Q_k^* - Q_k^{j*} \leq 2\left(\frac{\epsilon_r + 2\gamma d_{TV}\cdot\Omega}{1-\gamma}\right), \tag{13}$$

where we define $\epsilon_r \coloneqq \max_{s,a^{\mathrm{a}}}|r_k^{\mathrm{a}}(s,a^{\mathrm{a}}) - r_j^{\mathrm{a}}(s,a^{\mathrm{a}})|$, $d_{TV}(p_k,p_j) \coloneqq \frac{1}{2}\max_{s,a}\sum_{s'}|p_k(s'|s,a)-p_j(s'|s,a)|$, and $\Omega \coloneqq \max_{s,a^{\mathrm{a}}} Q_j^* + \frac{1}{2}$.

*3) GPI for Transfer in Ad Hoc MMDPs:*

**Theorem 2.** Suppose the agent has optimized policies in $n$ different ad hoc MMDPs within $\mathcal{M}^{\phi}\setminus M_k$, so that we have $n$ policies $\pi_1^{\mathrm{a}*},\ldots,\pi_n^{\mathrm{a}*}$. First we approximate the action-value function of these policies when executed in $M_k$ such that,

$$\mid Q_k^{j*}(s,a^{\mathrm{a}}) - \tilde{Q}_k^{j*}(s,a^{\mathrm{a}}) \mid \leq \epsilon \tag{14}$$

for all $s\in\mathcal{S}$, $a^{\mathrm{a}}\in\mathcal{A}$, where $\epsilon$ is the approximation error and $j\in\{1,\ldots,n\}$.

Now, given the set $\{\tilde{Q}_k^{1*},\tilde{Q}_k^{2*},\ldots,\tilde{Q}_k^{n*}\}$, we define the learner policy $\pi^{\mathrm{a}}$ as,

$$\pi^{\mathrm{a}}(s) \in \operatorname*{argmax}_{a^{\mathrm{a}}} \max_j \tilde{Q}_k^{j*}(s,a^{\mathrm{a}}). \tag{15}$$

with its corresponding action-value function denoted by $Q_k^{\pi^{\mathrm{a}}}$. Then we can bound its performance as,

$$\mid Q_k^*(s,a^{\mathrm{a}}) - Q_k^{\pi^{\mathrm{a}}}(s,a^{\mathrm{a}}) \mid \leq \frac{2}{1-\gamma}\left(\epsilon_r + 2\gamma d_{TV}\cdot\Omega + \epsilon\right) \tag{16}$$

where we define $\epsilon_r \coloneqq \max_{s,a^{\mathrm{a}}}|r_k^{\mathrm{a}}(s,a^{\mathrm{a}}) - r_j^{\mathrm{a}}(s,a^{\mathrm{a}})|$, $d_{TV}(p_k,p_j) \coloneqq \frac{1}{2}\max_{s,a}\sum_{s'}|p_k(s'|s,a)-p_j(s'|s,a)|$, and $\Omega \coloneqq \max_{s,a^{\mathrm{a}}} Q_j^* + \frac{1}{2}$. Here, $j$ is the same as in Equation (15).

## V. Experiments

### A. Environment

We empirically demonstrate GSAT's performance in a multi-agent object-collecting environment inspired by [9]. Our environment is depicted in Figure 1 and has $d$ different object types. As in [9], the state representations are agent-centric and toroidal, such that the agent is always in the upper left corner and the grid is wrapped around the edges of the environment. The representation has $(d+2)$ channels, i.e., one channel for each object type, one for all other agents, and one for walls.

We define the environment features as $\phi(s^i, \boldsymbol{a}, s^{i'}) : \mathcal{S}^i \times \mathcal{A} \times \mathcal{S}^i \mapsto [0, n]^d$, where $\mathcal{S}^i$ is the agent-centric state space of the agent $i$, $n = \min\{|\mathcal{N}|, 18\}$ represents the maximum number of objects of a given type that can be collected in a state transition, $\phi(s^i, \boldsymbol{a}, s^{i'}) = 1$ if a coin of type $i$ was collected during the transition from $s$ to $s'$, and $\phi(s^i, \boldsymbol{a}, s^{i'}) = 0$ otherwise. We assume all agents have the same state space. We use $d = 3$ objects (red, orange, yellow) for the presented results.

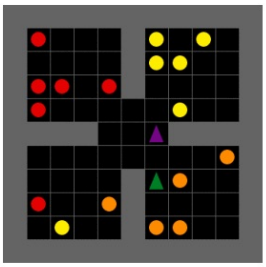

Fig. 1. Multi-agent collect game environment. The two agents are represented by triangles, where green is the learner and purple is the uncontrolled agent. Circles represent objects and different object types have different colours.

### B. Results

We compare GSAT's performance to the pre-trained learner policies of set $\Pi$ (i.e., the set of policies that yield the GPI policy) and an SFQL agent [9] trained from scratch with the new ad hoc team. Our choice of baselines is inspired by state-of-the-art AHT methods, such as PLASTIC-policy [5]. When confronted with a new ad hoc team, these methods use the pre-trained policy in $\Pi$ that is best suited for the new team, typically based on inferred similarity to the new teammates. Our results do not include this inference process and simply show the performance obtained by each pre-trained policy in $\Pi$. We include an SFQL agent to represent performance with no prior training.

The environment consists of two agents: the learner and an uncontrolled teammate. We pre-train two learner policies, $\{\pi_i^{a*}\}_{i=1}^2$, each optimized for the task $\mathbf{w} = [1, 1, 1]$ with given teammates, which characterize the set of source ad hoc teams, $\mathcal{M}_0$. The teammate policies operate with a random agent during training and optimize the following tasks: teammate for $\pi_1^{a*}$ is optimal for the task $\mathbf{w} = [1, 0, 0]$ and teammate for $\pi_2^{a*}$ is optimal for the task $\mathbf{w} = [0, 0, 1]$.

Figure 2 shows the mean return of GSAT when executed with the new ad hoc team for 5000 episodes, after 600 episodes of updating pre-trained SFs for the new ad hoc team (i.e., 300 episodes for each pre-trained policy and its corresponding SFs). We also include the mean return of each pre-trained policy and the performance of the SFQL baseline. We see that our method (GSAT) shows improved performance over each pre-trained policy, as expected given Theorem 1. We also see that SFQL requires about 2500 episodes before it surpasses the performance of GSAT (or about 1900 episodes when we include the required SF updating period of GSAT). However, Figure 3 shows that GSAT's performance strongly depends on the accuracy of the updated SFs, as reducing the allowed number of episodes for SF updating decreases performance.

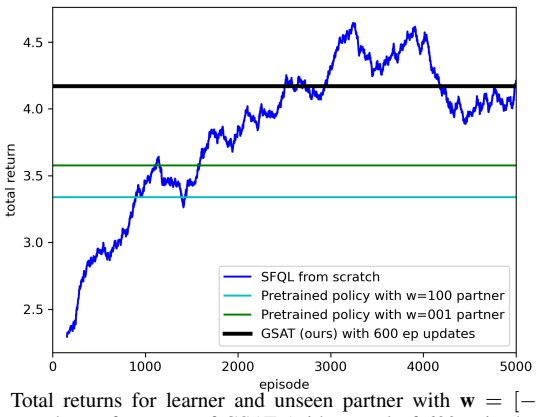


Fig. 2. Total returns for learner and unseen partner with $\mathbf{w} = [-1, 1, 0]$. We compare the performance of GSAT (with a total of 600 episode updates beforehand) with SFQL from scratch and each pre-trained policy. Fixed policy performance was averaged over all episodes.

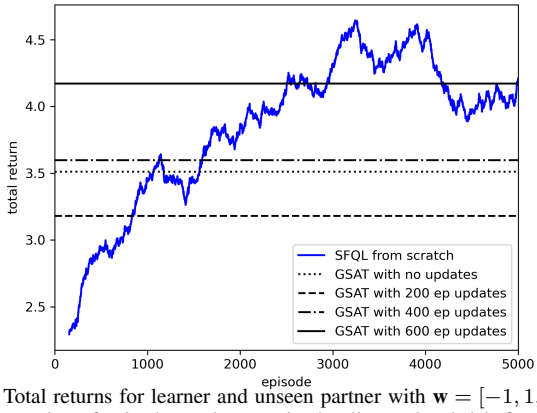


Fig. 3. Total returns for learner and unseen partner with $\mathbf{w} = [-1, 1, 0]$. We vary the number of episodes each pre-trained policy updated $\psi$ beforehand to account for the dynamic changes induced by the new teammate.

## VI. Conclusion

In this work, we proposed GSAT, a novel approach for AHT that leverages GPI and SFs. In contrast to previous methods that use a single policy from a pre-trained set, we leverage the entire set of pre-trained policies at every time step by applying a GPI policy with SFs. We empirically demonstrate that this more exhaustive use of prior knowledge improves performance in an object-collecting coordination environment relative to baselines. Building from existing work, we also derive performance bounds for our method. Future research directions include developing more sample efficient ways to update pre-trained SFs for a new task with different dynamics.

## REFERENCES


[1] P. Stone, G. Kaminka, S. Kraus, and J. Rosenschein, "Ad hoc autonomous agent teams: Collaboration without pre-coordination," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, pp. 1504–1509, Jul. 2010.

[2] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel, "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, (Richland, SC), p. 2085–2087, International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[3] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *J. Mach. Learn. Res.*, vol. 21, Jan. 2020.

[4] R. Mirsky, I. Carlucho, A. Rahman, E. Fosong, W. Macke, M. Sridharan, P. Stone, and S. V. Albrecht, "A survey of ad hoc teamwork research," in *Multi-Agent Systems* (D. Baumeister and J. Rothe, eds.), (Cham), pp. 275–293, Springer International Publishing, 2022.

[5] S. Barrett, A. Rosenfeld, S. Kraus, and P. Stone, "Making friends on the fly: Cooperating with new teammates," *Artificial Intelligence*, vol. 242, pp. 132–171, 2017.

[6] H. Li, T. Ni, S. Agrawal, F. Jia, S. Raja, Y. Gui, D. Hughes, M. Lewis, and K. Sycara, "Individualized mutual adaptation in human-agent teams," *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 6, pp. 706–714, 2021.

[7] A. Barreto, D. Borsa, J. Quan, T. Schaul, D. Silver, M. Hessel, D. Mankowitz, A. Zidek, and R. Munos, "Transfer in deep reinforcement learning using successor features and generalised policy improvement," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 501–510, PMLR, 10–15 Jul 2018.

[8] A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. P. van Hasselt, and D. Silver, "Successor features for transfer in reinforcement learning," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[9] A. Barreto, S. Hou, D. Borsa, D. Silver, and D. Precup, "Fast reinforcement learning with generalized policy updates," *Proceedings of the National Academy of Sciences*, vol. 117, pp. 30079–30087, Dec. 2020. Publisher: Proceedings of the National Academy of Sciences.

[10] S. Hansen, W. Dabney, A. Barreto, D. Warde-Farley, T. V. de Wiele, and V. Mnih, "Fast task inference with variational intrinsic successor features," in *International Conference on Learning Representations*, 2020.