Proceedings of Machine Learning Research vol vvv:1–14, 2025

Monte Carlo Tree Search for Co-safe Linear Inequaility Temporal Logic on Partially Observable Markov Decision Processes

Zetong Xuan Z.XUAN@UFL.EDU and **Yu Wang** YUWANG1@UFL.EDU
University of Florida, Gainesville, FL 32611, USA

Abstract

Partially observable Markov decision process (POMDP) models real-world autonomous systems that have states that are not fully observable due to system/environment uncertainty. Planning for POMDP requires the posterior estimation of the distribution of hidden states, that is, belief states. Task specifications on these belief states are more expressive than the task specifications on hidden states and are needed in many applications, such as drone surveillance. Thus, we propose a POMDP planning problem with co-safe inequality linear temporal logic (sc-iLTL) objectives. Sc-iLTL defines complex rule-based tasks that consider the specification of belief states where each atomic proposition is a linear inequality on belief space. Our solution is to transform the time-dependent sc-iLTL objective into a reachability problem by constructing the product of the belief MDP and a deterministic finite automaton (DFA) built from the sc-iLTL objective. Monte Carlo Tree Search (MCTS) finds the optimal policy on the product belief MDP, and then the optimal policy on POMDP can be recovered.

1. Introduction

The states of real-world autonomous systems such as self-driving cars Hubmann et al. (2018) and robots Bozkurt et al. (2021) are not fully observable due to system/environment uncertainty (e.g., sensor noise). For these applications, a widely used mathematical model for planning is the partially observable Markov decision process (POMDP) Kaelbling et al. (1998). A POMDP generalizes a Markov decision process (MDP), whose states are partially observable through a probabilistic relation to a set of observations. We call these states hidden states.

Estimating the distribution of the hidden states is a well-motivated approach for planning in POMDPs. Since we do not have direct access to the hidden states, many POMDP solvers base their control actions on posterior state estimation, i.e., beliefs. A belief state is the optimal estimation of the current distribution of hidden states, given all previous actions and observations. Thus, the belief can represent property like the probability of not visiting a dangerous hidden state (safety) or the probability of identifying a hidden state (confidence).

In many cases, the task specification on these belief states is more expressive than the task specification on hidden states. A reward function defined on hidden states always results in a linear reward function on belief states. On the contrary, a reward function defined on the belief state may not be representable via a reward function defined on hidden states. An example can be a reachability problem on the belief MDP, which may not have an equivalent reward function defined on hidden states. The application for such a reachability problem shall be extended to safety and other tasks shown in Section 3.1 and 6.

Linear inequality temporal logic (iLTL) Kwon and Agha (2004) can be used to specify complex rule-based tasks that consider the specification on belief states. Linear temporal logic (LTL) is a symbolic language to specify high-level tasks like reaching a sequence of goals or ordering a set of

events. We consider an iLTL where each automatic proposition is a linear inequality on the belief space. The planning problem with the iLTL objective is to find the optimal policy that maximizes the probability of satisfying the given iLTL objective. Namely, we want to find the optimal policy that maximizes the satisfaction probability of an objective that specifies the behavior of the path of belief states.

Memory dependency on the optimal policy is a challenge when planning for iLTL objectives. We transform the planning for the iLTL objective into planning for reachability that allows a memoryless policy to be optimal. We construct the product belief MDP from the belief MDP and a deterministic finite automaton (DFA). DFA is a finite state machine, and each DFA state represents the satisfaction status of the iLTL objective. Each state of the product belief MDP contains both a belief state and a DFA state. The transition in the product belief MDP represents both the change in belief and the change in the completion of the iLTL objective. The optimal policy that achieves the reachability objective on the product belief MDP would be the optimal policy for the iLTL objective on the belief MDP and the POMDP.

We use Monte Carlo Tree Search (MCTS) to find the optimal policy on the continuous-state belief MDPs. The planning for continuous-state MDPs is hard Lavaei et al. (2022). Many solving techniques simplify the planning in continuous state space by utilizing the fact that i) the belief state transition obeys the belief update, ii) the value function on the belief space is piece-wise linear or convex if the reward function is defined on hidden states. However, as we define the objective on the belief states but not hidden states, we can not use a solving technique like point-based methods Pineau et al. (2003); Bouton et al. (2020) that benefits from the second fact. Thus, we consider solving techniques like real-time dynamic programming Geffner and Bonet (1998); Bonet and Geffner (2009) and MCTS Silver and Veness (2010), which do not rely on piece-wise linear or convex assumptions. We prefer MCTS over real-time dynamic programming as the complexity is independent of the size of the belief space, considering the latter method discretizes the belief space.

We propose an MCTS method to derive the optimal policy for an iLTL objective on POMDP. By constructing the belief MDP of the POMDP, we first lift the planning problem to the belief space. Then, we build the product belief MDP of the belief MDP and the DFA of the temporal logic. The product transforms the time-dependent iLTL objective to a reachability objective on the product belief MDP. Finally, we propose an MCTS method to find the optimal memoryless policy, which can then be translated back to the optimal memory-based policy on the POMDP.

2. Preliminaries

This section formally defines POMDP, history MDP, and belief MDP Li (2024). These three models capture the same dynamics, that is, given an action, the probabilistic transition of the hidden states, and the probabilistic outcome of receiving an observation. The history MDP is a way to represent memory, where every state contains all previous actions and observations. Considering the exponential growth of history states with respect to time, the belief MDP serves as an equivalent of history MDP with fixed but continuous state space. In section 3, we formulate our problem using belief MDP, and then in section 4.3, we solve it in the form of history MDP.

2.1. POMDP

We model an autonomous system's dynamics in the unknown and partially observable environment by a POMDP, where the underlying dynamics is an MDP, but the control can only depend on observations probabilistically related to the hidden states.

Definition 1 *A POMDP is a tuple $\mathcal{M}_\mathcal{P} = (S, A, T_\mathcal{P}, s_{\text{init}}, \Omega, O)$, where i) S is a finite set of hidden states, ii) A is a finite set of actions, iii) Ω is a finite set of observations, where $\Omega(s)$ denotes the set of possible observations in the state $s \in S$, iv) $T_\mathcal{P} : S \times A \times S \to [0, 1]$ is the transition probability function such that for all $s \in S$, we have $\sum_{s' \in S} T_\mathcal{P}(s, a, s') = 1$, if $a \in A(s)$ and $\sum_{s' \in S} T_\mathcal{P}(s, a, s') = 0$, if $a \notin A(s)$, v) $O : S \times \Omega \to [0, 1]$ is the observation probability function such that for all $s \in S$, we have $\sum_{o \in \Omega} O(s, o) = 1$, vi) s_{init} is the initial distribution of the hidden state.*

We call a sequence of states $\sigma_\mathcal{P} : \mathbb{N} \to S$ a *path* of the POMDP if for any $t \in \mathbb{N}$, there exists $a \in A$ such that $T_\mathcal{P}(\sigma_\mathcal{P}(t), a, \sigma_\mathcal{P}(t + 1)) > 0$. In addition, A *history* is a sequence of actions and observations, $h_t = \{a_0, o_0, ...a_{t-1}, o_{t-1}\}$ and $h_0 = \emptyset$.

2.2. Belief MDP and history MDP

The states are not directly observable in a POMDP. But, from the history of observations and actions, one can derive a probabilistic estimation of the states called a belief $b \in B := \Delta(S)$, here we denote the set of probability distributions on S by $\Delta(S)$. We denote the mapping from history h_t to the belief state b_t as $\mathcal{B}(h)$. It can be derived inductively as follows. For all $s \in S$, $b_0(s) = s_{\text{init}}(s)$ and

$$b_t(s) = \frac{O(s, o_t) \sum_{s' \in S} b_{t-1}(s') T_\mathcal{P}(s', a_t, s)}{\sum_{s \in S} O(s, o_t) \sum_{s' \in S} b_{t-1}(s') T_\mathcal{P}(s', a_t, s)} \tag{1}$$

The transition of belief states with respect to actions can be treated as a continuous state MDP called *belief MDP*.

Definition 2 *The belief MDP $\mathcal{M}_\mathcal{B}$ of the POMDP $\mathcal{M}_\mathcal{P} = (S, A, T_\mathcal{P}, s_{\text{init}}, \Omega, O)$ is defined by $\mathcal{M}_\mathcal{B} = (B, A, T_\mathcal{B}, b_0)$ where i) $B := \Delta(S)$ is the continuous belief space, ii) A is the finite set of actions, iii) $b_0 = s_{\text{init}}$, iv) $T_\mathcal{B} : B \times A \times B \to [0, 1]$ is the transition probability function*

$$T_\mathcal{B}(b, a, b') = \sum_{o \in O} \sum_{s' \in S} \sum_{s \in S} \eta(b, o, b') O(s', o) T_\mathcal{P}(s, a, s') b(s) \tag{2}$$

with

$$\eta(b, o, b') = \begin{cases} 1, & \text{if the belief update for } b, o \text{ by (1) returns } b' \\ 0, & \text{otherwise.} \end{cases}$$

A sequence of belief states $\sigma : \mathbb{N} \to \Delta(S)$ is a *belief path* of the belief MDP if for any $t \in \mathbb{N}$, there exists $a \in A$ such that $T_\mathcal{B}(\sigma(t), a, \sigma(t + 1)) > 0$.

Whereas the belief MDP has a continuous state space, we can represent the same dynamics using the transition between discrete history states. This representation is more friendly to MCTS.

Definition 3 *The history MDP $\mathcal{M}_{\mathcal{H}}$ of the POMDP $\mathcal{M}_{\mathcal{P}} = (S, A, T_{\mathcal{P}}, s_{\text{init}}, \Omega, O)$ is defined by $\mathcal{M}_{\mathcal{H}} = (H, A, T_{\mathcal{H}}, o_0)$ where i) H is the discrete history state space containing all the possible history, ii) A is the finite set of actions, iii) $T_{\mathcal{H}} : H \times A \times H \to [0, 1]$ is the transition probability function $T_{\mathcal{H}}(h, a, hao) = \sum_{s' \in S} \sum_{s \in S} O(s', o) T_{\mathcal{P}}(s, a, s') \mathcal{B}(h)(s)$ iv) $\emptyset$ is the initial history state.*

A deterministic memoryless[1] *policy* is $\pi(b_t) \in A$, which uses belief as input and outputs an action. The policy $\pi(\mathcal{B}(h_t))$ returns an action give a history state.

3. Problem Formulation and Main Result

Now, we introduce co-safe linear inequality temporal logic (sc-iLTL) specifying POMDP tasks. First, we present an example to show that some tasks can be specified on the belief path σ but are hard to specify on the state path $\sigma_{\mathcal{P}}$. Then, we formally define our planning objective and show the main result of solving it.

3.1. Motivating example

The common way we specify the task on POMDP is through hidden states, which may not be expressive enough for a given task.

Example 1 *Consider a probing task in a grid world (inspired by Svoreňová et al. (2015)). The drone needs to locate a ground target on the grid using an imperfect sensor. The **sensor** only provides the respective quadrant of the target within the field of view, that is, adjacent to or under the drone. The set of observations is $\Omega = \{\text{SW}, \text{NW}, \text{NE}, \text{SE}, \text{None}\}$, where the first four observations stand for the respective quadrant and "None" means the target is not in the field of view. Suppose the target is at the adjacent grid north of the drone, then the sensor returns "NE" or "NW" with equal probability, and if the target is under the drone, then the sensor returns an observation in $\{\text{SW}, \text{NW}, \text{NE}, \text{SE}\}$ with equal probability. We can define achieving such a task as increasing the maximum element of the belief state to a confidence level c.*

$$\|b\|_\infty \geq c$$

It is difficult to specify this task using objectives defined on hidden states; meanwhile, it can be specified using objectives defined on belief states. Traditional planning objectives usually use reward $r(s, a) : S \times A \to \mathbb{R}$ or LTL with atomic propositions defined on hidden states. However, even if the drone is right above the target, $\|b\|_\infty$ may still be below the confidence c. This example requires the drone to approach the target in a specific way that can increase $\|b\|_\infty$ by belief update (1).

3.2. Sc-iLTL

We express this kind of task by an iLTL formula constructed via atomic propositions defined on belief states. The definition and verification of the iLTL formula Kwon and Agha (2004) are similar to the standard LTL formula Baier and Katoen (2008) except for atomic propositions. An iLTL formula is derived recursively from the rules

$$\varphi ::= \text{true} \mid \text{ineq} \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \bigcirc\varphi \mid \varphi_1 \mathcal{U} \varphi_2, \text{ ineq} \in \Lambda \tag{3}$$

1. Here, memoryless stands for only using current belief as input. We only consider deterministic policy in this work

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

In this work, we focus on the co-safe LTL formula, which can be verified via finite prefixes. We say that the iLTL formula φ is co-safe Lacerda et al. (2014) 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'. Sc-iLTL requires $\neg$ only to apply directly to atomic propositions, and the formula only uses the temporal operators $\bigcirc$, $\mathcal{U}$, and $\Diamond$. Let $\sigma : \mathbb{N} \to \Delta(S)$ be a belief path of the POMDP. The satisfaction (denoted by $\models$) of an sc-iLTL formula follows the standard rule for co-safe LTL Lacerda et al. (2014) except for

$$\sigma \models \text{ineq iff } p^T \sigma_0 > c \text{ with } p \in \mathbb{R}^{|S|}, c \in \mathbb{R}.$$

We translate sc-iLTL to deterministic finite automata (DFAs). *A DFA is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where i) Q is a finite set of automaton states, ii) Σ is a finite set of alphabets, iii) $\delta : Q \times \Sigma \to 2^Q$ is a (partial) transition function, iv) $q_{\text{init}} \in Q$ is the initial automaton state, v) $F \subseteq Q$ is a set of final automaton states.*

Given a sc-iLTL objective φ, we can construct an DFA $\mathcal{A}_\varphi$ (with labels $\Sigma = 2^\Lambda$) such that a belief path $\sigma \models \varphi$ if and only if σ is accepted by the DFA $\mathcal{A}_\varphi$. The DFA verifies the belief path by a labeling function $L : B \to 2^\Lambda$, where $\text{ineq} \in L(b)$ if and only if ineq holds on b. The labeling on the belief path moves the automation states. A belief 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_0 to a final state $q \in F$.

> **Problem Formulation:** For a POMDP $\mathcal{M}_\mathcal{P} = (S, A, T_\mathcal{P}, s_{\text{init}}, \Omega, O)$ and a sc-iLTL objective φ specifing the belief path σ. Find the optimal policy π that maximizes the satisfaction probability $Pr_\pi(\sigma \models \varphi)$.

Solving this planning problem with an objective specifying the belief path is challenging. By constructing the product of the belief MDP and the DFA, we can transform the planning into a maximum reachability problem on a continuous state belief MDP. However, solving the reachability problem for a continuous state MDP is hard.

We approach the reachability problem from the history MDP point of view. Since we specify the behavior of a finite prefix, we are dealing with a finite horizon. We only need to argue a finite number of belief states that are reachable from b_0. Thus, our problem can be treated as a reachability problem on a history MDP with finite state space.

MCTS Kocsis and Szepesvári (2006); Silver and Veness (2010) gives us the ability to solve the maximum reachability problem on the history MDP. Specifically, given a finite horizon POMDP, the value function estimated by MCTS converges in probability to the optimal value function. We can apply MCTS to our product belief MDP and find the maximal reachability probability with a convergence guarantee.

Theorem 4 *Our algorithm 1 finds the optimal policy on the POMDP that maximizes the satisfaction probability of the given sc-iLTL objective.*

4. MCTS for iLTL

In this section, we introduce our algorithm to solve the iLTL planning problem on POMDPs. First, we transform the iLTL planning problem into a reachability problem by constructing the product

belief MDP and show that both problems share the same optimal policy. Then, We apply MCTS to find the optimal policy.

4.1. Product belief MDP

We translate the planning problem for the POMDP into a reachability problem on the product belief MDP.

Definition 5 *A product belief MDP* $\mathcal{M}_{\mathcal{B}}^{\times} = (B^{\times}, A^{\times}, P^{\times}, b_0^{\times}, F^{\times})$ *of an belief MDP* $\mathcal{M}_{\mathcal{B}} = (B, A, T_{\mathcal{B}}, b_0)$ *and an DFA* $\mathcal{A} = (Q, \Sigma, \delta, q_{\text{init}}, F)$ *is defined by i) the set of states* $B^{\times} = B \times Q \cup \text{sink}$*, ii) the set of actions* $A^{\times} = A$*, iii) the the set of final states* $F^{\times} = \{\langle b, q\rangle \in B^{\times} | q \in F\}$*, iv) the initial product belief state* $b_0^{\times} = \langle b_0, q_{\text{init}}\rangle$*, v) the transition probability function*

$$T_{\mathcal{B}}^{\times}(b^{\times}, a, b^{\times'}) = \begin{cases} T_{\mathcal{B}}(b, a, b') & b^{\times} = \langle b, q\rangle \notin F^{\times},\ b^{\times'} = \langle b', q'\rangle,\ q' = \delta(q, L(b)) \\ 1 & b^{\times} \in F^{\times},\ b^{\times'} = \text{sink} \\ 1 & b^{\times} = \text{sink},\ b^{\times'} = \text{sink} \\ 0 & \text{else.} \end{cases} \tag{4}$$

The transitions of the product MDP $\mathcal{M}_{\mathcal{B}}^{\times}$ are derived by combining the transitions of the belief MDP $\mathcal{M}_{\mathcal{B}}$ and the DFA $\mathcal{A}$. The path on the product belief MDP falls to the sink once $F^{\times}$ is met. Each belief path $\sigma \models \varphi$ will have a corresponding path $\sigma^{\times}$ on the product belief MDP $\mathcal{M}^{\times}$ visiting $F^{\times}$ in finite steps and vice versa. Here, we use $\sigma^{\times} \models \Diamond F^{\times}$ to describe the event of visiting $F^{\times}$ in finite steps.

4.2. Reachability problem on the product belief MDP

To solve the reachability problem of the product belief MDP, one may consider building the following reward function

$$r(b^{\times}) = \begin{cases} 1 & b^{\times} \in F^{\times} \\ 0 & \text{else} \end{cases}, \tag{5}$$

such that the value function $V_{\pi^{\times}}(b_0^{\times}) = \mathbb{E}_{\pi} \sum_{i=0}^{\infty} r(b_i^{\times})$, which is the expected return of the cumulative reward along all paths starting at $b^{\times}$, equals the reachability probability.

Lemma 6 *Given a policy* $\pi^{\times}$*, belief state* b *and DFA state* q*,*

$$V_{\pi^{\times}}(\langle b, q\rangle) = Pr_{\pi^{\times}}(\sigma^{\times} \models \Diamond F^{\times} | \sigma_0^{\times} = \langle b, q\rangle). \tag{6}$$

The Lemma holds by the following Bellman equation. Given a policy, the value function satisfies the recursive formulation as the Bellman equation,

$$\begin{cases} V_{\pi^{\times}}(b^{\times}) = r(b^{\times}) + \sum_{b^{\times'}} P_{\mathcal{B}}^{\times}(b^{\times}, \pi(b^{\times}), b^{\times'}) V_{\pi^{\times}}(b^{\times'}) & b^{\times} \neq \text{sink} \\ V_{\pi^{\times}}(b^{\times}) = 1 & b^{\times} = \text{sink}. \end{cases} \tag{7}$$

The optimal value function satisfies the Bellman optimality equation

$$\begin{cases} V^*(b^{\times}) = r(b^{\times}) + \max_{a \in A} \sum_{b^{\times'}} T_{\mathcal{B}}^{\times}(b^{\times}, a, b^{\times'}) V^*(b^{\times'}) & b^{\times} \neq \text{sink} \\ V^*(b^{\times}) = 1 & b^{\times} = \text{sink} \end{cases} \tag{8}$$

The policy $\pi^\times(\langle b, q\rangle)$ requires us to update the belief states and the DFA state at the same time. Given current belief and DFA state b_t, q_t, the action is $a_t = \pi^\times(\langle b_t, q_t\rangle)$.

Lemma 7 *Given sc-iLTL objective φ and POMDP $\mathcal{M}_\mathcal{P}$, the optimal policy $\pi^\times(\langle b, q\rangle)$ on $\mathcal{M}^\times_\mathcal{B}$ maximizing reachability probability is the optimal policy $\pi^\times(\langle b, q\rangle)$ on $\mathcal{M}_\mathcal{P}$ maximizing satisfication probability of φ.*

Proof Given DFA $\mathcal{A}_\varphi$ with initial DFA state $q_0 = q_{\text{init}}$, following the same policy $\pi^\times$, the satisfaction probability on POMDP equals the reachability probability on product belief MDP, $Pr_{\pi^\times}(\sigma \models \varphi) = Pr_{\pi^\times}(\sigma^\times \models \Diamond F^\times | \sigma^\times_0 = \langle \sigma_0, q_0\rangle)$. Since each belief path σ has a unique corresponding path $\sigma^\times$ where $\sigma^\times_0 = \langle \sigma_0, q_0\rangle$. Lemma 6 and (13) finishes the proof. ■

4.3. Monte-Carlo tree search

Here, we use MCTS to find the optimal policy on the product belief MDP. We use a discrete history state to represent a continuous belief state similar to Silver and Veness (2010). In this way, MCTS only searches a finite number of belief states that are reachable from b_0.

Algorithm 1 MCTS for Product Belief MDP

Procedure Search($h^\times = \langle h, q\rangle$)
  repeat
    if $h = \emptyset$ **then**
      $s \sim s_{\text{init}}$
    else
      $s \sim \mathcal{B}(h)$
    end if
    Simulate($s, h^\times, 0$)
  until Timeout()
  return $\arg\max_a \hat{V}(h^\times a)$
end procedure

Procedure Rollout($s, h^\times, depth$)
  if $depth \geq d_{\text{max}}$ **or** $h^\times = \text{sink}$ **then**
    return 0
  end if
  $a = \pi_{rollout}(\mathcal{B}(h))$
  $s' \sim T_\mathcal{P}(s, a, \cdot)$
  $h^{\times\prime} \sim P^\times_\mathcal{H}(h^\times, a, \cdot)$
  return $r(\mathcal{B}(h)) + \text{Rollout}(s', h, \text{depth} + 1)$
end procedure

Procedure Simulate($s, h^\times, depth$)
  if $depth \geq d_{\text{max}}$ **or** $h^\times = \text{sink}$ **then**
    return 0
  end if
  if $h^\times \notin G$ **then**
    for all $a \in A$
      $G(h^\times, a)$
      $\leftarrow \langle \hat{V}_{init}(h^\times), N_{init}(h^\times), \mathcal{B}(h^\times)\rangle$
    end for
    return Rollout($s, h^\times, depth$)
  end if
  $a \leftarrow \arg\max_d \left(\hat{V}(h^\times, d) + c \cdot \sqrt{\frac{\log N(h^\times)}{N(h^\times, d)}}\right)$
  $s' \sim T_\mathcal{P}(s, a, \cdot)$
  $h^{\times\prime} \sim P^\times_\mathcal{H}(h^\times, a, \cdot)$
  R
  $\leftarrow r(\mathcal{B}(h)) + \text{Simulate}(s', h^{\times\prime}, depth + 1)$
  $N(h^\times) \leftarrow N(h) + 1$
  $N(h^\times, a) \leftarrow N(h, a) + 1$
  $\hat{V}(h^\times, a) \leftarrow \hat{V}(h, a) + \frac{R - \hat{V}(h^\times, a)}{N(h^\times, a)}$
  return R
end procedure

We use an equivalent product history state $h^\times = \langle h, q\rangle$ to represent each product belief state $b^\times = \langle \mathcal{B}(h), q\rangle$ by (1). The transition between the product history states is given as

$$T^\times_{\mathcal{H}}(h^\times, a, h^{\times\prime}) = \begin{cases} T^\times_{\mathcal{B}}(b^\times, a, b^{\times\prime}) & h^\times = \langle h, q\rangle,\ h^{\times\prime} = \langle hao, q'\rangle,\ q' = \delta(q, L(\mathcal{B}(h))), \\ & b^\times = \langle \mathcal{B}(h), q\rangle,\ b^{\times\prime} = \langle \mathcal{B}(hao), q'\rangle \\ 1 & h^\times = \langle h, q\rangle, q \in F,\ h^{\times\prime} = \text{sink} \\ 1 & h^\times = \text{sink},\ h^{\times\prime} = \text{sink} \\ 0 & \text{else}, \end{cases} \tag{9}$$

MCTS extends the UCB1 method from the Bandit problem to a decision tree. The decision tree is constructed based on the Bellman optimality equation (13). Each state node $G(h^\times) = \langle N(h^\times), \hat{V}(h^\times), \mathcal{B}(h^\times)\rangle$ [2] contains $N(h^\times)$ counts the number of times that $h^\times$ has been visited, $\hat{V}(h^\times)$ is the value estimation of $\mathcal{B}(h^\times)$ by the mean return of all simulations starting with $\mathcal{B}(h^\times)$. New nodes are initialized to $\langle \hat{V}_{init}(h^\times), N_{init}(h^\times), \mathcal{B}(h^\times)\rangle$ if the knowledge is available, and to $\langle 0, 0, \mathcal{B}(h^\times)\rangle$ otherwise. The children to state node $G(h^\times)$ are action nodes $G(h^\times a) = \langle N(h^\times, a), \hat{V}(h^\times a), \mathcal{B}(h^\times)\rangle$. The children to the action node are state node $G(h^{\times\prime})$ where $h^{\times\prime}$ is reachable from $h^\times$ via action a.

Each simulation on the tree starts in a hidden state $s \sim \mathcal{B}(h^\times)$. In the simulation, if we have knowledge of all the child nodes, actions are selected by $\arg\max_a \{\hat{V}(h^\times a) + c\sqrt{\frac{\log N(^\times)}{N(h^\times, a)}}\}$. Otherwise, a rollout policy $\pi_{rollout}$ (uniform random action selection) is applied. Each simulation will add one new node to the tree. When the search is complete, the algorithm returns the optimal action for $h^\times$.

5. Guaranteed Convergence to the Optimal Policy

The key challenge to finding the optimal policy is the large number of historical states, which grows exponentially with respect to the length of the path. MCTS handles such a large state space as it uses best-first search; thus, its complexity is not related to the size of the state space.

Further, the finding of the optimal policy on POMDP for the iLTL objective is guaranteed as follows. First, MCTS finds the optimal policy for the reachability problem on the product belief MDP. Then, the optimal policy on the original POMDP is recovered.

Theorem 8 *Apply MCTS to the product belief MDP $\mathcal{M}^\times_{\mathcal{B}}$, given a history state $h^\times$, the probability of MCTS failed to return the optimal action on the history state convergence to zero as*

$$\lim_{N(h^\times)\to\infty} Pr(\pi(\mathcal{B}(h^\times) \neq a^*(h^\times)) = 0 \tag{10}$$

and the estimation error of the value function is bounded by

$$|\mathbb{E}[V^*(\mathcal{B}(h^\times)) - \hat{V}(h^\times)]| \leq O\big(\frac{\log(N(h^\times)}{N(h^\times)}\big) \tag{11}$$

Proof This proof can be done by applying (Silver and Veness, 2010, Theorem 1) to the product belief MDP. We provided a detailed version of the proof in the appendix. Applying the UCB1

2. Here we use $\mathcal{B}(h^\times)$ to represent $\mathcal{B}(h)$ where h is inside $h^\times$

method to a non-stationary bandit problem shows the guaranteed convergence to an optimal value function and optimal action. Suppose we fix $V(b^{\times'})$, then the bandit problem is stationary and can be solved by the UCB1 method. However, when we search the root, value estimations on the follow-up nodes also change, so we call this non-stationary. Kocsis and Szepesvári (2006) transform an MDP problem into a non-stationary bandit problem. They utilize the convergence results on applying the UCB1 method to the non-stationary bandit problem in Auer et al. (2002) to get the result on MDPs.

MCTS finds the optimal policy for the reachability problem on the product belief MDP. Our product belief MDP alines the analysis in Kocsis and Szepesvári (2006) considering an undiscounted bounded reward function. (Kocsis and Szepesvári, 2006, Theorem 3,5) holds. MCTS solves the reachability problem of the product belief MDP. ■

Proof of main result

Theorem 8 guarantees MCTS return the optimal policy $\pi^{\times}(\langle b, q\rangle)$. Lemma 7 shows the optimal policy on product belief MDP and is the optimal policy on POMDP. Thus, MCTS finds the optimal policy for sc-iLTL objective φ on POMDP. ■

6. Case Study

To show the expressiveness of the iLTL objective, we create the drone-probing problem. We apply our MCTS algorithm on the problem 100 times. We evaluated the performance of the MCTS algorithm by the percentage of success simulations.

6.1. Drone-probing problem and iLTL objective

The drone-probing problem is in a 4×4 gride world with 256 hidden states. The **drone** on the grid has an action set $\{\mathrm{N}, \mathrm{S}, \mathrm{E}, \mathrm{W}, \mathrm{X}\}$ stands for moving to the four different directions or staying in the current grid. Given the action, the movement is deterministic, and the drone stays in the current grid if a movement hits the edge. The drone is initialized on the grid $(0, 0)$. The **ground target** will move randomly on the grid as We assign equal probability to all the possible following locations of the target. The target is initialized on the grid, excluding the corner with equal probability. The **sensor** has a limited field of view and only returns respective quadrant from $\Omega = \{\mathrm{SW}, \mathrm{NW}, \mathrm{NE}, \mathrm{SE}, \mathrm{None}\}$ as described in section 3.1.

The expressiveness of the sc-iLTL formula allows us to specify the following task. Suppose we want the drone to use its imperfect sensor to accurately locate the ground target and then move to the landing zone $(3, 3)$. Reaching the landing zone before getting an accurate measure is seen as a failure. Not reaching the landing zone until within the simulation horizon 100 is seen as a failure. We define the planning objective with the following iLTL formula,

$$\varphi = \Diamond\left(\bigvee_{i=1}^{625} \mathrm{ineq}_{\mathrm{measure},i}\right) \wedge \Diamond(\mathrm{ineq}_{\mathrm{goal}}) \wedge (\neg \mathrm{ineq}_{\mathrm{goal}} \mathcal{U} \bigvee_{i=1}^{625} \mathrm{ineq}_{\mathrm{measure},i}), \tag{12}$$

where $\Lambda = \{\mathrm{ineq}_{\mathrm{measure},1}, \mathrm{ineq}_{\mathrm{measure},2}, ..., \mathrm{ineq}_{\mathrm{measure},625}, \mathrm{ineq}_{\mathrm{goal}}\}$ is the set of atomic propositions, each $\mathrm{ineq}_{\mathrm{measure},i} := p_i^T b > 0.9$ is a linear inequality over the belief space measures the

confidence of the location of the ground target, p_i is a vector with all zero elements expect the i-th entry to be 1, b is the belief state, $\text{ineq}_{\text{goal}} := p_g^T b \geq 1$ stands for the event of reaching the landing zone.

It is hard to specify the task with the LTL formula. Satisfying $\Diamond(\bigvee_{i=1}^{625} \text{ineq}_{\text{measure},i})$ requires the drone to chase the target in a manner that can reason about the actual location of the target. Even if the drone is right above the target, $\|b\|_\infty$ may still be below the confidence 0.9.

6.2. Experiment

The experiments are conducted on a Windows 11 machine equipped with an Intel i9-14900K processor. The implementation is done using Python. We use the mona package Fuggitti (2019) to translate the LTL objective into DFA. The DFA has 4 states, thus increasing the number of all product history states by 4 times.

We run the experiment for 100 times. In each experiment, the drone and ground target are spawned based on the initial belief. Then, we apply MCTS to get an action. After receiving an observation and moving to the hidden state, we apply MCTS again to get action. For each MCTS search, we apply 2000 simulations and set the depth of the tree d_max to 20.

We show the performance of our algorithm using a histogram and the average change of $\|b\|_\infty$ in fig 1. 87 out of 100 experiments are successful. The drone needs an average of 40.71 steps to finish the task. 4 out of 13 failed experiments are due to not returning to the landing zone before the horizon 100. Other failed experiments are due to the randomness of MCTS output.

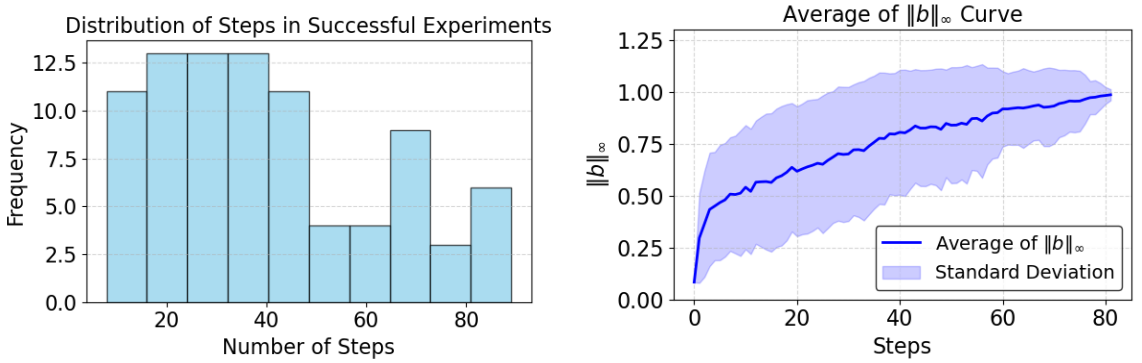

Figure 1: Histograms of steps in success runs (left) and change of average $\|b\|_\infty$ during success runs (right). Each action is returned by MCTS using 2000 simulations with depth $d_{max} = 20$. Our MCTS algorithm achieves 87% success rate for φ. We see the MCTS algorithm tries to increase $\|b\|_\infty$ up to the confidence predefined and finishes the task in an average of 40.71 steps.

7. Conclusion

This work proposes an iLTL objective that is more expressive than the LTL objective for planning on POMDPs. Specifically, we utilize inequality in belief space to specify tasks on POMDPs. Sc-iLTL is suitable for tasks related to safety or surveillance. The transformation of the planning for

the sc-iLTL objective into a reachability problem on the product belief MDP enables us to leverage MCTS to find optimal policies effectively. Experiments in the drone-probing problem demonstrate the expressiveness of the sc-iLTL objective and the performance of our MCTS method.

References

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47(2):235–256, May 2002. ISSN 1573-0565. doi: 10.1023/A:1013689704352.

Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. The MIT Press, 2008.

Blai Bonet and Hector Geffner. Solving POMDPs: RTDP-Bel vs. Point-based Algorithms. In *International Joint Conference on Artificial Intelligence*, July 2009.

Maxime Bouton, Jana Tumova, and Mykel J. Kochenderfer. Point-Based Methods for Model Checking in Partially Observable Markov Decision Processes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(06):10061–10068, April 2020. ISSN 2374-3468. doi: 10.1609/aaai.v34i06.6563.

Alper Kamil Bozkurt, Yu Wang, and Miroslav Pajic. Secure Planning Against Stealthy Attacks via Model-Free Reinforcement Learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

Francesco Fuggitti. Ltlf2dfa. March 2019. doi: 10.5281/zenodo.3888410.

Hector Geffner and Blai Bonet. Solving Large POMDPs using Real Time Dynamic Programming. 1998.

Constantin Hubmann, Jens Schulz, Marvin Becker, Daniel Althoff, and Christoph Stiller. Automated Driving in Uncertain Environments: Planning With Interaction and Uncertain Maneuver Prediction. *IEEE Transactions on Intelligent Vehicles*, 3(1):5–17, March 2018.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, May 1998.

Levente Kocsis and Csaba Szepesvári. Bandit Based Monte-Carlo Planning. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006*, pages 282–293, Berlin, Heidelberg, 2006. Springer. ISBN 978-3-540-46056-5. doi: 10.1007/11871842_29.

YoungMin Kwon and Gul Agha. Linear Inequality LTL (iLTL): A Model Checker for Discrete Time Markov Chains. In Jim Davies, Wolfram Schulte, and Mike Barnett, editors, *Formal Methods and Software Engineering*, pages 194–208, Berlin, Heidelberg, 2004. Springer. ISBN 978-3-540-30482-1. doi: 10.1007/978-3-540-30482-1_21.

Bruno Lacerda, David Parker, and Nick Hawes. Optimal and dynamic planning for Markov decision processes with co-safe LTL specifications. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1511–1516, September 2014. doi: 10.1109/IROS.2014.6942756.

Abolfazl Lavaei, Sadegh Soudjani, Alessandro Abate, and Majid Zamani. Automated verification and synthesis of stochastic hybrid systems: A survey. *Automatica*, 146:110617, December 2022. ISSN 0005-1098. doi: 10.1016/j.automatica.2022.110617.

Junchao Li. *Reinforcement Learning-Based Motion Planning in Partially Observable Environments for Complex Tasks*. PhD thesis, United States – Iowa, 2024.

Joelle Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*, August 2003.

David Silver and Joel Veness. Monte-Carlo Planning in Large POMDPs. In *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010.

Mária Svoreňová, Martin Chmelík, Kevin Leahy, Hasan Ferit Eniser, Krishnendu Chatterjee, Ivana Černá, and Calin Belta. Temporal logic motion planning using POMDPs with parity objectives: Case study paper. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, HSCC '15, pages 233–238, New York, NY, USA, April 2015. Association for Computing Machinery. ISBN 978-1-4503-3433-4. doi: 10.1145/2728606.2728617.

Appendix

In the Appendix, we apply the proof in Kocsis and Szepesvári (2006) to our product belief MDP.

MCTS treats the Bellman optimality equation

$$V^*(b^\times) = R(b^\times) + \max_{a \in A} \sum_{b^{\times\prime}} P^\times_{\mathcal{B}}(b^\times, a, b^{\times\prime}) V^*(b^{\times\prime}), \tag{13}$$

as a non-stationary bandit problem with $|A|$ arms. The bandit problem is defined by the sequence of random payoffs $X_{it}, i = 1, ..., K, t \geq 1$, where each i is the index of an action. Successive choose action i from the same state $b^\times$ yield the payoffs $X_{i1}, X_{i2}, ...$ The payoff sequence is non-stationary since during the MCTS, the value estimation $\hat{V}(h^{\times\prime})$ for successive node changes when we estimate $\hat{V}(h^\times)$.

MCTS uses UCB1 to choose the action with the best upper confidence bound:

$$I_t = \arg_{i \in \{1,...,K\}} \max \left\{ \bar{X}_{i,T_i(t-1)} + c_{t-1,T_i(t-1)} \right\} \tag{14}$$

where $T_i(n) = \sum_{s=1}^{n} \mathbb{I}(I_s = i)$ is the number of times action i was played up to time n, $I_t \in \{1, ..., K\}$ is the index of the action selected at time t, $c_{t,s} = 2C_p\sqrt{\ln t/s}$ is a bias sequence.

During the MCTS, the value estimation $\hat{V}(h^{\times\prime})$ for follow-up node changes when we estimate $\hat{V}(h^\times)$. Thus, the payoff sequence is non-stationary.

MCTS allows the payoff sequence X_{it} to be the non-stationary as long as it follows the drift condition. That is, the averages $\bar{X}_{in} := 1/n \sum_{t=1}^{n} X_{it}$ converges to the expectation μ_i

$$\begin{aligned} \mu_i &= \lim_{n \to \infty} \mu_{in} \\ \mu_{in} &= \mathbb{E}[\bar{X}_{in}] = \mu_i + \delta_{in}. \end{aligned} \tag{15}$$

For a finite horizon MDP with bounded reward function, or in our case, the product belief MDP with reward function being either 0 or 1, we can show that the drift condition always holds.

Lemma 9 *Consider the product belief MDP of the belief MDP and a DFA, let k be the length of the truncated finite sequence $\sigma_{\leq k} = b_0 b_1 \cdots b_k$ used to verify the sc-iLTL objective. Then the bias of the esimated expected payoff $\bar{X}_n$ is $O((k|A| \log n + |A|^k)/n)$*

If the drift condition holds, the expected estimation error is bounded as

Lemma 10 *(Kocsis and Szepesvári, 2006, Theorem 3) Let $\bar{X}_n = \sum_{i=1}^{K} \frac{T_i(n)}{n} \bar{X}_{i,T_i(n)}$, let μ^* be the expectation for μ_i where i is the optimal action and $\delta_n^* = \mu_n^* - \mu^*$,*

$$|\mathbb{E}[\bar{X}_n] - \mu^*| \leq |\delta_n^*| + O(\frac{K(C_p^2 \ln(n) + N_0)}{n}). \tag{16}$$

The estimated optimal payoff concentrates on the mean by

Lemma 11 *(Kocsis and Szepesvári, 2006, Theorem 5) Fix $\delta > 0$ and let $\Delta_n = 9\sqrt{2n \ln(2/\delta)}$. The following bounds hold true provided that n is sufficiently large: $Pr(n\bar{X}_n \geq n\mathbb{E}[\bar{X}_n] + \Delta_n) \leq \delta$, $Pr(n\bar{X}_n \leq n\mathbb{E}[\bar{X}_n] - \Delta_n) \leq \delta$.*

Proof of Lemma 9

(Kocsis and Szepesvári, 2006, Theorem 7) The proof is done by applying Lemma 10, 11 inductively on the length of the truncated finite sequence k. Consider setting the case $k = 1$. Hoeffding's inequality holds the assumptions on the payoffs.

Assume the result holds for the tree up to depth $d - 1$ and consider a tree of depth d. Consider the root node. The result followed by Lemma 10, 11 still holds. ■

With the guarantee of drift condition, the estimated optimal payoff concentrates around its mean,

Lemma 12 *(Kocsis and Szepesvári, 2006, Theorem 6) we can get the optimal action with an accurate guarantee*

$$\lim_{t\to\infty} Pr(I_t \neq i^*) = 0. \tag{17}$$

Proof of Theorem 8

Equation (17) holds for all nodes due to Lemma 9. Thus, MCTS returns the optimal action with probability 1 when n is sufficiently large. ■