Optimal Control of Logically Constrained Partially Observable and Multi-Agent Markov Decision Processes

Krishna C. Kalagarla, Dhruva Kartik, Dongming Shen, Rahul Jain, *Senior Member, IEEE*, Ashutosh Nayyar, *Senior Member, IEEE* and Pierluigi Nuzzo, *Senior Member, IEEE*

***Abstract*— Autonomous systems often have logical constraints arising, for example, from safety, operational, or regulatory requirements. Such constraints can be expressed using temporal logic specifications. The system state is often partially observable. Moreover, it could encompass a team of multiple agents with a common objective but disparate information structures and constraints. In this paper, we first introduce an optimal control theory for partially observable Markov decision processes (POMDPs) with finite linear temporal logic (LTL_f) constraints. We provide a structured methodology for synthesizing policies that maximize a cumulative reward while ensuring that the probability of satisfying a temporal logic constraint is sufficiently high. Our approach comes with guarantees on approximate reward optimality and constraint satisfaction. We then build on this approach to design an optimal control framework for logically constrained multi-agent settings with information asymmetry. We illustrate the effectiveness of our approach by implementing it on several case studies.**

***Index Terms*— Markov decision processes, Multi-agent systems, Partially observable Markov decision processes, Stochastic optimal control, Temporal logic**

I. Introduction

Autonomous systems are rapidly being deployed in many safety-critical applications like robotics, transportation, and advanced manufacturing. Markov decision processes (MDPs) [1] can model a wide range of sequential decision-making scenarios in these dynamically evolving environments. Traditionally, a reward structure is defined over the MDP state-action space, and is then maximized to achieve a desired objective. Formulating an appropriate reward function is critical, as an incorrect formulation can easily lead to unsafe and unforeseen behaviors. Designing reward functions for complex specifications can be exceedingly difficult and may not always be possible.

Increasing interest has been directed over the past decade toward leveraging tools from formal methods and temporal logic [2] to alleviate this difficulty. These tools allow unambiguously specifying, solving, and validating complex control and planning problems. Temporal logic formalisms are capable of capturing a wide range of task specifications, including surveillance, reachability, safety, and sequentiality. However, while certain objectives like safety or reachability are well expressed by temporal logic constraints, others, e.g., pertaining to system performance or cost, are often better framed as "soft" rewards to be maximized. In this paper, we focus on such composite tasks.

Consider, for example, an autonomous robot tasked with navigating through a warehouse with hazardous areas to perform inspections or repairs. We can express complex requirements for this robot such as "Always avoid hazardous areas," "Eventually perform inspections," or "Always return to a charging station when the battery is low," unambiguously via temporal logic. Model checking methods can be used to verify if a given controller satisfies these requirements [2]. We can also use algorithmic methods to synthesize a controller for the robot such that the resulting controller satisfies the requirements by construction [2]. This is the approach we adopt in the paper. Further, we wish to account for additional considerations, such as fuel efficiency or smoothness of motion, expressed by reward functions to be maximized.

While full state observability is assumed in environments modeled by MDPs, this assumption excludes many real-life scenarios where the state is only partially observed. These scenarios can instead be captured by partially observable Markov decision processes (POMDPs). For example, the warehouse robot only has a partial observation of the state since the sensor observations only provide a noisy estimate of it.

In this paper, we expand the traditional POMDP framework to incorporate temporal logic specifications. Specifically, we aim to synthesize policies such that the agent's cumulative reward is maximized while the probability of satisfying a given temporal logic specification is above a desired threshold. Due to the inherent partial observability, planning methods for MDPs with temporal logic specifications [3]–[5] are not directly applicable to our problem. Recasting our problem as an MDP would necessitate constructing beliefs over an augmented state space, which grows exponentially with the time horizon. This results in an extremely large belief space, making the synthesis methods developed for MDPs intractable

This work was supported in part under *NSF* grants ECCS 2025732, ECCS 1750041, CNS 1846524, and ECCS 2139982, under *ONR (Science of AI and Science of Autonomy Programs)* award N00014-20-1-2258, under an Okawa Research Grant, and under the USC Center for Autonomy and Artificial Intelligence.

This work was performed when all the authors were at the Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA 90089, USA. Email: {kalagarl, mokhasun, alvinshe, rahul.jain, ashutosh.nayyar, nuzzo}@usc.edu.

in the context of POMDPs. Consequently, we utilize scalable algorithms specifically designed for POMDPs to solve our problem effectively.

We consider finite linear temporal logic (LTL_f) [6], a temporal extension of propositional logic, to express complex tasks. LTL_f is a variant of linear temporal logic (LTL) [2] which is interpreted over finite length strings rather than infinite length strings. We can express the requirements of our warehouse robot example in LTL_f, respectively, as follows: $Always$ ¬hazard.nearby, $Eventually$ inspection.done, $Always$ (low.battery $\rightarrow$ $Eventually$ charging.station). We need to simultaneously consider the cumulative POMDP reward and the temporal logic satisfaction. For a given LTL_f specification, we construct a deterministic finite automaton (DFA) that accepts an agent's trajectory if and only if it satisfies the specification [7]. By augmenting the system state with the DFA's internal state, we can monitor both environmental and task status. This approach enables us to frame our planning problem as a standard reward-constrained POMDP problem. In the context of our motivating example, we combine the POMDP expressing the robot dynamics in the warehouse and the DFA expressing our specifications into a constrained POMDP problem containing details of both the POMDP and the DFA (thus the LTL_f specification).

We then tackle the challenge of constrained POMDP problems by proposing an iterative primal-dual scheme which solves a sequence of unconstrained POMDP problems using any off-the-shelf unconstrained POMDP solver [8], [9]. This approach effectively leverages the established techniques of unconstrained POMDP planning. The iterative scheme also incorporates regret bounds from no-regret online learning [10], providing guarantees on the near-optimality of the returned policy.

Finally, we extend our framework to a multi-agent setting with information asymmetry. We use the previously described approach for composition with the DFA to obtain a constrained multi-agent planning problem. In this case, our approach requires solving a sequence of unconstrained multi-agent problems. Under some mild assumptions, these unconstrained multi-agent problems can be viewed as single-agent POMDP problems via the *common information approach* [11]. Owing to the theoretical guarantees of the common information approach and our algorithm, the returned policies will still provide the desired guarantees on optimality and temporal logic satisfaction.

Our contributions can be summarized as follows: (i) We formulate a novel optimal control problem in terms of cumulative reward maximization in POMDPs under expressive LTL_f constraints. (ii) We design an iterative planning scheme which can leverage any off-the-shelf unconstrained POMDP solver to solve the problem. Differently from existing work on constrained POMDPs, our scheme uses a no-regret online learning approach to provide theoretical guarantees on the near-optimality of the returned policy. (iii) We extend the single-agent framework to a multi-agent setting with information asymmetry, where agents have a common reward objective but can have individual or even joint temporal logic requirements. Their different information structure makes the problem rather difficult. (iv) We conduct experimental studies on a suite of single and multi-agent environments to validate the algorithms developed and explore their effectiveness. To the best of our knowledge, this is the first effort on optimal control of logically constrained partially observed and multi-agent MDP models.

We presented some preliminary results on optimal control of POMDPs with LTL_f specifications in a previous paper [12]. In this paper, we expand on our previous results and show that our methodology is broader in scope, in that it encompasses other problem formulations and solution strategies. In particular, we extend our planning algorithm from a single-agent setting to a *multi-agent setting with information asymmetry*. Moreover, while we previously assumed access to an *exact* POMDP solver and an exact estimator of the expected total reward, we show that our method can also operate with an approximate POMDP solver and approximate estimates of the expected total reward. We then show that our method also supports cumulative reward constraints along with LTL_f requirements. Finally, we include all the proofs of our theorems.

II. Related Work

Synthesizing policies for MDPs such that they maximize the probability of satisfying a temporal logic specification has been extensively studied in the context of known [4], [13] and unknown [5], [14] transition probabilities.

Several approaches have also been proposed for maximizing the temporal logic satisfaction probability in POMDPs, albeit without considering an additional reward objective. Some of these methods include building a finite-state abstraction via approximate stochastic simulation over the belief space [15], grid-based discretization of the POMDP belief space [16], and restricting the space of policies to finite state controllers [17], [18]. Similarly to our work, leveraging well-studied unconstrained POMDP planners [19], [20] has also been proposed in this context, while deep learning approaches like those using recurrent neural networks [21], [22] have been lately explored.

Our approach further differs from a few other existing methods for solving constrained POMDPs. One of these methods [23] addresses a constrained POMDP by modeling it as a constrained belief MDP and employing an approximate linear program that operates on a selectively refined subset of reachable beliefs. However, the expanding size of the subset of reachable beliefs and the resulting increase in the complexity of the linear programs eventually render this approach computationally intractable. Another method addresses the scalability issues of the previous one using a primal-dual approach based on Monte Carlo Tree Search (MCTS) [24]. We solve the constrained POMDP problem using a similar method. A key difference is that, instead of using an MCTS approach, we use an approximate unconstrained POMDP solver, SARSOP [8], which returns policies along with bounds on their optimality gaps. Column generation algorithms [25] also use a primal-dual approach, but with a different dual parameter update procedure. While only convergence to optimality is discussed for these algorithms, our method, on the other hand, gives a precise relationship between the approximation error and the number of iterations.

A few methods have also appeared which address temporal logic satisfaction in a multi-agent setting, albeit leveraging different formulations and solution strategies. Some examples include the synthesis of a joint policy maximizing the temporal logic satisfaction probability for multiple agents with shared state space [26] and the decomposition of temporal logic specifications for scalable planning for a large number of agents under communication constraints [27], [28]. Some efforts have also leveraged the robustness semantics of temporal logics to guide reward shaping for multi-agent planning [29], [30]. To the best of our knowledge, our method is the first to consider a reward objective and a temporal logic constraint in the multi-agent stochastic setting with information asymmetry.

III. Preliminaries

We denote the POMDP by $\mathscr{M}$ and the LTL$_f$ specification by φ. The DFA equivalent to the LTL$_f$ specification is symbolized by $\mathscr{A}$ and the product POMDP constructed with $\mathscr{M}$ and $\mathscr{A}$ is designated as $\mathscr{M}^{\times}$. $\mathbb{P}^{\mathscr{M}}_{\varphi}(\mu)$ denotes the probability that a run of $\mathscr{M}$ satisfies φ under policy μ. The indicator function $\mathbb{1}_S(s)$ evaluates to 1 when $s \in S$ and 0 otherwise. The probability simplex over the set S is denoted by Δ_S. Finally, for a string s, $|s|$ denotes its length.

A. Labeled POMDPs

1) Model: A Labeled Partially Observable Markov Decision Process is defined as a tuple $\mathscr{M} = (\mathcal{S}, \mathcal{U}, P, \varpi, O, Z, AP, L, r^o, r^c, T)$, where $\mathcal{S}$ is a finite state space, $\mathcal{U}$ is a finite action space, $P_t(s, u; s')$ is the probability of transitioning from state s to state s' on taking action u at time t, $\varpi \in \Delta_{\mathcal{S}}$ is the initial state distribution, O is a finite observation space, $Z_t(s; o)$ is the probability of seeing observation o in state s at time t, AP is a set of atomic propositions, used to indicate the truth value of a predicate (property) of the state, e.g., the presence of an obstacle or goal state. $L : \mathcal{S} \to 2^{AP}$ is a labeling function which indicates the set of atomic propositions which are true in each state, e.g., $L(s) = \{a\}$ indicates that only the atomic proposition a is true in state s. $r^o_t(s, u)$ and $r^c_t(s, u)$ are the objective and constraint rewards obtained on taking action u in state s at time t. The state, action, and observation at time t are denoted by $\mathcal{S}_t$, $\mathcal{U}_t$, and O_t, respectively. At any given time t, the information available to the agent is the collection of all the observations $O_{0:t}$ and all the past actions $\mathcal{U}_{0:t-1}$. We denote this information with $I_t = \{O_{0:t}, \mathcal{U}_{0:t-1}\}$. We say that the system is time-invariant when the reward functions r^o_t, r^c_t and the transition and observation probability functions P_t and Z_t do not depend on time t. The POMDP runs for a random time horizon T. This random time may be determined exogenously (independently) of the POMDP or it may be a stopping time with respect to the information process $\{I_t : t \geq 0\}$.

2) Pure and Mixed Policies: A *control law* π_t maps the information I_t to an action in the action space $\mathcal{U}$. A *policy* $\pi := (\pi_0, \pi_1, \ldots)$ is then the sequence of laws over the entire horizon. We refer to such deterministic policies as pure policies and denote the set of all pure policies with $\mathcal{P}$.

A mixed policy μ is a distribution on a finite collection of pure policies. Under a mixed policy μ, the agent randomly selects a pure policy $\pi \in \mathcal{P}$ with probability $\mu(\pi)$ before the POMDP begins. The agent uses this randomly selected policy to select its actions during the course of the process. More formally, $\mu : \mathcal{P} \to [0, 1]$ is a mapping. The support of the mixture μ is defined as $\mathrm{supp}(\mu) := \{\pi \in \mathcal{P} : \mu(\pi) \neq 0\}$.

If the support size of a mixed policy is 1, then it is a pure policy. The set $\mathcal{M}_p$ of all mixed mappings is given by

$$\mathcal{M}_p := \left\{ \mu : |\mathrm{supp}(\mu)| < \infty, \sum_{\pi \in \mathrm{supp}(\mu)} \mu(\pi) = 1 \right\}. \quad (1)$$

Clearly, the set $\mathcal{M}_p$ of mixed policies is convex.

A *run* ξ of the POMDP is the sequence of states and actions $((\mathcal{S}_0, \mathcal{U}_0), (\mathcal{S}_1, \mathcal{U}_1), \cdots, (\mathcal{S}_T, \mathcal{U}_T))$ up to horizon T. The total expected objective reward for a policy μ is given by

$$\begin{aligned} \mathcal{R}^{\mathscr{M}}_o(\mu) &= \mathbb{E}^{\mathscr{M}}_{\mu}\left[\sum_{t=0}^{T} r^o_t(\mathcal{S}_t, \mathcal{U}_t)\right] \\ &= \sum_{\pi \in \mathrm{supp}(\mu)} \left[\mu(\pi)\mathbb{E}^{\mathscr{M}}_{\pi}\left[\sum_{t=0}^{T} r^o_t(\mathcal{S}_t, \mathcal{U}_t)\right]\right]. \end{aligned} \quad (2)$$

The total expected constraint reward $\mathcal{R}^{\mathscr{M}}_c(\mu)$ is similarly defined with respect to r^c. $\mathcal{R}^{\mathscr{M}}_o(\mu)$ and $\mathcal{R}^{\mathscr{M}}_c(\mu)$ are linear functions in μ.

Assumption 1: The POMDP $\mathscr{M}$ is such that for every pure policy π, the expected value of the random horizon T is bounded by a finite constant and the total expected constraint reward is also bounded above by a finite constant, i.e.,

$$\mathbb{E}^{\mathscr{M}}_{\pi}[T] < T_{\mathrm{MAX}} < \infty, \quad (3)$$

$$0 \leq \mathcal{R}^{\mathscr{M}}_c(\pi) \leq \mathbf{R}^{max}_c \quad \forall\ \pi. \quad (4)$$

The expectation in $\mathbb{E}^{\mathscr{M}}_{\pi}[T]$ is over the random state, action, and observation trajectory of POMDP $\mathscr{M}$ under the pure policy π.

Assumption 1 ensures that T is finite almost surely, i.e., $\mathbb{P}^{\mathscr{M}}_{\mu}[T < \infty] = 1$ and the total expected constraint reward satisfies $\mathcal{R}^{\mathscr{M}}_c(\mu) \leq \mathbf{R}^{max}_c$ for every policy μ.

B. Finite Linear Temporal Logic

We use LTL$_f$ [6], a variant of linear temporal logic (LTL) [2] interpreted over finite strings, to express complex task specifications. Given a set AP of atomic propositions, i.e., Boolean variables that have a unique truth value (true or false) for a given system state, LTL$_f$ formulae are constructed inductively as follows:

$$\varphi := \mathsf{true} \mid a \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathbf{X}\varphi \mid \varphi_1 \mathbf{U} \varphi_2,$$

where $a \in AP$, φ, φ_1, and φ_2 are LTL formulae, $\wedge$ and $\neg$ are the logic conjunction and negation, and $\mathbf{U}$ and $\mathbf{X}$ are the *until* and *next* temporal operators. Additional temporal operators such as *eventually* ($\mathbf{F}$) and *always* ($\mathbf{G}$) are derived as $\mathbf{F}\varphi := \mathsf{true}\mathbf{U}\varphi$ and $\mathbf{G}\varphi := \neg\mathbf{F}\neg\varphi$. For example, $\varphi = \mathbf{F}a \wedge (\mathbf{G}\neg b)$ expresses the specification that a state where atomic proposition a holds true has to be *eventually* reached by

the end of the trajectory and states where atomic proposition b holds true have to be *always* avoided.

LTL$_f$ formulae are interpreted over finite-length words $w = w_0 w_1 \cdots w_{last} \in (2^{AP})^*$, where each letter w_i is a set of atomic propositions and $last = |w| - 1$ is the index of the last letter of the word w. Given a finite word w and LTL$_f$ formula φ, we inductively define when φ is true for w at step i, $0 \leq i < |w|$, written $(w, i) \models \varphi$. Informally, a is true for (w, i) iff $a \in w_i$; $\mathbf{X}\varphi$ is true for (w, i) iff φ is true for $(w, i+1)$; $\varphi_1 \mathbf{U} \varphi_2$ is true for (w, i) iff there exits $k \geq i$ such that φ_2 is true for (w, k) and φ_1 is true for all j, $i \leq j < k$; $\mathbf{G}\varphi$ is true for (w, i) iff φ is true for all $(w, j), j \geq i$; $\mathbf{F}\varphi$ is true for (w, i) iff φ is true for some $(w, j), j \geq i$ (where 'iff' is shorthand for 'if and only if'). A formula φ is true in w, denoted by $w \models \varphi$, iff $(w, 0) \models \varphi$.

Given a POMDP $\mathscr{M}$ and an LTL$_f$ formula φ, a *run* $\xi = ((s_0, u_0), (s_1, u_1), \cdots, (s_T, u_T))$ of the POMDP under policy μ is said to satisfy φ if the word $w = L(s_0)L(s_1) \cdots \in (2^{AP})^{T+1}$ generated by the *run* satisfies φ. The probability that a run of $\mathscr{M}$ satisfies φ under policy μ is denoted by $\mathbb{P}^{\mathscr{M}}_{\varphi}(\mu)$. We refer to Section VII for various examples of LTL$_f$ specifications, especially those which cannot be easily expressed by standard reward functions.

C. Deterministic Finite Automata (DFA)

The language defined by an LTL$_f$ formula, i.e., the set of words satisfying the formula, can be captured by a Deterministic Finite Automaton (DFA) [7]. A DFA is a finite state machine that accepts or rejects a given string of symbols, by running through a state sequence uniquely determined by the string. It consists of a finite set of states, a set of input symbols, a transition function, a start state, and a set of accepting states [31]. We denote such a DFA by a tuple $\mathscr{A} = (Q, \Sigma, q_0, \delta, F)$, where Q is a finite set of states, Σ is a finite alphabet, q_0 is the initial state, $\delta : Q \times \Sigma \to Q$ is a transition function, and $F \subseteq Q$ is the set of accepting states. A *run* $\xi_{\mathscr{A}}$ of $\mathscr{A}$ over a finite word $w = w_0 \cdots w_n$, with $w_i \in \Sigma$, is a sequence of states, $q_0 q_1 \cdots q_{n+1} \in Q^{n+1}$ such that $q_{i+1} = \delta(q_i, w_i), i = 0, \cdots, n$. A *run* $\xi_{\mathscr{A}}$ is *accepting* if it ends in an accepting state, i.e., $q_{n+1} \in F$. A word $w \in \Sigma^*$ is accepted by $\mathcal{A}$ if and only if there exists an accepting *run* $\xi_{\mathscr{A}}$ of $\mathcal{A}$ on w. Finally, we say that an LTL$_f$ formula is equivalent to a DFA $\mathscr{A}$ if and only if the language defined by the formula is the language (i.e., the set of words) accepted by $\mathscr{A}$. For any LTL$_f$ formula φ over AP, we can construct an equivalent DFA with input alphabet 2^{AP} [7].

IV. Problem Formulation and Solution Strategy

Given a labeled POMDP $\mathscr{M}$ and an LTL$_f$ specification φ, our objective is to design a policy μ that maximizes the total expected objective reward $\mathcal{R}^{\mathscr{M}}_o(\mu)$ while ensuring that the probability $\mathbb{P}^{\mathscr{M}}_{\varphi}(\mu)$ of satisfying the specification φ is at least $1-\delta$ and the total expected constraint reward $\mathcal{R}^{\mathscr{M}}_c(\mu)$ exceeds a threshold ρ. $\mathcal{R}^{\mathscr{M}}_o(\mu), \mathbb{P}^{\mathscr{M}}_{\varphi}(\mu)$, and $\mathcal{R}^{\mathscr{M}}_c(\mu)$ are all used to express different requirements which we wish to satisfy. More formally, we would like to solve the following constrained optimization problem

$$\textbf{LTL}_f\textbf{-POMDP:} \quad \sup_{\mu \in \mathcal{M}_p} \ \mathcal{R}^{\mathscr{M}}_o(\mu) \quad \text{s.t.} \ \mathcal{R}^{\mathscr{M}}_c(\mu) \geq \rho, \ \mathbb{P}^{\mathscr{M}}_{\varphi}(\mu) \geq 1 - \delta. \tag{P1}$$

If (P1) is feasible, then we denote its optimal value with $\mathcal{R}^{\mathscr{M}}_*$. If (P1) is infeasible, then $\mathcal{R}^{\mathscr{M}}_* = -\infty$.

Remark 1: The results and experiments in this work are in the setting of POMDPs and multi-agent MDPs with finite state, action, and observation spaces.

A. Constrained Product POMDP

In (P1), we require a policy μ to maximize the objective reward and satisfy a reward constraint, both of which are expressed via the given POMDP $\mathscr{M}$. We also need the same policy μ to satisfy the LTL$_f$ constraint which is expressed via a DFA $\mathscr{A}$ (equivalent to the LTL$_f$ specification φ) instead of the POMDP $\mathscr{M}$. We thus construct a constrained product POMDP $\mathscr{M}^{\times}$ which allows us to address these different requirements via a single unified model.

Given the labeled POMDP $\mathscr{M}$ and a DFA $\mathscr{A}$ capturing the LTL$_f$ formula φ, we follow a construction previously proposed for MDPs [12] to obtain a constrained product POMDP $\mathscr{M}^{\times} = (\mathcal{S}^{\times}, \mathcal{U}^{\times}, P^{\times}, s_0^{\times}, r^{o\times}, r^{c\times}, r^f, \varpi, O, Z^{\times})$ which incorporates the transitions of $\mathscr{M}$ and $\mathscr{A}$, the observations and the reward functions of $\mathscr{M}$, and the acceptance set of $\mathscr{A}$.

In the constrained product POMDP $\mathscr{M}^{\times}$, $\mathcal{S}^{\times} = (\mathcal{S} \times Q)$ is the state space, $\mathcal{U}^{\times} = \mathcal{U}$ is the action space, and $s_0^{\times} = (s_0, q_0)$ is the initial state where the POMDP's initial state s_0 is drawn from the distribution ϖ and q_0 is the DFA's initial state. For each pair (s, s'), (q, q'), and u, we define the transition function $P_t^{\times}((s, q), u; (s', q'))$ at time t as

$$\begin{cases} P_t(s, u; s'), & \text{if } q' = \delta(q, L(s)), \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

The reward functions are defined as

$$r_t^{o\times}((s, q), u) = r_t^o(s, u), \ \forall s, q, u \tag{6}$$

$$r_t^{c\times}((s, q), u) = r_t^c(s, u), \ \forall s, q, u \tag{7}$$

$$r^f((s, q)) = \begin{cases} 1, & \text{if } q \in F \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

The reward functions $r^{o\times}$ and $r^{c\times}$ depend on the reward functions of the original POMDP $\mathscr{M}$. The reward function r^f is instead informed by the accepting states of the DFA $\mathscr{A}$. The observation space O is the same as in the original POMDP $\mathscr{M}$. The observation probability function $Z^{\times}((s, q); o)$ is defined to be equal to $Z(s; o)$ for every s, q, o. We denote the state of the product POMDP $\mathscr{M}^{\times}$ at time t with $X_t = (\mathcal{S}_t, Q_t)$ in order to avoid confusion with the state $\mathcal{S}_t$ of the original POMDP $\mathscr{M}$.

At any given time t, the information available to the agent is $I_t = \{O_{0:t}, \mathcal{U}_{0:t-1}\}$. Control laws and policies in the product POMDP are the same as in the original POMDP $\mathscr{M}$. We define three reward functions in the product POMDP: (i) a reward

$\mathcal{R}_o^{\mathscr{M}^\times}(\mu)$ associated with the objective reward r^o, (ii) a reward $\mathcal{R}_c^{\mathscr{M}^\times}(\mu)$ associated with the constraint reward r^c, and (iii) a reward $\mathcal{R}^f(\mu)$ associated with reaching an accepting state in the DFA $\mathscr{A}$. The expected reward $\mathcal{R}_o^{\mathscr{M}^\times}(\mu)$ is defined as

$$\mathcal{R}_o^{\mathscr{M}^\times}(\mu) = \mathbb{E}_\mu\left[\sum_{t=0}^{T} r_t^{o\times}(X_t, \mathcal{U}_t)\right]. \tag{9}$$

The expected reward $\mathcal{R}_c^{\mathscr{M}^\times}(\mu)$ is similarly defined as

$$\mathcal{R}_c^{\mathscr{M}^\times}(\mu) = \mathbb{E}_\mu\left[\sum_{t=0}^{T} r_t^{c\times}(X_t, \mathcal{U}_t)\right]. \tag{10}$$

The expected reward $\mathcal{R}^f(\mu)$ is defined as

$$\begin{aligned}\mathcal{R}^f(\mu) &= \mathbb{E}_\mu\left[r^f(X_{T+1})\right] \\ &= \sum_{\pi\in\text{supp}(\mu)} \mu(\pi)\mathbb{E}_\pi\left[r^f(X_{T+1})\right].\end{aligned} \tag{11}$$

Due to Assumption 1, the stopping time T is finite almost surely and therefore, the reward $\mathcal{R}^f(\mu)$ is well-defined. Further, (11), which follows from the definition of mixed policies, demonstrates that $\mathcal{R}^f(\mu)$ is a linear function of μ.

B. Constrained POMDP Formulation

In the product POMDP $\mathscr{M}^\times$, we are interested in solving the following constrained optimization problem

$$\textbf{C-POMDP:}\quad \begin{aligned}\sup_{\mu\in\mathcal{M}_p}\quad & \mathcal{R}_o^{\mathscr{M}^\times}(\mu) \\ \text{s.t.}\quad & \mathcal{R}_c^{\mathscr{M}^\times}(\mu) \geq \rho, \\ & \mathcal{R}^f(\mu) \geq 1-\delta\end{aligned} \tag{P2}$$

whose optimal value is denoted by $\mathcal{R}_*^{\mathscr{M}^\times}$.

Theorem 1 (Equivalence of Problems (P1) *and* (P2)*):* For any policy μ, we have

$$\begin{aligned}\mathcal{R}_o^{\mathscr{M}^\times}(\mu) &= \mathcal{R}_o^{\mathscr{M}}(\mu), \\ \mathcal{R}_c^{\mathscr{M}^\times}(\mu) = \mathcal{R}_c^{\mathscr{M}}(\mu),\ &\text{and } \mathcal{R}^f(\mu) = \mathbb{P}_\varphi^{\mathscr{M}}(\mu).\end{aligned}$$

Therefore, a policy μ^* is an optimal solution to Problem (P1) if and only if it is an optimal solution to Problem (P2), and therefore, $\mathcal{R}_*^{\mathscr{M}} = \mathcal{R}_*^{\mathscr{M}^\times} := \mathcal{R}^*$.

Proofs of all theorems and lemmas are available in the appendices.

V. No-regret Learning Approach for Solving the Constrained POMDP

Problem (P2) is a POMDP policy optimization problem with constraints. Since solving unconstrained optimization problems is generally easier than solving constrained optimization problems, we describe a general methodology that reduces the constrained POMDP optimization problem (P2) to a sequence of unconstrained POMDP problems. These unconstrained POMDP problems can be solved using any off-the-shelf POMDP solver. The main idea is to first transform Problem (P2) into a sup-inf problem using the Lagrangian function. This sup-inf problem can then be solved approximately using a no-regret online learning algorithm such as the exponentiated gradient (EG) algorithm [10].

The Lagrangian function $L(\mu, \boldsymbol{\lambda})$ associated with Problem (P2) with $\boldsymbol{\lambda} = (\lambda^f, \lambda^c)$ is

$$\mathcal{R}_o^{\mathscr{M}^\times}(\mu) + \lambda^c(\mathcal{R}_c^{\mathscr{M}^\times}(\mu) - \rho) + \lambda^f(\mathcal{R}^f(\mu) - 1 + \delta).$$

Let

$$l^* := \sup_\mu \inf_{\boldsymbol{\lambda}\geq 0} L(\mu, \boldsymbol{\lambda}). \tag{P3}$$

The constrained optimization problem in (P2) is equivalent to the sup-inf optimization problem above [32]. That is, if an optimal solution μ^* exists in problem (P2), then μ^* is a maximizer in (P3), and if (P2) is infeasible, then $l^* = -\infty$. Further, the optimal value of Problem (P2) is equal to l^*. Consider the following variant of (P3) wherein the Lagrange multiplier $\boldsymbol{\lambda}$ is bounded in the L^1 norm:

$$l_B^* := \sup_\mu \inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} L(\mu, \boldsymbol{\lambda}). \tag{P4}$$

Lemma 1: Let $\bar{\mu}$ be an ϵ-optimal policy in sup-inf problem (P4), i.e.,

$$l_B^* \leq \inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} L(\bar{\mu}, \boldsymbol{\lambda}) + \epsilon, \tag{12}$$

for some $\epsilon > 0$. Then, we have

$$\mathcal{R}_o^{\mathscr{M}^\times}(\bar{\mu}) \geq \mathcal{R}^* - \epsilon, \tag{13}$$

$$\mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu}) \geq \rho - \epsilon^f \quad \text{and} \tag{14}$$

$$\mathcal{R}^f(\bar{\mu}) \geq 1 - \delta - \epsilon^f, \tag{15}$$

where $\epsilon^f = \frac{\mathbf{R}_o^{max} - \mathcal{R}^* + \epsilon}{B}$ and $\mathbf{R}_o^{max} := \sup_\mu \mathcal{R}_o^{\mathscr{M}^\times}(\mu)$ is the maximum possible objective reward for unconstrained $\mathscr{M}^\times$.

Lemma 1 suggests that if we can find an ϵ-optimal mixed policy $\bar{\mu}$ of the sup-inf problem (P4), then the policy $\bar{\mu}$ is approximately optimal and approximately feasible in (P2), and therefore in Problem (P1) due to Theorem 1.

We use the exponentiated gradient (EG) algorithm [10] to find an ϵ-approximate policy $\bar{\mu}$ for Problem (P4). This algorithm is used for online convex optimization and has an L1-norm bounded decision space. It is also known to satisfy the no-regret property, i.e., the average gap between its cumulative loss and that of the optimal hindsight decision asymptotically approaches zero. We refer the reader to the literature [10] for more details on the algorithm.

Following the structure of the EG algorithm, our algorithm uses a sub-gradient of the following function of the dual variable:

$$f(\boldsymbol{\lambda}) = \sup_\mu L(\mu, \boldsymbol{\lambda}). \tag{16}$$

The sub-gradient of function $f(\cdot)$ at $\boldsymbol{\lambda}$ is given by $\left[\mathcal{R}_c^{\mathscr{M}^\times}(\mu_{\boldsymbol{\lambda}}) - \rho, \mathcal{R}^f(\mu_{\boldsymbol{\lambda}}) - 1 + \delta\right]^T$, where policy $\mu_{\boldsymbol{\lambda}}$ maximizes (16). Such a policy exists because the POMDP $\mathscr{M}^\times$ has finite state, observation, and action spaces [33]. The `EG-CPOMDP` algorithm, which is described in detail in Algorithm 1, uses this sub-gradient to iteratively update $\boldsymbol{\lambda}$. The

Algorithm 1 EG-CPOMDP Algorithm

Input: Constrained product POMDP $\mathscr{M}^\times$, learning rate η
Initialize $\boldsymbol{\lambda}_1 = [B/3, B/3]$
for $k = 1, \ldots, K$ **do**
$\mu_k \leftarrow \text{OPT}(\mathscr{M}^\times, \boldsymbol{\lambda}_k)$
$\hat{p}_k \leftarrow \text{EVAL}(\mu_k, r^f)$
$\hat{r}_k^{c\times} \leftarrow \text{EVAL}(\mu_k, r^{c\times})$

$\frac{\partial f}{\partial \lambda^f} \leftarrow \hat{p}_k - 1 + \delta$
$\frac{\partial f}{\partial \lambda^c} \leftarrow \hat{r}_k^{c\times} - \rho$

$\lambda_{k+1}^f \leftarrow B \frac{\lambda_k^f e^{-\eta(\hat{p}_k - 1 + \delta)}}{B + \lambda_k^f (e^{-\eta(\hat{p}_k - 1 + \delta)} - 1) + \lambda_k^c (e^{-\eta(\hat{r}_k^{c\times} - \rho)} - 1)}$
$\lambda_{k+1}^c \leftarrow B \frac{\lambda_k^c e^{-\eta(\hat{r}_k^{c\times} - \rho)}}{B + \lambda_k^f (e^{-\eta(\hat{p}_k - 1 + \delta)} - 1) + \lambda_k^c (e^{-\eta(\hat{r}_k^{c\times} - \rho)} - 1)}$
end for
Output: $\bar{\mu} = \frac{\sum_{k=1}^K \mu_k}{K}$, $\bar{\boldsymbol{\lambda}} = \frac{\sum_{k=1}^K \boldsymbol{\lambda}_k}{K}$

value of $\boldsymbol{\lambda}$ at the kth iteration is denoted by $\boldsymbol{\lambda}_k$ and the corresponding maximizing policy (that achieves $\sup_\mu L(\mu, \boldsymbol{\lambda}_k)$) is denoted by μ_k. In the kth iteration, computing the subgradient at $\boldsymbol{\lambda}_k$ involves two key steps: (a) $\text{OPT}(\mathscr{M}^\times, \boldsymbol{\lambda}_k)$ which solves an unconstrained POMDP problem with expected reward given by $L(\mu, \boldsymbol{\lambda}_k)$ and returns the maximizing policy μ_k; (b) $\text{EVAL}(\mu_k, r^{c\times})$, which estimates $\mathcal{R}_c^{\mathscr{M}^\times}(\mu_k)$, and $\text{EVAL}(\mu_k, r^f)$, which estimates $\mathcal{R}^f(\mu_k)$.

For a mixed policy $\mu_k = \sum_i \alpha_i \pi_i$, where π_i are pure policies, $\text{EVAL}(\mu_k, r)$ is evaluated as $\sum_i \alpha_i \text{EVAL}(\pi_i, r)$. The algorithm does not depend on which methods are used for solving the unconstrained POMDP and evaluating the constraints as long as they satisfy the following assumption.

Assumption 2: The POMDP solver $\text{OPT}(\mathscr{M}^\times, \boldsymbol{\lambda}_k)$ and the expected reward estimators $\text{EVAL}(\mu_k, r^{c\times})$ and $\text{EVAL}(\mu_k, r^f)$ used in Algorithm 1 are exact, i.e.,

$$\text{OPT}(\mathscr{M}^\times, \boldsymbol{\lambda}_k) = \arg\sup_\mu L(\mu, \boldsymbol{\lambda}_k) \quad \forall\ \boldsymbol{\lambda}_k, \text{ and } \forall\ \mu_k$$

$$\text{EVAL}(\mu_k, r^{c\times}) = \mathcal{R}_c^{\mathscr{M}^\times}(\mu_k) \text{ and } \text{EVAL}(\mu_k, r^f) = \mathcal{R}^f(\mu_k).$$

Remark 2: For solving an unconstrained POMDP, it is sufficient to consider pure policies and therefore, most solvers optimize only over the space of pure policies. Thus, the support size of μ_k in Algorithm 1 is 1.

Algorithm 1 runs for K iterations and returns a policy $\bar{\mu} = (1/K)\sum_{k=1}^K \mu_k$, which is a mixed policy that assigns a probability of $1/K$ to each μ_k for $k = 1, \cdots, K$. The following theorem states that the policy $\bar{\mu}$ obtained from Algorithm 1 is an approximately optimal and feasible policy for Problem (P2).

Theorem 2: Under Assumptions 1 and 2, if $\eta = \sqrt{\frac{\log 3}{2KB^2G^2}}$ in Algorithm 1, the policy $\bar{\mu}$ returned by Algorithm 1 satisfies:

$$\mathcal{R}_o^{\mathscr{M}}(\bar{\mu}) \geq \mathcal{R}^* - 2BG\sqrt{2\log 3/K}, \tag{17}$$
$$\mathcal{R}_c^{\mathscr{M}}(\bar{\mu}) \geq \rho - \left(\frac{\mathbf{R}_o^{max} - \mathcal{R}^* + 2BG\sqrt{2\log 3/K}}{B}\right),$$
$$\mathbb{P}_\varphi^{\mathscr{M}}(\bar{\mu}) \geq 1 - \delta - \left(\frac{\mathbf{R}_o^{max} - \mathcal{R}^* + 2BG\sqrt{2\log 3/K}}{B}\right),$$

where $G = \max\{\mathbf{R}_c^{max}, 1\}$.

If (P1) is feasible, then $\mathcal{R}^*$ is finite and Theorem 2 guarantees that Algorithm 1 returns an approximately optimal and approximately feasible policy $\bar{\mu}$. If (P1) is not feasible, then $\mathcal{R}^* = -\infty$, hence the 3 inequalities in Theorem 2 are trivially true. In either case, we can determine from Algorithm 1 how far $\bar{\mu}$ is from being feasible. This is because, in the kth iteration, the algorithm evaluates the constraints for policy μ_k. The average of these constraint values is exactly the constraint value for policy $\bar{\mu}$.

In Theorem 2, we make use of the assumption that Algorithm 1 has access to an exact unconstrained POMDP solver and an exact method for evaluating $\mathcal{R}_c^{\mathscr{M}^\times}(\mu)$ and $\mathcal{R}^f(\mu)$ (see Assumption 2). In practice, however, methods for solving POMDPs and evaluating policies are approximate. We next describe the performance of Algorithm 1 under the more practical assumption that the POMDP solver returns an ϵ-optimal policy and the evaluation function has an ϵ-error.

Assumption 3: The POMDP solver $\text{OPT}(\mathscr{M}^\times, \boldsymbol{\lambda}_k)$ and the expected reward estimators $\text{EVAL}(\mu_k, r^f)$ and $\text{EVAL}(\mu_k, r^{c\times})$ used in Algorithm 1 are approximate, i.e., for all $\boldsymbol{\lambda}_k$ and $\mu_k = \text{OPT}(\mathscr{M}^\times, \boldsymbol{\lambda}_k)$, we have

$$\sup_\mu L(\mu, \boldsymbol{\lambda}_k) - L(\mu_k, \boldsymbol{\lambda}_k) \leq \epsilon_{bp},$$
$$|\text{EVAL}(\mu, r^{c\times}) - \mathcal{R}_c^{\mathscr{M}^\times}(\mu)| \leq \frac{\epsilon_{est}}{2B} \quad \forall\ \mu, \qquad \text{and}$$
$$|\text{EVAL}(\mu, r^f) - \mathcal{R}^f(\mu)| \leq \frac{\epsilon_{est}}{2B}.$$

A result similar to Theorem 2 can be obtained under Assumption 3 by using similar arguments.

Theorem 3: Under Assumptions 1 and 3, if $\eta = \sqrt{\frac{\log 3}{2KB^2G^2}}$ in Algorithm 1, the policy $\bar{\mu}$ returned by Algorithm 1 satisfies:

$$\mathcal{R}_o^{\mathscr{M}}(\bar{\mu}) \geq \mathcal{R}^* - \left(2BG\sqrt{2\log 3/K} + \epsilon_{tot}\right), \tag{18}$$
$$\mathcal{R}_c^{\mathscr{M}}(\bar{\mu}) \geq \rho - \left(\frac{\mathbf{R}_o^{max} - \mathcal{R}^* + 2BG\sqrt{2\log 3/K} + \epsilon_{tot}}{B}\right),$$
$$\mathbb{P}_\varphi^{\mathscr{M}}(\bar{\mu}) \geq 1 - \delta - \left(\frac{\mathbf{R}_o^{max} - \mathcal{R}^* + 2BG\sqrt{2\log 3/K} + \epsilon_{tot}}{B}\right),$$

where $\epsilon_{tot} = \epsilon_{bp} + 2\epsilon_{est}$ and $G = \max\{\mathbf{R}_c^{max}, 1\}$.

In Lemma 1 and Theorems 2 and 3, we can replace $\mathcal{R}^*$ by its lower bound (e.g., replacing $\mathcal{R}^*$ by 0 in scenarios with non-negative objective rewards) to obtain slightly relaxed, yet more computable lower bounds for the performance of the returned policy.

Remark 3: Algorithm 1 uses K pure policies to generate the mixed policy $\bar{\mu}$. If we wish to obtain a mixed policy with a small support, we can define $\bar{\mu} = \sum_{k=1}^K w_k \mu_k$, where w_ks are a *basic feasible solution* (BFS) of the following linear

program:

$$\begin{aligned} \sup_{w\geq 0, \|w\|_1 \leq 1} \quad & \sum_{k=1}^{K} w_k \mathcal{R}_o^{\mathscr{M}^\times}(\mu_k) \\ \text{s.t.} \quad & \sum_{k=1}^{K} w_k \mathcal{R}_c^{\mathscr{M}^\times} \geq \rho - o(1/\sqrt{K}), \\ & \sum_{k=1}^{K} w_k \mathcal{R}^f(\mu_k) \geq 1 - \delta - o(1/\sqrt{K}). \end{aligned} \tag{BFS}$$

This leads to a mixed policy $\bar{\mu}$ whose support size is at most three.

We next consider three special cases of our problem with different types of time horizon T.

Remark 4: When the horizon T is a constant and the constraint reward r_t^c is non-negative, Assumption 1 is trivially true and, therefore, Theorem 2 holds.

Remark 5: Let $\{E_t : t = 0, 1, 2, ...\}$ be a sequence of i.i.d. Bernoulli random variables with $\mathbb{P}[E_0 = 1] = 1 - \gamma$, $\gamma < 1$. Let the geometrically-distributed time-horizon T be defined as

$$T = \min\{t : E_t = 1\}. \tag{19}$$

The random horizon T has an expected value of $\gamma/(1-\gamma)$ under any policy and thus satisfies Assumption 1 if the constraint reward r_t^c is non-negative. The following lemma shows that with such a geometric time horizon, the policy optimization problem for $L(\mu, \boldsymbol{\lambda}_k)$ is equivalent to solving an infinite horizon discounted-reward POMDP.

Lemma 2: For a given $\boldsymbol{\lambda}$, maximizing $L(\mu, \boldsymbol{\lambda})$ over μ under a geometrically distributed time horizon is equivalent to maximizing the following infinite horizon discounted reward

$$\mathbb{E}_\mu \left[\sum_{t=0}^{\infty} \gamma^t \left(r_t^{o\times}(X_t, \mathcal{U}_t) + \frac{\lambda^f (1-\gamma)}{\gamma} r^f(X_t) + \lambda^c r_t^{c\times}(X_t, \mathcal{U}_t) \right) \right].$$

Thus, by using appropriate POMDP solvers and a Monte Carlo method for constraint reward evaluation, we can implement Algorithm 1 for constant and geometrically distributed time horizons.

Goal-POMDPs: In Goal-POMDPs, there is a set of goal states GO and the POMDP run terminates once the goal state is reached. We assume that every possible action in every non-goal state results in a strictly positive cost (or strictly negative reward). The objective is to obtain a policy that maximizes the total expected reward while the probability of satisfying the specification φ is at least $1-\delta$. (In this section, for simplicity of discussion, we do not consider a constraint reward function.) Under this reward assumption, for any given policy, the expected time taken to reach the goal is infinite if and only if the total expected reward is negative infinity. Assumption 1 may not be true a priori in this setting. However, we can exclude every policy with negative infinite expected reward without any loss of optimality. After this exclusion of policies, Assumption 1 holds and all our results in the previous sections are applicable.

Consider the product POMDP $\mathscr{M}^\times = (\mathcal{S}^\times, \mathcal{U}^\times, P^\times, s_0^\times, r^{o\times}, r^f, \varpi, O, Z^\times)$. Let $GO \subseteq \mathcal{S}^\times$ be a set of goal states. For every non-goal state-action pair (x, u) in the product MDP, let $r^{o\times}(x, u) < 0$. This is a constrained Goal-POMDP, which can be solved using the Lagrangian approach discussed earlier. The problem of optimizing the Lagrangian function for a given λ can be reformulated as an unconstrained Goal-POMDP with minor modifications which can be solved using solvers like Goal-HSVI [34]. We make modifications to ensure that there is a single goal state and the rewards for every non-goal state action pair are strictly negative. The rewards and transitions until time T are the same as in the product MDP. We replace the goal states in GO with a unique goal state g. When a state $x \in GO$ is reached (at T), the process goes on for two more time steps $T+1$ and $T+2$. At time $T+1$, the agent receives a reward

$$\lambda(r^f(X_{T+1}) - 2 + \delta). \tag{20}$$

This ensures that the reward is strictly negative. At $T+2$, the agent reaches the goal state g. One can easily show that this modified Goal-POMDP is equivalent to optimizing the Lagrangian function for a given λ.

Remark 6: The framework developed in this section can easily be generalized to the setting where there are multiple LTL_f specifications and reward constraints.

VI. Multi-Agent Systems

In previous sections, we have discussed how we can incorporate LTL_f specifications in a single-agent POMDP and solve it using a Lagrangian approach. We will now consider a setting in which there are multiple agents and these agents select their actions based on different information. In this section, for the sake of simplicity in presentation, we will consider a single LTL_f specification, and no reward constraints.

We can solve a multi-agent problem with LTL_f constraints by first converting it into a problem with reward-like constraints as we did in Section IV-A. We can then use the exponentiated gradient algorithm to solve this constrained multi-agent problem. In the exponentiated gradient algorithm, we need to solve unconstrained multi-agent problems. One approach for solving a large class of unconstrained multi-agent problems is the *common information approach* [11]. This approach transforms the unconstrained multi-agent problem into a single-agent unconstrained POMDP (with enlarged state and action spaces) which, in principle, can be solved using POMDP solvers. While this approach is conceptually sound, it is computationally intractable in general. We now discuss a multi-agent model with a specific structure and illustrate how some of the computational issues can be mitigated.

We consider a class of labeled multi-agent systems which can be characterized by a tuple $\mathscr{M} = (N, \mathcal{S}, \mathcal{S}^{\text{loc}}, \mathcal{U}, P, P^{\text{loc}}, \varpi, AP, L, r, T)$. N is the number of agents in the system, $\mathcal{S}$ is a finite *shared* state space, and $\mathcal{U}$ is a finite *joint* action space. The shared state at time t is denoted by $\mathcal{S}_t$; the action of agent i at time t is denoted by $\mathcal{U}_t^i$; and $\mathcal{U}_t = (\mathcal{U}_t^1, \ldots, \mathcal{U}_t^N)$ denotes the joint action. $P_t(s, u; s')$ is the probability of transitioning from state s to state s' on taking joint action u. The shared state and the agents' actions can be observed by all the agents in

the system. Each agent i has an associated local state $\mathcal{S}_t^{\mathrm{loc,i}}$ at time t. The evolution of agent i's local state is captured by the local transition function $P_t^{\mathrm{loc,i}} : \mathcal{S}^{\mathrm{loc,i}} \times \mathcal{S} \times \mathcal{U} \to \Delta_{\mathcal{S}^{\mathrm{loc,i}}}$ where $P_t^{\mathrm{loc,i}}(s^{\mathrm{loc,i}}, s, u; s^{\mathrm{loc,i}'})$ is the probability of transitioning to local state $s^{\mathrm{loc,i}'}$ if the current local state is $s^{\mathrm{loc,i}}$, the current global state is s, and the joint action u is taken. For convenience, let us denote $\bar{\mathcal{S}} = \mathcal{S} \times \mathcal{S}^{\mathrm{loc}}$, where $\mathcal{S}^{\mathrm{loc}} = \prod_{i=1}^{N} \mathcal{S}^{\mathrm{loc,i}}$. Note that the local state transitions among agents are mutually independent given the shared state and action, and the state dynamics can be viewed as

$$\mathcal{S}_{t+1} = \zeta_t^g(\mathcal{S}_t, \mathcal{U}_t, W_t^g), \tag{21}$$

$$\mathcal{S}_{t+1}^{\mathrm{loc,i}} = \zeta_t^g(\mathcal{S}_t^{\mathrm{loc,i}}, \mathcal{S}_t, \mathcal{U}_t, W_t^i), \tag{22}$$

where ζ_t^g and ζ_t^i are fixed transformations and W_t^g, W_t^i are the noise variables associated with the dynamics. $\varpi \in \Delta_{\bar{\mathcal{S}}}$ is the initial state distribution. AP is a set of atomic propositions, $L : \bar{\mathcal{S}} \to 2^{AP}$ is a labeling function which indicates the set of atomic propositions which are true in each state, $r_t : \bar{\mathcal{S}} \times \mathcal{U} \to \mathbb{R}$ is a reward function. The system runs for a random time horizon T. This random time may be independent of the multi-agent system, or could be a stopping time with respect to the common information.

Information Structure: Agents have access to different information. The information that agent i can possibly use to select its actions at time t is given by

$$I_t^i = \{\mathcal{S}_{0:t}, \mathcal{U}_{0:t-1}, \mathcal{S}_{0:t}^{\mathrm{loc,i}}\}. \tag{23}$$

The *common information* available to all the agents at time t is denoted by

$$C_t = \{\mathcal{S}_{0:t}, \mathcal{U}_{0:t-1}\}, \tag{24}$$

and the *private information* that is available only to agent i at time t is denoted by

$$P_t^i = \{\mathcal{S}_{0:t}^{\mathrm{loc,i}}\}. \tag{25}$$

This information structure is referred to as the control-sharing information structure [35]. A *control law* π_t^i for agent i maps its information I_t^i to its action $\mathcal{U}_t^i$, i.e., $\mathcal{U}_t^i = \pi_t^i(I_t^i)$. The collection of control laws $\pi^i := (\pi_0^i, \pi_1^i, \dots,)$ over the entire horizon is referred to as agent i's *policy*. The set of all such policies for agent i is denoted by $\mathcal{P}^i$. The team's policy is denoted by $\pi = (\pi^1, \dots, \pi^N)$. We refer to such deterministic team policies as pure policies and denote the set of all pure team policies with $\mathcal{P}$.

The set of mixed policies is defined in the same manner as in (1). Given a mixed policy μ, the team randomly selects a pure policy π with probability $\mu(\pi)$. This random selection happens using shared randomness which can be achieved by the agents in the team through a pseudo-random number generator with a common seed.

A *run* ξ of the POMDP is the sequence of states and actions $(\bar{\mathcal{S}}_0, \mathcal{U}_0)(\bar{\mathcal{S}}_1, \mathcal{U}_1) \cdots (\bar{\mathcal{S}}_T, \mathcal{U}_T)$. The total expected reward associated with a team policy μ is given by

$$\begin{aligned}\mathcal{R}^{\mathscr{M}}(\mu) &= \mathbb{E}_{\mu}^{\mathscr{M}}\left[\sum_{t=0}^{T} r_t(\bar{\mathcal{S}}_t, \mathcal{U}_t)\right] \\ &= \sum_{\pi \in \mathrm{supp}(\mu)} \left[\mu(\pi)\mathbb{E}_{\pi}^{\mathscr{M}}\left[\sum_{t=0}^{T} r_t(\bar{\mathcal{S}}_t, \mathcal{U}_t)\right]\right].\end{aligned} \tag{26}$$

Let φ be an LTL_f specification defined using the labeling function L. The probability that a run of $\mathscr{M}$ satisfies φ under policy μ is denoted by $\mathbb{P}_{\varphi}^{\mathscr{M}}(\mu)$. We want to solve the following constrained optimization problem for the team of agents:

$$\textbf{LTL}_f\textbf{-MA:} \quad \begin{aligned} \sup_{\mu} \quad & \mathcal{R}^{\mathscr{M}}(\mu) \\ \text{s.t.} \quad & \mathbb{P}_{\varphi}^{\mathscr{M}}(\mu) \geq 1 - \delta. \end{aligned} \tag{MP1}$$

A. Constrained Product Multi-Agent Problem

We construct a constrained product multi-agent problem similar to the constrained product POMDP in Section IV-A. In this construction, the system state space is $\bar{\mathcal{S}}$. Let the DFA associated with the specification φ be $\mathscr{A} = (Q, \Sigma, q_0, \delta, F)$. Hence, the product state space is $\bar{\mathcal{S}} \times Q$. The action space is A. The transitions and rewards are constructed in the same manner discussed in Section IV-A. This leads to the following multi-agent problem with reward-like constraints:

$$\textbf{C-MA:} \quad \begin{aligned} \sup_{\mu} \quad & \mathcal{R}^{\mathscr{M}^{\times}}(\mu) \\ \text{s.t.} \quad & \mathcal{R}^{f}(\mu) \geq 1 - \delta. \end{aligned} \tag{MP2}$$

B. Global and Local Specifications

Consider that the specification φ admits additional structure. For each agent, $L^i : \mathcal{S} \times \mathcal{S}^{\mathrm{loc,i}} \to 2^{AP}$ is a labeling function and φ^i a local specification defined with respect to it. There is a shared labeling function $L^g : \mathcal{S} \to 2^{AP}$ and a shared specification φ^g defined with respect to L^g. The overall specification φ is the conjunction of the local and shared specifications, i.e.,

$$\varphi = \varphi^g \wedge \varphi^1 \wedge \cdots \wedge \varphi^N. \tag{27}$$

For each local specification, let the corresponding DFA be $\mathscr{A}^i = (Q^i, \Sigma^i, q_0^i, \delta^i, F^i)$ and for the shared specification, let the corresponding DFA be $\mathscr{A}^g = (Q^g, \Sigma^g, q_0^g, \delta^g, F^g)$.

Instead of using the product construction in the previous section, we can construct the product state as

$$\begin{aligned}(\bar{\mathcal{S}}_t, Q_t^g, Q_t^1, \dots, Q_t^N) &= (\mathcal{S}_t, Q_t, \mathcal{S}_t^{\mathrm{loc},1}, Q_t^1, \dots, \mathcal{S}_t^{\mathrm{loc,N}}, Q_t^N) \\ &=: (X_t^g, X_t^1, \dots, X_t^N).\end{aligned}$$

The shared product state X_t^g evolves as

$$\begin{aligned}X_{t+1}^g &= (\mathcal{S}_{t+1}, Q_{t+1}) \\ &= (\zeta_t^g(\mathcal{S}_t, \mathcal{U}_t, W_t^g), \delta^g(Q_t, L^g(\mathcal{S}_t))).\end{aligned} \tag{28}$$

The local product state X_t^i evolves as

$$\begin{aligned}X_{t+1}^i &= (\mathcal{S}_{t+1}^{\mathrm{loc,i}}, Q_{t+1}^i) \\ &= (\zeta_t^i(\mathcal{S}_t, \mathcal{S}_t^{\mathrm{loc,i}}, \mathcal{U}_t, W_t^i), \delta^i(Q_t^i, L^i(\mathcal{S}_t, \mathcal{S}_t^{\mathrm{loc,i}}))).\end{aligned} \tag{29}$$

The reward is

$$r_t^{\times}((\bar{s},q^g,q^1,\ldots,q^N),u) = r_t(\bar{s},u),\ \forall \bar{s},q,u \tag{30}$$
$$r^f((\bar{s},q^g,q^1,\ldots,q^N)) = \begin{cases}1, & \text{if } q^g \in F^g, q^i \in F^i\ \forall i\\ 0, & \text{otherwise.}\end{cases}$$

Under this specification structure and modified construction, the *product problem* also conforms to the control sharing model [35]. For this model, it has been shown that there exist optimal policies in which agent i uses *reduced* private information $\{\mathcal{S}^{\mathrm{loc,i}}_t, Q^i_t\}$ (as opposed to $\{\mathcal{S}^{\mathrm{loc,i}}_{0:t}, Q^i_{0:t}\}$) and the common information $C_t = \{\mathcal{S}_{0:t}, Q_{0:t}, \mathcal{U}_{0:t-1}\}$ to choose its actions. This can be formally stated as the lemma below.

Lemma 3 (Policy Space Reduction): In Problem (MP2), we can restrict our attention to control laws of the form below without loss of optimality

$$\mathcal{U}^i_t = \pi^i_t(\mathcal{S}^{\mathrm{loc,i}}_t, Q^i_t, \mathcal{S}_{0:t}, \mathcal{U}_{0:t-1}). \tag{31}$$

Proof: Given an arbitrary team policy π, we can construct a team policy $\bar{\pi}$ of the form above that achieves the same expected reward (including the constraint). See [35, Proposition 3] for a complete proof. ∎

Remark 7: The substantial reduction in policy space under Lemma 3 is achieved by the modified construction of the product multi-agent problem. Using the construction in Section VI-A would not have enabled us in achieving this reduction. A similar private information reduction can be achieved for models with transition independence [36].

C. Algorithm

We can use Algorithm 1 to solve Problem (MP2). The unconstrained solver OPT in Algorithm 1 is now an unconstrained multi-agent solver. One way to implement such a solver is to use the common information approach [11]. Under assumptions similar to the ones made in the previous sections, owing to similar reward structures in the constrained multi-agent problem, the results derived in the previous sections can be extended to this setting in a straightforward manner.

VII. Experiments

We consider a collection of gridworld problems in which an agent or a team of agents needs to maximize their reward while satisfying an LTL_f specification. We consider three types of tasks, i.e., reach-avoid, ordered, and reactive tasks. In all of our experiments, we use the SARSOP [8] solver for finding the optimal policy μ_k and a default discount factor of 0.99. To estimate the constraint function, we use Monte Carlo simulations. We use the online tool $\mathrm{LTL_f2DFA}$ [37] based on MONA [38] to generate an equivalent DFA for an LTL_f formula.

For each location (i,j) in the grid, $L[(i,j)]$ is the label assigned to it by the labeling function L (see Section III-A). The images corresponding to the various models indicate the grid space and the associated labeling function, e.g., in Fig. 1a, we have $L[(5,2)] = \{b\}, L[(1,6)] = \{b\}, L[(7,7)] = \{a\}$ and $L[(i,j)] = \{\}$ for all other grid locations (i,j). In all single agent models, the agent starts from the grid location $(0,0)$. Further, the default reward for all actions is 0 in all grid locations, unless specified otherwise.

In the experiments of Section VII-A and VII-B, the agent's transitions in the gridworld are stochastic, i.e., if the agent decides to move in a certain direction, it moves in that direction with probability 0.95 and, with probability 0.05, it randomly moves one step in any direction that is not opposite to its intended direction. Further, the agent's observation of its current location is noisy – it is equally likely to be the agent's true current location or any location neighboring its current location.

For each model discussed below, we use Algorithm 1 to generate a mixed policy $\bar{\mu}$. The corresponding reward $\mathcal{R}^{\mathscr{M}}(\bar{\mu})$ and constraint $\mathcal{R}^f(\bar{\mu})$ are shown in Table II. We observe that the probability of satisfying the constraint generally exceeds the required threshold. Occasionally, the constraint is violated albeit only by a small margin. This is consistent with our result in Theorem 2. We also observe that the agent behaves in a manner that achieves high reward in all of these models.

A. Single-Agent Location Uncertainty with Reach-Avoid Task

In this task, we are interested in reaching a goal state a and always avoiding unsafe states b. This can be specified using LTL_f as $\varphi_1 = \mathbf{F}a \wedge (\mathbf{G}\neg b)$. This task was performed on model $\mathscr{M}_1$ with discounted reward $r((1,6),u) = 3, r((4,3),u) = 3$, and $r((7,7),u) = 1$ for all actions u.

In this experiment, we observe two characteristic behaviors. The agent reaches the goal state a and remains there. This behavior ensures that the specification is met but the reward is relatively lower. Following the other behavior, the agent goes towards the location $(4,3)$ and tries to remain there to obtain higher reward. However, since the obstacle is very close and the transitions are stochastic, it is prone to violating the constraint. Nonetheless, this violation is rare enough and the overall satisfaction probability exceeds the desired threshold.

(a) Model $\mathscr{M}_1$. (b) Model $\mathscr{M}_2$.

Fig. 1: Single-agent location uncertainty.

B. Single-Agent Location Uncertainty with Reactive Task

In this task, there are four states of interest: a, b, c, and d. The agent must eventually reach a or b. However, if it reaches b, then it must visit c without visiting d. This can be expressed as $\varphi_2 = \mathbf{F}(a \vee b) \wedge \mathbf{G}(b \rightarrow (\neg d \mathbf{U} c))$. This task was performed on model $\mathscr{M}_2$ in Fig. 1b with discounted reward $r((3,0),u) = 1$ and $r((3,3),u) = 2$ for all actions u.

In this experiment, under the policy returned by Algorithm 1 for $B = 10$ and $1 - \delta = 0.8$, the agent goes to a and

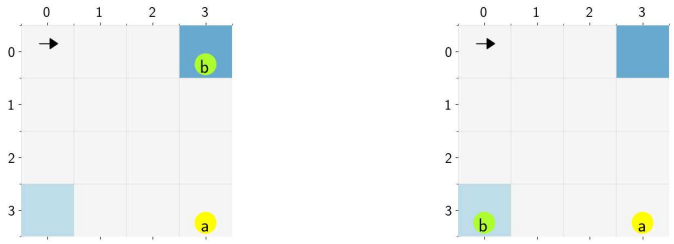
TABLE I: Results for different B and δ for the setting of Section VII-B.

$\mathcal{R}^{\mathscr{M}}(\bar{\mu})$	$\mathcal{R}^{f}(\bar{\mu})$	$1-\delta$	B	η	K
1.33	0.49	0.5	5	2	100
1.31	0.51	0.5	50	2	100
1.31	0.52	0.5	500	2	100
1.14	0.69	0.7	5	2	100
1.12	0.71	0.7	50	2	100
1.11	0.72	0.7	500	2	100
0.95	0.89	0.9	5	2	100
0.94	0.90	0.9	50	2	100
0.92	0.92	0.9	500	2	100

remains there, thus satisfying the constraint. Occasionally, the agent also goes to state b and remains there to obtain a larger reward. This behavior violates the constraint, since if the agent ever visits b, it must eventually go to c. We observe that this occurrence is rare enough and the overall satisfaction probability is close to the desired threshold.

We extend our experiment by varying B and δ, presenting the results in Table I. All the final policies either exceed or narrowly miss the desired satisfaction probability threshold $1-\delta$. Notably, as B increases while keeping δ constant, the satisfaction probability tends to rise with minimal impact on the total objective value. This observations aligns with the bounds presented in Theorem 3.

In the experiments of Section VII-C, VII-D, and VII-E, the agents' transitions in the gridworld are deterministic.



(a) Obstacle at $(0,3)$. (b) Obstacle at $(3,0)$.

Fig. 2: Model $\mathscr{M}_3$: Reach-avoid task.

C. Single-Agent Predicate Uncertainty with Reach-Avoid Task

In this task, the reach-avoid specification is the same as φ_1 in Section VII-A and was performed on model $\mathscr{M}_3$ in Fig. 2 with discounted reward $r((3,0),u)=2$ and $r((0,3),u)=4$ for all actions u. However, the agent does not know which location to avoid, as there is uncertainty in the location of the object that the agent has to avoid. There are two possible locations for object b: $(3,0)$ and $(0,3)$. In both cases, whenever the agent is far away (Manhattan distance greater than 1) from the object b, it gets an observation 'F' indicating that it is *far* with probability 1. When the object is at the bottom left and the agent is adjacent to it, the agent gets an observation 'C' with probability 0.9 indicating that the object is *close*. However, if object b is at the top right and the agent is adjacent to it, the agent gets an observation 'C' only with probability 0.1. Therefore, the detection capability of the agent is stronger when the object is in the bottom-left location as opposed to when it is in the top-right location.

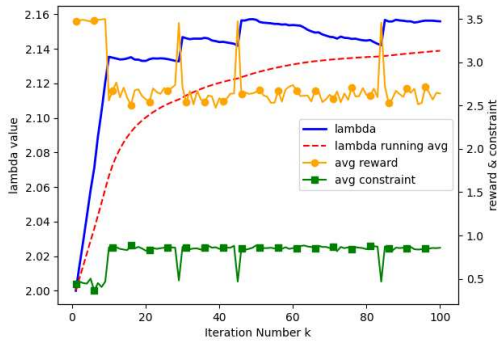


Fig. 3: This plot illustrates how the Lagrange multiplier λ_k, the reward $\mathcal{R}^{\mathscr{M}}(\mu_k)$, and the probability of satisfaction $\mathbb{P}^{\mathscr{M}}_{\varphi}(\mu_k)$ evolve with k for the experiment in Section VII-C.

In this experiment, the agent receives high reward when it remains in the top-right corner compared to a moderate reward in the bottom-left corner. Further, the agent's detection capability is better when it is in the bottom-left region than when it is in the top-right region. Thus, it generally first heads towards the location a (since it has to eventually visit it) via the bottom-left region without hitting the corner and acquires information on where the object is located. After reaching a, it goes to the top-right corner if the obstacle is *not* located there, and bottom-left corner otherwise. We see rare instances where the agent completely ignores the constraint and just maximizes the reward.

Figure 3 illustrates the performance of the various policies μ_k. The Lagrange multiplier λ_k shows a decreasing trend as long as the constraint continues to be met (which happens for the majority of the iterations). However, the multiplier eventually diminishes to a point where the constraint is breached and we observe noticeable increase in reward. These spikes contribute to the overall average reward. Given the significant constraint violation, the Lagrange multiplier experiences an uptick. This cycle ensures that constraint violation occurs rarely. Since we pick a policy randomly and with uniform distribution, the average error probability is still close to the threshold (see Table II).

D. Goal-POMDP Specification Uncertainty with Random Ordered Task

In this task, the agent needs to visit state a and b strictly in that order or *vice versa*. Thus, the uncertainty lies in the specification which the agent is required to satisfy. Either specification is chosen with uniform probability at the start (indicated by the truth value of c) and kept fixed. The information regarding the truth value of c is unknown to the agent until it is revealed by an observation from a particular gridworld state. This setting can be specified using LTL_f as $\varphi_4 = (c \rightarrow (\neg b \mathbf{U}(a \wedge \mathbf{F}b))) \wedge (\neg c \rightarrow (\neg a \mathbf{U}(b \wedge \mathbf{F}a)))$.

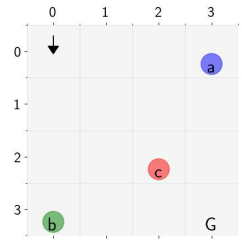


Fig. 4: Model $\mathscr{M}_4$: Random ordered task with goal.

This task was performed on model $\mathscr{M}_4$ in Fig. 4 with the observation corresponding to grid location $(2, 2)$ providing the truth value of c. In addition, the agent seeks to further reach the goal state, i.e., grid location $(3, 3)$. The agent receives a reward of -1 for all state-action pairs before reaching a goal state and 0 reward in the goal state. In the experiment, the agent first visits c, then visits a and b in the order corresponding to the truth value of c, and finally remains in the goal state.

E. Multi-Agent System Collision Avoidance with Random Ordered Task and One-Way Lane

In this task, there are two agents A_1 and A_2, whose locations are known to each other. Agent A_1 has one goal location a while agent A_2 has two goal locations b and c. An agent receives a positive reward only when it is at its goal locations. We require agent A_2 to visit both its goal locations, but in a specific order (i.e., b to c or c to b). The goal locations are known to both agents but the desired visitation order is known only to agent A_2. This order is randomly chosen at the beginning and kept fixed (indicated by the truth value of o). We require the agents to avoid collisions between themselves (indicated by the truth value of col) and to always stay within the lanes (indicated by the truth value of s). This requirements can be specified using LTL$_f$ as $\varphi_5 = (o \rightarrow (\neg b \mathbf{U}(c \wedge \mathbf{F}b))) \wedge (\neg o \rightarrow (\neg c \mathbf{U}(b \wedge \mathbf{F}c))) \wedge (\mathbf{G}(s \wedge \neg col))$.

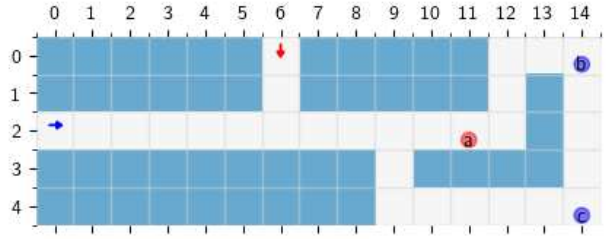


Fig. 5: Multi-agent system collision avoidance benchmark with random ordered tasks, one-way lane, and model $\mathscr{M}_5$.

This task was performed on model $\mathscr{M}_5$ with a larger grid size of 5×15 and a complex lane structure. The lanes are marked by the grey grid cells in Fig. 5. Further, we allow only one-way movement in some lanes. In model $\mathscr{M}_5$, agent A_2 (blue arrow), starting from $(2, 0)$, has two goal locations b (0, 14) and c (4, 14), while agent A_1 (red arrow), starting from $(0, 6)$, has one goal location a (2, 11). Each agent receives a (discounted) reward of 1 for each time step in their own goal locations. The environment allows only one-way movement in the lane from $(4, 14)$ to $(0, 14)$. Thus, if o indicates that agent A_2 has to visit c before b, it can do so through the one-way lane between c and b. If o indicates the other order of visitation, then agent A_2 has to take the longer route from b to c which passes through agent A_1's goal location, i.e., (2, 11). Our result matches this expectation, with the agents moving in a manner so as to avoid collision and maximize the total reward.

TABLE II: Reward and constraint performance of the policy $\bar{\mu}$ under various models and specifications.

Model	**Spec**	$\mathcal{R}^{\mathscr{M}}(\bar{\mu})$	$\mathcal{R}^{f}(\bar{\mu})$	$1-\delta$	B
$\mathscr{M}_1$	φ_1	0.95	0.70	0.70	8
$\mathscr{M}_2$	φ_2	1.01	0.79	0.80	10
$\mathscr{M}_3$	φ_1	2.73	0.81	0.85	20
$\mathscr{M}_4$	φ_4	-14	0.87	0.9	100
$\mathscr{M}_5$	φ_5	1.59	0.72	0.7	50

VIII. Conclusions

We provided a methodology for designing agent policies that maximize the total expected reward while ensuring that the probability of satisfying a linear temporal logic (LTL$_f$) specification is sufficiently high. We constructed a constrained product POMDP by augmenting the system state with the state of the DFA associated with the LTL$_f$ specification. Solving this constrained product POMDP is equivalent to solving the original problem. We provided a constrained POMDP solver based on the exponentiated gradient (EG) algorithm and derived approximation bounds for it. We then extended our methodology to a multi-agent setting with information asymmetry. For these various settings, we computed near optimal policies that satisfy the LTL$_f$ specification with sufficiently high probability. We observed in our experiments that our approach results in policies that effectively balance information acquisition (exploration), reward maximization (exploitation), and satisfaction of the specification, which is difficult to achieve using classical POMDP planning.

References

[1] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1994.

[2] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT press, 2008.

[3] K. C. Kalagarla, R. Jain, and P. Nuzzo, "Optimal Control of Discounted-Reward Markov Decision Processes Under Linear Temporal Logic Specifications," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 1268–1274.

[4] S. Sickert, J. Esparza, S. Jaax, and J. Křetínskỳ, "Limit-Deterministic Büchi Automata for Linear Temporal Logic," in *International Conference on Computer Aided Verification*. Springer, 2016, pp. 312–332.

[5] E. M. Hahn, M. Perez, S. Schewe, F. Somenzi, A. Trivedi, and D. Wojtczak, "Omega-Regular Objectives in Model-Free Reinforcement Learning," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2019, pp. 395–412.

[6] G. De Giacomo and M. Y. Vardi, "Linear temporal logic and linear dynamic logic on finite traces," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[7] S. Zhu, L. M. Tabajara, J. Li, G. Pu, and M. Y. Vardi, "Symbolic LTLf Synthesis," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017, pp. 1362–1369.

[8] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Robotics: Science and systems*, vol. 2008. Citeseer, 2008.

[9] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," *Advances in neural information processing systems*, vol. 23, 2010.

[10] E. Hazan *et al.*, "Introduction to online convex optimization," *Foundations and Trends in Optimization*, vol. 2, no. 3-4, pp. 157–325, 2016.

[11] A. Nayyar, A. Mahajan, and D. Teneketzis, "Decentralized Stochastic Control with Partial History Sharing: A Common Information Approach," *IEEE Transactions on Automatic Control*, vol. 58, no. 7, pp. 1644–1658, 2013.

[12] K. C. Kalagarla, K. Dhruva, D. Shen, R. Jain, A. Nayyar, and P. Nuzzo, "Optimal Control of Partially Observable Markov Decision Processes with Finite Linear Temporal Logic Constraints," in *Uncertainty in Artificial Intelligence*. PMLR, 2022, pp. 949–958.

[13] X. C. D. Ding, S. L. Smith, C. Belta, and D. Rus, "LTL control in uncertain environments with probabilistic satisfaction guarantees," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 3515–3520, 2011.

[14] A. K. Bozkurt, Y. Wang, M. M. Zavlanos, and M. Pajic, "Control synthesis from linear temporal logic specifications using model-free reinforcement learning," in *2020 IEEE International Conference on Robotics and Automation*, 2020, pp. 10 349–10 355.

[15] S. Haesaert, P. Nilsson, C. I. Vasile, R. Thakker, A.-a. Aghamohammadi, A. D. Ames, and R. M. Murray, "Temporal logic control of POMDPs via label-based stochastic simulation relations," *IFAC-PapersOnLine*, vol. 51, no. 16, pp. 271–276, 2018.

[16] G. Norman, D. Parker, and X. Zou, "Verification and control of partially observable probabilistic real-time systems," in *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 2015, pp. 240–255.

[17] M. Ahmadi, R. Sharan, and J. W. Burdick, "Stochastic finite state control of POMDPs with LTL specifications," *arXiv preprint arXiv:2001.07679*, 2020.

[18] R. Sharan and J. Burdick, "Finite state control of POMDPs with LTL specifications," in *2014 American Control Conference*. IEEE, 2014, pp. 501–508.

[19] J. Liu, E. Rosen, S. Zheng, S. Tellex, and G. Konidaris, "Leveraging Temporal Structure in Safety-Critical Task Specifications for POMDP Planning," *RSS Workshop Robotics for People*, 2021.

[20] M. Bouton, J. Tumova, and M. J. Kochenderfer, "Point-based methods for model checking in partially observable Markov decision processes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[21] S. Carr, N. Jansen, and U. Topcu, "Verifiable RNN-based policies for POMDPs under temporal logic constraints," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 4121–4127.

[22] S. Carr, N. Jansen, R. Wimmer, A. Serban, B. Becker, and U. Topcu, "Counterexample-Guided Strategy Improvement for POMDPs Using Recurrent Neural Networks," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019, pp. 5532–5539.

[23] P. Poupart, A. Malhotra, P. Pei, K.-E. Kim, B. Goh, and M. Bowling, "Approximate linear programming for constrained partially observable Markov decision processes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[24] J. Lee, G.-H. Kim, P. Poupart, and K.-E. Kim, "Monte-Carlo tree search for constrained POMDPs," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[25] E. Walraven and M. T. Spaan, "Column generation algorithms for constrained POMDPs," *Journal of artificial intelligence research*, vol. 62, pp. 489–533, 2018.

[26] L. Hammond, A. Abate, J. Gutierrez, and M. Wooldridge, "Multi-Agent Reinforcement Learning with Temporal Logic Specifications," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021, pp. 583–592.

[27] W. Wang, G. F. Schuppe, and J. Tumova, "Decentralized Multi-agent Coordination under MITL Tasks and Communication Constraints," in *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2022, pp. 320–321.

[28] J. Eappen and S. Jagannathan, "DistSPECTRL: Distributing specifications in multi-agent reinforcement learning systems," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2022, pp. 233–250.

[29] N. Zhang, W. Liu, and C. Belta, "Distributed Control using Reinforcement Learning with Temporal-Logic-Based Reward Shaping," in *Learning for Dynamics and Control Conference*. PMLR, 2022.

[30] C. Sun, X. Li, and C. Belta, "Automata guided semi-decentralized multi-agent reinforcement learning," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 3900–3905.

[31] M. Sipser, "Introduction to the Theory of Computation," *ACM Sigact News*, vol. 27, no. 1, pp. 27–29, 1996.

[32] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[33] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995, vol. 1.

[34] K. Horák, B. Bosanskỳ, and K. Chatterjee, "Goal-HSVI: Heuristic Search Value Iteration for Goal POMDPs." in *IJCAI*, 2018.

[35] A. Mahajan, "Optimal Decentralized Control of Coupled Subsystems With Control Sharing," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2377–2382, 2013.

[36] D. Kartik, S. Sudhakara, R. Jain, and A. Nayyar, "Optimal communication and control strategies for a multi-agent system in the presence of an adversary," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 4155–4160.

[37] F. Fuggitti, "LTLf2DFA," Mar. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3888410

[38] N. Klarlund and A. Møller, *MONA Version 1.4 User Manual*, BRICS, Department of Computer Science, University of Aarhus, January 2001, notes Series NS-01-1. Available from `http://www.brics.dk/mona/`.

APPENDIX I
PROOF OF THEOREM 1

For any policy μ, we have

$$\begin{aligned}\mathcal{R}_o^{\mathscr{M}^\times}(\mu) &= \mathbb{E}_\mu\left[\sum_{t=0}^{T} r_t^{o\times}(X_t,\mathcal{U}_t)\right]\\ &= \mathbb{E}_\mu\left[\sum_{t=0}^{T} r_t^{o\times}((\mathcal{S}_t,Q_t),\mathcal{U}_t)\right]\\ &\overset{a}{=} \mathbb{E}_\mu\left[\sum_{t=0}^{T} r_t^{o}(\mathcal{S}_t,\mathcal{U}_t)\right] = \mathcal{R}_o^{\mathscr{M}}(\mu).\end{aligned}$$

The equality in (a) follows from the definition of $r_t^{o\times}$ in (6). We can similarly show that $\mathcal{R}_c^{\mathscr{M}^\times}(\mu) = \mathcal{R}_c^{\mathscr{M}}(\mu)$. Further, using (31), we have

$$r^f(X_{T+1}) = r^f((\mathcal{S}_{T+1},Q_{T+1})) = \mathbb{1}_F(Q_{T+1}).$$

By construction of the product POMDP dynamics, a run $X_{0:T}$ of the product POMDP satisfies φ if and only $Q_{T+1} \in F$. Hence,

$$\mathcal{R}^f(\mu) = \mathbb{E}_\mu\left[r^f(X_{T+1})\right] = \mathbb{P}_\varphi^{\mathscr{M}}(\mu).$$

APPENDIX II
PROOF OF LEMMA 1

By definition of the Lagrangian function for Problem (P2), we have:

$$\begin{aligned}\mathcal{R}^* &= l^*\\ &\leq l_B^*\\ &\leq \inf_{\boldsymbol{\lambda}\geq 0,\|\boldsymbol{\lambda}\|_1\leq B} L(\bar{\mu},\boldsymbol{\lambda}) + \epsilon\\ &= \epsilon + \mathcal{R}_o^{\mathscr{M}^\times}(\bar{\mu}) + \\ &\inf_{\boldsymbol{\lambda}\geq 0,\|\boldsymbol{\lambda}\|_1\leq B}\left[\lambda^f(\mathcal{R}^f(\bar{\mu})-1+\delta) + \lambda^c(\mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu})-\rho)\right].\end{aligned} \tag{32}$$

There are two possible cases:
(i) $\min\{\mathcal{R}^f(\bar{\mu})-1+\delta, \mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu})-\rho\} \geq 0$ and
(ii) $\min\{\mathcal{R}^f(\bar{\mu})-1+\delta, \mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu})-\rho\} < 0$.

If case (i) is true, then (14) and (15) are trivially satisfied. Further, in this case,

$$\inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} \left[\lambda^f(\mathcal{R}^f(\bar{\mu}) - 1 + \delta) + \lambda^c(\mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu}) - \rho)\right] = 0.$$

Therefore, $\mathcal{R}^* \leq \mathcal{R}_o^{\mathscr{M}^\times}(\bar{\mu}) + \epsilon$ holds, hence (13) is satisfied.

If case (ii) is true, we have

$$\begin{aligned}&\inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} \left[\lambda^f(\mathcal{R}^f(\bar{\mu}) - 1 + \delta) + \lambda^c(\mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu}) - \rho)\right] \\ &= B\min\{\mathcal{R}^f(\bar{\mu}) - 1 + \delta, \mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu}) - \rho\} < 0 \end{aligned} \tag{33}$$

The results in (32) and (33) together imply that $\mathcal{R}^* \leq \mathcal{R}_o^{\mathscr{M}^\times}(\bar{\mu}) + \epsilon$ and hence, (13) is satisfied. Further, by substituting (33) in (32) we get

$$\begin{aligned}B\min\{\mathcal{R}^f(\bar{\mu}) - 1 + \delta, \mathcal{R}_c^{\mathscr{M}^\times}(\bar{\mu}) - \rho\} &\geq \mathcal{R}^* - \mathcal{R}_o^{\mathscr{M}^\times}(\bar{\mu}) - \epsilon \\ &\geq \mathcal{R}^* - \mathbf{R}_o^{max} - \epsilon,\end{aligned}$$

where the last inequality holds because $\mathbf{R}_o^{max}$ is the maximum possible reward. Hence, (14) and (15) are satisfied.

Appendix III
Proof of Theorem 2

Consider the dual of (P4) and define

$$u_B^* := \inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} \sup_{\mu} L(\mu, \boldsymbol{\lambda}). \tag{P7}$$

Then, we obtain $l_B^* \overset{a}{\leq} u_B^*$ and

$$\begin{aligned}u_B^* &= \inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} \sup_{\mu} L(\mu, \boldsymbol{\lambda}) \\ &\leq \sup_{\mu} L(\mu, \bar{\boldsymbol{\lambda}}) \\ &= L(\mu_{\bar{\boldsymbol{\lambda}}}, \bar{\boldsymbol{\lambda}}) \\ &\overset{b}{=} \frac{1}{K}\sum_{k=1}^K L(\mu_{\bar{\boldsymbol{\lambda}}}, \boldsymbol{\lambda}_k) \\ &\overset{c}{\leq} \frac{1}{K}\sum_{k=1}^K L(\mu_k, \boldsymbol{\lambda}_k) \\ &\overset{d}{\leq} \frac{1}{K}\inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} \sum_{k=1}^K L(\mu_k, \boldsymbol{\lambda}) + 2BG\sqrt{2\log 3/K} \\ &\overset{e}{=} \inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} L(\bar{\mu}, \boldsymbol{\lambda}) + 2BG\sqrt{2\log 3/K}. \end{aligned} \tag{34}$$

The inequality in (a) holds because of weak duality [32]. The equality in (b) holds because $L(\cdot,\cdot)$ is a bilinear (more precisely, bi-affine) function. The inequality in (c) holds because μ_k is the maximizer associated with $\boldsymbol{\lambda}_k$. Inequality (d) follows from the no-regret property of the EG algorithm [10, Corollary 5.7]. The equality in (e) is again a consequence of the bilinearity of $L(\cdot,\cdot)$. Combining (34) with Lemma 1 proves the theorem.

Appendix IV
Proof of Theorem 3

Define $\hat{L}(\mu, \boldsymbol{\lambda}) = \mathcal{R}^{\mathscr{M}^\times}(\mu) + \lambda^f(\textsc{eval}(\mu, r^f) - 1 + \delta) + \lambda^c(\textsc{eval}(\mu, r^{c\times}) - \rho)$. Following initial arguments similar to those in the proof of Theorem 2, we obtain:

$$\begin{aligned}l_B^* &\leq \frac{1}{K}\sum_{k=1}^K L(\mu_{\bar{\boldsymbol{\lambda}}}, \boldsymbol{\lambda}_k) \\ &\overset{a}{\leq} \frac{1}{K}\sum_{k=1}^K L(\mu_{\boldsymbol{\lambda}_k}, \boldsymbol{\lambda}_k) \\ &\overset{b}{\leq} \frac{1}{K}\sum_{k=1}^K L(\mu_k, \boldsymbol{\lambda}_k) + \epsilon_{bp} \\ &\overset{c}{\leq} \frac{1}{K}\sum_{k=1}^K \hat{L}(\mu_k, \boldsymbol{\lambda}_k) + \epsilon_{bp} + \epsilon_{est} \\ &\overset{d}{\leq} \frac{1}{K}\inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} \sum_{k=1}^K \hat{L}(\mu_k, \boldsymbol{\lambda}) + 2BG\sqrt{\frac{2\log 3}{K}} + \epsilon_{bp} + \epsilon_{est} \\ &\overset{e}{=} \inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} \hat{L}(\bar{\mu}, \boldsymbol{\lambda}) + 2BG\sqrt{2\log 3/K} + \epsilon_{bp} + \epsilon_{est} \\ &\overset{f}{\leq} \inf_{\boldsymbol{\lambda}\geq 0, \|\boldsymbol{\lambda}\|_1 \leq B} L(\bar{\mu}, \boldsymbol{\lambda}) + 2BG\sqrt{2\log 3/K} + \epsilon_{bp} + 2\epsilon_{est}. \end{aligned} \tag{35}$$

The inequality in (a) holds because $\mu_{\boldsymbol{\lambda}_k}$ is the maximizer associated with $\boldsymbol{\lambda}_k$. The inequality in (b) holds by the bounded sub-optimality of μ_k. Inequality (c) follows from the bounded error of the estimator. Inequality (d) follows from the no-regret property of the EG algorithm. The equality in (e) is again a consequence of the bilinearity of $\hat{L}(\cdot)$. Finally, inequality (f) follows from the bounded error of the estimator. Combining (35) with Lemma 1 proves the theorem.

Appendix V
Proof of Lemma 2

The rewards $\mathcal{R}_o^{\mathscr{M}^\times}(\mu)$ and $\mathcal{R}^f(\mu)$ in the corresponding product POMDP are given by

$$\begin{aligned}\mathcal{R}_o^{\mathscr{M}^\times}(\mu) &= \mathbb{E}_\mu\left[\sum_{t=0}^T r_t^{o\times}(X_t, \mathcal{U}_t)\right] \\ &= (1-\gamma)\mathbb{E}_\mu\left[\sum_{k=0}^\infty \gamma^k \sum_{t=0}^k r_t^{o\times}(X_t, \mathcal{U}_t)\right] \\ &= (1-\gamma)\mathbb{E}_\mu\left[\sum_{t=0}^\infty \sum_{k=0}^\infty \gamma^{k+t} r_t^{o\times}(X_t, \mathcal{U}_t)\right] \\ &= \mathbb{E}_\mu\left[\sum_{t=0}^\infty \gamma^t r_t^{o\times}(X_t, \mathcal{U}_t)\right], \\ \mathcal{R}^f(\mu) &= \mathbb{E}_\mu\left[r^f(X_{T+1})\right] \\ &= (1-\gamma)\mathbb{E}_\mu\left[\sum_{t=0}^\infty \gamma^t r^f(X_{t+1})\right] \\ &= \frac{(1-\gamma)}{\gamma}\mathbb{E}_\mu\left[\sum_{t=1}^\infty \gamma^t r^f(X_t)\right]. \end{aligned}$$

TABLE III: Performance values and hyper-parameters.

Model	Spec	$\|S\|$	$\|Q\|$	$\mathcal{R}^{\mathscr{M}}(\bar{\mu})$	$\mathcal{R}^{f}(\bar{\mu})$	$1-\delta$	B	η	K	$simu$	T_{solve}	T_{simu}	T_{total}
$\mathscr{M}_1$	φ_1	64	3	0.95	0.70	0.70	8	2	50	100	17299	7825	25125
$\mathscr{M}_2$	φ_2	16	4	1.01	0.79	0.80	10	2	100	200	109	718	828
$\mathscr{M}_3$	φ_1	32	3	2.73	0.81	0.85	20	0.02	100	200	370	21676	22046
$\mathscr{M}_4$	φ_4	16	3	-14	0.87	0.9	100	2	20	100	79	14	93
$\mathscr{M}_5$	φ_5	116	5	1.59	0.72	0.7	50	2	50	50	11192	41705	53108

Similarly, we have $\mathcal{R}_c^{\mathscr{M}^\times}(\mu) = \mathbb{E}_\mu\left[\sum_{t=0}^{\infty}\gamma^t r_t^{c\times}(X_t,\mathcal{U}_t)\right]$. Therefore,

$$L(\mu,\boldsymbol{\lambda}) = -\frac{\lambda^f(1-\gamma)}{\gamma}\mathbb{E}[r^f(X_0)] - \lambda^f(1-\delta) - \lambda^c\rho + \mathbb{E}_\mu\left[\sum_{t=0}^{\infty}\gamma^t\left(r_t^{o\times}(X_t,\mathcal{U}_t) + \frac{\lambda^f(1-\gamma)}{\gamma}r^f(X_t) + \lambda^c r_t^{c\times}(X_t,\mathcal{U}_t)\right)\right].$$

APPENDIX VI
HYPER-PARAMETERS AND RUNTIMES

The parameter δ in all the experiments is chosen in the following manner: (i) We first solve a POMDP problem aiming to maximize the probability of satisfaction of the LTL_f constraint. Let this probability be p_{max}. (ii) Since any threshold $1-\delta$ larger than p_{max} is infeasible, we choose δ such that $1-\delta$ is around $0.9p_{max}$. η and B are hyperparameters in our experiments. The value of η suggested by Theorem 2 is guaranteed to result in convergence, but in practice, slightly larger values of η can lead to faster convergence.

In Table III, we provide additional hyper-parameters that were used in our experiments. The parameter $simu$ denotes the number of Monte-Carlo simulations that were used to estimate the constraint in each iteration. T_{solve} is the total time (over K iterations) spent in solving the unconstrained POMDP using the SARSOP solver [8]. T_{simu} is the total time spent in simulating policies generated by the SARSOP solver. T_{total} is the overall computation time for that model. The runtime (see Table III) for our models is drastically different due to differences in three factors: (i) the state size, (ii) the DFA size, and (iii) the complexity of the POMDP problem.

Krishna C. Kalagarla received B.S. degrees in Electrical Engineering and Computer Science from the Indian Institute of Technology, Kanpur, India, in 2016 and the Ph.D. degree in Electrical and Computer Engineering from the University of Southern California (USC), Los Angeles, in 2023. He is currently a Postdoctoral Researcher at the Electrical and Computer Engineering Department, University of New Mexico, Albuquerque. His research interests include formal methods, stochastic control, online learning, reinforcement learning, and explainable AI. He is a recipient of the USC Annenberg fellowship.

Dhruva Kartik received the B.Tech. degree in electronics and communication engineering from the Indian Institute of Technology, Guwahati, India, in 2015, and the Ph.D. degree in electrical engineering from the University of Southern California (USC), Los Angeles, in 2021. He was a Postdoctoral Researcher at USC in 2022. He is currently an Applied Scientist at Amazon. His research interests are decentralized stochastic control, decision-making in sensing and communication systems, game theory, and reinforcement learning.

Dongming Shen received B.S. degrees in Computer Science and Applied and Computational Mathematics from the University of Southern California, Los Angeles, in 2023. His research interests include decision making and reinforcement learning.

Rahul Jain (Senior Member, IEEE), is a Professor of Electrical and Computer Engineering and Computer Science at the University of Southern California (USC), a Director of the USC Center for Autonomy and Artificial Intelligence, and a visiting research scientist at Google. He received his Ph.D. (EECS) from the University of California, Berkeley. He has received numerous awards including the NSF CAREER award, the ONR Young Investigator award, and an IBM Faculty award. His interests span reinforcement learning, stochastic control, statistical learning, and game theory.

Ashutosh Nayyar (Senior Member, IEEE) is an Associate Professor of Electrical and Computer Engineering at the University of Southrn California. He received the M.S. degree in electrical engineering and computer science, the M.S. degree in applied mathematics, and the Ph.D. degree in electrical engineering and computer science from the University of Michigan, Ann Arbor, MI, USA, in 2008, 2011, and 2011, respectively. His research interests are in decentralized stochastic control, decentralized decision-making in sensing and communication systems, reinforcement learning, game theory, mechanism design, and electric energy systems.

Pierluigi Nuzzo (Senior Member, IEEE), is the K.C. Dahlberg Early Career Chair and an Associate Professor of Electrical and Computer Engineering and Computer Science at the University of Southern California, Los Angeles, where he co-directs the Center for Autonomy and Artificial Intelligence (AI). He received the Ph.D. in Electrical Engineering and Computer Sciences from the University of California, Berkeley, and B.S. and M.S. degrees in electrical and computer engineering from the University of Pisa and the Sant'Anna School of Advanced Studies, Pisa, Italy. His interests revolve around methodologies and tools for high-assurance design of cyber-physical systems (CPSs) and systems-on-chip, including the application of formal methods and optimization theory to problems in CPSs, electronic design automation (EDA), autonomy, security, and AI. His awards include the NSF CAREER Award, the DARPA Young Faculty Award, the Early-Career Awards from the IEEE Council on EDA and the Technical Committee on CPSs, the Okawa Research Grant, the UC Berkeley EECS David J. Sakrison Memorial Prize, and several best paper and design competition awards.