How hard is it to solve constrained sc-LTL planning problems?

Zetong Xuan and Yu Wang

***Abstract*— We study the synthesis of optimal policies for planning problems on Markov decision processes with both objectives and safety constraints specified in co-safe linear temporal logic (sc-LTL). Our problems are inherently non-Markovian due to the complexity of the sc-LTL specification and may require policy randomization to balance the objective and constraint. We propose a novel approach that reduces the constrained sc-LTL planning problem to a constrained reachability problem on an extended model. We then show that a class of switching policies constructed from stationary policies for the individual sc-LTL specifications is sufficient for optimality for the constrained reachability problem. Our finding enables a tractable linear program to compute the optimal policy. A grid world case study demonstrates that our switching policies can achieve the optimal trade-off between the objective and the safety constraint and validates both optimality and tractability.**

I. Introduction

Modern autonomous systems often operate under dynamic and time-dependent safety constraints while accomplishing complex logic-based tasks [1]–[3]. A fundamental challenge lies in balancing task efficiency with formal safety guarantees, as these objectives may be contradictory [4]. For instance, drones must periodically return to their base for inspections and battery recharging to ensure operational safety and reliability. However, achieving high efficiency in complex tasks like surveillance may justify tolerating a certain level of risk, including the potential loss of drones. This motivates the need for planning that maximizes task success subject to formal logic constraints.

We study constrained planning problems in which both the objective and safety constraints are expressed as co-safe linear temporal logic (sc-LTL) [5]. sc-LTL specializes standard linear temporal logic (LTL) [1] to its co-safe fragments, whose specifications define properties over finite traces and are well suited for specifying mission goals and safety constraints. For example, an sc-LTL specification can require sequential reachability, such as "visit region A, then region B" or conditional objectives, such as "if region C is visited, then region D must be visited afterwards". In a variety of planning settings, sc-LTL has been employed as task objective, including in probabilistic systems such as MDPs and partially observable MDPs [6], [7], robotic planning [8], and hybrid dynamical systems [9].

Solving the constrained sc-LTL planning problem defined on Markov decision processes (MDPs) is challenging because the optimal solution can be a stochastic and non-Markovian policy. Stochastic policies are often required to balance conflicting objectives (e.g., task success vs. safety) [4], which contrasts with unconstrained sc-LTL planning where a purely deterministic policy can be optimal. The non-Markovian nature arises from two sources. First, since both the objective and the constraints are specified using sc-LTL specifications that involve complex sequential ordering, the optimal policy must depend on more than just the current state. Second, even when both the objective and the constraint reduce to simple reachability tasks, the optimal policy may still be non-Markovian. This is because satisfying one sc-LTL specification may depend on a finite prefix that overlaps or conflicts with the prefix required to satisfy the other.

Despite the complexity of the optimal policy, we discovered a simple underlying structure. The optimal policy for constrained sc-LTL planning can be viewed as a switching policy with three modes: (1) a mixed policy that balances the objective and safety constraints, (2) a pure optimistic policy that prioritizes the objective, and (3) a pure pessimistic policy that prioritizes safety. The mixed policy addresses the inherent trade-off between objective achievement and constraint satisfaction, while the switching mechanism exploits the fact that sc-LTL specifications can be satisfied by a finite prefix of a path. Once one specification is satisfied, the policy can switch to focusing exclusively on the other.

Building on this insight into the structure of the optimal policy, we develop a tractable solution pipeline for constrained sc-LTL planning. First, in Section IV, we reduce the problem to a constrained reachability problem on an extended model of the MDP. This reduction ensures that both the optimistic and pessimistic policies become stationary deterministic policies in the extended model, making them easy to compute. Second, in Section V, we prove that a switching policy, constructed by switching among the mixed, optimistic, and pessimistic policies (all stationary in the extended model), is sufficient for optimality. This result lies outside the scope of existing constrained MDP theory [4], whose standard assumptions, such as a discount factor or an absorbing MDP with a single target set, do not hold in our case. Finally, in Section VI, we develop an algorithm based on linear programming (LP) to compute the optimal switching policy in the extended model, and in Section VII we validate the approach on a grid world case study. The policy is then mapped back to the original MDP, yielding the desired non-Markovian optimal policy.

Related work

Our work addresses constrained sc-LTL planning on general MDPs, without assuming the feasibility of the constraint,

Zetong Xuan and Yu Wang are with Department of Mechanical & Aerospace Engineering, University of Florida, Gainesville, FL 32611, USA

any particular dependency between objective and constraint formulation, or structural assumptions on the dynamics. In comparison, most existing works on planning with temporal logic constraints either consider temporal logic only as a constraint (not as an objective), assume a fixed dependency between the objective and constraint specifications, or focus on specific dynamical systems.

Early studies focus on minimizing cost while strictly satisfying a temporal logic specification. Representative works include incremental or optimal control for MDPs and multi-agent systems [6], [10], [11], as well as approximate or sampling-based dynamic programming methods for scalability [12]–[14]. These approaches typically assume that the temporal logic specification is feasible and must be satisfied almost surely.

Other works consider scenarios where the temporal logic constraint can be relaxed. In this setting, the constraint is either relaxed or replaced with a cost-based surrogate. For example, [15] proposes a relaxed product MDP specification that balances cost, satisfaction probability, and violation penalty. Similarly, several works [13], [16], [17] incorporate cost-based constraints when temporal logic specifications cannot be fully enforced. These works do not consider multiple temporal logic specifications at the same time, and thus are less expressive than our formulation.

More general specifications consider planning problems where both the objective and the constraint are specified by temporal logic specifications. However, their satisfaction outcomes are not independent but often semantically coupled, and thus cannot represent cases where the objective and constraint are logically independent. [18] studied model-based reinforcement learning for mean-payoff optimization under ω-regular constraints, which contains LTL specifications. [19] proposed a model-free reinforcement learning framework that maximizes a weighted sum of satisfaction probabilities over multiple LTL objectives. [20] considered lexicographic preferences among temporal logic specifications, where safety is prioritized over task completion and reward collection.

Other related efforts explore signal temporal logic planning in continuous systems. However, these approaches typically assume deterministic dynamics and thus can not be generalized to general MDPs. Examples include safe control under signal temporal logic constraints [21] and recurrent-set based synthesis [22].

II. Preliminaries

This section introduces preliminaries on labeled Markov decision processes and co-safe linear temporal logic.

A. Labeled Markov decision processes

We use labeled Markov decision processes (LMDPs) to model planning problems where each decision has a potentially probabilistic outcome. LMDPs augment standard Markov decision processes [1] with state labels, enabling assigning properties, such as safety and liveness, to a sequence of states.

Definition 1: A labeled Markov decision process is a tuple $\mathcal{M} = (S, A, T, \beta, \Lambda, L)$ where

- S is a finite set of states,
- A is a finite set of actions where $A(s)$ denotes the set of allowed actions in state $s \in S$,
- $T : S \times A \times S \to [0, 1]$ is the transition probability function such that for all $s \in S$,

$$\sum_{s' \in S} T(s, a, s') = \begin{cases} 1, & a \in A(s) \\ 0, & a \notin A(s) \end{cases},$$

- $\beta \in \Delta(S)$ is the initial distribution, here we denote the set of probability distributions on S by $\Delta(S)$,
- Λ is a finite set of atomic propositions,
- $L : S \to 2^{\Lambda}$ is a labeling function.

The semantic structure Λ and the function L of the model M enable high-level symbolic descriptions of objectives and constraints that can reflect the context of the planning tasks. Specifically, given the control policies π, we can sample (subject to environmental stochasticity) the resulting individual path $\sigma \sim M_\pi$. A path of the LMDP $\mathcal{M}$ is an infinite state sequence $\sigma = s_0 s_1 s_2 \cdots$ such that for all $i \geq 0$, there exists $a_i \in A(s)$ and $s_i, s_{i+1} \in S$ with $P(s_i, a_i, s_{i+1}) > 0$. The semantic path corresponding to σ is given by $L(\sigma) = L(s_0)L(s_1)\cdots$, derived using the labeling function $L(s)$. Given a path σ, the ith state is denoted by $\sigma[i] = s_i$. We denote the prefix by $\sigma[{:}i] = s_0 s_1 \cdots s_i$ and suffix by $\sigma[i{+}1{:}] = s_{i+1} s_{i+2} \cdots$.

We consider multiple classes of policies in this work. A policy is a function that maps a prefix (or a fragment of it) and the current time step to a distribution over actions. When the input is restricted to the current state $\sigma[i]$ only, we say the policy is *stationary*. When the input is restricted to the current state $\sigma[i]$ and the current time step i, we say the policy is *Markovian*. Otherwise, we say the policy is *non-Markovian*. Moreover, if the output distribution is always concentrated on a single action, the policy is said to be deterministic. Otherwise, we say the policy is stochastic.

B. Co-safe linear temporal logic and deterministic finite automata

In an LMDP $\mathcal{M}$, whether a given semantic path $L(\sigma)$ satisfies a property such as avoiding unsafe states can be expressed using linear temporal logic. LTL can specify the change of labels along the path by connecting Boolean variables over the labels with two propositional operators, negation ($\neg$) and conjunction ($\wedge$), two temporal operators, next ($\bigcirc$) and until ($\cup$).

Definition 2: The LTL specification is defined by the syntax

$$\varphi ::= \text{true} \,|\, \alpha \,|\, \varphi_1 \wedge \varphi_2 \,|\, \neg\varphi \,|\, \bigcirc \varphi \,|\, \varphi_1 \cup \varphi_2, \alpha \in \Lambda. \quad (1)$$

Satisfaction of an LTL specification φ on a path σ of an MDP (denoted by $\sigma \models \varphi$) is defined as, $\alpha \in \Lambda$ is satisfied on σ if $\alpha \in L(\sigma[1])$, $\bigcirc\varphi$ is satisfied on σ if φ is satisfied on $\sigma[1{:}]$, $\varphi_1 \cup \varphi_2$ is satisfied on σ if there exists i such that $\sigma[i{:}] \models \varphi_2$ and for all $j < i$, $\sigma[j{:}] \models \varphi_1$.

Other propositional and temporal operators can be derived from previous operators, e.g., (or) $\varphi_1 \vee \varphi_2 \coloneqq \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, (eventually) $\Diamond\varphi \coloneqq \text{true} \cup \varphi$, and (always) $\Box\varphi \coloneqq \neg\Diamond\neg\varphi$.

In this work, we focus on the co-safe fragment of the LTL specification, which can be verified via finite prefixes. We say the LTL specification φ is co-safe [23] if all the infinite sequences $\sigma \models \varphi$ satisfy: there exists a truncated finite sequence $\sigma_{\leq k} = b_0 b_1 \cdots b_k$, $k \in \mathbb{N}$ such that $\sigma_{\leq k} \cdot \sigma' \models \varphi$ for any infinite sequence σ'.

We use deterministic finite automata (DFAs) as a graph representation of an sc-LTL objective, thus simplifying the complex rule-based objective into reachability. Given an sc-LTL objective φ, we can construct a DFA $\mathcal{A}_\varphi$ (with labels $\Sigma = 2^\Lambda$) such that a path $\sigma \models \varphi$ if and only if σ is accepted by the DFA $\mathcal{A}_\varphi$ which has a reachability objective.

Definition 3: A deterministic finite automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_{\text{init}}, Z)$ where

- Q is a finite set of automaton states,
- Σ is a finite set of alphabets,
- $\delta : Q \times \Sigma \to 2^Q$ is a (partial) transition function,
- $q_{\text{init}} \in Q$ is the initial automaton state,
- $Z \subseteq Q$ is a set of final automaton states.

The DFA verifies the path σ by its semantic path $L(\sigma)$. Each label on the path moves the automation states through transition δ. A path σ is accepted by $\mathcal{A}_\varphi$ if and only if there exists $k \in \mathbb{N}$ such that the prefix $\sigma_{\leq k}$ moves the initial automaton state q_{init} to a final state $q \in Z$.

We use $P_\pi(\sigma \models \phi)$ to denote the probability of an MDP path σ under policy π satisfies ϕ. For a state s, we introduce the shorthand $P_\pi(s \models \phi) := P_\pi(\sigma \models \phi |\, \sigma[0] = s)$.

III. Problem Formulation

We consider planning problems where both the objective and the safety constraint are specified using co-safe LTL specifications over system paths. Formally, the problem is defined as

$$\textbf{Objective:} \quad \max_{\pi \in \Pi} P_\pi(\sigma \models \phi) \quad \text{for } \sigma \sim M_\pi \tag{2}$$

$$\textbf{Constraint:} \quad P_\pi(\sigma \models \psi) \geq \tau, \tag{3}$$

where Π is the set of all admissible polices.

This problem is challenging because the optimal policy may be both non-Markovian and stochastic. Existing methods, such as dynamic programming and reinforcement learning, which search over the space of Markovian policies, are fundamentally incapable of synthesizing such policies.

We next present a minimal example that demonstrates how, even in simple settings, the optimal policy may require history dependence and randomization, highlighting the need to consider non-Markovian and stochastic policies in the general case.

Example 1: Consider an MDP with five states and initial distribution $\beta(s_{\text{init},1}) = \beta(s_{\text{init},2}) = 0.5$. The transition dynamics are shown in Fig 1. From $s_{\text{init},1}$, action α leads deterministically to s_ϕ (which self-loops), and action β reaches $s_{\psi,1}$ with probability 0.1 and returns back to $s_{\text{init},1}$ with probability 0.9. From $s_{\psi,1}$, the agent returns deterministically to $s_{\text{init},1}$. From $s_{\text{init},2}$, action α leads to s_ϕ, and action β leads to $s_{\psi,2}$ (which self-loops).

We consider a constrained sc-LTL planning problem where both the objective ϕ and constraint ψ are reachability properties: $\phi = \Diamond Z_\phi$ and $\psi = \Diamond Z_\psi$ with $Z_\phi = \{s_\phi\}$ and $Z_\psi = \{s_{\phi,1}, s_{\phi,2}\}$. The constraint threshold is $\tau = 0.9$. The resulting problem takes the form:

$$\begin{aligned} &\max_\pi \; P_\pi(\sigma \models \Diamond Z_\phi) \\ &\text{s.t.} \quad P_\pi(\sigma \models \Diamond Z_\psi) \geq \tau. \end{aligned} \tag{4}$$

We construct a stochastic, non-Markovian policy π_h as follows. 1) At $s_{\text{init},1}$, the policy selects β until the path visits $s_{\psi,1}$, and choose α thereafter. 2) At $s_{\text{init},2}$, take action α with probability 0.2 and β with probability 0.8.

Such a policy can satisfy the constraint while achieving the highest possible objective function

$$\begin{aligned} P_{\pi_h}(\sigma \models \Diamond Z_\phi) &= 0.5 P_{\pi_h}(\sigma \models \Diamond Z_\phi | \sigma[0] = s_{\text{init},1}) \\ &\quad + 0.5 P_{\pi_h}(\sigma \models \Diamond Z_\phi | \sigma[0] = s_{\text{init},2}) \\ &= 0.5 \times 1 + 0.5 \times 0.2 = 0.6, \\ P_{\pi_h}(\sigma \models \Diamond Z_\psi) &= 0.5 P_{\pi_h}(\sigma \models \Diamond Z_\psi | \sigma[0] = s_{\text{init},1}) \\ &\quad + 0.5 P_{\pi_h}(\sigma \models \Diamond Z_\psi | \sigma[0] = s_{\text{init},2}) \\ &= 0.5 \times 1 + 0.5 \times 0.8 = 0.9. \end{aligned} \tag{5}$$

In contrast, any Markovian policy from $s_{\text{init},1}$ must commit to β a fixed number of times, thus cannot ensure reachability to both Z_ϕ and Z_ψ. Similarly, any deterministic policy at $s_{\text{init},2}$ must choose either α or β exclusively. To satisfy the constraint, β must be selected, which contributes nothing to the objective.

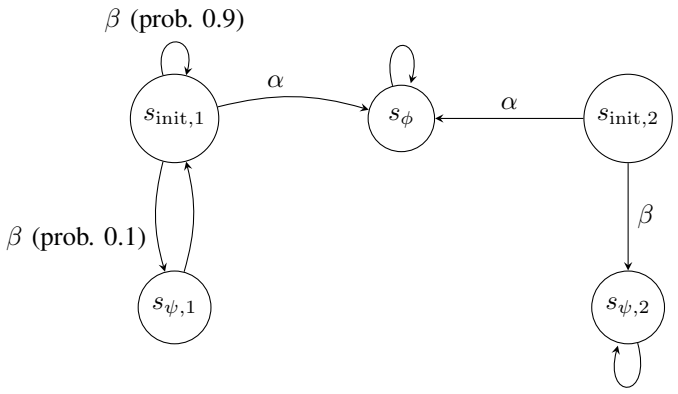

Fig. 1. An MDP with five states and two initial states: $s_{\text{init},1}$ and $s_{\text{init},2}$, each selected with probability 0.5. From $s_{\text{init},1}$, action α leads deterministically to s_ϕ (which self-loops), while action β reaches $s_{\psi,1}$ with probability 0.1 and loops back to $s_{\text{init},1}$ with probability 0.9. From $s_{\psi,1}$, the process returns deterministically to $s_{\text{init},1}$. A separate branch begins at $s_{\text{init},2}$, with α leading to s_ϕ and β leading to self-looping $s_{\psi,2}$. We consider the sc-LTL objective $\max_\pi P_\pi(\sigma \models \Diamond\{s_\phi\})$, subject to the constraint $P_\pi(\sigma \models \Diamond\{s_{\psi_1}, s_{\psi_2}\}) \geq 0.9$. As discussed in Equation (5), any Markovian or deterministic policy performs worse than the non-Markovian, stochastic policy we construct for this example.

IV. Eliminating Non-Markovianity via Double Product MDP Construction

This section enables the application of solving techniques designed for Markovian policies to the constrained sc-LTL problem by eliminating the planning problem's non-Markovianity. Specifically, we show that by transforming the original problem into an equivalent constrained reachability problem on an extended model of the LMDP, a stationary policy becomes sufficient for optimality. This construction enables us to leverage existing methods, such as linear programming, which are applicable to stationary policies, to solve the transformed problem and subsequently recover the optimal policy for the original sc-LTL planning.

The main challenge lies in the fact that, in the original LMDP, even when both the objective and the constraint are expressed as reachability, the optimal policy is not necessarily Markovian, as illustrated earlier in Fig. 1. However, we show that a similar constrained reachability formulation admits a stationary policy to be the optimal policy on a specially constructed product MDP.

A. Double product MDP construction

We adopt the construction of an extended model for the unconstrained LTL planning problem [1] to our constrained problem. Given the constrained sc-LTL problem with specifications ϕ and ψ, we construct a double product MDP that encodes the satisfaction status of both specifications. This construction transforms both ϕ and ψ into two reachability objectives $\Diamond Z_\phi$ and $\Diamond Z_\psi$. The double product MDP is defined as follows:

Definition 4: A double product MDP $\mathcal{M}^\times = (S^\times, A^\times, T^\times, \beta^\times, Z_\phi^\times, Z_\psi^\times)$ of an LMDP $\mathcal{M} = (S, A, T, \beta, \Lambda, L)$ and a DFA $\mathcal{A}_\phi = (Q_\phi, \Sigma_\phi, \delta_\phi, q_{\text{init},\phi}, Z_\phi)$ and a DFA $\mathcal{A}_\psi = (Q_\psi, \Sigma_\psi, \delta_\psi, q_{\text{init},\psi}, Z_\psi)$ is defined by

- the set of states $S^\times := S \times Q_\phi \times Q_\psi$,
- the set of actions $A^\times := A$,
- one target set $Z_\phi^\times := \{\langle s, q_\phi, q_\psi\rangle \in S^\times | q_\phi \in Z_\phi\}$,
- another target set $Z_\psi^\times := \{\langle s, q_\phi, q_\psi\rangle \in S^\times | q_\psi \in Z_\psi\}$,
- the initial distribution $\beta^\times(\langle b_0, q_{\text{init},\phi}, q_{\text{init},\psi}\rangle) := \beta(s)$,
- the transition probability function

$$T^\times(s^\times, a, s^{\times\prime}) = \begin{cases} T(s,a,s') & s^\times = \langle s, q_\phi, q_\psi\rangle, \\ & s^{\times\prime} = \langle s', \delta_\phi(q_\phi, L(s)), \delta_\psi(q_\psi, L(s))\rangle, \\ 0 & \text{else.} \end{cases}$$

This construction captures the transitions of the LMDP, and two parallel DFAs are actuated by the labeling on the LMDP. As a result, every LMDP path σ corresponds uniquely to a double product MDP path $\sigma^\times$.

B. Sufficiency of stationary policy

The construction of the double product MDP enables us to formulate the sc-LTL control problem into the following constrained reachability problem,

$$\begin{aligned} &\max_{\pi_h} \; P_{\pi_h}(\sigma \models \Diamond Z_\phi^\times) \\ &\textbf{s.t.} \quad P_{\pi_h}(\sigma \models \Diamond Z_\psi^\times) \geq \tau, \end{aligned} \tag{6}$$

Although it is counterintuitive, given that reachability in the original MDP may require non-Markovian behavior, we show that a stationary policy is sufficient to achieve optimality in the double product MDP:

Theorem 1: The optimal policy among all stationary policies is the optimal policy for the constrained reachability problem on the double product MDP. (The proof will be shown in the Section V-D.)

This result is enabled by the structure of the double product MDP. Each product state $s^\times = \langle s, q_\phi, q_\psi\rangle$ encodes not only the current LMDP state s, but also the satisfaction status of both sc-LTL specifications via the DFA states q_ϕ and q_ψ. The following example shows how the DFA states modify Example 1.

Example 2: As shown in Fig. 2, after visiting a state in Z_ψ, the path may return to $s_{\text{init},1}$. But now with an updated DFA state q'_ψ, the path will return to a new product state $\langle s_{\text{init},1}, q_\phi, q'_\psi\rangle$ that is distinguishable from the original one. In this way, the necessary history information is implicitly encoded in the state space, and a stationary policy on the double product MDP can replicate the behavior of a non-Markovian policy in the original model.

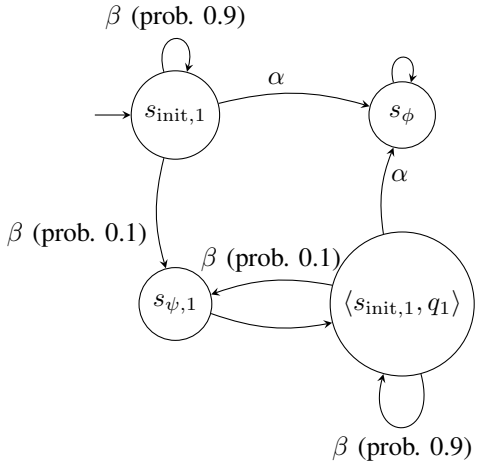

Fig. 2. The augmented logic state q_1 encodes the history information that $s_{\psi,1}$ has been visited.

C. Recovering the optimal policy on the original LMDP

Furthermore, we show that solving the constrained reachability problem on the double product MDP yields a solution to the original constrained sc-LTL planning problem.

Lemma 1: Optimal non-Markovian policy for the constrained sc-LTL problem on the MDP can be recovered from the optimal policy for the constrained reachability problem on the double product MDP.

Proof: Given a policy π, the reachability probability to either one of the target sets $Z_\phi^\times$, $Z_\psi^\times$ is equivalent to the satisfaction probability of either one of the sc-LTL objectives ϕ,ψ,

$$\begin{aligned} P_\pi(\sigma \models \phi) &= P_\pi(\sigma^\times \models \Diamond Z_\phi^\times) \\ P_\pi(\sigma \models \psi) &= P_\pi(\sigma^\times \models \Diamond Z_\psi^\times). \end{aligned} \tag{7}$$

■

With this theoretical foundation, we can apply LP-based techniques or solve the Lagrangian Bellman equation in the double product MDP, and map the resulting stationary policy back to a non-Markovian policy in the original MDP.

For simplicity, we henceforth drop the superscript $\times$ when referring to the double product MDP.

V. Optimality of stationary policies

In this section, we establish that a stationary policy suffices to be the solution to the constrained reachability problem on the double product MDP. At the end of this section, we formally prove Theorem 1.

Although Example 1 shows that stationary policies are not sufficient to achieve optimality for the constrained reachability problem on a general MDP, we identify a class of non-Markovian policies that can. This class, denoted $\pi_{s\to s}$, switches between different stationary sub-policies depending on which target set (if any) has been reached, thereby resolving the temporal dependency between the two reachability objectives. The switching condition is naturally encoded in the DFA states of the double product MDP, rendering $\pi_{s\to s}$ stationary in that space.

Remark 1: Classical constrained MDP theory does not cover our case. In the constrained MDP literature (e.g., [4]), similar results typically rely on the fact that the occupancy measure of any arbitrary policy can be matched by that of a stationary policy. Since both the objective and constraint are linear in the occupancy measure, this matching implies optimality of stationary policies. However, this matching in occupancy measure typically requires either (i) a discount factor or (ii) structural assumptions on the MDP, such as being an absorbing MDP and having a single target set. Neither assumption holds in our case: we consider undiscounted reachability with two interacting target sets, where behavior after reaching one set still influences the other.

A. A switching policy $\pi_{s\to s}$ for the general MDP

As illustrated in Fig. 1, the optimal policy to the constrained reachability problem may still be non-Markovian due to inherent temporal dependency between the two reachability objectives.

We propose a class of policies that switch between different stationary policies based on which target has been visited:

- π_{mix} is a stationary policy applied before any states in Z_ϕ or Z_ψ are visited. We also refer to it as the pre-switching policy.
- π_ψ is a stationary policy applied once a states in Z_ϕ is visited, the policy encourages to hit Z_ψ,
- π_ϕ is a stationary policy applied once a states in Z_ψ is visited, the policy encourages to hit Z_ϕ,

That is, a path on the double product MDP, if the initial state is not in Z_ϕ or Z_ψ, the path will begin with π_{mix} and switches to either π_ϕ or π_ψ depending on which target is visited first as shown in Fig. 3. We denote this switching policy by $\pi_{s\to s}$. It is not necessary to specify the policy after both target sets have been visited, since at that point both ϕ and ψ can be seen as satisfied.

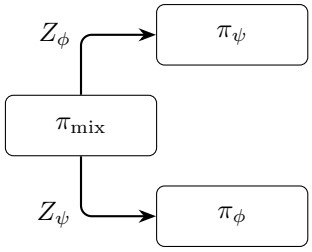

Fig. 3. Illustration of the switching policy $\pi_{s\to s}$. The pre-switching policy π_{mix} is followed until a target set is reached. Hitting Z_ϕ triggers a switch to π_ψ, while hitting Z_ψ triggers a switch to π_ϕ.

Importantly, this switching policy is realizable as a stationary policy on the double product MDP. Because the DFA state tracks which (if any) target set has been visited. Specifically, for all states $\langle s, q_\phi, q_\psi\rangle$, The switching policy $\pi_{s\to s}$ is defined as:

$$\pi_{s\to s}(\langle s, q_\phi, q_\psi\rangle) = \begin{cases} \pi_{\text{mix}}(\langle s, q_\phi, q_\psi\rangle), & \text{if } q_\phi \notin Z_\phi \text{ and } q_\psi \notin Z_\psi \\ \pi_\psi(\langle s, q_\phi, q_\psi\rangle), & \text{if } q_\phi \in Z_\phi \text{ and } q_\psi \notin Z_\psi \\ \pi_\phi(\langle s, q_\phi, q_\psi\rangle), & \text{if } q_\phi \notin Z_\phi \text{ and } q_\psi \in Z_\psi \end{cases}. \quad (8)$$

To formally express the performance of this switching policy, we introduce the concept of the dominating class of policies.

Definition 5: A class of policies $\overline{U}$ is said to be a dominating class of policies for the constrained planning problem for a given initial distribution β, if for any policy π there exists a policy $\overline{\pi} \in \overline{U}$ such that

$$P_{\overline{\pi}}(\sigma \models \phi) \geq P_\pi(\sigma \models \phi)$$
$$P_{\overline{\pi}}(\sigma \models \psi) \geq P_\pi(\sigma \models \psi). \quad (9)$$

Theorem 2: The class of our proposed switching policy $\pi_{s\to s}$ is a dominating class of policies for the constrained reachability problem on the LMDP.

(The proof for this theorem will be shown at the end of Section V-C.)

To prove Theorem 2, we proceed in two steps. Given an arbitrary non-Markovian policy π_h, we first construct a switching policy $\pi_{h\to s}$ that is guaranteed to perform at least as well as π_h on both the objective and the constraint. The switching policy $\pi_{h\to s}$ is defined as follows:

- Use π_h before visiting any state in Z_ϕ or Z_ψ.
- Switch to $\pi_\psi := \arg_\pi \max P_\pi(\sigma \models \Diamond Z_\phi)$ after entering Z_ϕ.
- Switch to $\pi_\phi := \arg_\pi \max P_\pi(\sigma \models \Diamond Z_\psi)$ after entering Z_ψ.

We then show that replacing π_h (the pre-switching component) with a stationary policy π_{mix} does not degrade performance on either the objective or the constraint.

B. Monotonic policy improvement via switching

We first show that switching to an optimal stationary reachability policy after visiting a target cannot degrade performance, but only increases the performance.

Lemma 2: The class of our proposed switching policy $\pi_{h\to s}$ is a dominating class of policies for the constrained reachability problem.

The proof proceeds by decoupling the interaction between the two reachability specifications. We show that the objective function can be decomposed into two components: (1) the occupancy measure over paths that have not yet reached either Z_ϕ or Z_ψ, and (2) the conditional reachability probability starting from states within Z_ψ. By replacing the second term with the maximum unconstrained reachability probability to Z_ϕ, we strictly increase (or preserve) the objective function, without affecting the constraint function. An analogous argument applies when analyzing the constraint function with respect to Z_ψ. The key quantity in this decomposition is the *occupancy measure*, which describes how often each state-action pair is visited before hitting either target set.

Definition 6: Let π be a policy and $Z \subseteq S$ be a target set.

For initial state $i \in S$, we define the following: (1) The probability that a path σ starting from i stays within $S \setminus Z$ for t steps and visits state $j \in S \setminus Z$ at step t is given by

$$\xi_\pi(i,j,t,Z) = \\ P_\pi\left(\sigma[t]=j,\ \sigma[n]\in S\backslash Z,\ 0\le n\le t-1 \mid \sigma[0]=i\right).$$

(2) The *occupancy measure of state-action pair* (j,a) before reaching either target set is defined as the expected number of times (j,a) is visited:

$$\mu_\pi(j,a) := \\ \pi(j,a)\sum_{i\in S}\sum_{t=0}^{\infty}\beta(i)\,\xi_\pi(i,j,t,Z_\phi\cup Z_\psi). \tag{10}$$

(3) The *state occupancy measure* before reaching either target sets is then:

$$\mu_\pi(j) := \sum_{a\in A(j)}\mu_\pi(j,a). \tag{11}$$

Proof of Lemma 2: Given an arbitrary policy π_h, the reachability probability to the target set $Z_\phi \subseteq F$ can be expressed as the weighted sum of all occupancy measures $\mu_{\pi_h}(i,a)$.

$$\begin{aligned}
&P_{\pi_h}(\sigma \models \Diamond Z_\phi)\\
&= \sum_{i\in Z_\phi}\beta(i) + \sum_{i\in Z_\psi\setminus Z_\phi}\beta(i)P_{\pi_h}(i\models\Diamond Z_\phi)\\
&+ \sum_{i\in S_?}\beta(i)P_{\pi_h}(i\models\Diamond Z_\phi)\\
&= \sum_{i\in Z_\phi}\beta(i) + \sum_{i\in Z_\psi\setminus Z_\phi}\beta(i)P_{\pi_h}(i\models\Diamond Z_\phi)\\
&+ \sum_{i\in S_?}\sum_{a\in A(i)}\sum_{j\in Z_\phi}\mu_{\pi_h}(i,a)T(i,a,j)\\
&+ \sum_{i\in S_?}\sum_{a\in A(i)}\sum_{j\in Z_\psi\setminus Z_\phi}\mu_{\pi_h}(i,a)T(i,a,j)P_{\pi_h}(h,j\models\Diamond Z_\phi),
\end{aligned} \tag{12}$$

where $S_? := S\backslash Z_\psi\backslash Z_\phi$ and $P_{\pi_h}(h,j \models \Diamond Z_\phi)$ denotes the probability of eventually reaching Z_ϕ when the run has already produced history consistent with h and is currently at state j, continuing under π_h. If π_h is Markovian, this reduces to $P_{\pi_h}(j \models \Diamond Z_\phi)$. The first equality holds as we partition the unconstrained reachability to Z_ϕ based on the initial distribution β. The second equality holds as we further decompose $\sum_{i\in S_?}\beta(i)P_{\pi_h}(i\models\Diamond Z_\phi)$ based on whether the path directly visits Z_ϕ or visits Z_ψ before visiting Z_ϕ.

For any policy π_h, there exists a switching policy $\pi_{h\to s}$, such that

$$P_{\pi_{h\to s}}(\sigma\models\Diamond Z_\phi) \ge P_{\pi_h}(\sigma\models\Diamond Z_\phi). \tag{13}$$

This inequality holds by the construction of $\mu_{\pi_{h\to s}}(s,a)$,

$$\begin{aligned}
&P_{\pi_{h\to s}}(\sigma\models\Diamond Z_\phi)\\
&= \sum_{i\in Z_\phi}\beta(i) + \sum_{i\in Z_\psi\setminus Z_\phi}\beta(i)P_{\max}(i\models\Diamond Z_\phi)\\
&+ \sum_{i\in S_?}\sum_{a\in A(i)}\sum_{j\in Z_\phi}\mu_{\pi_h}(i,a)T(i,a,j)\\
&+ \sum_{i\in S_?}\sum_{a\in A(i)}\sum_{j\in Z_\psi\setminus Z_\phi}\mu_{\pi_h}(i,a)T(i,a,j)P_{\max}(j\models\Diamond Z_\phi)\\
&\ge P_{\pi_h}(\sigma\models\Diamond Z_\phi),
\end{aligned}$$

where $P_{\max}(j \models \Diamond Z_\phi)$ denotes the maximum reachability probability from state i to target set Z_ϕ. As a known fact, the maximum reachability probability can be achieved by a stationary policy.

We can directly apply the same procedure to $P_{\pi_h}(\sigma \models \Diamond Z_\psi)$ yield

$$P_{\pi_{h\to s}}(\sigma\models\Diamond Z_\psi) \ge P_{\pi_h}(\sigma\models\Diamond Z_\psi). \tag{14}$$

The above two inequalities for objective and constraint hold at the same time for one specific $\pi_{h\to s}$. The reason is π_ψ does not affect $P_{\pi_{h\to s}}(\sigma \models \Diamond Z_\phi)$ and π_ϕ does not affect $P_{\pi_{h\to s}}(\sigma \models \Diamond Z_\psi)$.

By definition, we say the class of our proposed switching policy $\pi_{h\to s}$ is a dominating class of policies for the constrained reachability problem. ■

C. Stationary policy realization without performance loss

We have shown replacing π_h after hitting one of the target sets, such that switching only increases the performance of the policy.

We now show that the initial non-Markovian component of π_h in the switching policy $\pi_{h\to s}$ can be replaced by a stationary policy π_{mix} without degrading performance.

The construction of such π_{mix} is based on the MDP theory. Given policy π_h, the set of *transient* states contains all states that only have a finite expected number of visits,

$$\overline{X} := \{x | \mu_{\pi_h}(x) < \infty\}. \tag{15}$$

The set of *recurrent* states contains all states that have an infinite expected number of visits,

$$\underline{X} := \{x | \mu_{\pi_h}(x) = \infty\}. \tag{16}$$

Now we construct π_{mix}, which induces a Markov chain, such that the partition of recurrent and transient states is the same as the partition done by policy π_h. For all transient state (under π_h) $x \in \overline{X}$, Let π_{mix} be a stationary policy satisfying

$$\pi_{\text{mix}}(x,a) = \frac{\mu_{\pi_h}(x,a)}{\mu_{\pi_h}(x)}, \tag{17}$$

where denominator $\mu_{\pi_h}(x)$ is zero, $\pi_s(x,a)$ is chosen arbitrarily. For all recurrent state (under π_h) $x \in \underline{X}$, let π_{mix} be a stationary policy such that $\pi_{\text{mix}}(x,a) > 0$ for all state-action pair $\{(x,a) | \mu_{\pi_h}(x,a) > 0\}$.

Lemma 3: Under the construction of policy π_{mix},

1) if $\mu_{\pi_h}(x) < \infty$ then $\mu_{\pi_{\text{mix}}}(x,a) = \mu_{\pi_h}(x,a)$,
2) if $\mu_{\pi_h}(x) = \infty$ then $\mu_{\pi_{\text{mix}}}(x) = \infty$.

(The proof for this lemma will be shown in the Appendix.) Therefore, the occupancy measure before hitting target sets under policy π_h can be replicated by the stationary policy π_{mix}.

Before switching, using the stationary π_{mix} to replace the non-Markovian $\pi_{h\to s}$ does not change the reachability probability.

Lemma 4: Given an arbitrary π_h and the construction of the switching policy $\pi_{h\to s}$ from π_h, there exists a switching policy $\pi_{s\to s}$ such that

$$\begin{aligned} P_{\pi_{s\to s}}(\sigma \models \Diamond Z_\phi) &= P_{\pi_{h\to s}}(\sigma \models \Diamond Z_\phi) \\ P_{\pi_{s\to s}}(\sigma \models \Diamond Z_\psi) &= P_{\pi_{h\to s}}(\sigma \models \Diamond Z_\psi). \end{aligned} \tag{18}$$

(The proof will be shown at the end of this subsection.)

Proof: By (14), the objective function $P_{\pi_{h\to s}}(\sigma \models \Diamond Z_\phi)$ is determined by finite occupancy measure $\mu_{\pi_h} < \infty$. By Lemma 3, since $\mu_{\pi_h} = \mu_{\pi_{\text{mix}}}$ when $\mu_{\pi_h} < \infty$, we have

$$P_{\pi_{s\to s}}(\sigma \models \Diamond Z_\phi) = P_{\pi_{h\to s}}(\sigma \models \Diamond Z_\phi). \tag{19}$$

The same result holds for the constrained function

$$P_{\pi_{s\to s}}(\sigma \models \Diamond Z_\psi) = P_{\pi_{h\to s}}(\sigma \models \Diamond Z_\psi). \tag{20}$$

■

Furthermore, we can show that for any π_h, there always exists a $\pi_{s\to s}$ that is at least as good as π_h in performance.

Proof of Theorem 2: Given an arbitrary non-Markovian policy π_h, by Lemma 2, the switching policy $\pi_{h\to s}$ is always at least as good as the policy π_h. By Lemma 4, the switching policy policy $\pi_{s\to s}$ is no worse than the policy $\pi_{h\to s}$. Thus, the switching policy $\pi_{s\to s}$ is a dominating class of policies for the constrained reachability problem. ■

D. Optimality of stationary policy

Now we finish the proof for our main result, Theorem 1.

Proof of Theorem 1: By Theorem 2, on the general MDP, the switching policy $\pi_{s\to s}$ is dominating. Since such switching is encoded by the construction of the double product MDP, the stationary policy $\pi_{s\to s}$ is dominating on the double product MDP. ■

VI. Linear Programming Formulation for Constrained sc-LTL Planning

In this section, we present a linear programming formulation for solving the constrained sc-LTL planning problem. As established in Section IV-B, the original constrained sc-LTL problem can be transformed into an equivalent constrained reachability problem on the double product MDP, for which an optimal stationary policy exists. Our LP formulation computes this optimal stationary policy by combining a pre-switching policy π_{mix} with two stationary sub-policies: an optimistic policy π_ψ that prioritizes the objective, and a pessimistic policy π_ϕ that prioritizes safety. The overall procedure, from problem transformation to LP solution and policy composition, is summarized in Algorithm 1. Here, we ignore the process of how to obtain π_ψ and π_ϕ since it is a well-known result [1].

A. Removing self-loop

This subsection corresponds to Step 10–13 of Algorithm 1, where we partition the state space to avoid states with infinite occupancy measures caused by self-loops. The occupancy measure is defined only for states with finite values, which are obtained by partitioning the state space into F, $S^?$, and S^0, where

$$F := Z_\phi \cup Z_\psi, \tag{21}$$

$$S^0 := \{s \mid P_{\max}(s \models \Diamond\{Z_\phi \cup Z_\psi\}) = 0\}, \tag{22}$$

$$S^? := S \setminus \left(F \cup S^0\right). \tag{23}$$

The set S^0 can be found via a backward reachability graph search; a path starting from these states can never visit states $Z_\phi \cup Z_\psi$ under any policies. We exclue $\mu(s,a)$ with $s \in S^0 \cup F$ from the decision variable X.

B. Representing the constrained reachability problem in a linear form

This subsection corresponds to Step 14–18 of Algorithm 1, where we solve the LP for the pre-switching policy π_{mix}. Solving for π_{mix} requires introducing a decision variable X, from which we define two linear functions $R_\phi(X)$ and $C_\phi(X)$ representing the objective and constraint satisfaction

probabilities, respectively. This allows us to formulate the constrained reachability problem in standard LP form:

$$\begin{aligned} &\max_{X} R_\phi(X) \\ \mathbf{s.t.}\quad &C_\phi(X) - \tau \geq 0 \\ &X \geq 0. \end{aligned} \tag{24}$$

Here for all $s \in S$, $\pi(s) \in \Delta(s)$, under the policy π, we use the weighted sum of all $\mu(i,a)$, to represent the objective function as

$$\begin{aligned} &P_{\pi_{s\to s}}(\sigma \models \Diamond Z_\phi) \\ &= \sum_{i\in S^?} \sum_{a\in A(i)} \sum_{j\in Z_\phi} \mu'(i,a)T(i,a,j) + \sum_{j\in Z_\phi} \beta(j) \\ &+ \sum_{i\in S^?} \sum_{a\in A(i)} \sum_{j\in Z_\psi} \mu'(i,a)T(i,a,j)P_{\max}(j \models \Diamond Z_\phi) \\ &+ \beta(j) \sum_{j\in Z_\phi} P_{\max}(j \models \Diamond Z_\phi). \end{aligned} \tag{25}$$

Meanwhile, the constraint function is expressed as

$$\begin{aligned} &P_{\pi_{s\to s}}(\sigma \models \Diamond Z_\psi) \\ &= \sum_{i\in S^?} \sum_{a\in A(i)} \sum_{j\in Z_\psi} \mu'(i,a)T(i,a,j) + \sum_{j\in Z_\psi} \beta(j) \\ &+ \sum_{i\in S^?} \sum_{a\in A(i)} \sum_{j\in Z_\psi} \mu'(i,a)T(i,a,j)P_{\max}(j \models \Diamond Z_\phi) \\ &+ \beta(j) \sum_{j\in Z_\phi} P_{\max}(j \models \Diamond Z_\phi) \\ &\geq \tau. \end{aligned} \tag{26}$$

Furthermore, as in [24], the occupancy measure must satisfy the following flow constraint:

$$\beta(j) + \underbrace{\sum_{i\in S^?} \sum_{a\in A(i)} \mu'(i,a)T(i,a,j)}_{\text{inflow}} = \underbrace{\sum_{a\in A} \mu'(j,a)}_{\text{outflow}}. \tag{27}$$

C. Composing the switching policy

Finally, once π_{mix}, π_ψ, and π_ϕ are computed, we compose the switching policy $\pi_{s\to s}$ as in Step 21–26 of Algorithm 1.

Algorithm 1 Solving Constrained sc-LTL Planning

1: **Input:** LMDP $\mathcal{M}$, sc-LTL specifications ϕ, ψ, constraint threshold τ
2: **Output:** Optimal policy $\pi_{s\to s}$ for the constrained sc-LTL planning problem
3: *// Construct double product MDP*
4: Construct DFAs $\mathcal{A}_\phi$, $\mathcal{A}_\psi$ corresponding to ϕ, ψ
5: Construct the double product MDP $\mathcal{M}^\times = \mathcal{M} \times \mathcal{A}_\phi \times \mathcal{A}_\psi$
6: Define target sets $Z_\phi^\times$, $Z_\psi^\times$ in the double product MDP
7: *// Solve unconstrained reachability subproblems*
8: Compute $\pi_\phi := \arg\max_\pi P_\pi(\sigma \models \Diamond Z_\psi^\times)$
9: Compute $\pi_\psi := \arg\max_\pi P_\pi(\sigma \models \Diamond Z_\phi^\times)$
10: *// Partition state space*
11: Compute $F := Z_\phi^\times \cup Z_\psi^\times$
12: Let $S_0 := \{s^\times \in S^\times \mid P_{\max}(s^\times \models \Diamond F) = 0\}$
13: Let $S_? := S^\times \setminus (F \cup S_0)$
14: *// Solve LP for pre-switching policy*
15: Define decision variable $\mu(i,a)$ for $i \in S_?$
16: Add flow constraints for all $j \in S_?$:
17: $\quad \sum_a \mu(j,a) = \beta(j) + \sum_{i,a} \mu(i,a)T(i,a,j)$
18: Add constraint satisfaction: $C_\psi(\mu) \geq \tau$
19: Maximize objective: $R_\phi(\mu)$
20: Solve LP to obtain μ^* and corresponding stationary policy π_{mix}
21: *// Compose switching policy*
22: Define $\pi_{s\to s}(\langle s, q_\phi, q_\psi\rangle)$ as:
23: $\quad$ **if** $q_\phi \notin Z_\phi^\times$ and $q_\psi \notin Z_\psi^\times$ **then** $\pi_{\text{mix}}(s)$
24: $\quad$ **else if** $q_\phi \in Z_\phi^\times$ and $q_\psi \notin Z_\psi^\times$ **then** $\pi_\psi(s)$
25: $\quad$ **else if** $q_\phi \notin Z_\phi^\times$ and $q_\psi \in Z_\psi$ **then** $\pi_\phi(s)$
26: **return** $\pi_{s\to s}$

VII. Case Study

To demonstrate how our method balances between objective satisfaction and safety constraint, we construct a constrained planning scenario where satisfying both specifications requires non-trivial trade-off behavior. The experiment is conducted on a 4×5 grid world with stochastic dynamics and strategically placed barriers.

A. Grid environment and sc-LTL objective

To evaluate how our method balances between objective satisfaction and safety constraints, As illustrated in Fig. 4, we design a grid world environment where the two sc-LTL specifications are intentionally in probabilistic conflict. The key idea is to place absorbing *barrier states* in locations that force any policy pursuing one specification to incur a non-negligible risk of failing the other. As a result, no policy can satisfy both specifications with probability 1. Achieving a better success rate for one inevitably reduces the probability of satisfying the other, thereby creating a setting where a trade-off is necessary.

- **Dynamics:** A 4×5 grid world. Actions $\mathcal{A} = \{\mathrm{N}, \mathrm{S}, \mathrm{E}, \mathrm{W}\}$ attempt a unit move to the adjacent cell in the nominated direction; if the adjacent cell lies outside the grid, the agent remains in place. On *white* states, all transitions are deterministic (no slip). On *gray* states, each action goes to its intended successor with probability 0.8 and to each orthogonal successor with probability 0.1. The initial position is $(0,0)$.
- **Objective:** $\phi = \Diamond(A \wedge \Diamond B)$, requiring the agent to visit a cell labeled A and subsequently a cell labeled B.
- **Safety constraint:** $\psi = \Diamond C$, requiring that the agent visit a cell labeled C at least once.
- **Labels:** A at $(0,1)$, B at $(3,1)$, and C at $(3,3)$.
- **Barrier states (black cells):** $(0,4)$, $(1,0)$, $(1,2)$, $(1,4)$, $(2,0)$, $(2,2)$, $(2,4)$, $(3,0)$, $(3,2)$, $(3,4)$. These barriers are absorbing: once entered, the agent cannot leave, and both φ and ψ fail.

This spatial arrangement forces the trade-off: reaching C reliably requires traversing a risky corridor adjacent to barriers, which can jeopardize the sequence $A \to B$. Conversely, choosing a safer route to A and B bypasses C.

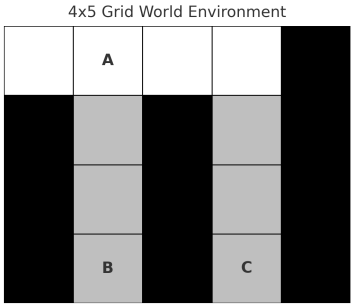

Fig. 4. 4×5 grid world environment. White cells have deterministic transitions. Gray cells have stochastic transitions (vertical moves may slip left/right into adjacent barriers). Black cells are absorbing barriers. Letters A, B, and C indicate labeled target cells. The gray corridor is adjacent to barriers, so any policy that attempts to reach B or C incurs a strictly positive probability of absorption. Therefore, in the planning problem where reaching B and reaching C are specified separately as the objective and the constraint, respectively, the inherent trade-off must be explicitly accounted for.

B. Experiment

We evaluate the approach on the conflict-rich $4{\times}5$ grid world introduced above. All experiments were implemented in Python on a Windows 11 machine equipped with an Intel i9-14900K processor. The Mona package [25] was used to translate the sc-LTL objectives into DFAs.

The experiment follows the pipeline in Algorithm 1. Specifically:

- $\mathcal{M}$ is the 4×5 grid world with barrier states.
- $\phi = \Diamond(A \wedge \Diamond B)$, DFA size $|\mathcal{A}_\phi| = 3$.
- $\psi = \Diamond C$, DFA size $|\mathcal{A}_\psi| = 2$.
- The constraint threshold $\tau \in [0, 1]$ is varied to control the desired safety level.

Following Algorithm 1, we first compute the unconstrained sub-policies π_ϕ and π_ψ by value iteration, and then, for each threshold $\tau \in [0, 1]$, solve the linear program to obtain the mixed (pre-switching) policy π_{mix}, which is finally composed with the sub-policies into the switching policy $\pi_{s\to s}$.

A key structural finding is that π_{mix} randomizes at a single state only, namely at state $(0, 1)$, while all other visited states remain deterministic. Moreover, outside the state $(0, 1)$, the chosen actions are invariant with respect to τ. Fig. 5(a)–(b) visualize π_{mix} at two representative thresholds ($\tau = 0.20$ and 0.60). The green arrows indicate the two outgoing actions being mixed, with line width proportional to the mixing weights. As τ increases, the mixing probabilities at $(0, 1)$ smoothly shift to satisfy the constraint more, while the red arrows elsewhere (deterministic actions) are unchanged. Fig. 5(c)–(e) show the sub-policies π_ϕ and π_ψ used to construct $\pi_{s\to s}$.

Fig. 6 shows the trade-off between the objective function and constraint function obtained by sweeping τ with step 0.01. The instance is feasible up to $\tau = 0.64$. When $\tau > 0.64$ no policy can satisfy the constraint. For small thresholds (approximately $\tau \leq 0.21$), the resulting policy $\pi_{s\to s}$ coincides with the unconstrained deterministic policy

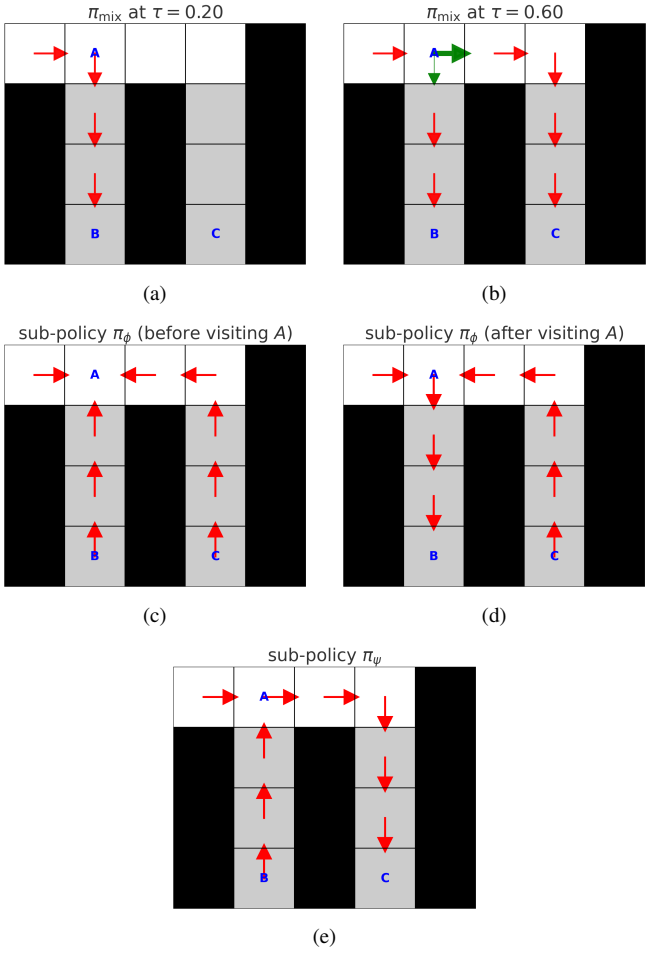

Fig. 5. (a)–(b) Pre-switching policy π_{mix} obtained from the linear program at two constraint tolerances ($\tau = 0.20$ and 0.60). Randomization occurs only at $(0, 1)$. As τ increases, the mixing probabilities at $(0, 1)$ adjust, while actions elsewhere remain deterministic and unchanged. (c) Sub-policy π_ϕ before visiting A (move toward A). (d) Sub-policy π_ϕ after visiting A (move toward B). (e) Sub-policy π_ψ (reach C). Red arrows denote deterministic actions; green arrows denote mixed actions with line width proportional to mixing probabilities. Together these sub-policies realize the switching policy $\pi_{s\to s}$, whose illustration is given in Fig. 3.

π_ϕ. As τ increases toward the feasibility limit, the solution reallocates probability mass to improve $P(\sigma \models \psi)$ at the expense of $P(\sigma \models \phi)$, producing the expected trade-off.

C. Complexity and runtime

The complexity of Algorithm 1 is polynomial in the number of decision variables and constraints. We denote this by $\text{poly}(m, n)$. For the double product MDP built from an LMDP with size $(|S|, |\mathcal{A}|)$ and DFAs of sizes $|Q_\phi|$ and $|Q_\psi|$, we have

$$m = |S|\,|\mathcal{A}|\,|Q_\phi|\,|Q_\psi|, \qquad n = |S|\,|Q_\phi|\,|Q_\psi|.$$

Algorithm 1 computes the mixed, optimistic, and pessimistic policies. Solving for the mixed policy dominates in cost because it uses more decision variables. We solve all LPs with the HiGHS linear optimizer under default settings. On our machine, one LP solve per τ takes $\approx 0.002\,\text{s}$ on average, and sweeping $\tau \in [0, 0.64]$ with step 0.01 takes $\approx 0.13\,\text{s}$.

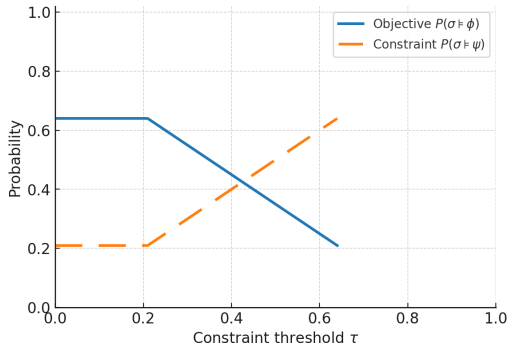

Fig. 6. Trade-off between the objective and the constraint as the threshold τ varies. Curves are computed via Algorithm 1 for $\tau \in [0, 1]$ with a step size of 0.01. The problem is feasible up to $\tau = 0.64$; beyond this, no policy can satisfy the constraint. For small thresholds (approximately $\tau \leq 0.21$), the solution coincides with the unconstrained deterministic policy π_ϕ. As τ increases, the policy shifts to satisfy ψ, trading off the performance in satisfying ϕ.

VIII. Conclusion

We have addressed the challenging problem of solving constrain sc-LTL planning problem. Our work establishes that, despite the non-Markovian nature of the optimal policy in the original MDP, one can construct an equivalent optimal stationary policy on a double product MDP. This structural insight enables us to solve the problem via a linear program over reachability objectives and constraints. We demonstrated that switching between different stationary policies suffices to capture the trade-offs between objectives and constraints, and this switching is inherently captured in the double product MDP's structure. Our results lay the groundwork for scalable and theoretically sound methods for constrained temporal logic planning in MDPs.

References

[1] C. Baier and J.-P. Katoen, *Principles of Model Checking*. Cambridge, MA: The MIT Press, 2008.

[2] G. Fainekos, H. Kress-Gazit, and G. Pappas, "Temporal Logic Motion Planning for Mobile Robots," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 2020–2025.

[3] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, Dec. 2009.

[4] E. Altman, *Constrained Markov Decision Processes: Stochastic Modeling*, 1st ed. Boca Raton, FL: Routledge, 2021.

[5] O. Kupferman and M. Y. Vardi, "Model Checking of Safety Properties," *Formal Methods in System Design*, vol. 19, no. 3, pp. 291–314, Nov. 2001.

[6] A. Ulusoy, T. Wongpiromsarn, and C. Belta, "Incremental control synthesis in probabilistic environments with Temporal Logic constraints," in *Proceedings of the 51st IEEE Conference on Decision and Control*, Maui, HI, Dec. 2012, pp. 7658–7663.

[7] Z. Xuan and Y. Wang, "Control Synthesis in Partially Observable Environments for Complex Perception-Related Objectives," *IEEE Control Systems Letters*, vol. 9, pp. 1724–1729, 2025.

[8] V. Nenchev and C. Belta, "Receding horizon robot control in partially unknown environments with temporal logic constraints," in *2016 European Control Conference*, Aalborg, Denmark, Jun. 2016, pp. 2614–2619.

[9] A. Bhatia, L. E. Kavraki, and M. Y. Vardi, "Motion planning with hybrid dynamics and temporal goals," in *49th IEEE Conference on Decision and Control*, Atlanta, GA, Dec. 2010, pp. 1108–1115.

[10] X. Ding, S. L. Smith, C. Belta, and D. Rus, "Optimal control of markov decision processes with linear temporal logic constraints," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1244–1257, May 2014.

[11] S. L. Smith, J. Tůmová, C. Belta, and D. Rus, "Optimal path planning for surveillance with temporal-logic constraints," *The International Journal of Robotics Research*, vol. 30, no. 14, pp. 1695–1708, Dec. 2011.

[12] L. Li and J. Fu, "Sampling-based approximate optimal temporal logic planning," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation*, Singapore, May 2017, pp. 1328–1335.

[13] ——, "Approximate dynamic programming with probabilistic temporal logic constraints," in *Proceedings of the 2019 American Control Conference*, Philadelphia, PA, Jul. 2019, pp. 1696–1703.

[14] K. Deng, Y. Chen, and C. Belta, "An Approximate Dynamic Programming Approach to Multiagent Persistent Monitoring in Stochastic Environments With Temporal Logic Constraints," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4549–4563, Sep. 2017.

[15] M. Cai, S. Xiao, Z. Li, and Z. Kan, "Optimal Probabilistic Motion Planning With Potential Infeasible LTL Constraints," *IEEE Transactions on Automatic Control*, vol. 68, no. 1, pp. 301–316, Jan. 2023.

[16] M. Cohen and C. Belta, "Temporal logic guided safe model-based reinforcement learning," in *Adaptive and Learning-Based Control of Safety-Critical Systems*. Springer International Publishing, 2023, pp. 165–192.

[17] D. Aksaray, Y. Yazıcıoğlu, and A. S. Asarkaya, "Probabilistically guaranteed satisfaction of temporal logic constraints during reinforcement learning," in *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Prague, Czech Republic, Sep. 2021, pp. 6531–6537.

[18] J. Kretínský, G. A. Pérez, and J.-F. Raskin, "Learning-Based Mean-Payoff Optimization in an Unknown MDP under Omega-Regular Constraints," *LIPIcs, Volume 118, CONCUR 2018*, vol. 118, pp. 8:1–8:18, 2018.

[19] 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*, Richland, SC, May 2021, pp. 583–592.

[20] A. K. Bozkurt, Y. Wang, and M. Pajic, "Model-free learning of safe yet effective controllers," in *Proceedings of the 2021 60th IEEE Conference on Decision and Control*, Austin, TX, Dec. 2021, pp. 6560–6565.

[21] S. Jha, V. Raman, D. Sadigh, and S. A. Seshia, "Safe autonomy under perception uncertainty using chance-constrained temporal logic," *Journal of Automated Reasoning*, vol. 60, no. 1, pp. 43–62, 2018.

[22] W. Liu, N. Mehdipour, and C. Belta, "Recurrent Neural Network Controllers for Signal Temporal Logic Specifications Subject to Safety Constraints," *IEEE Control Systems Letters*, vol. 6, pp. 91–96, 2022.

[23] B. Lacerda, D. Parker, and N. Hawes, "Optimal and dynamic planning for markov decision processes with co-safe ltl specifications," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, IL, Sep. 2014, pp. 1511–1516.

[24] C. Derman, *Finite State Markovian Decision Processes*. New York, NY: Academic Press, 1970, p. 58.

[25] N. Klarlund and A. Møller, "Mona version 1.4 user manual," *B R I C S Notes Series*, no. NS-01-1, 2001.

Appendix

In the Appendix, we formally proof Lemma 3. The proof is based on the the fact that all recurrent states in the MDP are contained in end components.

Definition 7: Let $\mathcal{M} = (S, A, T, \beta)$ be an MDP. An *end component* of $\mathcal{M}$ is a pair (T, A_T) such that:

- $T \subseteq S$ is non-empty, and $A_T : T \to 2^A$ assigns to each $s \in T$ a non-empty set $A_T(s) \subseteq A(s)$;
- Each end component is closure. For all $s \in T$ and $a \in A_T(s)$, every successor s' with $T(s, a, s') > 0$ is also in T;
- Each end component is strong connectivity. In the directed graph induced by T and actions A_T, every state

in T is reachable from every other state in T via actions in A_T.

We denote by $\mathrm{EC}(\mathcal{M})$ the set of all end components of $\mathcal{M}$. Let $\mathrm{EC}_{\pi_h}(\mathcal{M}) \subseteq \mathrm{EC}(\mathcal{M})$ denote the set of end components that are valid under policy π_h. $\mathrm{EC}_{\pi_h}(\mathcal{M})$ contains all the recurrent states under policy π_h.

Proof of Lemma 3: (1) For all transient (under π_h) states $x \in \overline{X}$, both $\mu_{\pi_{\mathrm{mix}}}(x,a)$ and $\mu_{\pi_h}(x,a)$ satisfies the same flow constraint,

$$\begin{aligned}
\mu_{\pi_h}(x) &= \beta(x) + \sum_{t=1}^{\infty} \sum_{a \in A(x)} P_\beta^\pi(X_t = x, A_t = a) \\
&= \beta(x) \\
&+ \sum_{y \in \overline{X}} \sum_{t=1}^{\infty} \sum_{a \in A(y)} P_\beta^\pi(X_{t-1} = y, A_{t-1} = a) T(y,a,x) \\
&= \beta(x) + \sum_{y \in \overline{X}} \sum_{a \in A(y)} \mu_{\pi_h}(y,a) T(y,a,x) \\
&= \beta(x) + \sum_{y \in \overline{X}} \sum_{a \in A(y)} \mu_{\pi_h}(y) \pi_{\mathrm{mix}}(y,a) T(y,a,x). \quad (28)
\end{aligned}$$

Similarly,

$$\mu_{\pi_{\mathrm{mix}}}(x) = \beta(x) + \sum_{y \in \overline{X}} \sum_{a \in A(y)} \mu_{\pi_{\mathrm{mix}}}(y) \pi_{\mathrm{mix}}(y,a) T(y,a,x). \quad (29)$$

Since the flow constrain for either μ_{π_h} or $\mu_{\pi_{\mathrm{mix}}}$, is only determined by the one-step transition $\pi_{\mathrm{mix}}(y,a)T(y,a,x)$.

In matrix form, let μ and β be row vectors with $R^{1 \times |\overline{X}|}$,

$$\mu_{\pi_h} = \beta + \mu_{\pi_h} P_{\overline{X} \to \overline{X}}^{\pi_{\mathrm{mix}}} \quad (30)$$

$$\mu_{\pi_{\mathrm{mix}}} = \beta + \mu_{\pi_{\mathrm{mix}}} P_{\overline{X} \to \overline{X}}^{\pi_{\mathrm{mix}}}. \quad (31)$$

where $P_{\overline{X} \to \overline{X}}^{\pi_{\mathrm{mix}}}$ is the submatrix of the transition matrix under policy π_{mix}, restricted to states in in $\overline{X}$. Since all states in $\overline{X}$ are transient states induced by π_{mix}. Thus, the matrix $(I - P_{\overline{X} \to \overline{X}}^{\pi_{\mathrm{mix}}})$ is invertible.

Therefore, both μ_{π_h} and $\mu_{\pi_{\mathrm{mix}}}$ becomes the unique solution to the same equation,

$$\begin{aligned}
\mu_{\pi_h} &= \beta (I - P_{\overline{X} \to \overline{X}}^{\pi_{\mathrm{mix}}})^{-1} \\
&= \mu_{\pi_{\mathrm{mix}}}, \quad (32)
\end{aligned}$$

(2) Now we show that all the recurrent states under policy π_h will be recurrent states under policy π_{mix}. The second part follows from the construction of π_{mix} and the properties of end components.

First, π_{mix} reaches every end component in $\mathrm{EC}_{\pi_h}(\mathcal{M})$ that are reachable under policy π_h. This follows from the fact that the occupancy measure on transient states under π_{mix} is identical to that under π_h, so every transient path leading to an end component under π_h is also realized under π_{mix}.

Second, by construction of π_{mix}, all state-action pairs in each EC of $\mathrm{EC}_{\pi_h}(\mathcal{M})$ are chosen with non-zero probability and no other actions are taken. Because each EC is strongly connected, this ensures that every state-action pair in the EC is visited infinitely often under π_{mix}.

Consequently, the recurrent behavior and the occupancy measure of π_{mix} coincide with those of π_h. ■