Ensuring Logic in the Fog: Sound POMDP Synthesis with LTL Objectives

Can Zhou[1] , **Yulong Gao**[1] , **Pian Yu**[2]
[1]Imperial College London, United Kingdom
[2]University College London, United Kingdom
{c.zhou24, yulong.gao}@imperial.ac.uk, pian.yu@ucl.ac.uk

Abstract

Synthesising autonomous agents that can navigate uncertain environments while adhering to complex temporal constraints remains a fundamental challenge. While Linear Temporal Logic (LTL) provides a rigorous language for specifying such tasks, the inherent undecidability of qualitatively verifying LTL satisfaction in partially observable Markov decision processes renders quantitative synthesis difficult, especially when designing reliable reward signals for approximate solvers. In this paper, we bridge this gap with a novel, sound reward-shaping mechanism that dynamically generates belief-dependent rewards grounded in certified LTL satisfaction. By integrating this mechanism into an enhanced Monte Carlo Planning framework, we empower agents to navigate the 'fog' of partial observability with a search process focused on maximising verifiable success. Our experiments demonstrate that this approach not only thrives in scenarios where existing solvers fail but also maintains effectiveness and scalability across diverse benchmark domains.

1 Introduction

Partially Observable Markov Decision Processes (POMDPs) provide a principled framework for sequential decision-making amidst the fog of uncertainty, accounting for both stochastic dynamics and incomplete state information [Kaelbling *et al.*, 1995; Sebastian *et al.*, 2005]. When coupled with ω-regular properties like Linear Temporal Logic (LTL) [Pnueli, 1977], they allow for the synthesis of policies that satisfy complex, temporally extended goals over infinite traces in uncertain, partially observable environments. Therefore, POMDP-LTL synthesis has attracted considerable interest across AI, robotics, and control [Sharan and Burdick, 2014; Bouton *et al.*, 2020; Kalagarla *et al.*, 2024].

Despite its potential, quantitative POMDP-LTL synthesis, aiming to maximise satisfaction probability, remains notably challenging. Optimal policy synthesis for general planning tasks in infinite-horizon POMDPs is undecidable, even for simple reachability objectives, as it necessitates reasoning over an uncountable belief space and may require unbounded memory [Chatterjee *et al.*, 2016]. Practical approaches therefore rely on tractable approximations that reformulate the problem as reward maximisation, with reward functions encoding task satisfaction, and apply approximate solvers such as Point-Based Value Iteration (PBVI) [Pineau *et al.*, 2003] and Partially Observable Monte Carlo Planning (POMCP) [Silver and Veness, 2010].

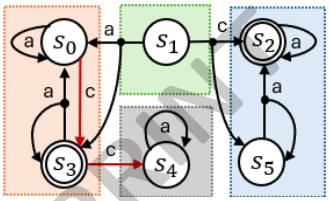

Figure 1: A POMDP where states sharing the same observation are indicated by a common background colour. This colour-coding convention is maintained throughout all subsequent figures.

However, constructing appropriate reward signals for full LTL is non-trivial, since its semantics concern non-terminating behaviour over infinite traces. Existing approaches either restrict attention to finite-trace LTL (LTL_f), which reduces satisfaction to a one-shot reachability problem [Kalagarla *et al.*, 2022; Liu *et al.*, 2021], or apply optimistic reward shaping based on state-space accepting maximal end components (AMECs) [Bouton *et al.*, 2020]. Nevertheless, due to partial observability, POMDP policy synthesis must operate over the belief space, and many objectives such as recurrence and persistence require full LTL. Furthermore, an optimistic reward for POMDP-LTL may assign a positive satisfaction signal to a belief state even when no single belief-based policy (defined as a policy shared by all states within the belief's support) can realise the implied satisfaction probability. This challenge, which we term the *common policy issue*, is illustrated in the following example.

Example 1. *Consider the POMDP in Figure 1 with action set $\{a, c\}$ and initial belief $b_0 = \{s_1 : 1.0\}$. The LTL objective $\varphi = \Box\Diamond(s_2 \vee s_3)$, requiring the system to visit states s_2 or s_3 infinitely often. Under full observability, two AMECs exist: 1. $\{s_0, s_3\}$, which achieves almost-sure satisfaction (i.e. probability one) by alternating actions a and c, and 2. $\{s_2, s_5\}$, which succeeds by repeatedly playing a. This suggests that all beliefs supported on AMECs yield satisfaction*

with probability one. However, with partial observability, no belief-based policy supported on $\{s_0, s_3\}$ can satisfy φ almost surely: state s_0 requires action c, but taking c in s_3 leads to the sink s_4. Conversely, beliefs supported on $\{s_2, s_5\}$ can still satisfy φ almost surely by playing a.

Furthermore, assigning positive reward one to all states within the state-space AMECs (i.e., $\{s_0, s_2, s_3, s_5\}$) incorrectly suggests that actions a and c are equally optimal at b_0. In reality, c is the uniquely optimal action.

More fundamentally, encoding exact LTL satisfaction as a reward signal is undecidable for POMDPs because of the probability leakage inherent in belief transitions, as shown in Example 1 for beliefs supported on $\{s_0, s_3\}$. For such beliefs, deciding whether a policy with positive satisfaction probability exists is undecidable [Baier *et al.*, 2008]. These theoretical barriers make it difficult to construct sound reward signals. Without these, approximate solvers struggle to converge on reliable, belief-based policies for LTL objectives.

Contributions. We bridge the theoretical and practical gaps in quantitative POMDP-LTL synthesis by introducing a novel mechanism for sound reward shaping. In contrast to fixed scalar reward structures, our approach defines reward assignment rules that integrate with an enhanced online POMCP planner. This allows the system to dynamically evaluate rewards and guide the search toward maximising certified satisfaction, thereby enabling reliable belief-based policy synthesis. To the best of our knowledge, this is the first sound and practical anytime synthesis algorithm for full LTL in POMDPs. Our main contributions are summarised as follows: 1) **Sound Reward Shaping:** We introduce a sound reward shaping mechanism over the belief space by leveraging belief support structures, grounded in the insight that almost-sure satisfaction is the only certifiable property in POMDP-LTL. A pruning strategy streamlines reward construction, and the resulting rewards provide a certified lower bound on the true LTL satisfaction probability. 2) **Sound Anytime Synthesis:** We integrate our reward assignment rules into an enhanced POMCP algorithm specialised for anytime LTL synthesis. Once the search reaches a certified belief region, the agent may continue exploring for higher satisfaction or switch to a verified policy that guarantees satisfaction. This offers the first sound and practical method for real-time POMDP-LTL synthesis. 3) **Empirical Validation:** We show that our method synthesises reliable belief-based policies in scenarios where existing approaches fail. We further show its effectiveness and scalability on three standard POMDP benchmarks with diverse LTL tasks.

Related Work. Owing to the undecidability of quantitative synthesis for POMDPs with LTL, prior work has turned to tractable approximations. One possible approach is to restrict policies to randomised finite-state controllers (FSCs) [Chatterjee *et al.*, 2015; Sharan and Burdick, 2014; Ahmadi *et al.*, 2020; Carr *et al.*, 2020; Carr *et al.*, 2021]. However, such finite-memory restrictions impose key limitations: optimisation remains only locally optimal for any fixed memory size, and while performance is highly sensitive to that size, the minimal required memory is undecidable [Madani *et al.*, 1999]. A complementary line of work [Liu *et al.*, 2021; Kalagarla *et al.*, 2022; Kalagarla *et al.*, 2024] considers belief-based policies for finite-trace LTL (LTL$_f$). However, many complex planning tasks require infinite-horizon recurrence and persistence, necessitating full LTL. [Bouton *et al.*, 2020] addresses this by applying PBVI with reward signals derived from state-based AMECs. As illustrated in Example 1, such state-based reward shaping can be overly optimistic under partial observability. A detailed discussion of related work is provided in Appendix A [1].

2 Preliminaries and Problem Statement

This section first outlines the preliminaries of LTL and its equivalent Limit-Deterministic Büchi Automaton (LDBA) representation. We then provide the foundations of POMDPs and the POMCP algorithm for online planning. Finally, the problem under consideration is formulated.

2.1 Linear Temporal Logic

LTL [Pnueli, 1977] extends propositional logic by introducing temporal operators that enable the specification of task requirements over time. The syntax of an LTL path formula over a finite set of atomic propositions ($\mathtt{AP}$) is defined inductively as follows:

$$\varphi ::= \mathsf{true} \mid a \in \mathtt{AP} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi \mid \varphi \mathsf{U} \varphi,$$

where $\bigcirc$ (Next) and U (Until) are temporal operators. Additional derived temporal operators include: $\Diamond\varphi \equiv \mathsf{true}\mathsf{U}\varphi$ (Eventually) and $\Box\varphi \equiv \neg(\Diamond\neg\varphi)$ (Always). Standard Boolean operators, such as $\vee$ (or) and $\rightarrow$ (implies), are used conventionally. The detailed semantics with satisfaction relation of LTL can be found in [Baier and Katoen, 2008]. For a formula φ and an infinite word $w = w_0 w_1 w_2 \cdots \in (2^{\mathtt{AP}})^\omega$, we write $\mathcal{L}(\varphi)$ to denote the language of φ, which is the set of infinite words over $(2^{\mathtt{AP}})^\omega$ that satisfy φ.

Every LTL formula admits an equivalent representation as a Nondeterministic Büchi Automaton (NBA) [Vardi and Wolper, 1994] that accepts the language $\mathcal{L}(\varphi)$. More recently, a subclass of NBAs, known as limit-deterministic Büchi automata, has shown particularly suitable for the quantitative analysis of probabilistic systems [Sickert *et al.*, 2016]. In this paper, we also make use of LDBAs to support our approach.

Definition 1 (LDBA [Sickert *et al.*, 2016])**.** *An LDBA $\mathcal{A}$ is defined as a tuple $\mathcal{A} = (\Sigma, Q, q_0, \Delta, \mathtt{Acc})$ where Σ is an alphabet, $Q = Q_i \uplus Q_{acc}$ is the set of states partitioned into two disjoint sets Q_i and Q_{acc}. $q_0 \in Q_i$ is the initial state. $\Delta = \Delta_i \uplus \Delta_{acc}$ where: $\Delta_i : Q_i \times (\Sigma \cup \{\epsilon\}) \to 2^Q$, $\Delta_{acc} : Q_{acc} \times \Sigma \to 2^{Q_{acc}}$ is singleton-valued, and $\mathtt{Acc} \subseteq Q_{acc}$ is the accepting set.*

An infinite word $w \in \Sigma^\omega$ is recognised by an LDBA $\mathcal{A}$ if there exists an infinite run $r = r_0 r_1 r_2 \cdots \in Q^\omega$, beginning from q_0 and satisfying $r_{i+1} \in \Delta(r_i, \omega_i)$ for all $i \geq 0$ as well as $\mathtt{Inf}(r) \cap \mathtt{Acc} \neq \emptyset$, where $\mathtt{Inf}(r)$ denotes transitions in the accepting set appearing infinitely often in r. The set of infinite words accepted by the LDBA $\mathcal{A}$ is denoted by $\mathcal{L}(\mathcal{A})$. An LTL formula φ over $\mathtt{AP}$ can be efficiently translated into an equivalent LDBA $\mathcal{A}$ with $\Sigma = 2^{\mathtt{AP}}$, such that $\mathcal{L}(\mathcal{A}) =$

[1] An extended version with the appendix is available at https://arxiv.org/pdf/2605.12581

$\mathcal{L}(\varphi)$, using state-of-the-art tools such as Owl [Sickert *et al.*, 2016] and Rabinizer4 [Křetínský *et al.*, 2018].

2.2 Partially Observable MDPs

In this paper, we consider the following mathematical model:

Definition 2 (Labelled POMDPs)**.** *A* labelled POMDP *is defined as a tuple $\mathcal{M} = (S, S_0, A, O, T, Z, L)$, where S, S_0, A, and O are finite sets of states, initial states, actions, and observations, respectively. The state transition function $T : S \times A \to \mathtt{Dist}(S)$, where $T(s,a)(s')$ denotes the probability of transitioning to state s' from state s by taking action a. The observation function $Z : S \times A \to \mathtt{Dist}(O)$, where $Z(s',a)(o)$ denotes the probability of observing o after taking action a and reaching state s'. The labelling function $L : S \to 2^{\mathtt{AP}}$ maps each state $s \in S$ to the set of atomic propositions in $\mathtt{AP}$ that hold true at s.*

A belief state b of $\mathcal{M}$ is a probability distribution over its state space S. Let b_0 denote the initial belief state and let B be the set of all possible belief states for a given POMDP $\mathcal{M}$. The *belief support* of a belief state $b \in B$ is the set of states with positive belief, defined as $\mathtt{Supp}(b) := \{\, s \in S \mid b(s) > 0\}$. The set of all possible belief supports in $\mathcal{M}$ is then given by $B_{\mathtt{Supp}} := \big\{\, \Theta \subseteq S \;\big|\; \exists o \in O,\ \forall s \in \Theta,\ o \in \mathtt{obs}(s)\big\}$, where $\mathtt{obs} : S \to 2^O$. We write $b(\Theta) := \sum_{s\in\Theta} b(s)$ for the probability mass assigned by $b \in B$ to $\Theta \subseteq S$, called its *belief mass*. The enabled action set at state s is defined as $A_{\mathrm{E}}(s) := \{a \in A \mid \mathtt{Post}_s(a) \neq \emptyset\}$ where $\mathtt{Post}_s(a) = \mathtt{Supp}(T(s,a))$. We then define $A_{\mathrm{E}}(\Theta) := \bigcap_{s\in\Theta} A_{\mathrm{E}}(s)$. Without loss of generality, we assume that states with the same observation admit the same set of enabled actions; that is, for any $s, s' \in \Theta$, $A_{\mathrm{E}}(s) = A_{\mathrm{E}}(s') = A_{\mathrm{E}}(\Theta)$.

Paths. A *history* at time t is a finite sequence $h_t = (a_0, o_1, \ldots, a_{t-1}, o_t) \in (A \times O)^t$ capturing the agent's observable interaction with the environment. A *belief path* is an infinite sequence $\rho = (b_0, a_0, o_1, b_1, a_1, o_2, \ldots)$ in which each belief $b_{t+1} = \tau(b_t, a_t, o_{t+1})$ is obtained via Bayesian update. The set of such paths starting from b_0 is denoted by $\mathrm{BPaths}(b_0)$. For any ρ, the set of *compatible state paths* $\mathrm{SPaths}(\rho) \subseteq \mathcal{S}^\omega$ contains all sequences $(s_0, s_1, \ldots)$ such that $s_0 \sim b_0$, $s_{t+1} \sim T(s_t, a_t)$, and $o_{t+1} \sim Z(s_{t+1}, a_t)$, consistent with the induced belief evolution.

Policy and LTL Satisfaction. We consider a deterministic belief-based policy $\pi \in \Pi : B \to A$ that maps a belief state to an action. Denote by Π the set of these policies. Under a given policy π, the probability that an LTL specification φ holds at belief state b is denoted by

$$\Pr^{\pi}_{\mathrm{b}}(\varphi) = \Pr\{\rho_{\mathrm{b}} \in \mathrm{BPaths(b)}, \rho_{\mathrm{s}} \in \mathrm{SPaths}(\rho_{\mathrm{b}}) \mid \rho_{\mathrm{s}} \models \varphi\},$$

which measures all belief executions induced by π whose compatible underlying state paths satisfy φ. Accordingly, the satisfaction probability for a POMDP $\mathcal{M}$ (under the policy π) with initial belief b_0 is defined as $\Pr^{\pi}_{\mathcal{M}}(\varphi) := \Pr^{\pi}_{\mathrm{b}_0}(\varphi)$.

Undecidability. Optimal policy synthesis for POMDPs is generally undecidable under Büchi objectives induced by LTL. [Baier *et al.*, 2008] show that, for Büchi objectives, both qualitative verification of positive (non-zero) satisfaction probability and quantitative verification are undecidable. The only property that can be certified is almost-sure satisfaction.

Product POMDP. Analogous to the product construction for MDPs [Baier and Katoen, 2008], the product POMDP $\mathcal{M}^\times$ synchronises the evolution of the POMDP $\mathcal{M}$ with the LDBA $\mathcal{A}_\varphi$ generated from the LTL φ. This allows satisfaction of φ to be checked by tracking accepting runs of the LDBA along paths of $\mathcal{M}$.

Definition 3 (Product POMDP)**.** *Given a labelled POMDP $\mathcal{M} = (S, S_0, A, O, T, Z, L)$ and an LDBA $\mathcal{A}_\varphi = (\Sigma, Q, q_0, \Delta, \mathtt{Acc})$ from the given LTL formula φ, the product POMDP is a tuple $\mathcal{M}^\times = (S^\times, S_0^\times, A^\times, O, T^\times, Z^\times, L^\times, \mathtt{Acc}^\times)$ with state-space $S^\times = S \times Q$. The set of accepting states is given by $\mathtt{Acc}^\times = \{(s,q) \in S^\times \mid q \in \mathtt{Acc}\}$.*

Given the initial belief state b_0 of $\mathcal{M}$, we have $b_0^\times((s, \Delta(q_0, \mathcal{L}(s)))) = b_0(s)$, $\forall s \in S_0$. The set of all possible belief states of $\mathcal{M}^\times$ is denoted by $B^\times$. The complete definition of the product POMDP is provided in Appendix B.

2.3 Partially Observable Monte-Carlo Planning

POMCP [Silver and Veness, 2010] is a widely adopted online planning algorithm for POMDPs, as it effectively mitigates the *curse of dimensionality* through state sampling and the *curse of history* via history sampling with a black-box simulator. At each step t, it uses Monte Carlo Tree Search [Coulom, 2006] to build and explore a search tree rooted at $\mathcal{T}(h_t) = \langle \mathcal{N}(h_t), \mathcal{V}(h_t), \mathcal{P}(h_t)\rangle$, where $\mathcal{N}(h_t)$ tracks the number of visits to the history h_t, $\mathcal{V}(h_t)$ estimates the expected return, and $\mathcal{P}(h_t)$ holds particles that approximate the belief state b_t. Due to space limitations, details on how POMCP works are provided in Appendix C.

2.4 Problem Statement

We study the following policy synthesis problem:

Problem 1. *Given a POMDP $\mathcal{M}$ and an LTL specification φ, synthesise a deterministic belief-based policy π that maximises the probability of satisfying φ, i.e. $\max_{\pi\in\Pi} \Pr^{\pi}_{\mathcal{M}}(\varphi)$.*

Following the standard product construction for LTL and probabilistic systems [Baier and Katoen, 2008], we form the product POMDP $\mathcal{M}^\times = (S^\times, S_0^\times, A^\times, O, T^\times, Z^\times, L^\times, \mathtt{Acc}^\times)$ of the POMDP $\mathcal{M}$ with the LDBA $\mathcal{A}_\varphi$ derived from φ (Definition 3). This synchronises the executions of $\mathcal{M}$ and $\mathcal{A}_\varphi$ and reduces LTL satisfaction in $\mathcal{M}$ to the Büchi objective $\Box\Diamond\,\mathtt{Acc}^\times$ in $\mathcal{M}^\times$:

$$\max_{\pi\in\Pi} \Pr^{\pi}_{\mathcal{M}}(\varphi) = \max_{\pi^\times\in\Pi^\times} \Pr^{\pi^\times}_{\mathcal{M}^\times}(\Box\Diamond\,\mathtt{Acc}^\times).$$

Accordingly, we further reformulate Problem 1 as an equivalent reward-based policy synthesis problem over $\mathcal{M}^\times$. The goal is to design a belief-based reward function that induces an optimal policy whose expected reward is equal to the maximum satisfaction probability of Büchi objective.

Problem 2 (Reward-Based Policy Synthesis)**.** *Given the product POMDP $\mathcal{M}^\times$ induced by a POMDP $\mathcal{M}$ and an LTL objective φ, design a belief-based reward function $R : B^\times \to [0,1]$ and synthesis a policy $\pi^\times : B^\times \to A^\times$ such that*

$$\max_{\pi^\times\in\Pi^\times} \mathbb{E}^{\pi^\times}_{\mathcal{M}^\times}\left[\sum_{t=0}^{\infty} R(b_t^\times)\right] = \max_{\pi^\times\in\Pi^\times} \Pr^{\pi^\times}_{\mathcal{M}^\times}(\Box\Diamond\,\mathtt{Acc}^\times).$$

3 Solution Technique

Solving Problem 2 presents significant theoretical hurdles. Encoding LTL satisfaction via a fixed scalar reward is infeasible since assigning an exact reward is equivalent to computing exact satisfaction probabilities, which is known to be undecidable (Section 1). Moreover, even if such a reward were available, maximising the expected reward within a POMDP remains undecidable [Madani *et al.*, 1999]. Practical methods must therefore rely on approximations in both reward shaping and policy synthesis.

We address this gap by proposing a sound approximation framework designed for practical POMDP-LTL synthesis. Exploiting the fact that almost-sure satisfaction is the only certifiable property, we first show that the exact reward signal in Problem 2 can be soundly approximated by the belief mass over certified satisfaction regions derived from the belief-support structure (Section 3.1). We then develop a belief-dependent reward-shaping mechanism that uses a set of assignment rules to generate sound rewards dynamically, together with a pruning strategy that streamlines reward construction (Section 3.2). Finally, we integrate this mechanism into an enhanced POMCP algorithm that steers the search toward maximising certified satisfaction and synthesising reliable belief-based policies (Section 3.3).

3.1 Sound Approximation of Rewards Signal

For POMDPs with Büchi objectives, while determining exact or zero satisfaction is undecidable, almost-sure satisfaction fortunately admits a decidable property. Specifically, it depends only on the belief support rather than the precise probability distribution [Chatterjee *et al.*, 2007]: if a belief b is almost-sure winning, then any belief b' with $\texttt{Supp}(b') = \texttt{Supp}(b)$ is also almost-sure winning. This allows the continuous belief space to be discretised into a finite belief-support MDP (BSMDP), enabling structural evaluation of certified satisfaction over belief support. In this abstraction, each state corresponds to a belief support $\Theta \subseteq 2^{S^\times}$, and transitions mimic the belief executions in the product POMDP.

Definition 4 (BSMDP [Junges *et al.*, 2021])**.** *Given a product POMDP $\mathcal{M}^\times = (S^\times, S_0^\times, A^\times, O, T^\times, Z^\times, L^\times, \texttt{Acc}^\times)$, let $B^\times_{\texttt{Supp}}$ be belief supports induced by $\mathcal{M}^\times$. The corresponding BSMDP is the tuple $\mathcal{M}^\times_{B_{\texttt{Supp}}} = (B^\times_{\texttt{Supp}}, \Theta_0, T^\times_{\texttt{Supp}}, A^\times)$, where Θ_0 is the initial belief support, and the transition function $T^\times_{\texttt{Supp}}$ is defined as $\Theta' \in T^\times_{\texttt{Supp}}(\Theta, a \in A^\times_{\mathrm{E}}(\Theta))$ if*

$$\Theta' \in \left\{ \cup_{s\in\Theta} \texttt{Post}_s(a) \cap \{s \mid o \in \texttt{obs}(s)\} \mid o \in O^\times \right\}. \quad (1)$$

The BSMDP serves as a formal bridge between qualitative almost-sure satisfaction and general quantitative analysis. By reasoning over the BSMDP, we can identify specific winning supports: subsets of state from which a belief-support based policy exists to satisfy the Büchi objective with probability 1. These winning supports enable the construction of a sound reward signal that, for any belief, lower-bounds the true satisfaction probability by the portion of belief mass guaranteed to satisfy the objective almost surely. We now formalise the notion of winning supports and classify beliefs based on their underlying support structure.

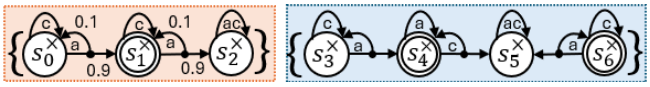

Figure 2: Partially winning belief (supports).

Definition 5 (Winning Support and Winning Region)**.** *A subset $\Theta \subseteq S^\times$ is a winning support if there exists a belief-support policy $\hat{\pi}^\times : B^\times_{\texttt{Supp}} \to A^\times$ beginning with Θ such that every state $s^\times \in \Theta$ satisfies the LTL objective with probability one under $\hat{\pi}^\times$. We denote the union of all winning supports as a winning region $\mathcal{W}^\times := \bigcup \Theta$.*

Definition 6 (Belief Categorisation)**.** *A belief $b^\times \in B^\times$ can be categorised by the relationship between its support $\texttt{Supp}(b^\times)$ and $\mathcal{W}^\times$ as follows:*

1. ***Almost-sure Winning:*** *if $\texttt{Supp}(b^\times) \in \mathcal{W}^\times$;*
2. ***Partially Winning:*** *if $\exists \Theta \in \mathcal{W}^\times$ s.t. $\Theta \subset \texttt{Supp}(b^\times)$;*
3. ***Non-Winning:*** *if $\forall \Theta \in \mathcal{W}^\times$ and $\Theta \not\subseteq \texttt{Supp}(b^\times)$.*

Note that even if the initial belief is non-winning, the agent may reach a (partially) winning belief during belief evolution.

Example 2. *In the left of Figure 2, beliefs supported on $\{s_0^\times, s_1^\times, s_2^\times\}$ are not almost-sure winning, but they contains a winning support $\{s_1^\times\}$ under action c. The satisfaction probability for any belief supported by $\{s_0^\times, s_1^\times, s_2^\times\}$ is lower-bounded by the belief mass assigned to $\{s_1^\times\}$ by repeating c.*

Notably, winning supports are defined with respect to belief-support policies. Since a partially winning belief may contain multiple winning supports (each associated with a distinct policy), obtaining the tightest sound lower bound requires selecting the specific winning support that maximises the belief mass. Simply taking the union of all winning states would result in the *common policy issue* (as discussed in Example 1), where no single policy can satisfy the entire set. This principle is illustrated in the following example.

Example 3. *Consider beliefs supported by $\{s_3^\times, s_4^\times, s_5^\times, s_6^\times\}$ in the right of Figure 2. The winning supports and their admissible actions are: $\{s_3^\times\}$ with $\{a\}$, $\{s_4^\times\}$ with $\{a\}$, $\{s_3^\times, s_4^\times\}$ with $\{a\}$, and $\{s_6^\times\}$ with $\{c\}$. These cannot be merged into $\{s_3^\times, s_4^\times, s_6^\times\}$ when computing a sound lower bound, as no single policy is winning for all three states. The tightest lower bound is therefore determined by whichever has the larger belief mass: $\{s_3^\times, s_4^\times\}$ or $\{s_6^\times\}$.*

We now define $K_{b^\times}$ as the collection of all winning supports contained within the support of a belief state $b^\times$ in the product POMDP $\mathcal{M}^\times$. For instance, in Example 3, a belief $b^\times$ with support $\{s_3^\times, s_4^\times, s_5^\times, s_6^\times\}$ would have the set of winning supports $K_{b^\times} = \{\{s_3^\times\}, \{s_4^\times\}, \{s_3^\times, s_4^\times\}, \{s_6^\times\}\}$. Then a sound belief-based reward signal can be defined as the maximum belief mass assigned to any winning support, reducing Problem 2 to the following sound approximation:

Theorem 1. *Given a product POMDP $\mathcal{M}^\times$, the maximum expected reward in Problem 2 is lower-bounded by a belief-based policy $\tilde{\pi}^\times : B^\times \to A^\times$ that maximises the probability*

of reaching a (partially) winning belief $b^\times$ with maximal belief mass over its winning supports $K_{b^\times}$:

$$\max_{\pi^\times \in \Pi^\times} \mathbb{E}^{\pi^\times}_{\mathcal{M}^\times}\left[\sum_{t=0}^{\infty} R(b^\times_t)\right] \geq \max_{\tilde{\pi}^\times \in \Pi^\times} \mathbb{E}^{\tilde{\pi}^\times}_{\mathcal{M}^\times}\left[\max_{\Theta \in K_{b^\times}} b^\times(\Theta)\right],$$

where $b^\times(\Theta)$ is the belief mass assignment to the set of states contained in Θ.

Tightness. The lower bound in Theorem 1 attains the maximum certified LTL satisfaction probability. The remaining gap to the true value reflects satisfaction probability that cannot be certified in finite time, arising from the undecidable leakage illustrated in Example 1.

3.2 Sound Reward Shaping

Theorem 1 allows sound reward signals to be constructed by identifying winning supports in the BSMDP. However, the state space of BSMDP grows exponentially with the state space of the product POMDP, rendering all winning supports and their associated winning policies computationally expensive. To maintain tractability, we extract a sufficient subset of winning supports, together with their almost-sure or partially winning beliefs, by pruning the search space according to two principles: (i) restricting the state space of the BSMDP to obtain a sub-BSMDP; and (ii) focusing on structurally relevant regions within this sub-BSMDP. A sound reward-shaping mechanism is then derived from this refined subset.

Pruned Search Regions. Recall that in fully observable product MDPs with a Büchi objective, any satisfying run eventually enters and remains within a state-based AMEC. We hence restrict the BSMDP construction to belief supports that intersect the set of underlying state AMECs $\mathcal{E}^\times_S$ in $\mathcal{M}^\times$:

$$\hat{B}^\times_{\text{Supp}} := \{\Theta \in B^\times_{\text{Supp}} \mid \Theta \cap \mathcal{E}^\times_S \neq \emptyset\}. \tag{2}$$

This restriction gives an over-approximation of the belief-support states relevant for satisfaction under belief-based policies, while substantially reducing the BSMDP state space. Moreover, since every winning support ensures almost-sure satisfaction, any belief-support path must almost surely enter a trapping region of the BSMDP in which each support state recurs infinitely often with satisfaction probability one. In the BSMDP, such closed recurrent accepting regions correspond to belief-support accepting MECs (BS-AMECs), defined formally below.

Definition 7 (BS-(A)MECs)**.** *A belief-support end component (BS-EC) of an BSMDP $\mathcal{M}^\times_{B_{\text{Supp}}}$ is a sub-MDP (E, A_E) such that the digraph induced by (E, A_E) is strongly connected. A belief-support MEC (BS-MEC) is an BS-EC (E, A_E) that is not strictly contained in any other BS-EC (E', A'_E). We call a BS-MEC (E, A_E) accepting (BS-AMEC) if all belief supports $\Theta \in E$ are winning.*

Sufficient Winning Region. We therefore restrict the search for the set of winning-supports $\hat{\mathcal{W}}^\times$ ($\hat{\mathcal{W}}^\times \subseteq \mathcal{W}^\times$) to those contained in the BS-AMECs of the sub-BSMDP $\hat{\mathcal{M}}^\times_{B_{\text{Supp}}}$, constructed over belief supports that intersect the state-space AMECs defined in Eq. (2). Any other winning support will almost surely reach $\hat{\mathcal{W}}^\times$ (thus $\hat{\mathcal{W}}^\times$ is sufficient).

Figure 3: BS-MEC structure under differing acceptance conditions.

Algorithm 1 Compute BS-AMECs

Input: BS-MECs $(\mathcal{E}^\times_{B_{\text{Supp}}}, A_{\mathcal{E}^\times})$ of sub-BSMDP $\hat{\mathcal{M}}^\times_{B_{\text{Supp}}}$
Output: Sufficient winning region $\hat{\mathcal{W}}^\times$

```
Ŵ^× ← ∅
for all (E, A_E) ∈ (E^×_{B_Supp}, A_{E^×}) do
    if ∃ Θ ∈ E s.t. Θ ⊆ Acc^× then
        Ŵ^× ← Ŵ^× ∪ {Θ ∈ E}
    else
        Construct the product MDP M^××
        Compute the winning state set S^××_w of M^××
        if S^××_w = S^×× then
            Ŵ^× ← Ŵ^× ∪ {Θ ∈ E}
```

Determining, however, whether a BS-MEC is accepting is non-trivial, since the presence of $\texttt{Acc}^\times$ alone is insufficient. As illustrated in Figure 3, a BS-MEC may contain no winning support, may be only partially accepting, or is a BS-AMEC (from left to right in Figure 3). Hence, belief-support information alone is inadequate: one must verify the existence of a belief-support-based policy under which all underlying states are winning. To this end, Algorithm 1 identifies BS-AMECs by constructing an auxiliary product MDP $\mathcal{M}^{\times\times} = (S^{\times\times}, A^{\times\times}, T^{\times\times}, \texttt{Acc}^{\times\times})$, where $S^{\times\times} = \{(s^\times, \Theta) \mid s^\times \in \Theta, \Theta \in E\}$, $A^{\times\times}(s^\times, \Theta) = A_{\mathcal{E}^\times}(\Theta)$, $T^{\times\times}((s^\times, \Theta), a)((s'^\times, \Theta')) = T^\times(s^\times, a)(s'^\times)$, and $\texttt{Acc}^{\times\times} = \{(s^\times, \Theta) \in S^{\times\times} \mid s^\times \in \texttt{Acc}^\times\}$. One can see that $\mathcal{M}^{\times\times}$ couples each BS-MEC with the underlying state-space dynamics, permitting joint reasoning over belief-support policies and state transitions. A BS-MEC is accepting iff all states in this product MDP are winning.

Theorem 2. *Given the sub-BSMDP $\hat{\mathcal{M}}^\times_{B_{\text{Supp}}}$ and its BS-MECs $(\mathcal{E}^\times_{B_{\text{Supp}}}, A_{\mathcal{E}^\times})$, Algorithm 1 returns a sound and complete union set $\hat{\mathcal{W}}^\times$ over all BS-AMECs.*

Proof scratch. Soundness. If there exists $\Theta \in E$ with $\Theta \subseteq \texttt{Acc}^\times$, then E is a winning support and (E, A_E) is a BS-AMEC by the MEC property. Otherwise, consider the randomised belief-support policy $\bar{\pi}^\times$ that selects uniformly from A_E. If $S^{\times\times}_w = S^{\times\times}$, every BSCC induced by $\bar{\pi}^\times$ is accepting. By Lemma 1 of [Chatterjee *et al.*, 2010], a corresponding deterministic policy $\tilde{\pi}^\times$ exists; hence (E, A_E) is a BS-AMEC. *Completeness.* Immediate from Definitions 5 and 7. A full proof is provided in Appendix D. □

Sound Reward Assignment Rules. We then define the sound reward structure based on the sufficient winning region $\hat{\mathcal{W}}^\times$ computed by Algorithm 1. We call a belief support $\Theta \in \hat{\mathcal{W}}^\times$ a *certified winning support*. Furthermore, Θ is called a *certified partially winning support* if there exists

a certified winning support $\Theta' \in \hat{\mathcal{W}}^{\times}$ such that $\Theta' \subset \Theta$. The set of all certified partially winning support is denoted by $\hat{\mathcal{W}}^{\times}_{\text{PW}}$. Since a belief $b^{\times}$ with certified partially winning support $\texttt{Supp}(b^{\times}) \in \hat{\mathcal{W}}^{\times}_{\text{PW}}$ may contain multiple certified winning supports (see Example 3), we let $\hat{K}_{b^{\times}}$ be the collection of all such supports contained within $\texttt{Supp}(b^{\times})$.

Following the lower bound established in Theorem 1, the sound reward function $\hat{R}(\cdot)$ is defined as:

$$\hat{R}(b^{\times}) = \begin{cases} 1, & \text{if } \texttt{Supp}(b^{\times}) \in \hat{\mathcal{W}}^{\times}, \\ \max\limits_{\Theta \in \hat{K}_{b^{\times}}} b^{\times}(\Theta), & \text{if } \texttt{Supp}(b^{\times}) \in \hat{\mathcal{W}}^{\times}_{\text{PW}}, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where $b^{\times}(\Theta)$ is the belief mass assignment to the set Θ.

3.3 Online POMDP Planning

To synthesise reliable belief-based policies for POMDP-LTL using the proposed sound reward function $\hat{R}(\cdot)$ in Eq. (3), we develop an enhanced POMCP algorithm as an anytime approximation method. Standard POMCP, however, is not directly applicable, as it relies on Monte Carlo simulations that assign rewards to individual state particles, which is incompatible with our belief-based reward signal. The termination conditions for the Monte Carlo simulation also require rigorous treatment. While entering a belief with a certified winning support is terminal as it already yields the maximum reward, entering a belief with a certified partially winning support is not necessarily so. In the latter case, the agent may continue to explore beliefs that offer higher satisfaction probabilities, a distinction further illustrated in Example 4.

Example 4. *Consider a belief node $b^{\times}$ in the Monte Carlo search tree with ideal distribution $\{s_0^{\times} : 0.8, s_1^{\times} : 0.1, s_2^{\times} : 0.1\}$ in the left of Figure 2. It is clear that $b^{\times}$ is partially winning since its support contains a winning support $\{s_1^{\times}\}$. If simulations terminate whenever a particle reaches any (certified) winning support, the resulting satisfaction probability of $b^{\times}$ is 0.1 by the belief mass on $\{s_1^{\times}\}$. However, if simulations do not terminate early and all particles continue exploring, taking action a leads to the belief $\{s_0^{\times} : 0.08, s_1^{\times} : 0.73, s_2^{\times} : 0.19\}$, which yields a higher lower bound as 0.73.*

We address reward assignment for particles by augmenting each POMCP tree node $\mathcal{T}(h_{t+k})$, which represents the belief $b^{\times}_{t+k}$ at time t and simulation depth k, with its ground-truth belief support $\Theta(h_{t+k}) = \texttt{Supp}(b^{\times}_{t+k})$. This support is directly recoverable from the history. Intuitively, the search tree simulates BSMDP: each particle path induces a belief-support trace determined only by the history, allowing belief supports to be recovered without explicitly tracking belief distributions. Hence, for each particle $s^{\times}_{t+k}$, we assign reward 1 if the associated belief support $\Theta(h_{t+k})$ is in $\hat{\mathcal{W}}^{\times}$, or if it is in $\hat{\mathcal{W}}^{\times}_{\text{PW}}$ and the particle lies in a certified winning support $\Theta \in \hat{K}_{b^{\times}_{t+k}}$. All remaining particles receive reward 0.

To handle termination for a POMCP tree node with a certified partially winning support, the Upper Confidence Bound (UCB) rule evaluates both terminal and exploratory

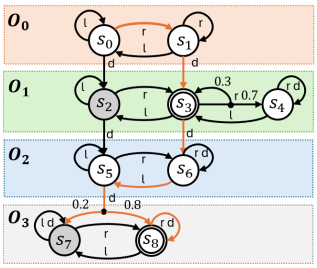

Figure 4: Motivating Toy Example.

options. Consider, for instance, a belief $b^{\times}$ with support $\{s_3^{\times}, s_4^{\times}, s_5^{\times}, s_6^{\times}\}$, as shown on the right of Figure 2. As discussed in Example 3, the tightest lower bound for satisfaction is determined by the maximum belief mass assigned to either $\{s_3^{\times}, s_4^{\times}\}$ or $\{s_6^{\times}\}$. Consequently, the UCB selection mechanism identifies the optimal path by switching between: 1) the terminal policy for $\{s_3^{\times}, s_4^{\times}\}$ if it carries the dominant mass; 2) the terminal policy for $\{s_6^{\times}\}$ if it is more probable; or 3) any available exploration actions if they promise higher satisfaction through further search. Exploration proceeds until a certified winning support is reached or the maximum depth is exceeded. This mechanism allows the planner to terminate at any time during execution by selecting an optimal belief-support policy; the resulting satisfaction probability is then guaranteed by the lower bound $\max_{\Theta \in \hat{K}_{b^{\times}}} b^{\times}(\Theta)$. The full planning algorithm are given in Appendix E.2.

4 Experiments

We implemented our framework in Python[2]. In the following, we first evaluate a motivating toy example demonstrating that existing methods fail even in simple cases, while our approach succeeds in synthesising reliable belief-based policies. We further assess the effectiveness of our method on three standard POMDP benchmarks under various LTL tasks. The implementation details, experimental setup and additional results are provided in Appendix E and F.

4.1 Motivating Toy Example

Consider the POMDP in Figure 4 with initial belief $b_0 = \{s_0 : 1\}$ and actions left (l), right (r), and down (d). The LTL specification is $\varphi = \Box\neg(s_2 \vee s_7) \wedge \Box\Diamond(s_3 \vee s_8)$, which requires visiting s_3 or s_8 infinitely often while always avoiding s_2 and s_7. We compare our approach against optimistic reward shaping [Bouton *et al.*, 2020] and a deterministic FSC variant [Sharan and Burdick, 2014].

Our method. Our analysis identifies a single winning support, $\{s_8\}$. The sound reward shaping assigns to each belief a value equal to its belief mass on $\{s_8\}$. Running our solver obtains the following belief-based policy and history: $b_0 : \{s_0 : 1\} \xrightarrow{r} b_1 : \{s_1 : 1\} \xrightarrow{d} b_2 : \{s_3 : 1\} \xrightarrow{d} b_3 : \{s_6 : 1\} \xrightarrow{l} b_4 : \{s_5 : 1\} \xrightarrow{d} b_5 : \{s_7 : 0.2, s_8 : 0.8\} \xrightarrow{\{s_8\}}$. At b_5, the policy switches to the belief-support policy for $\{s_8\}$ (e.g.

[2]The source code is provided in https://github.com/Safe-Autonomy-and-Intelligence-Lab/POMDP-LTL

alternating r and d), ensuring $\Pr^{\pi}_{b_0}(\varphi) \geq 0.8$. Experimental details and POMCP values are provided in Appendix F.2.

Existing approaches. Using the optimistic reward shaping in [Bouton *et al.*, 2020] with their PBVI solver reports a satisfaction probability of 1 and forces early termination at the belief $b = \{s_3 : 1\}$, preventing sufficient exploration of the reachable belief space. Hence, the solver fails to construct α-vectors valid for successor beliefs. Since the only exit actions encoded in these α-vectors are r and d, with l absent for any possible satisfaction, execution ultimately results in zero LTL satisfaction. We next consider the deterministic one-memory FSC variant in [Sharan and Burdick, 2014], which collapses to a memoryless observation-based policy. However, no FSC provides positive satisfaction for b_0. For observation o_0, action r leads to $\{s_1 : 1\}$, which self-loops under r; action d violates $\Box\neg(s_2 \vee s_7)$; and action l keeps the system at $\{s_0 : 1\}$. Larger memory may help, but selecting a suitable memory size is undecidable, making FSC synthesis brittle in practice. Conversely, our deterministic belief-based policy remains executable and offers a meaningful certified lower bound.

4.2 Benchmark Domains and LTL Objectives

Hallway. We evaluate two Hallway models, HW_1 and HW_2, from [Littman *et al.*, 1995]. The agent moves in an office environment with uncertain location and four orientations, using stochastic forward and left turn actions. A short range sensor reports adjacent walls with some error. Three locations are labelled A, B and C. We consider two LTL objectives: $\varphi_1 = \Diamond A \wedge \Box\neg B$, requiring the agent to eventually reach A while avoiding B at all times, and $\varphi_2 = \Diamond\Box C$, requiring the agent to reach C and remain there indefinitely.

Rock Sample. We follow the rock sample model from [Junges *et al.*, 2021], where a robot moves on an $n \times n$ grid with two rocks, each either good G or bad B with initially unknown quality. The robot can move, sense adjacent cells, and sample rocks. We consider two LTL tasks: $\varphi_3 = \Diamond(G \wedge \Diamond E) \wedge \Box\neg B$, requiring it to first sample a good rock, then reach in an exit state E while never reaching or sampling a bad rock, and $\varphi_4 = \Diamond\Box G$, requiring the robot to collect a good rock and remain there indefinitely.

Grid World. The $Grid(n, n)$ model from [Littman *et al.*, 1995] is an $n \times n$ grid where each cell is labelled as a trap (T), goal 1 (G_1), or goal 2 (G_2). The agent moves left, right, or down, with a 0.2 chance of movement failure, and receives observations of neighbouring cells. We consider three LTL tasks. The first is $\varphi_5 = \Diamond G_1 \wedge \Box\neg T$, requiring the agent to eventually reach G_1 while always avoiding traps. The second is $\varphi_6 = \Box(G_2 \rightarrow \Diamond G_1) \wedge \Box\neg T$, requiring it to always avoid traps and, whenever it reaches G_2, eventually reach G_1. The third is $\varphi_7 = \Box\Diamond G_1 \wedge \Box\neg T$, requiring it to visit G_1 infinitely often while avoiding traps.

4.3 Results and Discussion

Table 1 summarises the performance of our approach across all tasks. For each case, we report the product model details ($|S^{\times}|/|O^{\times}|/|T^{\times}|$), the LTL objective ($\varphi_i$) with its satisfaction probability ($\Pr^{\pi}_{\mathcal{M}}(\varphi)$), the time to construct the sound reward function (SR (s)), and the policy synthesis time using the enhanced POMCP solver (POMCP (s)). Further details on the original models and additional experimental results are provided in Appendix F.4.

Domain	$\lvert S^{\times}\rvert/\lvert O^{\times}\rvert/\lvert T^{\times}\rvert$	SR (s)	POMCP (s)	$\Pr^{\pi}_{\mathcal{M}}(\varphi)$
HW				
HW1 φ_1	180 / 24 / 624	0.015	74.967	0.254
φ_2	180 / 24 / 684	0.020	52.759	0.707
HW2 φ_1	276 / 24 / 975	0.547	64.356	0.247
φ_2	276 / 24 / 1067	0.743	44.643	0.464
RS				
4 φ_3	1024 / 112 / 7936	42.231	79.102	0.453
φ_4	768 / 112 / 6208	0.078	91.788	0.469
5 φ_3	1600 / 132 / 12480	58.246	121.315	0.326
φ_4	1200 / 132 / 9760	0.114	120.307	0.455
7 φ_3	3136/ 244 / 24640	105.475	129.259	0.180
φ_4	2352 / 244 / 19264	0.210	160.291	0.003
Grid				
50 φ_5	7500 / 1250 / 30990	10.987	82.198	0.334
φ_6	7500 / 1250 / 30990	10.597	0.246	1.000
φ_7	5000 / 1250 / 20660	1.843	67.959	0.530
100 φ_5	30000 / 5000 / 120840	81.361	99.407	0.272
φ_6	30000 / 5000 / 120840	82.334	0.248	1.000
φ_7	20000 / 5000 / 80560	30.603	107.366	0.308
120 φ_5	43200 / 7200 / 173580	191.431	182.667	0.000
φ_6	43200 / 7200 / 173580	188.264	0.267	1.000
φ_7	28800 / 7200 / 115720	66.314	183.521	0.058

Table 1: Numerical Results for POMDP-LTL Policy Synthesis.

The results show that our method synthesises POMDP-LTL policies effectively and efficiently, scaling to product models with over forty thousand states, with both sound reward construction and planning completing in about 190 seconds on a single CPU thread. Our approach is particularly effective for LTL tasks such as the reach-persistence objective φ_4 and the recurrence objective φ_7. Although the underlying state-space AMECs grow proportionally with the model size, they remain confined to a small region of the overall state space, which enables efficient construction of sound reward signals. For example, in the Rock Sample domain, as the model scales, the construction time for the task φ_4 increases only from 0.078 to 0.210 seconds.

For the POMCP results, although theory ensures $\Pr_{\mathcal{M}}(\varphi_5) \geq \Pr_{\mathcal{M}}(\varphi_7)$ as every trace satisfying φ_5 also satisfies φ_7, we observed $\Pr_{\mathcal{M}}(\varphi_5) < \Pr_{\mathcal{M}}(\varphi_7)$ in practice. This mismatch arises from the differing sizes of their LDBAs: $|\mathcal{A}_{\varphi_5}| = 3$ versus $|\mathcal{A}_{\varphi_7}| = 2$. The larger automaton for φ_5 results a larger product POMDP, requiring more simulations and deeper search for similar approximation quality.

5 Conclusion

This paper presented the first sound and practical anytime belief-based policy synthesis algorithm for full LTL in POMDPs. We approximate undecidable quantitative POMDP-LTL synthesis by seeking policies that maximise verifiable success. Central to this approach is a sound belief-dependent reward-shaping mechanism, integrated into an enhanced POMCP algorithm for reliable policy synthesis. For future work, we plan to extend this sound reward mechanism to a model-free setting, enabling safe POMDP reinforcement learning under LTL objectives.

References

[Ahmadi *et al.*, 2020] Mohamadreza Ahmadi, Rangoli Sharan, and Joel W Burdick. Stochastic finite state control of POMDPs with LTL specifications. *arXiv preprint arXiv:2001.07679*, 2020.

[Baier and Katoen, 2008] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.

[Baier *et al.*, 2008] Christel Baier, Nathalie Bertrand, and Marcus Größer. On decision problems for probabilistic Büchi automata. In *Proceedings of the International Conference on Foundations of Software Science and Computational Structures*, pages 287–301. Springer, 2008.

[Bouton *et al.*, 2020] Maxime Bouton, Jana Tumova, and Mykel J Kochenderfer. Point-based methods for model checking in partially observable Markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10061–10068, 2020.

[Carr *et al.*, 2020] Steven Carr, Nils Jansen, and Ufuk Topcu. Verifiable RNN-based policies for POMDPs under temporal logic constraints. *arXiv preprint arXiv:2002.05615*, 2020.

[Carr *et al.*, 2021] Steven Carr, Nils Jansen, and Ufuk Topcu. Task-aware verifiable RNN-based policies for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, 72:819–847, 2021.

[Chatterjee *et al.*, 2007] Krishnendu Chatterjee, Laurent Doyen, Thomas A Henzinger, and Jean-François Raskin. Algorithms for ω-regular games with imperfect information. *Logical Methods in Computer Science*, 3, 2007.

[Chatterjee *et al.*, 2010] Krishnendu Chatterjee, Laurent Doyen, Hugo Gimbert, and Thomas A Henzinger. Randomness for free. In *Proceedings of the International Symposium on Mathematical Foundations of Computer Science*, pages 246–257. Springer, 2010.

[Chatterjee *et al.*, 2015] Krishnendu Chatterjee, Martin Chmelik, Raghav Gupta, and Ayush Kanodia. Qualitative analysis of POMDPs with temporal logic specifications for robotics applications. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 325–330. IEEE, 2015.

[Chatterjee *et al.*, 2016] Krishnendu Chatterjee, Martin Chmelik, and Mathieu Tracol. What is decidable about partially observable Markov decision processes with ω-regular objectives. *Journal of Computer and System Sciences*, 82(5):878–911, 2016.

[Coulom, 2006] Rémi Coulom. Efficient selectivity and backup operators in Monte-Carlo tree search. In *Proceedings of the International Conference on Computers and Games*, pages 72–83. Springer, 2006.

[Junges *et al.*, 2021] Sebastian Junges, Nils Jansen, and Sanjit A Seshia. Enforcing almost-sure reachability in POMDPs. In *Proceedings of the International Conference on Computer Aided Verification*, pages 602–625. Springer, 2021.

[Kaelbling *et al.*, 1995] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Partially observable markov decision processes for artificial intelligence. In *Proceedings of the International Workshop on Reasoning with Uncertainty in Robotics*, pages 146–163. Springer, 1995.

[Kalagarla *et al.*, 2022] Krishna C Kalagarla, Kartik Dhruva, Dongming Shen, Rahul Jain, Ashutosh Nayyar, and Pierluigi Nuzzo. Optimal control of partially observable markov decision processes with finite linear temporal logic constraints. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 949–958. PMLR, 2022.

[Kalagarla *et al.*, 2024] Krishna C Kalagarla, Dhruva Kartik, Dongming Shen, Rahul Jain, Ashutosh Nayyar, and Pierluigi Nuzzo. Optimal control of logically constrained partially observable and multiagent Markov decision processes. *IEEE Transactions on Automatic Control*, 70(1):263–277, 2024.

[Křetínskỳ *et al.*, 2018] Jan Křetínskỳ, Tobias Meggendorfer, Salomon Sickert, and Christopher Ziegler. Rabinizer 4: From LTL to your favourite deterministic automaton. In *Proceedings of International Conference on Computer Aided Verification*, pages 567–577. Springer, 2018.

[Littman *et al.*, 1995] Michael L Littman, Anthony R Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *Machine Learning Proceedings 1995*, pages 362–370. Elsevier, 1995.

[Liu *et al.*, 2021] Jason Liu, Eric Rosen, Suchen Zheng, Stefanie Tellex, and George Konidaris. Leveraging temporal structure in safety-critical task specifications for POMDP planning. In *Proceedings of the RSS Workshop Robot. People*, 2021.

[Madani *et al.*, 1999] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. *Aaai/iaai*, 10(315149.315395), 1999.

[Pineau *et al.*, 2003] Joelle Pineau, Geoff Gordon, Sebastian Thrun, et al. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 3, pages 1025–1032, 2003.

[Pnueli, 1977] Amir Pnueli. The temporal logic of programs. In *Proceedings of 18th Annual Symposium on Foundations of Computer Science*, pages 46–57. ieee, 1977.

[Sebastian *et al.*, 2005] Thrun Sebastian, Burgard Wolfram, and Fox Dieter. *Probabilistic Robotics*. The MIT Press, Cambridge, 2005.

[Sharan and Burdick, 2014] Rangoli Sharan and Joel Burdick. Finite state control of POMDPs with LTL specifications. In *Proceedings of 2014 American Control Conference*, pages 501–508. IEEE, 2014.

[Sickert *et al.*, 2016] Salomon Sickert, Javier Esparza, Stefan Jaax, and Jan Křetínskỳ. Limit-deterministic Büchi

automata for linear temporal logic. In *Proceedings of the International Conference on Computer Aided Verification*, pages 312–332. Springer, 2016.

[Silver and Veness, 2010] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. *Proceedings of the Advances in Neural Information Processing Systems*, 23, 2010.

[Vardi and Wolper, 1994] Moshe Y Vardi and Pierre Wolper. Reasoning about infinite computations. *Information and computation*, 115(1):1–37, 1994.