Multi-Agent Reinforcement Learning with Temporal Logic Specifications

Lewis Hammond
University of Oxford
lewis.hammond@cs.ox.ac.uk

Alessandro Abate
University of Oxford
aabate@cs.ox.ac.uk

Julian Gutierrez
Monash University
julian.gutierrez@monash.edu

Michael Wooldridge
University of Oxford
mjw@cs.ox.ac.uk

ABSTRACT

In this paper, we study the problem of learning to satisfy temporal logic specifications with a group of agents in an unknown environment, which may exhibit probabilistic behaviour. From a learning perspective these specifications provide a rich formal language with which to capture tasks or objectives, while from a logic and automated verification perspective the introduction of learning capabilities allows for practical applications in large, stochastic, unknown environments. The existing work in this area is, however, limited. Of the frameworks that consider full linear temporal logic or have correctness guarantees, all methods thus far consider only the case of a single temporal logic specification and a single agent. In order to overcome this limitation, we develop the first multi-agent reinforcement learning technique for temporal logic specifications, which is also novel in its ability to handle multiple specifications. We provide correctness and convergence guarantees for our main algorithm – Almanac (Automaton/Logic Multi-Agent Natural Actor-Critic) – even when using function approximation. Alongside our theoretical results, we further demonstrate the applicability of our technique via a set of preliminary experiments.

KEYWORDS

multi-agent reinforcement learning; temporal logic; automata; formal methods; multi-objective reinforcement learning

ACM Reference Format:
Lewis Hammond, Alessandro Abate, Julian Gutierrez, and Michael Wooldridge. 2021. Multi-Agent Reinforcement Learning with Temporal Logic Specifications. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 10 pages.

1 INTRODUCTION

Much recent work from the control and machine learning communities has considered the task of learning to satisfy temporal logic specifications in unknown environments [9, 17, 19, 21, 22, 29–31, 38, 45, 52, 53, 56]. In these frameworks the agent is given a goal, typically specified using Linear Temporal Logic (LTL), and the dynamics of the agent's environment are assumed to be captured by some unknown Markov Decision Process (MDP). The task of the agent is then to learn a policy that maximises the probability of satisfying the LTL specification. Importantly, the proposed Reinforcement Learning (RL) algorithms are *model-free*, and so do not require evaluating the LTL specification against a model of the MDP (as is typically done in probabilistic model-checking, for instance), allowing for greater flexibility and scalability. These techniques have several advantages. From the perspective of RL, LTL forms an expressive and compact language with which to express infinite-horizon rewards that may be non-Markovian or exhibit special logical structure, and has provided a basis for new reward signal languages [23, 33]. From the perspective of logic, control and automated verification, the introduction of learning allows system designers to ensure that agents satisfy certain desirable properties in large, stochastic, unknown environments [22].

The existing work in this area is, however, limited. Of the frameworks that consider full LTL or have correctness guarantees, all methods thus far consider only the case of a single specification and agent. Modern AI and control systems on the other hand are increasingly multi-agent and often multi-objective. Simply applying single-agent learning algorithms in a multi-agent setting can lead to poor performance and a lack of convergence [11, 60]. Furthermore, even in the single-agent setting, no previous work has provided any correctness guarantees when using function approximation, which is crucial for scenarios that require both rigour *and* scalability.

1.1 Related Work

Our contributions in this paper draw on many areas. The most closely related of these is a recent line of work investigating the problem of learning to satisfy temporal logic specifications in MDPs. These works can in turn be partitioned by whether they focus on full LTL or on a fragment of LTL. Within the former category, early approaches used model-based algorithms and encoded LTL specifications using Deterministic Rabin Automtata (DRAs) [17, 45]. To overcome scalability issues resulting from DRAs and model-based algorithms, later works employed model-free algorithms and Limit-Deterministic Büchi Automata (LDBAs) [9, 19, 21, 38] and in some cases function approximation [22]. However, these works only consider the case of a single specification and a single agent, and none have provided correctness guarantees when using function approximation.

Other works have instead restricted their attention to *fragments* of LTL [30, 31, 56]. One strand of research uses 'reward machines' (finite state transducers) to capture finite-horizon objectives to allow for a natural decomposition of tasks [52], and also for the introduction of multiple objectives [53]. Concurrently with this work,

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

one recent effort has sought to generalise reward machines to the multi-agent and multi-objective case [29]. However, this approach simply optimises the conjunction of all objectives via a single reward machine and independent Q-learning, which is well-known to suffer from convergence issues and sub-optimality in the multi-agent setting [11]. Besides not supporting full LTL, the methods that use function approximation lack theoretical guarantees.

Similar problems to the one we tackle in this work have been considered by the verification community. Brázdil et al. propose a Probably Approximate Correct (PAC) Q-learning algorithm for unbounded reachability properties in tabular settings with a single agent and single objective [10]. Probabilistic or statistical model-checking algorithms have also been proposed for Markov Games (MGs), although so far these only handle known models and highly restricted forms of game, such as the turn-based two-player case [4], or those that are composed of two coalitions of players and can thus be reduced to a two-player game [28]. Related paradigms such as *rational verification* [57] and *rational synthesis* [15] only consider non-stochastic games without learning agents.

Finally, our work can also be viewed in the context of the RL, Multi-Agent RL (MARL), and game theory literature [3, 8, 12, 32, 42, 55]. The main algorithm we develop in this paper, Almanac (Automaton/Logic Multi-Agent Natural Actor-Critic), builds upon natural actor-critic algorithms [6, 40, 51] and generalises this to the multi-agent setting via the derivation of a multi-agent natural gradient. Multi-agent actor-critic algorithms enjoy state-of-the-art performance [16, 34] and have also been the focus of efforts to provide theoretical guarantees of convergence [39, 43, 59]. We refer the reader to Zhang et al. for a recent survey of MARL [58] and to Nowé et. al for a more game-theoretic perspective [37]. All of these works, however, use traditional scalar reward functions, whereas we focus on satisfying temporal logic formulae that provide a rich and rigorous language in which to express complex tasks and specifications over potentially infinite horizons.

1.2 Contribution

We overcome the limitations described above by proposing *the first multi-agent reinforcement learning algorithm for temporal logic specifications* with correctness and convergence guarantees, even when using function approximation. Generalising from the single-objective, single-agent, non-approximate framework to the multi-objective, multi-agent, approximate setting is far from trivial and introduces several new challenges. We provide theoretical solutions to these challenges in the form of a new algorithm, Almanac, which provably converges to either locally or globally optimal joint policies with respect to multiple LTL specifications, depending on whether agents use local or global policies, respectively (the notions of local and global are made precise in later sections). We also evaluate our algorithm against ground-truth probabilities using PRISM, a state-of-the-art probabilistic model-checker [27].

We proceed as follows. In Section 2 we provide the requisite technical background on MARL and LTL and in Section 3 we formalise our problem statement. We then introduce our full algorithm in Section 4 and report briefly on our experiments in Section 5. Full proofs are relegated to the supplementary material.[1]

[1] Available online at `https://arxiv.org/abs/2102.00582`.

2 PRELIMINARIES

Unless otherwise indicated we use superscripts $i \in N$ to denote affiliation with a player i, or $j \in M$ to denote affiliation with a specification φ^j, and with subscripts $t \in \mathbb{N}$ to index variables through time. We denote true or optimal versions of functions or quantities using superscripts *, and approximate versions using superscripts $\hat{}$.

2.1 Multi-Agent Reinforcement Learning

Markov games (MGs), also known as (concurrent) stochastic games, are the *lingua franca* of MARL, in much the same way that MDPs are for standard RL [32]. In this setting the game proceeds, at each time step t, from a state s_t by each player i selecting an action a^i_t, after which a new state s_{t+1} is reached and individual rewards r^i_{t+1} are received. Formally, we have the following statement.

Definition 1. A (finite) **Markov Game (MG)** is a tuple $G = (N, S, A, T, \gamma, R)$ where $N = \{1, \dots, n\}$ is a set of players, S is a (finite) state space, $A = \{A^1, \dots, A^n\}$ is a set of finite action spaces, $T : S \times A^1 \times \cdots \times A^n \times S \to [0, 1]$ is a stochastic transition function, $\gamma \in (0, 1)$ is an (optional) discount rate, and $R = \{R^1, \dots, R^n\}$ is a set of reward functions defined as $R^i : S \times A^1 \times \cdots \times A^n \times S \to \mathbb{R}$. A (memoryless) **policy** $\pi^i : S \times A^i \to [0, 1]$ maps states to a distribution over player i's actions. If the range of π^i is in fact $\{0, 1\}$ then we say that π is **deterministic**. A **joint policy** $\pi = (\pi^1, \dots, \pi^n)$ is the combined policy of all players in N. We denote also by $\pi^{-i} = (\pi^1, \dots, \pi^{i-1}, \pi^{i+1}, \dots, \pi^n)$ the joint policy without player i.

Each player's objective in an MG is to maximise their cumulative discounted expected reward over time, given that the other players are playing some joint policy π^{-i}. Observe that given a starting state s a joint policy π induces a Markov chain $\Pr^\pi_G(\cdot|s)$ over the states of the MG. By taking the expectation over time of this Markov chain we define the *value function* as $V^{i*}_\pi(s) := \mathbb{E}_\pi[\sum^\infty_{t=0} \gamma^t r^i_{t+1} \mid s]$ where $R^i(s_t, a^1, \dots, a^n, s_{t+1}) = r^i_{t+1}$. The core solution concept in MGs is that of a Markov Perfect Equilibrium (MPE) [35]. Informally, in the games we consider, an MPE is a set of memoryless strategies that forms a Nash Equilibrium when starting from any state.

Definition 2. Consider an MG G. For each agent i and joint policy π^{-i}, a policy π^i is a **best response** to π^{-i} if it is in the set $BR^i(\pi^{-i}) := \{\pi^i : V^{i*}_{(\pi^i,\pi^{-i})}(s) = \max_{\bar{\pi}^i} V^{i*}_{(\bar{\pi}^i,\pi^{-i})}(s), \forall s \in S\}$. A joint policy $\pi = (\pi^1, \dots, \pi^n)$ in G is a **Markov Perfect Equilibrium (MPE)** if $\pi^i \in BR^i(\pi^{-i})$ for all $i \in N$. If, in addition, we have that $V^{i*}_\pi(s) = \max_{\bar{\pi}} V^{i*}_{\bar{\pi}}(s)$ for all $s \in S$ and for all $i \in N$, then we say that π is **team-optimal**. If there exists a team-optimal joint policy in G, then we call G a **common-interest game**.

Intuitively, a common-interest game captures a setting in which there is a joint policy under which 'everyone is happy'. In many games no such policy exists, and so a trade-off may be necessary. We may thus instead wish to maximise a weighted sum of rewards $V^*_\pi(s) = \sum_{i\in N} w[i] V^{i*}_\pi(s)$. In this general and popular setting (which generalises both team and common-interest games) that we adopt for the remainder of the paper we describe a joint policy π as *locally optimal* if it forms an MPE and *globally optimal* if that MPE is team-optimal. Maximising a weighted sum of rewards means that there is always a *deterministic* optimal joint policy [49] (as we can view a joint policy as a policy for a single agent in an MDP), and hence a deterministic MPE, in the games we consider.

2.2 Linear Temporal Logic

When defining specifications for a system (e.g., tasks for an agent), a natural idea is to introduce requirements on the possible traces that may arise as the system executes over time. LTL captures this idea and provides a logic for reasoning about the properties of such traces [41], which here we view as infinite paths ρ through a state space S, where each $\rho[t] \in S$ for $t \in \mathbb{N}$ and $\rho[t..]$ denotes the path ρ from time t onwards. Additionally, we introduce a set of atomic propositions AP and a labelling function $L : S \to \Sigma$ where $\Sigma = 2^{AP}$.

Definition 3. The syntax of **Linear Temporal Logic (LTL)** formulae is defined recursively using the following operators:

$$\varphi ::= \top \mid \alpha \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathsf{X}\,\varphi \mid \varphi \,\mathsf{U}\, \varphi$$

where $\alpha \in AP$ is an atomic proposition and $\top$ is read as 'true'. The semantics of said formulae are also defined recursively:

$$\begin{aligned}
&\rho \models \top \\
&\rho \models \alpha && \Leftrightarrow\ \alpha \in L(\rho[0]) \\
&\rho \models \psi_1 \wedge \psi_2 && \Leftrightarrow\ \rho \models \psi_1 \text{ and } \rho \models \psi_2 \\
&\rho \models \neg\psi && \Leftrightarrow\ \rho \not\models \psi \\
&\rho \models \mathsf{X}\,\psi && \Leftrightarrow\ \rho[1..] \models \psi \\
&\rho \models \psi_1 \,\mathsf{U}\, \psi_2 && \Leftrightarrow\ \exists t \in \mathbb{N} \text{ s.t. } \rho[t..] \models \psi_2, \forall t' \in [0,t),\ \rho[t'..] \models \psi_1
\end{aligned}$$

Alongside the standard operators from propositional logic, from which we may derive $\vee$, $\to$, and $\leftrightarrow$, we have the temporal operators X ('next') and U ('until'), from which we may derive $\mathsf{F}\,\varphi \equiv \top \,\mathsf{U}\, \varphi$ ('finally') and $\mathsf{G}\,\varphi \equiv \neg\,\mathsf{F}\,\neg\varphi$ ('globally'). An LTL formula φ thus describes a set of infinite traces $\{\rho \in S^\omega : \rho \models \varphi\}$ through S. Alternatively, one may encode such a set by using an *automaton*.

Definition 4. A **Non-deterministic Büchi Automaton (NBA)** is a tuple $B = (Q, q_0, \Sigma, F, \delta)$ where Q is a finite set of states, $q_0 \in Q$ is the initial state, $\Sigma = 2^{AP}$ is a finite alphabet over a set of atomic propositions AP, $F \subseteq Q$ is a set of accepting states, and $\delta : Q \times \Sigma \to 2^Q$ is a (non-deterministic) transition function. We say that an **infinite word** $w \in \Sigma^\omega$ is **accepted** by B if there exists an **infinite run** $\rho \in Q^\omega$ such that $\rho[0] = q_0$, $\rho[j+1] \in \delta(\rho[j], \omega[j])$ for all $j \in \mathbb{N}$, and we have $\inf(\rho) \cap F \neq \emptyset$, where $\inf(\rho)$ is the set of states in Q that are visited infinitely often on run ρ.

In this work, we use a specific variant of NBAs, called Limit-Deterministic Büchi Automata (LDBAs). Intuitively, LDBAs relegate all non-determinism to a set of $\mathcal{E}$-transitions between two halves of the automaton, an initial component Q_I and an accepting component $Q_A \supseteq F$. This level of non-determinism is, perhaps surprisingly, sufficient for encoding any LTL formula. We refer the reader to Sickert et al. for details of this LTL-to-LDBA conversion process [46], which often yields smaller automata for formulas with deep nesting of modal operators compared to other approaches.

Definition 5. A **Limit-Deterministic Büchi Automaton (LDBA)** is an NBA $B = (Q, q_0, \Sigma \cup \{\mathcal{E}\}, \delta, F)$ where Q can be partitioned into two disjoint subsets Q_I and Q_A such that: $|\delta(q, \alpha)| = 1$ for every $q \in Q$ and every $\alpha \in \Sigma$; $\delta(q, \mathcal{E}) = \emptyset$ for every $q \in Q_A$; $\delta(q, \alpha) \subseteq Q_A$ for every $q \in Q_A$ and every $\alpha \in \Sigma$; and $F \subseteq Q_A$.

3 PROBLEM STATEMENT

We now combine MARL and LTL to consider the task of learning to satisfy temporal logic specifications with maximal probability in unknown multi-agent environments. The problem we seek to address in this work is:

> Given an (unknown) environment with a team of n agents characterised as an MG G, and a set of m LTL specifications, compute (without first learning a model) a joint policy π that maximises a weighted sum of the probabilities of satisfying each of the LTL specifications.

To formalise this problem, we first define the satisfaction probability of π in G with respect to an LTL specification φ.

Definition 6. Given an MG G and a joint policy π, denote by $\Pr^\pi_G(\cdot|s)$ the induced Markov chain over the states of G starting from s. Then, given an LTL formula φ, the **satisfaction probability** of π in G with respect to φ starting from a state s is given by $\Pr^\pi_G(s \models \varphi) \coloneqq \Pr^\pi_G(\{\rho : \rho \models \varphi\} \mid \rho[0] = s)$.

Thus, our problem can be formally expressed as computing a policy π^* in an unknown MG G, given a set of LTL specifications $\{\varphi^j\}_{0 \le j \le m}$ and vector of weights w of length m, such that:

$$\pi^* \in \operatorname*{argmax}_{\pi} \sum_j w[j] \Pr^{\pi}_{G}(s \models \varphi^j) \qquad \forall s \in S$$

This forms a natural extension of the single-agent single-objective case, in which one agent seeks to compute a policy that maximises the probability of satisfying a single LTL specification.

Our solution to this problem crucially relies on the definition of a *product game* which, while never explicitly constructed, defines the full environment over which our agents learn. Note that in the following definition we consider an MG with a generic discount rate γ and reward functions $R^\otimes$, though in our algorithm we redefine these to capture the original LTL specification, as in similar single-agent constructions [46]. The idea behind this construction is that by learning to act optimally in the (implicit) product game, agents learn to satisfy the LTL specification(s) in the original game.

Definition 7. Given an LDBA $B = (Q, q_0, \Sigma \cup \{\mathcal{E}\}, \delta, F)$ associated with a set of agents $N^B \subseteq N$, where $Q = Q_I \cup Q_A$, a (finite) MG $G = (N, S, A, T, \gamma, R)$, and a labelling function $L : S \to \Sigma$, the resulting **Product MG** is a tuple $G \otimes B = G_B = (N, S^\otimes, A^\otimes, T^\otimes, \gamma, R^\otimes)$ where: $S^\otimes = S \times Q$ is a product state space; $A^\otimes = \{A^1_\otimes, \dots, A^n_\otimes\}$ where each $A^i_\otimes = A^i \cup \{\mathcal{E}_{q'} \mid \exists q \in Q_I, q' \in \delta(q, \mathcal{E})\}$ for $i \in N^B$ and $A^i_\otimes = A^i$ otherwise; $T^\otimes : S^\otimes \times A^1_\otimes \times \cdots \times A^n_\otimes \times S^\otimes \to [0,1]$ is a stochastic transition function such that $T^\otimes((s,q), a^1, \dots, a^n, (s', q')) =$

$$\begin{cases} T(s, a^1, \dots, a^n, s') & \text{if } \forall i \in N^B, a^i \in A^i, q' \in \delta(q, L(s')) \\ 1 & \text{if } \exists i \in N^B \text{ s.t. } a^i = \mathcal{E}_{q'}, q' \in \delta(q, \mathcal{E}), s = s' \\ 0 & \text{otherwise} \end{cases}$$

and $R^\otimes$ is a set of reward functions $\{R^1_\otimes, \dots, R^n_\otimes\}$ such that $R^i_\otimes : S^\otimes \times A^1_\otimes \times \cdots \times A^n_\otimes \times S^\otimes \to \mathbb{R}$ for each $i \in N$. A (memoryless) **policy** $\pi^i : S^\otimes \times A^i_\otimes \to [0,1]$ for a player i in the product MG is defined as before, using $S^\otimes$ and $A^\otimes$.

We also extend the initial state distribution ζ to $\zeta^{\otimes}$ in the product game, where $\zeta^{\otimes}(s, q_0^1, \dots, q_0^m) = \zeta(s)$ for all $s \in S$ and is equal to 0 for all other $s^{\otimes} \in S^{\otimes}$. We write $G \otimes B^1 \otimes \cdots \otimes B^m = G_{B^1,\dots,B^m}$ for the product of G with multiple automata $B^1, \dots, B^m$, defined by sequentially taking individual products (as a product MG is simply another MG). In fact, given G, this operation can easily be seen to be associative (up to the ordering of elements forming a product state) if we assume that: $L^j(s, q^1, \dots, q^{j-1}) = L^j(s) \subseteq \Sigma^j$ only depends on the state of G for each labelling function L^j (for automaton B^j); and that $\mathcal{E}$-transitions can be made for multiple automata at the same time step, i.e., there is no order in which $\mathcal{E}$-transitions are prioritised between groups N^{B^j} when defining the new product transition function $T^{\otimes}$. At each time step t every agent in some set N^{B^j} has the opportunity to make an $\mathcal{E}$-transition at which point their corresponding automaton state q_t^j changes to q_{t+1}^j with probability one and other elements of the product state remain the same. If no $\mathcal{E}$-transitions are made by any agent in any set N^{B^j} then the transition probabilities are simply defined by the original transition function. Previous works have considered a similar multi-objective product construction, though only in the simpler case of a single agent [13].

4 AUTOMATON/LOGIC MULTI-AGENT NATURAL ACTOR-CRITIC

We now present our solution to the problem statement, in the form of our algorithm, Almanac (Automaton/Logic Multi-Agent Natural Actor-Critic). Almanac falls into a category of model-free RL algorithms known as *actor-critic* methods [6, 26, 40], whereby a policy π (the actor) is optimised via gradient descent using the value function V_π (the critic) which is updated via bootstrapping. These two functions are typically learnt separately and simultaneously using a two-timescale approach in which the critic is updated faster than the actor in order to learn the value function with respect to the current policy. Such methods form a powerful, flexible, and highly scalable class of algorithms which can be applied to a wide range of (MA)RL problems and regularly achieve state-of-the-art performance [16, 34]. We begin by introducing a novel temporal difference (TD) algorithm with state-dependent discounts such that the learnt critics capture the LTL specifications. We then combine this with a natural policy gradient scheme to update the actors, forming our full algorithm. In the final subsection we provide proof sketches of correctness and convergence, with full proofs available in the supplementary material.

4.1 Patient Temporal Difference Learning

We wish to solve the problem of learning a critic V_π given any fixed joint policy π such that for any $s \in S$:

$$\pi^* \in \operatorname*{argmax}_{\pi} V_\pi^*(s^{\otimes}) \Rightarrow \pi^* \in \operatorname*{argmax}_{\pi} \sum_j w[j] \Pr_G^{\pi}(s \models \varphi^j), \quad (1)$$

where $s^{\otimes} = (s, q_0^1, \dots, q_0^m)$. The main idea is that by defining a new reward function $R^{\otimes}$ and discount rate Γ we can learn a value function V_π in the product MG $G_{B^1,\dots,B^m}$ (where B^j is the LDBA corresponding to φ^j) such that any policy π that maximises V_π^* in $G_{B^1,\dots,B^m}$ is guaranteed to maximise $\sum_j w[j] \Pr_G^{\pi}(s \models \varphi^j)$ when projected down into the original game G. In this way, the states of the automata $B^1, \dots, B^m$ can be thought of as a finite memory for π in the original game.

Remark 1. An MPE in our setting is simply a Subgame Perfect Equilibrium (SPE) in which all players use memoryless strategies, where a subgame in an MG is defined by a starting state [18]. If (1) holds, then any joint policy $\pi^* \in \operatorname{argmax}_\pi V_\pi^*(s^{\otimes})$ forms an MPE in the product game, but when viewed in terms of the original game, a policy $\pi^* \in \operatorname{argmax}_\pi \sum_j w[j] \Pr_G^{\pi}(s \models \varphi^j)$ for all s is merely an SPE, as the policies of each agent are no longer memoryless.

The problem defining $R^{\otimes}$ and Γ such that the limit V_π^* of the learnt value function V_π satisfies (1) is trickier than it might initially seem. Previous approaches for MDPs have either been open to counterexamples in which an agent learns to prioritise the length of the path taken to satisfy φ over the probability of satisfying it [21], or involved constructions that hinder learning by increasing the state-space size [38], increasing reward sparsity [19], or increasing learning rates [9]. We propose a novel solution that is far simpler and more natural. Given a state $s^{\otimes} = (s, q^1, \dots q^m)$ the basic idea is, for each automaton, to issue a reward when $q^j \in F^j$ and to use a *state-dependent* discount factor which is equal to 1 when no reward is seen and equal to a constant $\gamma_V \in (0, 1)$ otherwise. Formally, for each specification φ^j we define $R_{\otimes}^j$ and Γ^j as follows:

$$R_{\otimes}^j(s^{\otimes}) := \begin{cases} 1 & \text{if } q^j \in F^j \\ 0 & \text{otherwise}, \end{cases} \qquad \Gamma^j(s^{\otimes}) := \begin{cases} \gamma_V & \text{if } R_{\otimes}^j(s^{\otimes}) = 1 \\ 1 & \text{otherwise}. \end{cases} \quad (2)$$

We then define $V_\pi^{j*}(s^{\otimes}) := \mathbb{E}_\pi\big[\sum_{t=0}^{\infty} \Gamma_{1:t}^j R_{\otimes}^j(s_{t+1}^{\otimes}) \mid s^{\otimes}\big]$ where $\Gamma_{1:t}^j = \prod_{\tau=1}^{t} \Gamma^j(s_\tau^{\otimes})$ for $t \geq 1$ and and $\Gamma_{1:0}^j = 1$. While earlier works have considered a similar solution in MDPs [20, 21], they fail to mention that the problem with this scheme is that if used to update a value function naively (such as in the vanilla Q-learning algorithms these works make use of) the update process can converge to the wrong values. This lack of convergence arises because of possible loops in the product MG that do not contain any rewarding states. To overcome this limitation we use a *patient TD* scheme whereby agents update their estimates of the value function only once a reward is seen (or when the next state has value 0), meaning that value estimates for states on such loops cannot be artificially inflated.

We begin by considering the standard TD(0) update rule with learning rate $\alpha = \{\alpha_t\}_{t \in \mathbb{N}}$ and fixed policy π given by [48]:

$$V_\pi^j(s_t^{\otimes}) \leftarrow (1 - \alpha_t) V_\pi^j(s_t^{\otimes}) + \alpha_t \big[R_{\otimes}^j(s_{t+1}^{\otimes}) + \Gamma_{t+1:t+1}^j V_\pi^j(s_{t+1}^{\otimes})\big],$$

where $R_{\otimes}^j(s_{t+1}^{\otimes}) + \Gamma(s_{t+1}^{\otimes}) V_\pi^j(s_{t+1}^{\otimes}) =: G_{t:t+1}^j$ is a *one-step target*, though one may also use a *k-step target* instead, given by:

$$\begin{aligned} G_{t:t+k}^j &:= R_{\otimes}^j(s_{t+1}^{\otimes}) + \Gamma_{t+1:t+1}^j R_{\otimes}^j(s_{t+2}^{\otimes}) + \cdots \\ &\quad + \Gamma_{t+1:t+k-1}^j R_{\otimes}^j(s_{t+k}^{\otimes}) + \Gamma_{t+1:t+k}^j V_\pi^j(s_{t+k}^{\otimes}). \end{aligned}$$

It is well-known that using longer trajectories as targets can improve bootstrapping as much more can be learnt from a single episode [49]. Our motivation is different: by not immediately updating $V_\pi(s_t^{\otimes})$ when we either do not see a reward, or when the value of the successor state $V_\pi(s_{t+1}^{\otimes})$ is non-zero, then we avoid increasing the values of zero-reward loop states in terms of themselves. Instead, we set k 'on the fly' to be the least k such that either $R^j_\otimes(s^\otimes_{t+k}) > 0$ or $V^j_\pi(s^\otimes_{t+k}) = 0$, i.e.,we wait to update $V^j_\pi(s^\otimes_t)$ until we see a reward. Observe that for any $0 < l < k$ we have $R^j_\otimes(s^\otimes_{t+l}) = 0$, $\Gamma^j(s^\otimes_{t+l}) = 1$, and $\Gamma^j(s^\otimes_{t+k}) = \gamma_V$, hence:

$$G_{t:t+k} = R^j_\otimes(s^\otimes_{t+k}) + \Gamma^j(s^\otimes_{t+k})V^j_\pi(s^\otimes_{t+k}) = R^j_\otimes(s^\otimes_{t+k}) + \gamma_V V^j_\pi(s^\otimes_{t+k}).$$

Almanac implements an (approximate) patient TD scheme to learn each value function V^j_π in the product MG under a joint policy π by maintaining a temporary set of *zero-reward states* $\{s^\otimes_t, \dots, s^\otimes_{t+k-1}\}$ (with respect to $R^j_\otimes$) whose values it waits to update. The convergence of this update rule is proven in Theorem 1. What remains to show here is that the resulting $V_\pi := \sum_j w[j]V^j_\pi$ satisfies (1), which gives us a critic for correctly *capturing* the given LTL specifications. In the next subsection we show how this critic can be learnt synchronously alongside an actor (i.e., a joint policy) for optimally *satisfying* said specifications.

Proposition 1. *Given an MG G and LTL objectives $\{\varphi^j\}_{1\le j\le m}$ (each equivalent to an LDBA B^j), let $G_B = G \otimes B^1 \otimes \cdots \otimes B^m$ be the resulting product MG with newly defined reward functions $R^j_\otimes$ and state-dependent discount functions Γ^j given by (2). Then there exists some $0 < \gamma_V < 1$ such that (1) is satisfied by the patient value function $V_\pi := \sum_j w[j]V^j_\pi$.*

Proof (Sketch). We begin by observing that:

$$V^*_\pi(s^\otimes) = \sum_j w[j] \sum_\rho \left[Pr^\pi_{G_B}(\rho|s^\otimes) \sum_{t=0}^{\infty} \Gamma^j_{1:t} R^j_\otimes(s^\otimes_{t+1}) \right].$$

We denote the number of times a path ρ in G_B passes through the accepting set F^j of automaton B^j by $F^j(\rho)$, and let $F(\rho) = \sum_j F^j(\rho)$. Then when $F^j(\rho) = \infty$ we have that $\sum_{t=0}^\infty \Gamma^j_{1:t} R^j_\otimes(s^\otimes_{t+1}) = \frac{1}{1-\gamma_V}$ and when $F^j(\rho) = f^j < \infty$ we have that $\sum_{t=0}^\infty \Gamma^j_{1:t} R^j_\otimes(s^\otimes_{t+1}) = (1-\gamma_V^{f^j})\frac{1}{1-\gamma_V}$. We show that if $\pi \notin \operatorname{argmax}_\pi \sum_j w[j] \Pr^\pi_G(s \models \varphi^j)$ then there exists $0 < \gamma_V < 1$ such that $\pi \notin \operatorname{argmax}_\pi V^*_\pi(s^\otimes)$ and for some $\pi' \in \operatorname{argmax}_\pi \sum_j w[j] \Pr^\pi_G(s \models \varphi^j)$, we have $V^*_{\pi'}(s^\otimes) > V^*_\pi(s^\otimes)$. Define the sets $\mathrm{fin}^j(s^\otimes) := \{\rho : \rho[0] = s^\otimes \wedge F^j(\rho) \neq \infty\}$ and $\mathrm{inf}^j(s^\otimes) := \{\rho : \rho[0] = s^\otimes \wedge F^j(\rho) = \infty\}$. Then we have:

$$V^*_{\pi'}(s^\otimes) \ge \sum_j w[j] \left[\sum_{\rho\in\mathrm{inf}^j(s^\otimes)} Pr^{\pi'}_{G_B}(\rho|s^\otimes)\frac{1}{1-\gamma_V} \right] = \frac{a^\top w}{1-\gamma_V},$$

where a is a vector such that $a[j] = \sum_{\rho\in\mathrm{inf}^j(s^\otimes)} Pr^{\pi'}_{G_B}(\rho|s^\otimes) = Pr^{\pi'}_{G_B}\big(\mathrm{inf}^j(s^\otimes)\big)$. Similarly, we have:

$$V^*_\pi(s^\otimes) \le \frac{b^\top w}{1-\gamma_V} + (1-b)^\top w \frac{1-\gamma_V^f}{1-\gamma_V},$$

where $b[j] = \sum_{\rho\in\mathrm{inf}^j(s^\otimes)} Pr^\pi_{G_B}(\rho|s^\otimes) = Pr^\pi_{G_B}(\mathrm{inf}^j(s^\otimes))$ and $f = \max_j \max_{\rho\in\mathrm{fin}^j(s^\otimes)} F^j(\rho)$. Given that $\sum_j w[j]\Pr^{\pi'}_G(s \models \varphi^j) > \sum_j w[j]\Pr^\pi_G(s \models \varphi^j)$ by assumption, then by a straightforward extension of the result (see the supplementary material and [46]) that there exists a canonical extension of π to G_B such that $\Pr^\pi_G(s \models \varphi) = \Pr^\pi_{G_B}(\{\rho : F(\rho) = \infty\}|s^\otimes)$, we have $1 \ge a^\top w > b^\top w \ge 0$. The proof is concluded by setting $\gamma_V > \sqrt[f]{\frac{1-a^\top w}{1-b^\top w}}$. □

Finally, we remark that as well as a 'patient' value function V, we also learn a (standard) 'hasty' value function U. This is because, if for some specification φ and two possible joint policies π and π', we have that $\Pr^\pi_G(s \models \varphi) = \Pr^{\pi'}_G(s \models \varphi)$ then agents have no reason to use π over π', even if π results in a much more efficient trajectory. In many ways this is a feature and not a bug; LTL has no emphasis on 'hastiness' by design. In reality, however, whilst we may not want to forsake the satisfaction of a constraint for the sake of speed, we would like the agents to find the most efficient policy that maximises the probability of satisfying the constraint. We solve this problem by learning two value functions and then using *lexicographic RL* [44, 47] to maximise the hasty objective subject to maximising the patient objective, and thus satisfying (1). Further details are provided in the following section.

4.2 Multi-Agent Natural Actor-Critic

For the remainder of the paper, we assume that each agent's policy π^i is parameterised by $\theta^i \in \Theta^i$, potentially using a *non-linear* function approximator giving $\pi^i(a|s;\theta^i)$, and that the value functions are linearly approximated using some state basis functions $\phi(s^\otimes)$ and parameters v and u, such that $\hat{V}(s^\otimes) = \phi(s^\otimes)^\top v$ and $v = \sum_j w[j]v^j$ (and likewise for U). Note that these assumptions subsume the tabular setting which most prior work on RL with LTL specifications has focused on. We use $\theta = [\theta^{1\top}, \dots, \theta^{n\top}]^\top$ to denote the joint set of parameters for all agents and write joint actions as $a = (a^1, \dots, a^n)$. We also replace π by θ in our notation for clarity where appropriate.

When formulated in terms of our parametrisation, and given that (1) holds, our task can be viewed as optimising the objective function $J(\theta) := \sum_{s^\otimes} \zeta^\otimes(s^\otimes)V_\theta(s^\otimes) = \sum_j w[j]J^j(\theta)$ where $J^j(\theta) := \sum_{s^\otimes} \zeta^\otimes(s^\otimes)V^j_\theta(s^\otimes)$ is the objective for specification φ^j. Then, within $\operatorname{argmax}_\theta J(\theta)$, we also wish to select the parameters that maximise the objective function with respect to our 'hasty' value function, which is given by $K(\theta) := \sum_{s^\otimes} \zeta^\otimes(s^\otimes)U_\theta(s^\otimes) = \sum_j w[j]K^j(\theta)$. The following derivations are provided for J, the case for K is analogous. We can improve θ with respect to an objective function J using the gradient $\nabla_\theta J(\theta)$ which, by the policy gradient theorem [50], is:

$$\nabla_\theta J(\theta) = \sum_j w[j] \sum_{s^\otimes} d^j_{\theta,\zeta}(s^\otimes) \sum_a Q^j_\theta(s^\otimes, a)\nabla_\theta \pi(a|s^\otimes;\theta)$$

where $d^j_{\theta,\zeta}(s^\otimes)$ is the (patient) discounted state distribution in the product game for specification φ^j, given initial distribution $\zeta^\otimes$:

$$d^j_{\theta,\zeta}(s^\otimes) := \mathbb{E}_{s^\otimes_0\sim\zeta^\otimes}\left[\sum_\rho \Pr^\theta_{G_B}(\rho|s^\otimes_0) \left(\frac{1}{\sum_{t=0}^\infty \Gamma^j_{0:t}} \sum_{t=0}^\infty \Gamma^j_{0:t}\mathbb{I}(\rho[t] = s^\otimes) \right) \right]$$

where $\mathbb{I}$ denotes an indicator function. Where unambiguous we write simply d^j instead of $d^j_{\theta,\zeta}$.

Remark 2. Producing unbiased samples with respect to each $d^j(s^\otimes)$ in order to estimate the gradients used in policy evaluation and improvement raises several difficulties [36, 51]. It is possible, however, to instead use trajectories from the *undiscounted* MG distribution $d(s^\otimes)$ that are truncated after each transition to a state $s^\otimes$ with probability $1 - \Gamma^j(s^\otimes)$ [5], or to re-weight updates as a function of $\Gamma^j(s^\otimes)$ [51]. We combine these two approaches in Almanac.

A known problem with 'vanilla' gradients is that they can sometimes be inefficient due to large plateaus in the optimisation space, leading to small gradients and thus incremental updates. A solution to this problem is instead to use the *natural* policy gradient which is invariant to the parametrisation of the policy, and can be computed by applying the inverse Fisher matrix to the vanilla gradient [2, 24]. For each specification j, the natural gradient of $J^j(\theta)$ can be shown [40] to equal $\tilde{\nabla}_\theta J^j(\theta) = G^j(\theta)^{-1}\nabla_\theta J^j(\theta) = x_{V^j}$, where $G^j(\theta)$ is the Fisher information matrix and x_{V^j} satisfies:

$$\psi_\theta(a|s^\otimes)^\top x_{V^j} = Q^j_\theta(s^\otimes, a) - V^j_\theta(s^\otimes) =: A^j_\theta(s^\otimes, a).$$

Here, A^j_θ denotes the advantage function for specification j (for the hasty advantage function we use Z^j_θ) and $\psi_\theta(a|s^\otimes) = \nabla_\theta \log \pi(a|s^\otimes;\theta)$ denotes the score function. Using a similar line of reasoning a derivation of a natural policy gradient for the multi-agent case is a simple exercise (omitted here due to space constraints).

Lemma 1. *Let G_B be some (product) MG. Then for any set of parameters $\{\theta^i\}_{i\in N}$ and any player $i \in N$, the natural policy gradient for player i with respect to each $J^j(\theta) = \sum_{s^\otimes} \zeta^\otimes(s^\otimes) V^j_\theta(s^\otimes)$ is given by $\tilde{\nabla}_{\theta^i} J^j(\theta) = x^i_{V^j}$, where $x^i_{V^j}$ is a parameter satisfying $\psi^i_{\theta^i}(a^i|s^\otimes)^\top x^i_{V^j} = \nabla_{\theta^i} \log \pi^i(a^i|s^\otimes;\theta)^\top x^i_{V^j} = A^j_\theta(s^\otimes, a)$.*

This implies that the gradient x^i_V with respect to our weighted combination of objectives can be found by minimising the loss $L^i_V(x^i_V;\theta,\nu_{\theta,\zeta}) \coloneqq \sum_j w[j] L^i_{V^j}(x^i_V;\theta,\nu^j_{\theta,\zeta})$ where:

$$L^i_{V^j}(x^i_V;\theta,\nu^j_{\theta,\zeta}) \coloneqq \mathbb{E}_{(s^\otimes,a)\sim\nu^j_{\theta,\zeta}}\left[\left|\psi^i_{\theta^i}(a^i|s^\otimes)^\top x^i_V - A^j_\theta(s^\otimes,a)\right|\right]$$

and $\nu^j_{\theta,\zeta}(s^\otimes, a) \coloneqq d^j_{\theta,\zeta}(s^\otimes)\pi(a|s^\otimes;\theta)$. Similarly we may define $\mu_{\theta,\zeta}(s^\otimes, a) = c_{\theta,\zeta}(s^\otimes)\pi(a|s^\otimes;\theta)$ as the hasty state-action distribution under a joint policy θ and with initial distribution $\zeta^\otimes$, where:

$$c_{\theta,\zeta}(s^\otimes) \coloneqq \mathbb{E}_{s^\otimes_0\sim\zeta^\otimes}\left[\sum_\rho \Pr^\theta_{G_B}(\rho|s^\otimes_0)\left(\frac{1}{1-\gamma_U}\sum_{t=0}^\infty \gamma^t_U \mathbb{I}(\rho[t] = s^\otimes)\right)\right],$$

which does not depend on the specification φ^j because of the constant discount rate. As with d^j we drop subscripts for c, ν^j, and μ where unambiguous. Recall that our secondary objective is to optimise θ^i according to $K(\theta)$, given our lexicographic prioritisation of $J(\theta)$. In other words, we wish to follow the hasty natural gradient x^i_U subject to following the patient natural gradient x^i_V. Formally, the gradient we seek is given by:

$$x^i_* \in \operatorname{argmin}_{x^i_U \in \operatorname{argmin}_{x^i_V} L^i_V(x^i_V;\theta,\nu)} L^i_U(x^i_U;\theta,\mu), \tag{3}$$

where L^i_U is defined analogously to L^i_V. Note that both L^i_V and L^i_U are convex, and so we can find some x^i_* satisfying (3) by simply first following $\nabla_{x^i} L^i_V(x^i;\theta,\nu)$ until this gradient is zero, and then following $\nabla_{x^i} L^i_U(x^i;\theta,\mu)$ subject to the constraint that $\nabla_{x^i} L^i_V(x^i;\theta,\nu) = 0$. We use a multi-timescale *lexicographic* approach to perform this operation simultaneously and compute x^i_*, which we then use to update θ^i. More specifically, we minimise $L^i_V(x^i;\theta,\nu)$ on a faster timescale and so guarantee its convergence to some l^i before $L^i_U(x^i;\theta,\mu)$ has converged. On a slower timescale we solve the Lagrangian dual corresponding to the constrained optimisation problem of minimising $L^i_U(x^i;\theta,\mu)$ such that $L^i_V(x^i;\theta,\nu) - l^i \leq 0$:

$$\max_{\lambda^i\geq 0}\min_{x^i} L^i_U(x^i;\theta,\mu) + \lambda^i\left[L^i_V(x^i;\theta,\nu) - l^i\right]. \tag{4}$$

To form the gradients of L^i_V and L^i_U we use an unbiased estimate of each A^j_θ and Z^j_θ using samples of the TD error [6] which can be trivially extended to the k-step version $\delta^{V^j}_{t:t+k} = G^{V^j}_{t:t+k} - V^j_\theta(s^\otimes_t)$ [49]. We compute v by minimising the following loss for each v^j (the case for the u is analogous):

$$L^j_v(v^j;\theta,d^j) = \mathbb{E}_{s^\otimes\sim d^j}\left[\left(V^j_\theta(s^\otimes_t) - \phi(s^\otimes)^\top v^j\right)^2\right].$$

This can again be solved via gradient updates that use the temporal difference $\delta^{V^j}_{t:t+k}$, corresponding to the linear semi-gradient temporal difference algorithm [49]:

$$\begin{aligned}\hat{\delta}^{V^j}_{t:t+k} &\leftarrow r^j_{t+k} + \gamma_V \phi(s^\otimes_{t+k})^\top v^j - \phi(s^\otimes_t)^\top v^j \\ v^j &\leftarrow v^j + \alpha_t \hat{\delta}^{V^j}_{t:t+k}\phi(s^\otimes_t)\end{aligned} \tag{5}$$

$$\begin{aligned}\hat{\delta}^{U^j}_{t:t+1} &\leftarrow r^j_t + \gamma_U \phi(s^\otimes_{t+1})^\top u^j - \phi(s^\otimes_t)^\top u^j \\ u^j &\leftarrow u^j + \alpha_t \hat{\delta}^{U^j}_{t:t+1}\phi(s^\otimes_t).\end{aligned} \tag{6}$$

Using these quantities we then update the natural gradient x^i and Lagrange multiplier λ^i:

$$\begin{aligned}x^i &\leftarrow \Omega_{x^i}\left[x^i + \Big(\sum_j w[j](\beta^V_t + \beta^U_t\lambda^i)\chi^{V^j}_t + \beta^U_t\chi^{U^j}_t\Big)\psi^i_{\theta^i}(a^i_t|s^\otimes_t)\right], \\ \lambda^i &\leftarrow \Omega_\lambda\left[\lambda^i + \eta_t\Big(\sum_j w[j]\Gamma^j_{1:t}\left|\psi^i_{\theta^i}(a^i_t|s^\otimes_t)^\top x^i_V - \hat{\delta}^{V^j}_{t:t+1}\right| - l^i\Big)\right],\end{aligned} \tag{7}$$

where $\chi^{V^j}_t \coloneqq \Gamma^j_{1:t}\operatorname{sgn}\big(\hat{\delta}^{V^j}_{t:t+1} - \psi^i_{\theta^i}(a^i_t|s^\otimes_t)^\top x^i\big)$ is used to form an estimate of $-\nabla_{x^i} L^i_{V^j}(x^i;\theta,\nu^j_{\theta,\zeta})$ in a piecewise fashion. Finally we update the policy parameters using:

$$\theta^i \leftarrow \Omega_{\theta^i}\left[\theta^i + \iota_t x^i\right]. \tag{8}$$

The functions $\Omega_{x^i}, \Omega_\lambda, \Omega_{\theta^i}$ are projections onto X^i, $[0,\infty)$, and Θ^i respectively, which are commonly used within stochastic approximation to ensure boundedness of iterates, and is also standard in the literature on natural actor-critic algorithms [6, 7]. We assume that learning rates $\alpha, \beta^V, \beta^U, \eta, \iota$ obey the following relationship:

$$\begin{gathered}\sum_{t=0}^\infty \alpha_t = \sum_{t=0}^\infty \beta^V_t = \sum_{t=0}^\infty \beta^V_t = \sum_{t=0}^\infty \eta_t = \sum_{t=0}^\infty \iota_t = \infty, \\ \sum_{t=0}^\infty \left[(\alpha_t)^2 + (\beta^V_t)^2 + (\beta^U_t)^2 + (\eta_t)^2 + (\iota_t)^2\right] < \infty, \\ \lim_{t\to\infty}\frac{\beta^V_t}{\alpha_t} = \lim_{t\to\infty}\frac{\beta^U_t}{\beta^V_t} = \lim_{t\to\infty}\frac{\eta_t}{\beta^U_t} = 0.\end{gathered} \tag{9}$$

Intuitively, this means that critics update on the fastest timescale, followed by the patient updates to the natural gradient, the hasty updates to the natural gradient, and then the Lagrange multipliers. These updates occur in an inner loop, and the policy parameters themselves are updated on an outer loop, once the natural gradients have converged. The full procedure is shown in Algorithm 1.

Algorithm 1 ALMANAC

Input: specifications $\{\varphi^j\}_{0 \le j \le m}$, discount rates γ_V, γ_U, learning rates $\alpha, \beta^V, \beta^U, \eta, \iota$, reset probability p
Output: policy π^i_*

```
convert each φ^j into an LDBA B^j
initialise parameters θ^i, x^i, {v^j}_{1≤j≤m}, {u^j}_{1≤j≤m}, λ^i
while θ^i not converged do
    while x^i not converged do
        initialise t ← 0, end ← ⊥, and Z^j ← ∅ for each j
        sample s_0^⊗ ~ ζ^⊗
        while end = ⊥ do
            Z^j ← Z ∪ {s_t^⊗} for each j
            sample a_t^i ~ π_θ^i(·|s_t^⊗)
            observe s_{t+1}^⊗ and r_{t+1}^j for each j
            if r_{t+1}^j > 0 or φ(s_{t+1}^⊗)^⊤ v^j = 0 then
                for s_k^⊗ ∈ Z^j do update v^j using (5)
                Z^j ← ∅
            update u^j using (6) for each j
            update x^i and λ^i using (7)
            with probability p set end ← ⊤
    update θ^i using (8)
return π^i
```

4.3 Convergence and Correctness

By making use of results from the stochastic approximation and RL literature we provide an asymptotic convergence guarantee to locally or globally optimal joint policies with respect to multiple LTL specifications, depending on whether agents use local or global policies respectively. We assume that the following conditions hold:

(1) S and A are finite, and all reward functions are bounded.
(2) The Markov chain induced by any θ is irreducible over $S^\otimes$.
(3) $\pi^i(a^i | s^\otimes; \theta^i)$ is continuously differentiable $\forall i, s^\otimes, a^i$
(4) Let Φ be the $|S^\otimes| \times c$ matrix with rows $\phi(s^\otimes)$. Then Φ has full rank, $c \le |S|$, and $\nexists w \in W$ such that $\Phi w = 1$.
(5) $\mathbb{E}_t\big[L^i_{V^j}(x^{i*}_t; \theta_t, v^j_*)\big] \le e^j_{approx}$, where e^j_{approx} is some constant, thus $\mathbb{E}_t\big[L^i_V(x^{i*}_t; \theta_t, v_*)\big] \le e_{approx} := \sum_j w[j] e^j_{approx}$.
(6) $\exists \sigma < \infty$ s.t. $\log \pi^i(a^i | s^\otimes; \theta)$ is a *σ-smooth* in θ^i $\forall i, s^\otimes, a^i$.
(7) The *relative condition number* is finite.
(8) $\pi^i(\cdot | s^\otimes; \theta^i)$ is initialised as the uniform distribution $\forall i, s^\otimes$.

Conditions 1–4 are standard within the literature on the convergence of actor critic algorithms [6, 26]. Conditions 5–8 are taken from recent work on the convergence of natural policy gradient methods by Agarwal et al. [1]. Of particular note is condition 5, where $e_{approx} = 0$ when π^i is a sufficiently rich class, such as an over-parametrised neural network. We recall that if $\log \pi^i(a^i | s^\otimes; \theta)$ is a *σ-smooth function* of θ^i then for any $\theta^i_1, \theta^i_2 \in \Theta^i$ we have:

$$\left\| \nabla_{\theta^i} \log \pi^i(a | s^\otimes; \theta^i_1) - \nabla_{\theta^i} \log \pi^i(a | s^\otimes; \theta^i_2) \right\|_2 \le \alpha \left\| \theta^i_1 - \theta^i_2 \right\|_2.$$

Regarding 6 we define $\Sigma^\nu(\theta^i) := \mathbb{E}_{(s^\otimes, a) \sim \nu}\big[\psi^i_{\theta^i}(a^i_t | s^\otimes_t) \psi^i_{\theta^i}(a^i_t | s^\otimes_t)^\top\big]$ where ν is some state-action distribution. Then the average *relative condition number* [1] is defined and bounded as follows for each player i and each specification φ^j:

$$\mathbb{E}\left[\sup_{x^i} \frac{x^{i\top} \Sigma_{v^j_*}(\theta^i_t) x^i}{x^{i\top} \Sigma_\xi(\theta^i_t) x^i} \right] \le \kappa,$$

where ξ is some initial state-action distribution and:

$$v^j_* := v^j_{\theta,\xi}(s^\otimes, a) = \sum_{(s^\otimes_0, a_0) \in S^\otimes \times A} \xi^\otimes(s^\otimes_0, a_0) \sum_\rho \Pr^\theta_{G_B}(\rho | s^\otimes_0, a_0)$$
$$\cdot \left[\frac{1}{\sum_{t=0}^\infty \Gamma^j_{0:t}} \sum_{t=0}^\infty \Gamma^j_{0:t} \mathbb{I}\big(\rho[t, t+0.5] = (s^\otimes, a)\big) \right]$$

and $\rho[t + 0.5]$ refers to the action taken along the trajectory ρ at time t. Due to space limitations we refer the interested reader to the cited works above for further discussion of these conditions.

Our proof follows the recent work of Agarwal et al. [1]. We begin with a variant of the well-known performance difference lemma [25], using which we prove an analogue of the 'no regret' lemma from Agarwal et al. which is in turn based on the mirror-descent approach of Even-Dar et al. [14]. The proofs are similar to the originals, and so we relegate them to the supplementary material.

Lemma 2. *Suppose that $V_\theta(s^\otimes) \ge V_{\theta'}(s^\otimes)$ for some state $s^\otimes$ and two policies π and π' parametrised by θ and θ' respectively. Then:*

$$V_\theta(s^\otimes) - V_{\theta'}(s^\otimes) \le \sum_j w[j] \left(\mathbb{E}_\rho \left[\sum_{t=0}^\infty \Gamma^j_{0:t} A^j_{\theta'}(s^\otimes_t, a_t) \,\middle|\, F^j(\rho) = \infty \right] \right).$$

Lemma 3. *Consider a sequence of natural gradient updates $\{x^i_t\}_{0 \le t \le T}$ found by* ALMANAC *such that $\|x^i_t\|_2 \le X$ for all t. Let us write $\iota_{0:T} = \sum_{t=0}^T \iota_t$, and recall that $F^j(\rho)$ is the number of times a path ρ in G_B passes through the accepting set F^j of automaton B^j. Let us write $\mathbb{E}_{\rho^*}$ instead of $\mathbb{E}_{\rho \sim \Pr^{\theta_*}_{G_B}(\cdot | s^\otimes), s^\otimes \sim \zeta^\otimes}$ and define e^j_t by:*

$$e^j_t := \mathbb{E}_{\rho^*} \left[\sum_{\tau=0}^\infty \Gamma^j_{0:\tau} \left(A^j_{\theta_t}(s^\otimes_\tau, a_\tau) - \psi^i_{\theta^i_t}(a^i_\tau | s^\otimes_\tau)^\top x^i_t \right) \,\middle|\, F^j(\rho) = \infty \right],$$

where τ indexes ρ, i.e., $\rho[\tau] = s^\otimes_\tau$. Then we have:

$$V_{\theta_*}(s^\otimes) - \lim_{T \to \infty} \mathbb{E}_{t \sim \iota_T}\big[V_{\theta_t}(s^\otimes)\big] = \lim_{T \to \infty} \mathbb{E}_{t \sim \iota_T} \Big[\sum_j w[j] e^j_t \Big],$$

where we define the distribution ι_T over t with $\iota_T(t) := \frac{\iota_t}{\iota_{0:T}}$.

Finally, we use these results to prove that ALMANAC converges to either locally or globally optimal joint policies (i.e., either an SPE or a team-optimal SPE in the original MG) depending on whether agents use local or global policy parameters. By *local* policy parameters we mean that the parameters θ^i stored and updated by agent i only define π^i, and thus $\pi(a | s^\otimes; \theta) = \prod_i \pi^i(a^i | s^\otimes; \theta^i)$ is limited in its representational power due to its factorisation. If, instead, agents share a random seed and each $\theta^i = \theta$ is sufficient to parametrise the whole joint policy π (hence *global*) then at each timestep every agent i can sample the same full joint action $a = (a^1, \dots, a^n)$ and simply perform its own action a^i. As rewards are shared between agents then this means that updates to each agent's version of v, u, and x^i will also be identical, and therefore so too will updates to $\theta^i = \theta$. Though more expensive in terms of computation and memory, the use of global parameters guarantees convergence to the globally optimal joint policy.

Theorem 1. *Given an MG G and LTL objectives $\{\varphi^j\}_{1\leq j\leq m}$ (each equivalent to an LDBA B^j), let $G_B = G \otimes B^1 \otimes \cdots \otimes B^m$ be the resulting product MG with newly defined reward functions $R^j_\otimes$ and state-dependent discount functions Γ^j. Assume that γ_V satisfies Proposition 1, that the learning rates $\alpha, \beta^V, \beta^U, \eta, \iota$ are as in (9) and that conditions 1–8 hold. Then if each agent i uses* local *(*global*) parameters θ^i with* local *policy $\pi^i_{\theta^i}$ (*global *policy $\pi^i_{\theta^i} = \pi_\theta$) then as $T \to \infty$,* Almanac *converges to within*

$$\lim_{T\to\infty} \mathbb{E}_{t\sim\iota_T}\left[\sum_j w[j]\sqrt{e^j_{approx}}\frac{M^j}{(1-\gamma_V)P^j}\right]$$

of a local *(*global*) optimum of $\sum_j w[j] \Pr^\pi_G(s \models \varphi^j)$, where P^j and M^j are constants.*

Proof (Sketch). The proof proceeds via a multi-timescale stochastic approximation analysis and is asymptotic in nature [7]. We consider convergence of the critics, natural gradients, and actor in three steps, dividing our attention between the local and global settings, where required. **Step 1.** The convergence proof for the critics follows that of Tsitsiklis and Van Roy [54]. The hasty critic recursion is simply the classic linear semi-gradient temporal difference algorithm [49] which is known to converge to the unique TD fixed point with probability 1. A similar argument can be made for the patient critic. By waiting to update v^j until seeing a reward, we ensure that a discount is applied and thus that the patient critic recursion forms a contraction. The proof follows immediately from previous work [54], but using a k-step version of the relevant Bellman equation. **Step 2.** Due to the learning rates chosen according to (9) we may consider the more slowly updated parameters fixed for the purposes of analysing the convergence of more quickly updated parameters [7]. As the critic updates fastest we may consider it converged, and since the policy is only updated in the outer loop then it is fixed with respect to the natural gradient and Lagrange multiplier updates. We show that these updates form unbiased estimates of the relevant gradients and thus discrete approximations of the following ODEs:

$$\begin{aligned}
\dot{x}^i_t &= \Omega_{x^i}\left[-\nabla_{x^i} L^i_V(x^i;\theta,\nu)\right]\\
\dot{x}^i_t &= \Omega_{x^i}\left[-\nabla_{x^i}\left(L^i_U(x^i;\theta,\mu) + \lambda^i\left(L^i_V(x^i;\theta,\nu) - l^i\right)\right)\right],\\
\dot{\lambda}^i_\tau &= \Omega_{\lambda^i}\left[\nabla_{\lambda^i}\left(L^i_U(x^i(\lambda^i_\tau);\theta,\mu) + \lambda^i\left[L^i_V(x^i(\lambda^i_\tau);\theta,\nu) - l^i\right)\right)\right],
\end{aligned}$$

on timescales β^V, β^U, and η, respectively. Due to the convexity of L^i_V and L^i_U it can be shown that the recursions above lexicographically miminise L^i_V and then L^i_U and hence that the gradient x^i_* satisfies (3) [44]. **Step 3.** Finally we use Lemma 3 and bound each term e^j_t by $\sqrt{e^j_{approx}}\frac{M^j}{(1-\gamma_V)P^j}$ where M^j and P_j are constants. In particular, we have: $M^j := \max_k \mathbb{E}_{\rho^\zeta_*}\left[M^j_\rho(k) \mid F^j(\rho) = \infty\right]$ where $M^j_\rho(k)$ is the number of steps along trajectory ρ between the k^{th} reward and preceding reward, and $P^j := \min\left(\sum_\rho \Pr^{\theta_*}_{G_B} \mathbb{I}(F^j(\rho) = \infty), 1\right)$. The proof structure follows that of Agarwal et al. [1] with minor variations to handle our use of multiple agents and multiple state-dependent discount rates. □

	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
1	0.13	0.20	0.16	0.21	0.14	0.13	0.17	0.22	0.19	–
2	0.54	0.26	0.19	0.12	0.30	0.19	0.36	0.29	0.38	–
3	0.48	0.23	0.25	0.10	0.20	0.15	0.10	0.31	–	–
4	0.44	0.21	0.02	0.16	0.22	0.26	0.23	0.34	–	–
5	0.17	0.30	0.10	0.13	0.22	0.06	0.30	–	–	–
1	0.14	0.07	0.11	0.17	0.18	0.09	0.24	0.14	0.25	–
2	0.15	0.07	0.15	0.34	0.20	0.15	0.17	0.06	–	–
3	0.15	0.14	0.12	0.25	0.23	0.52	0.28	–	–	–
4	0.12	0.25	0.23	0.23	0.22	–	–	–	–	–
5	0.23	0.21	0.28	0.45	0.01	–	–	–	–	–

Table 1: Average errors across a number of states (columns), agents (rows), and specifications (top and bottom).

5 EXPERIMENTS

Evaluating our proposed algorithm is non-trivial for several reasons. The first is its novelty; it is designed specifically to satisfy the non-Markovian, infinite-horizon specifications that other MARL algorithms are unable to learn, making a direct comparison less meaningful. The second is that the satisfaction of the specifications we wish to evaluate our algorithm against cannot be estimated simply from samples. For example, ψ may be true at every state in a set of samples despite $\mathsf{G}\,\psi$ being false with probability 1. Using a probabilistic model-checker instead raises a third and final difficulty, as even state-of-the-art tools are unable to handle the size of games or number of specifications that Almanac is applicable to.

Despite this, we provide an initial set of results in which we benchmark an implementation[2] of our algorithm against ground-truth models exported to PRISM, a probabilistic model-checker [27]. These results serve to demonstrate Almanac's empirical convergence properties, and how this performance varies as a function of the size of the state space, the number of actors, and the number of specifications (though, unfortunately, PRISM only supports multi-objective synthesis with two specifications). For each of these combinations, we randomly generated ten MGs and sample the specifications and weights. We then ran our algorithm for 5000 episodes and exported the resulting policy, game structure, and specifications to PRISM. The differences between the weighted sum of satisfaction probabilities resulting from Almanac and the ground-truth optimal quantities are displayed in Table 1. We ran PRISM with a maximum of 16GB of memory, 100,000 value iteration steps, and twelve hours of computation, but for some combinations this was insufficient.

ACKNOWLEDGMENTS

The authors thank Hosein Hasanbeig, Joar Skalse, Alper Kamil Bozkurt, Kaiqing Zhang, Salomon Sickert, and the anonymous reviewers for helpful comments. Hammond acknowledges the support of an EPSRC Doctoral Training Partnership studentship (Reference: 2218880) and the University of Oxford ARC facility.[3] Wooldridge and Abate acknowledge the support of the Alan Turing Institute.

[2]Our code can be found online at `https://github.com/lrhammond/almanac`.
[3]Details available at `http://dx.doi.org/10.5281/zenodo.22558`.

REFERENCES

[1] Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. 2020. Optimality and Approximation with Policy Gradient Methods in Markov Decision Processes. In *Proceedings of Thirty Third Conference on Learning Theory (Proceedings of Machine Learning Research, Vol. 125)*, Jacob Abernethy and Shivani Agarwal (Eds.). PMLR, 64–66.
[2] Shun'ichi Amari. 1998. Natural Gradient Works Efficiently in Learning. *Neural Computation* 10, 2 (1998), 251–276.
[3] Gurdal Arslan and Serdar Yuksel. 2017. Decentralized Q-learning for Stochastic Teams and Games. *IEEE Trans. Automat. Control* 62, 4 (2017), 1545–1558.
[4] Pranav Ashok, Jan Křetínský, and Maximilian Weininger. 2019. Pac Statistical Model Checking for Markov Decision Processes and Stochastic Games. In *Computer Aided Verification*. Springer International Publishing, 497–519.
[5] Dimitri P. Bertsekas and John N. Tsitsiklis. 1996. *Neuro-dynamic Programming*. Athena Scientific.
[6] Shalabh Bhatnagar, Richard S. Sutton, Mohammad Ghavamzadeh, and Mark Lee. 2009. Natural Actor–critic Algorithms. *Automatica* 45, 11 (2009), 2471–2482.
[7] Vivek S. Borkar. 2008. *Stochastic Approximation*. Hindustan Book Agency.
[8] Michael Bowling and Manuela Veloso. 2001. Rational and Convergent Learning in Stochastic Games. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2* (Seattle, WA, USA) *(IJCAI'01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1021–1026.
[9] Alper Kamil Bozkurt, Yu Wang, Michael M. Zavlanos, and Miroslav Pajic. 2019. Control Synthesis from Linear Temporal Logic Specifications Using Model-free Reinforcement Learning. *arXiv:1909.07299* (2019).
[10] Tomáš Brázdil, Krishnendu Chatterjee, Martin Chmelík, Vojtěch Forejt, Jan Křetiínský, Marta Kwiatkowska, David Parker, and Mateusz Ujma. 2014. Verification of Markov Decision Processes Using Learning Algorithms. In *Automated Technology for Verification and Analysis*. Springer International Publishing, 98–114.
[11] Lucian Busoniu, Robert Babuska, and Bart De Schutter. 2008. A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 2 (2008), 156–172.
[12] Vincent Conitzer and Tuomas Sandholm. 2003. Awesome: A General Multiagent Learning Algorithm That Converges in Self-play and Learns a Best Response against Stationary Opponents. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning (ICML'03)*. AAAI Press, Washington, DC, USA, 83–90.
[13] Kousha Etessami, Marta Kwiatkowska, Moshe Y. Vardi, and Mihalis Yannakakis. 2007. Multi-objective Model Checking of Markov Decision Processes. In *Tools and Algorithms for the Construction and Analysis of Systems*. Springer Berlin Heidelberg, 50–65.
[14] Eyal Even-Dar, Sham. M. Kakade, and Yishay Mansour. 2009. Online Markov Decision Processes. *Mathematics of Operations Research* 34, 3 (2009), 726–736.
[15] Dana Fisman, Orna Kupferman, and Yoad Lustig. 2010. Rational Synthesis. In *Tools and Algorithms for the Construction and Analysis of Systems*. Springer Berlin Heidelberg, 190–204.
[16] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual Multi-agent Policy Gradients. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 2974–2982.
[17] Jie Fu and Ufuk Topcu. 2014. Probably Approximately Correct Mdp Learning and Control with Temporal Logic Constraints. In *Robotics: Science and Systems X, University of California, Berkeley, USA, July 12-16, 2014*, Dieter Fox, Lydia E. Kavraki, and Hanna Kurniawati (Eds.).
[18] Drew Fudenberg and Jean Tirole. 1991. *Game Theory*. The MIT Press.
[19] Ernst M. Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. 2019. Omega-regular Objectives in Model-free Reinforcement Learning. In *Tools and Algorithms for the Construction and Analysis of Systems*. Springer International Publishing, 395–412.
[20] Ernst M. Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. 2020. Reward Shaping for Reinforcement Learning with Omega-regular Objectives. *arXiv:2001.05977* (2020).
[21] Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. 2019. Certified Reinforcement Learning with Logic Guidance. *arXiv:1902.00778* (2019).
[22] Mohammadhosein Hasanbeig, Daniel Kroening, and Alessandro Abate. 2020. Deep Reinforcement Learning with Temporal Logics. In *Lecture Notes in Computer Science*. Springer International Publishing, 1–22.
[23] Kishor Jothimurugan, Rajeev Alur, and Osbert Bastani. 2019. A Composable Specification Language for Reinforcement Learning Tasks. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 13041–13051.
[24] Sham Kakade. 2001. A Natural Policy Gradient. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic* (Vancouver, British Columbia, Canada) *(NIPS'01)*. MIT Press, Cambridge, MA, USA, 1531–1538.
[25] Sham Kakade and John Langford. 2002. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML '02)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 267–274.
[26] Vijay R. Konda and John N. Tsitsiklis. 2000. Actor-critic Algorithms. In *Advances in Neural Information Processing Systems 12*, S. A. Solla, T. K. Leen, and K. Müller (Eds.). MIT Press, 1008–1014.
[27] Marta Kwiatkowska, Gethin Norman, and David Parker. 2011. Prism 4.0: Verification of Probabilistic Real-time Systems. In *Proc. 23rd International Conference on Computer Aided Verification (CAV'11) (LNCS, Vol. 6806)*, G. Gopalakrishnan and S. Qadeer (Eds.). Springer, 585–591.
[28] Marta Kwiatkowska, Gethin Norman, David Parker, and Gabriel Santos. 2019. Equilibria-based Probabilistic Model Checking for Concurrent Stochastic Games. In *Lecture Notes in Computer Science*. Springer International Publishing, 298–315.
[29] Borja G. León and Francesco Belardinelli. 2020. Extended Markov Games to Learn Multiple Tasks in Multi-Agent Reinforcement Learning. In *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020) (Frontiers in Artificial Intelligence and Applications, Vol. 325)*, Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang (Eds.). IOS Press, 139–146.
[30] Xiao Li, Yao Ma, and Calin Belta. 2018. Automata Guided Reinforcement Learning with Demonstrations. *arXiv:1809.06305* (2018).
[31] Xiao Li, Cristian-Ioan Vasile, and Calin Belta. 2017. Reinforcement Learning with Temporal Logic Rewards. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.
[32] Michael L. Littman. 1994. Markov Games As a Framework for Multi-agent Reinforcement Learning. In *Proceedings of the Eleventh International Conference on International Conference on Machine Learning* (New Brunswick, NJ, USA) *(ICML'94)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 157–163.
[33] Michael L. Littman, Ufuk Topcu, Jie Fu, Charles Isbell, Min Wen, and James MacGlashan. 2017. Environment-independent Task Specifications Via Gltl. *arXiv:1704.04341* (2017).
[34] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent Actor-critic for Mixed Cooperative-competitive Environments. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6382–6393.
[35] Eric Maskin and Jean Tirole. 2001. Markov Perfect Equilibrium. *Journal of Economic Theory* 100, 2 (2001), 191–219.
[36] Chris Nota and Philip S. Thomas. 2020. Is the Policy Gradient a Gradient?. In *Proceedings of the 19th International Conference on Autonomous Agents and Multi-Agent Systems* (Auckland, New Zealand) *(AAMAS '20)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 939–947.
[37] Ann Nowé, Peter Vrancx, and Yann-Michaël De Hauwere. 2012. Game Theory and Multi-agent Reinforcement Learning. In *Adaptation, Learning, and Optimization*. Springer Berlin Heidelberg, 441–470.
[38] Ryohei Oura, Ami Sakakibara, and Toshimitsu Ushio. 2020. Reinforcement Learning of Control Policy for Linear Temporal Logic Specifications Using Limit-deterministic Generalized Büchi Automata. *arXiv:2001.04669* (2020).
[39] Julien Perolat, Bilal Piot, and Olivier Pietquin. 2018. Actor-critic Fictitious Play in Simultaneous Move Multistage Games. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 84)*, Amos Storkey and Fernando Perez-Cruz (Eds.). PMLR, Playa Blanca, Lanzarote, Canary Islands, 919–928.
[40] Jan Peters and Stefan Schaal. 2008. Natural Actor-critic. *Neurocomputing* 71, 7-9 (2008), 1180–1190.
[41] Amir Pnueli. 1977. The Temporal Logic of Programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS '77)*. IEEE Computer Society, USA, 46–57.
[42] H.L. Prasad, Prashanth L.A., and Shalabh Bhatnagar. 2015. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems* (Istanbul, Turkey) *(AAMAS '15)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1371–1379.
[43] Guannan Qu, Adam Wierman, and Na Li. 2020. Scalable Reinforcement Learning of Localized Policies for Multi-agent Networked Systems. *arXiv:1912.02906* (2020).
[44] Mark J. Rentmeesters, Wei K. Tsai, and Kwei-Jay Lin. 1996. A Theory of Lexicographic Multi-criteria Optimization. In *Proceedings of ICECCS 1996: 2nd IEEE International Conference on Engineering of Complex Computer Systems*. IEEE Comput. Soc. Press.
[45] Dorsa Sadigh, Eric S. Kim, Samuel Coogan, S. Shankar Sastry, and Sanjit A. Seshia. 2014. A Learning Based Approach to Control Synthesis of Markov Decision Processes for Linear Temporal Logic Specifications. In *53rd IEEE Conference on Decision and Control, CDC 2014, Los Angeles, CA, USA, December 15-17, 2014*. 1091–1096.

[46] Salomon Sickert, Javier Esparza, Stefan Jaax, and Jan Křetínský. 2016. Limit-deterministic Büchi Automata for Linear Temporal Logic. In *Computer Aided Verification*. Springer International Publishing, 312–332.

[47] Joar Skalse, Lewis Hammond, and Alessandro Abate. 2021. Lexicographic Multi-objective Reinforcement Learning. *Forthcoming* (2021).

[48] Richard S. Sutton. 1988. Learning to Predict by the Methods of Temporal Differences. *Machine Learning* 3, 1 (1988), 9–44.

[49] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning*. The MIT Press.

[50] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems* (Denver, CO) *(NIPS'99)*. MIT Press, Cambridge, MA, USA, 1057–1063.

[51] Philip S. Thomas. 2014. Bias in Natural Actor-critic Algorithms. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14)*. JMLR.org, Beijing, China, I–441–I–448.

[52] Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. 2018. Using Reward Machines for High-level Task Specification and Decomposition in Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, Stockholmsmässan, Stockholm Sweden, 2107–2116.

[53] Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Valenzano, and Sheila A. McIlraith. 2018. Teaching Multiple Tasks to an Rl Agent Using Ltl. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems* (Stockholm, Sweden) *(AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 452–461.

[54] John N. Tsitsiklis and Benjamin Van Roy. 1997. An Analysis of Temporal-difference Learning with Function Approximation. *IEEE Trans. Automat. Control* 42, 5 (1997), 674–690.

[55] Xiaofeng Wang and Tuomas Sandholm. 2002. Reinforcement Learning to Play an Optimal Nash Equilibrium in Team Markov Games. In *Proceedings of the 15th International Conference on Neural Information Processing Systems (NIPS'02)*. MIT Press, Cambridge, MA, USA, 1603–1610.

[56] Min Wen and Ufuk Topcu. 2016. Probably Approximately Correct Learning in Stochastic Games with Temporal Logic Specifications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, New York, New York, USA, 3630–3636.

[57] Michael Wooldridge, Julian Gutierrez, Paul Harrenstein, Enrico Marchioni, Giuseppe Perelli, and Alexis Toumi. 2016. Rational Verification: From Model Checking to Equilibrium Checking. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, Phoenix, Arizona, 4184–4190.

[58] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. 2019. Multi-agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. *arXiv:1911.10635* (2019).

[59] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. 2018. Fully Decentralized Multi-agent Reinforcement Learning with Networked Agents. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, Stockholmsmässan, Stockholm Sweden, 5872–5881.

[60] Martin Zinkevich, Amy Greenwald, and Michael L. Littman. 2005. Cyclic Equilibria in Markov Games. In *Proceedings of the 18th International Conference on Neural Information Processing Systems* (Vancouver, British Columbia, Canada) *(NIPS'05)*. MIT Press, Cambridge, MA, USA, 1641–1648.