To appear in the proceedings of the 13th International Workshop on the Algorithmic Foundations of Robotics (WAFR), 2018

Online Partial Conditional Plan Synthesis for POMDPs with Safe-Reachability Objectives

Yue Wang, Swarat Chaudhuri, and Lydia E. Kavraki

Department of Computer Science, Rice University
{yw27,swarat,kavraki}@rice.edu

Abstract. The framework of Partially Observable Markov Decision Processes (POMDPs) offers a standard approach to model uncertainty in many robot tasks. Traditionally, POMDPs are formulated with optimality objectives. However, for robotic domains that require a correctness guarantee of accomplishing tasks, boolean objectives are natural formulations. We study POMDPs with a common boolean objective: *safe-reachability*, which requires that, with a probability above a threshold, the robot eventually reaches a goal state while keeping the probability of visiting unsafe states below a different threshold. The solutions to POMDPs are policies or conditional plans that specify the action to take contingent on every possible event. A full policy or conditional plan that covers all possible events is generally expensive to compute. To improve efficiency, we introduce the notion of *partial conditional plans* that only cover a sampled subset of all possible events. Our approach constructs a partial conditional plan parameterized by a *replanning probability*. We prove that the probability of the constructed partial conditional plan failing is bounded by the replanning probability. Our approach allows users to specify an appropriate bound on the replanning probability to balance efficiency and correctness. We validate our approach in several robotic domains. The results show that our approach outperforms a previous approach for POMDPs with safe-reachability objectives in these domains.

1 Introduction

Planning robust executions under uncertainty is a fundamental concern in robotics. POMDPs [29] provide a standard framework for modeling many robot tasks under uncertainty. The solutions to POMDPs are *policies* [29] or *conditional plans* [9] that specify the actions to take under *all possible* events during execution.

Traditionally, the goal of solving POMDPs is to find optimal solutions that maximize (discounted) rewards [1, 3, 9, 12, 16, 17, 23, 24, 30]. While this purely quantitative formulation is suitable for many applications, some robotic settings demand synthesis concerning *boolean* requirements. For example, consider a robot with imperfect actuation and perception working in an office environment with uncertain obstacles such as floor signs and furniture (Fig. 1). Due to uncertainty, the locations of the robot and the obstacles are partially observable, and the robot's action effects and observations are both probabilistic. In this probabilistic

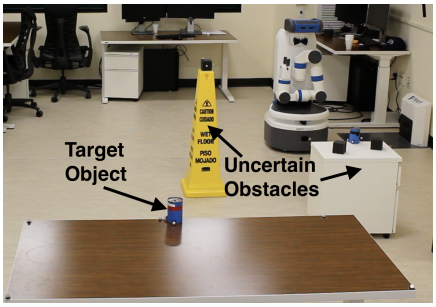


Fig. 1: A robot with imperfect actuation and perception needs to pick up the blue can on the table, while avoiding collisions with uncertain obstacles such as floor signs and file cabinets.

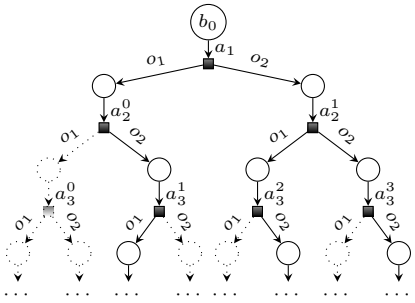


Fig. 2: A full conditional plan γ_k contains both solid and dotted branches. $a_1, a_2^0, \ldots$ are actions. o_1 and o_2 are observations. A partial conditional plan γ_k^p contains only solid branches.

setting, a reasonable task requirement for the robot is to eventually pick up the target object with a probability above a threshold while keeping the probability of collision below a different threshold. This task requirement is naturally formulated as a boolean objective written in a temporal logic. Moreover, formulating boolean requirements implicitly as quantitative objectives does not always yield good solutions for certain domains [32]. Therefore, POMDPs with explicit boolean objectives are better formulations than quantitative POMDPs in these domains.

Policy synthesis for POMDPs with boolean objectives has been studied in previous works [4, 5, 31], where the goal is to satisfy a temporal property with *probability 1* (almost-sure satisfaction). A more general policy synthesis for POMDPs with boolean objectives is to synthesize policies that satisfy a temporal property with *a probability above a threshold*. In this work, we study this problem for the special case of *safe-reachability* objectives, which require that with a probability above a threshold, a goal state is eventually reached while keeping the probability of visiting unsafe states below a different threshold. Many robot tasks such as the one in Fig. 1 can be formulated as a safe-reachability objective.

Our previous work [33] has presented a method called Bounded Policy Synthesis (BPS) for POMDPs with safe-reachability objectives. BPS computes a valid policy over the *goal-constrained belief space* rather than the entire belief space to improve efficiency. The goal-constrained belief space only contains beliefs visited by desired executions achieving the safe-reachability objective and is generally much smaller than the original belief space. BPS is an *offline* synthesis method that computes a full policy before execution. Another category of approaches to planning under uncertainty is *online planning* that interleaves planning and execution [3, 8, 13, 14, 17, 28, 30]. Offline synthesis offers a strong correctness guarantee, but it is difficult to scale. Online planning is much more scalable and works well when replanning is likely to succeed, but it often fails when replanning is difficult or infeasible in some states, making it hard to ensure correctness.

In this work, our goal is to scale up our previous BPS method further through online planning and achieve a good balance between efficiency and correctness.

Specifically, we present a method called *Online Partial Conditional Plan Synthesis* (OPCPS) for POMDPs with safe-reachability objectives, based on the new notion of *partial conditional plans*. A partial conditional plan only contains a sampled subset of all possible events and approximates a full policy. OPCPS computes a partial conditional plan parameterized by a *replanning probability*, i.e., the probability of the robot encountering an event not covered by the partial conditional plan, thus requiring replanning. We offer a theoretical analysis of this replanning probability framework, showing that the probability of the constructed partial conditional plan failing is bounded by the replanning probability. OPCPS allows users to specify an appropriate bound on the replanning probability to balance efficiency and correctness: for domains where replanning is likely to succeed, increasing the bound usually leads to better scalability, and for domains where replanning is difficult or impossible in some states, users can decrease the bound and allocate more computation time to achieve a higher success rate.

To further improve performance, OPCPS updates the replanning probability bound instead of using the same bound during computation. This bound update enables quicker detection of the current partial conditional plan meeting the bound and avoids unnecessary computation. For a better safety guarantee, OPCPS checks whether the successor belief of every uncovered observation of the constructed partial conditional plan satisfies the safety requirement. Thus OPCPS guarantees that the robot still satisfies the safety requirement when replanning fails. Section 3.1 has more details on the bound update and the safety guarantee of OPCPS.

We evaluate OPCPS in the kitchen domain presented in [33] and the Tag domain [23]. We also validate OPCPS on a Fetch robot for the domain shown in Fig. 1. The results demonstrate that OPCPS scales better than BPS and can solve problems that are beyond the capabilities of BPS within the time limit.

Related Work The analysis of POMDPs can be divided into three categories. In the first category, the objective is to find optimal solutions concerning quantitative rewards. Many previous POMDP algorithms [1, 3, 9, 16, 17, 23, 30] focus on maximizing (discounted) rewards. In the second category, the objective combines the quantitative rewards of the traditional POMDPs with notions of risk and cost. Recently, there has been a growing interest in constrained POMDPs [11,15,25,32], chance-constrained POMDP [27], and risk-sensitive POMDPs [10,19] that handle cost/risk constraints explicitly. The third category consists of POMDPs with high-level boolean requirements written in a temporal logic. Recent work [4,5,31] has investigated the *almost-sure satisfaction* of POMDPs with temporal properties, where the goal is to check whether a given temporal property can be satisfied with probability 1. A more general policy synthesis problem of POMDPs with safe-reachability objectives has been introduced in our previous work [33]. It has been shown that for robotic domains that require a correctness guarantee of accomplishing tasks, POMDPs with safe-reachability provide a better guarantee of safety and reachability than the quantitative POMDP formulations [33].

In this work, we focus on POMDPs with safe-reachability objectives and evaluate our previous BPS approach [33]. While BPS synthesizes a full policy

(conditional plan) offline that covers all possible events, our approach is an online method that interleaves the computation of a partial conditional plan and execution. Since a partial conditional plan only contains a sampled subset of all possible events, our method achieves better scalability than BPS and can solve problems that are beyond the capabilities of BPS within the time limit.

This idea of partial conditional plans resembles the state-of-the-art online POMDP algorithm based on Determinized Sparse Partially Observable Tree (DESPOT) [3, 30]. Both DESPOT and our partial conditional plans contain a subset of all possible observations to improve efficiency. There are two major differences between our method and DESPOT: first, DESPOT handles POMDPs with (discounted) rewards while our approach solves POMDPs with safe-reachability objectives. Second, DESPOT contains all action branches while our approach constructs partial conditional plans (Fig. 2) that only contains one action per step, which is part of the desired execution satisfying the safe-reachability objective.

2 Problem Formulation

We follow the notation in [33] for POMDPs with safe-reachability objectives.

Definition 1 (POMDP). *A POMDP is a tuple* $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$*, where* $\mathcal{S}$ *is a* finite *set of states,* $\mathcal{A}$ *is a* finite *set of actions,* $\mathcal{T}$ *is a probabilistic transition function,* $\mathcal{O}$ *is a* finite *set of observations, and* $\mathcal{Z}$ *is a probabilistic observation function.* $\mathcal{T}(s, a, s') = Pr(s'|s, a)$ *specifies the probability of moving to state* $s' \in \mathcal{S}$ *after taking action* $a \in \mathcal{A}$ *in state* $s \in \mathcal{S}$*.* $\mathcal{Z}(s', a, o) = Pr(o|s', a)$ *specifies the probability of observing* $o \in \mathcal{O}$ *after taking action* $a \in \mathcal{A}$ *and reaching* $s' \in \mathcal{S}$*.*

Due to uncertainty, states are partially observable and typically we maintain a probability distribution (*belief*) over all states $b : \mathcal{S} \mapsto [0, 1]$ with $\sum_{s\in\mathcal{S}} b(s) = 1$. The set of all beliefs $\mathcal{B} = \{b : \mathcal{S} \mapsto [0, 1] \mid \sum_{s\in\mathcal{S}} b(s) = 1\}$ is the *belief space*.

The belief space transition function $\mathcal{T}_\mathcal{B} : \mathcal{B} \times \mathcal{A} \times \mathcal{O} \to \mathcal{B}$ is *deterministic*. $b_a^o = \mathcal{T}_\mathcal{B}(b, a, o)$ is the successor belief for a belief $b \in \mathcal{B}$ after taking an action $a \in \mathcal{A}$ and receiving an observation $o \in \mathcal{O}$, defined according to Bayes rule: $\forall\ s' \in \mathcal{S},\ b_a^o(s') = \frac{\mathcal{Z}(s',a,o) \sum_{s\in\mathcal{S}} \mathcal{T}(s,a,s')b(s)}{\Pr(o|b,a)}$, where $\Pr(o|b, a) = \sum_{s'\in\mathcal{S}} \mathcal{Z}(s', a, o) \sum_{s\in\mathcal{S}} \mathcal{T}(s, a, s')b(s)$ is the probability of receiving the observation o after taking the action a in the belief b.

Definition 2 (Plan). *A* k*-step* plan *is a sequence* $\sigma = (b_0, a_1, o_1, \ldots, a_k, o_k, b_k)$ *such that for all* $i \in (0, k]$*, the belief updates satisfy the transition function* $\mathcal{T}_\mathcal{B}$*, i.e.,* $b_i = \mathcal{T}_\mathcal{B}(b_{i-1}, a_i, o_i)$*, where* $a_i \in \mathcal{A}$ *is an action and* $o_i \in \mathcal{O}$ *is an observation.*

Definition 3 (Safe-Reachability Objective). *A* safe-reachability objective *is a tuple* $\mathcal{G} = (Dest, Safe)$*, where* $Safe = \{b \in \mathcal{B} \mid \sum_{s \text{ violates safety}} b(s) < \delta_2\}$ *is a set of safe beliefs and* $Dest = \{b \in Safe \mid \sum_{s \text{ is a goal state}} b(s) > 1 - \delta_1\} \subseteq Safe$ *is a set of goal beliefs.* δ_1 *and* δ_2 *are small values that represent tolerance.*

$\mathcal{G}$ compactly represents the set $\Omega_{\mathcal{G}}$ of valid plans:

Definition 4 (Valid Plan). *A k-step plan $\sigma = (b_0, a_1, o_1, \ldots, a_k, o_k, b_k)$ is* valid *w.r.t. a safe-reachability objective $\mathcal{G} = (Dest, Safe)$ if b_k is a goal belief ($b_k \in Dest$) and all beliefs visited before step k are safe beliefs ($\forall\, i \in [0, k), b_i \in Safe$).*

Partial Conditional Plan Computing an exact policy over the entire belief space $\mathcal{B}$ is intractable, due to the curse of dimensionality [21]: $\mathcal{B}$ is a high-dimensional space with an infinite number of beliefs. To make the problem tractable, we can exploit the *reachable belief space* $\mathcal{B}_{b_0}$ [16, 23]. $\mathcal{B}_{b_0}$ only contains beliefs reachable from the initial belief b_0 and is generally much smaller than $\mathcal{B}$.

Our previous BPS work [33] has shown that the performance of policy synthesis for POMDPs with safe-reachability objectives can be further improved based on the notion of a *goal-constrained belief space* $\mathcal{B}_{\mathcal{G}}$. $\mathcal{B}_{\mathcal{G}}$ combines the reachable belief space $\mathcal{B}_{b_0}$ and the set $\Omega_{\mathcal{G}}$ of valid plans defined by the safe-reachability objective $\mathcal{G}$. $\mathcal{B}_{\mathcal{G}}$ only contains beliefs reachable from the initial belief b_0 under a valid plan $\sigma \in \Omega_{\mathcal{G}}$ and is generally much smaller than the reachable belief space $\mathcal{B}_{b_0}$.

Previous results [6, 18, 22] have shown that the problem of policy synthesis for POMDPs is generally undecidable. However, when restricted to a bounded horizon, this problem becomes PSPACE-complete [20, 21]. Therefore, BPS computes a bounded policy π over the goal-constrained belief space $\mathcal{B}_{\mathcal{G}}$ within a bounded horizon h. This bounded policy π is essentially a set of conditional plans [9]:

Definition 5 (Conditional Plan). *A k-step conditional plan $\gamma_k \in \Gamma_k$ is a tuple $\gamma_k = (b, a, \nu_k)$, where $b \in \mathcal{B}$ is a belief, $a \in \mathcal{A}$ is an action and $\nu_k : \mathcal{O} \mapsto \Gamma_{k-1}$ is an* observation strategy *that maps an observation $o \in \mathcal{O}$ to a $(k-1)$-step conditional plan $\gamma_{k-1} = (b', a', \nu_{k-1}) \in \Gamma_{k-1}$, where $b' = \mathcal{T}_{\mathcal{B}}(b, a, o)$ is the successor belief.*

Fig. 2 shows an example k-step conditional plan $\gamma_k = (b_0, a_1, \nu_k)$. γ_k defines a set Ω_{γ_k} of k-step plans $\sigma_k = (b_0, a_1, o_1, \ldots, a_k, o_k, b_k)$. For each plan $\sigma_k \in \Omega_{\gamma_k}$, the action a_1 at step 1 is chosen by the k-step conditional plan γ_k, the action a_2 at step 2 is chosen by the $(k-1)$-step conditional plan $\gamma_{k-1} = \nu_k(o_1)$, ..., and the action a_k at step k is chosen by the one-step conditional plan $\gamma_1 = \nu_2(o_{k-1})$.

Definition 6 (Valid Conditional Plan). *A k-step conditional plan γ_k is* valid *w.r.t. a safe-reachability objective $\mathcal{G}$ if every plan in Ω_{γ_k} is valid ($\Omega_{\gamma_k} \subseteq \Omega_{\mathcal{G}}$).*

It is clear that the number of valid plans in a valid k-step conditional plan γ_k grows exponentially as the horizon k increases. To address this challenge, our method computes *partial* conditional plans to approximate full conditional plans:

Definition 7 (Partial Conditional Plan). *A k-step* partial *conditional plan is a tuple $\gamma_k^p = (b, a, \mathcal{O}_k^p, \nu_k^p)$, where $b \in \mathcal{B}$ is a belief, $a \in \mathcal{A}$ is an action, $\mathcal{O}_k^p \subseteq \mathcal{O}$ is a subset of the observation set $\mathcal{O}$, and $\nu_k^p : \mathcal{O}_k^p \mapsto \Gamma_{k-1}^p$ is a* partial observation strategy *that maps an observation $o \in \mathcal{O}_k^p$ to a $(k-1)$-step partial conditional plan $\gamma_{k-1}^p = (b', a', \mathcal{O}_{k-1}^p, \nu_{k-1}^p)$, where $b' = \mathcal{T}_{\mathcal{B}}(b, a, o)$ is the successor belief.*

Similarly, γ_k^p defines a set $\Omega_{\gamma_k^p}$ of k-step plans σ_k in belief space.

Definition 8 (Valid Partial Conditional Plan). *A k-step partial conditional plan γ_k^p is* valid *w.r.t. a safe-reachability objective $\mathcal{G}$ if every plan in $\Omega_{\gamma_k^p}$ is valid.*

Replanning Probability Since a partial conditional plan $\gamma_k^p = (b, a, \mathcal{O}_k^p, \nu_k^p)$ only contains a subset of all observation branches at each step (see Fig. 2), during online execution, it is possible that an observation branch $o \in \mathcal{O} - \mathcal{O}_k^p$ that is not part of the partial conditional plan is visited. In this case, we need to recursively compute a new partial conditional plan for this new branch o. However, since γ_k^p does not consider all possible observation branches, it is possible that the action chosen by γ_k^p is *invalid* for the new observation branch o. As a result, there are no partial conditional plans for the new observation branch o and execution fails.

To preserve correctness, we would like to bound the *failure probability* $p_{\text{fail}}(\gamma_k^p) = \Pr(\text{failure}|\gamma_k^p)$ measured under a valid partial conditional $\gamma_k^p = (b, a, \mathcal{O}_k^p, \nu_k^p)$. However, computing p_{fail} is costly because it requires checking whether the action a chosen by γ_k^p is valid for every uncovered observation branch $o \in \mathcal{O} - \mathcal{O}_k^p$, which essentially computes a *full* conditional plan. Alternatively, we can easily compute the *replanning probability* $p_{\text{replan}}(\gamma_k^p) = \Pr(\text{replanning}|\gamma_k^p)$ of reaching an uncovered observation branch $o \in \mathcal{O} - \mathcal{O}_k^p$ and requiring *replanning*:

$$p_{\text{replan}}(\gamma_k^p) = \sum_{o \in \mathcal{O}_k^p} \Pr(o|b,a) p_{\text{replan}}(\nu_k^p(o)) + \sum_{o \in \mathcal{O} - \mathcal{O}_k^p} \Pr(o|b,a) \tag{1}$$

For the base case $k = 1$, $p_{\text{replan}}(\gamma_1^p) = \sum_{o \in \mathcal{O} - \mathcal{O}_1^p} \Pr(o|b,a)$.

The following theorem states that for a *valid* partial conditional plan γ_k^p, the failure probability $p_{\text{fail}}(\gamma_k^p)$ is bounded by the replanning probability $p_{\text{replan}}(\gamma_k^p)$:

Theorem 1. *For any valid partial conditional plan* γ_k^p, $p_{fail}(\gamma_k^p) \leq p_{replan}(\gamma_k^p)$.

Theorem 1 can be proved by induction (see Appendix A).

Problem Statement Given a POMDP P, an initial belief b_0, a replanning probability bound $\delta_{p_{\text{replan}}}$, a safe-reachability objective $\mathcal{G}$ and a horizon bound h, our goal is to synthesize a *valid* k-step ($k \leq h$) partial conditional plan $\gamma_k^p = (b_0, a, \mathcal{O}_k^p, \nu_k^p)$ with a replanning probability $p_{\text{replan}}(\gamma_k^p) \leq \delta_{p_{\text{replan}}}$.

Since the replanning probability $p_{\text{replan}}(\gamma_k^p)$ is bounded by $\delta_{p_{\text{replan}}}$, by Theorem 1, γ_k^p guarantees achieving the given safe-reachability objective with a probability at least $1 - \delta_{p_{\text{replan}}}$. Note that when $p_{\text{replan}}(\gamma_k^p) = 0$, γ_k^p is a full conditional plan.

3 Online Partial Conditional Plan Synthesis

Fig. 3 shows the overall structure of OPCPS. OPCPS follows the typical online planning paradigm [26] that interleaves synthesis of valid partial conditional plans and execution. If there are no valid partial conditional plans within the horizon bound, execution fails. Otherwise, OPCPS follows the generated partial conditional plan until a goal belief is reached or a new observation $o \in \mathcal{O} - \mathcal{O}_k^p$ is received. In the latter case, OPCPS recursively replans for the observation o.

In partial conditional plan synthesis (Fig. 4), we replace the policy generation component in BPS [33] with a new partial conditional plan generation (the green

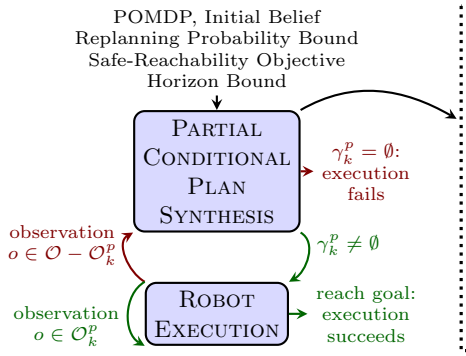


Fig. 3: OPCPS

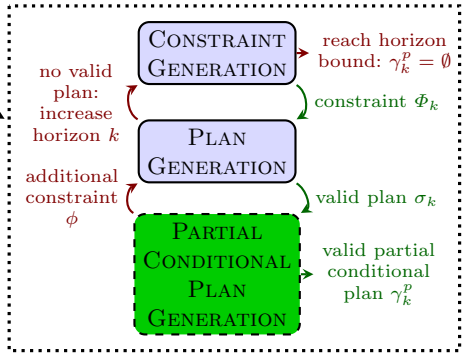


Fig. 4: Partial conditional plan synthesis

dashed component). For completeness, we offer a brief summary of the constraint generation and plan generation components in BPS. See [33] for more details.

In constraint generation (Fig. 4), given a POMDP P, an initial belief b_0 and a safe-reachability objective $\mathcal{G} = (\mathit{Dest}, \mathit{Safe})$, we first construct a constraint Φ_k to symbolically encode the goal-constrained belief space over a bounded horizon k based on the encoding from Bounded Model Checking [2]. Φ_k compactly represents the requirement of reaching a goal belief $b \in \mathit{Dest}$ safely in k steps.

Then in plan generation (Fig. 4), we compute a valid *plan* σ_k by checking the satisfiability of Φ_k through an SMT solver [7]. If Φ_k is satisfiable, the SMT solver returns a valid plan $\sigma_k = (b_0^{\sigma_k}, a_1^{\sigma_k}, o_1^{\sigma_k}, \ldots, b_k^{\sigma_k})$. This valid plan σ_k only covers a particular observation $o_i^{\sigma_k}$ at step i. In partial conditional plan generation (Fig. 4), we generate a valid partial conditional plan γ_k^p with a replanning probability $p_{\text{replan}}(\gamma_k^p) \leq \delta_{p_{\text{replan}}}$ from this valid plan σ_k by sampling a subset $\mathcal{O}_k^p \subseteq \mathcal{O}$ of observations at each step, where $\delta_{p_{\text{replan}}}$ is the given replanning probability bound. If this partial conditional plan generation fails, we construct an additional constraint ϕ to block invalid plans and force the SMT solver to generate another better plan. If Φ_k is unsatisfiable, we increase the horizon and repeat the above steps until a *valid* partial conditional plan is found or a given horizon bound is reached. Next we describe the new partial conditional plan generation component.

3.1 Partial Conditional Plan Generation

In partial conditional plan generation (Algorithm 1), we construct a valid partial conditional plan γ_k^p that satisfies the given bound $\delta_{p_{\text{replan}}}$ from a valid plan σ_k. For each step i, we first recursively construct a next-step conditional plan γ_{next}^p for $o_i^{\sigma_k}$ (line 5). If the replanning probability $p_{\text{replan}}(\gamma_k^p)$ is greater than the bound $\delta_{p_{\text{replan}}}$ (line 9), we add more observation branches to γ_k^p by sampling a new observation o' according to the probability of occurrence (line 11) and recursively constructing a next-step partial conditional plan γ_{next}^p for o' (line 13). This is another partial conditional plan synthesis problem with a new initial belief b' (line 12), and can be solved recursively using the algorithm shown in Fig. 4.

If we successfully construct a valid γ_{next}^p for o', we add o' to γ_k^p (line 8 or 16). Otherwise, this input plan σ_k cannot be an element of a *valid* partial conditional

Algorithm 1: PartialConditionalPlanGeneration

Input: POMDP $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$, Replanning Probability Bound $\delta_{p_{\text{replan}}}$, Safe-Reachability Objective $\mathcal{G} = (\mathit{Dest}, \mathit{Safe})$, Valid k-Step Plan $\sigma_k = (b_0^{\sigma_k}, a_1^{\sigma_k}, o_1^{\sigma_k}, \ldots, b_k^{\sigma_k})$, Step i, Horizon Bound h

Output: *Valid* partial conditional plan γ_k^p with replanning probability $p_{\text{replan}}(\gamma_k^p) \leq \delta_{p_{\text{replan}}}$, Constraint ϕ for blocking invalid plans

1 **if** $i > k$ **then** /* Reach end of the plan */
2 $\quad \gamma_k^p \leftarrow (b_k^{\sigma_k}, \emptyset, \emptyset, \emptyset)$ /* Terminal belief: γ_k^p specifies nothing */
3 $\quad$ **return** $\gamma_k^p, \emptyset$
4 $\mathcal{O}_k^p \leftarrow \emptyset$, $\delta'_{p_{\text{replan}}} \leftarrow \delta_{p_{\text{replan}}}$, $b \leftarrow b_{i-1}^{\sigma_k}$, $a \leftarrow a_i^{\sigma_k}$, $o' \leftarrow o_i^{\sigma_k}$ /* Initialize */
/* Recursively process next step */
5 $\gamma_{\text{next}}^p \leftarrow$ PartialConditionalPlanGeneration$(P, \delta'_{p_{\text{replan}}}, \mathcal{G}, \sigma_k, i+1, h)$
6 **if** $\gamma_{next}^p = \emptyset$ **then** /* Construction failed */
7 $\quad$ Construct ϕ using Formula 2, **return** $\emptyset$, ϕ
8 $\mathcal{O}_k^p \leftarrow \mathcal{O}_k^p \cup \{o'\}$, $\nu_k^p(o') \leftarrow \gamma_{\text{next}}^p$, $\gamma_k^p \leftarrow (b, a, \mathcal{O}_k^p, \nu_k^p)$ /* Add o' to γ_k^p */
9 **while** $p_{replan}(\gamma_k^p) > \delta_{p_{replan}}$ **do**
10 $\quad \delta'_{p_{\text{replan}}} \leftarrow \delta'_{p_{\text{replan}}} + \frac{\Pr(o'|b,a)(\delta'_{p_{\text{replan}}} - p_{\text{replan}}(\nu_k^p(o')))}{\sum_{o \in \mathcal{O} - \mathcal{O}_k^p - \{o'\}} \Pr(o|b,a)}$ /* Bound update */
11 $\quad o' \leftarrow$ sampled observation in $\mathcal{O} - \mathcal{O}_k^p$ based on the probability of occurrence
12 $\quad b' \leftarrow \mathcal{T}_{\mathcal{B}}(b, a, o')$ /* Get new initial belief */
$\quad$ /* Recursively construct a next-step partial conditional plan */
13 $\quad \gamma_{\text{next}}^p \leftarrow$ PartialConditionalPlanSynthesis$(P, b', \delta'_{p_{\text{replan}}}, \mathcal{G}, i, h)$
14 $\quad$ **if** $\gamma_{next}^p = \emptyset$ **then** /* Construction failed */
15 $\quad\quad$ Construct ϕ using Formula 2, **return** $\emptyset$, ϕ
16 $\quad \mathcal{O}_k^p \leftarrow \mathcal{O}_k^p \cup \{o'\}$, $\nu_k^p(o') \leftarrow \gamma_{\text{next}}^p$ /* Add o' to γ_k^p */
17 **foreach** *observation* $o \in \mathcal{O} - \mathcal{O}_k^p$ **do** /* Final safety check */
18 $\quad b' \leftarrow \mathcal{T}_{\mathcal{B}}(b, a, o)$ /* Try observation o */
19 $\quad$ **if** $b' \notin \mathit{Safe}$ **then** /* Violates safety */
20 $\quad\quad$ Construct ϕ using Formula 2, **return** $\emptyset$, ϕ
21 **return** γ_k^p, $\emptyset$

plan γ_k^p ($\sigma_k \notin \Omega_{\gamma_k^p}$). Therefore, the prefix $(b_0^{\sigma_k}, a_1^{\sigma_k}, o_1^{\sigma_k}, \ldots, b_{i-1}^{\sigma_k}, a_i^{\sigma_k})$ of the input plan σ_k is invalid for the current horizon k and we construct the following additional constraint ϕ to block invalid plans:

$$\neg((b_0 = b_0^{\sigma_k}) \wedge (a_i = a_i^{\sigma_k}) \wedge \left(\bigwedge_{m=1}^{i-1} (a_m = a_m^{\sigma_k}) \wedge (o_m = o_m^{\sigma_k}) \wedge (b_m = b_m^{\sigma_k})\right)) \quad (2)$$

ϕ blocks the invalid plans that have this prefix and avoids unnecessary checks of these plans (checking σ_k has already shown that these plans are invalid).

Updating Replanning Probability Bound As we add more observation branches to the current partial conditional plan $\gamma_k^p = (b, a, \mathcal{O}_k^p, \nu_k^p)$, we update the replanning probability bound $\delta'_{p_{\text{replan}}}$ (line 10) for the remaining uncovered observation branches $\mathcal{O} - \mathcal{O}_k^p$ to avoid unnecessary computation.

Initially, $\mathcal{O}_k^p$ is empty and $\delta'_{p_{\text{replan}}}$ is the input bound $\delta_{p_{\text{replan}}}$ (line 4). $\delta'_{p_{\text{replan}}}$ bounds the replanning probability $p_{\text{replan}}(\nu_k^p(o))$ of the next-step partial conditional plan $\nu_k^p(o)$ for every remaining uncovered observation $o \in \mathcal{O} - \mathcal{O}_k^p$. $\delta'_{p_{\text{replan}}}$ guarantees that the replanning probability $p_{\text{replan}}(\gamma_k^p)$ satisfies the original bound $\delta_{p_{\text{replan}}}$, i.e., $p_{\text{replan}}(\gamma_k^p) = \sum_{o \in \mathcal{O}} \Pr(o|b,a) p_{\text{replan}}(\nu_k^p(o)) \leq \sum_{o \in \mathcal{O}} \Pr(o|b,a) \delta'_{p_{\text{replan}}} \leq \delta'_{p_{\text{replan}}} = \delta_{p_{\text{replan}}}$ since $p_{\text{replan}}(\nu_k^p(o)) \leq \delta'_{p_{\text{replan}}}$ based on the definition of $\delta'_{p_{\text{replan}}}$.

During partial conditional plan generation, after adding a new observation $o' \in \mathcal{O} - \mathcal{O}_k^p$ to the partial conditional plan γ_k^p (line 8 or 16), we update $\delta'_{p_{\text{replan}}}$ to avoid unnecessary computation. Suppose we construct a new next-step partial conditional plan γ_{next}^p with the same replanning probability α for every remaining uncovered observation $o \in \mathcal{O} - \mathcal{O}_k^p - \{o'\}$. Then the replanning probability of the observation branches $\mathcal{O} - \mathcal{O}_k^p$ is $\Pr(o'|b,a) p_{\text{replan}}(\nu_k^p(o')) + \alpha \sum_{o \in \mathcal{O} - \mathcal{O}_k^p - \{o'\}} \Pr(o|b,a) \leq \sum_{o \in \mathcal{O} - \mathcal{O}_k^p} \Pr(o|b,a) \delta'_{p_{\text{replan}}}$. Therefore $\alpha \leq \delta'_{p_{\text{replan}}} + \frac{\Pr(o'|b,a)(\delta'_{p_{\text{replan}}} - p_{\text{replan}}(\nu_k^p(o')))}{\sum_{o \in \mathcal{O} - \mathcal{O}_k^p - \{o'\}} \Pr(o|b,a)}$. Then the new bound for the remaining uncovered observation $o \in \mathcal{O} - \mathcal{O}_k^p - \{o'\}$ should be $\delta'_{p_{\text{replan}}} + \frac{\Pr(o'|b,a)(\delta'_{p_{\text{replan}}} - p_{\text{replan}}(\nu_k^p(o')))}{\sum_{o \in \mathcal{O} - \mathcal{O}_k^p - \{o'\}} \Pr(o|b,a)}$ and this new bound is at least $\delta'_{p_{\text{replan}}}$ since $p_{\text{replan}}(\nu_k^p(o')) \leq \delta'_{p_{\text{replan}}}$ according to the definition of $\delta'_{p_{\text{replan}}}$. When the replanning probability bound becomes larger, computing a partial conditional plan is usually less expensive. Therefore, updating the replanning probability bound (line 10) usually improves efficiency and still makes the current partial conditional plan γ_k^p satisfy the original bound $\delta_{p_{\text{replan}}}$.

Safety Guarantee After we construct a valid partial conditional plan $\gamma_k^p = (b, a, \mathcal{O}_k^p, \nu_k^p)$, if the uncovered observation set is not empty ($\mathcal{O} - \mathcal{O}_k^p \neq \emptyset$), then the replanning probability $p_{\text{replan}}(\gamma_k^p) > 0$. Though this replanning probability is bounded by the given bound $\delta_{p_{\text{replan}}}$ and by Theorem 1, we know that the execution failure probability $p_{\text{fail}}(\gamma_k^p)$ is also bounded by $\delta_{p_{\text{replan}}}$. However, if $p_{\text{replan}}(\gamma_k^p) > 0$, during execution the robot might receive an uncovered observation $o \in \mathcal{O} - \mathcal{O}_k^p$ and there are no valid partial conditional plans for this observation o. Then execution fails due to unsuccessful replanning. In this case, though we cannot achieve the safe-reachability objective, a guarantee of the robot still satisfying the safety requirement is preferable to the situation where the robot violates the safety requirement. Our approach OPCPS can provide this safety guarantee by checking whether the successor belief of every uncovered observation $o \in \mathcal{O} - \mathcal{O}_k^p$ of the constructed partial conditional plan γ_k^p is a *safe* belief (lines 17-20).

4 Experiments

We test OPCPS on the kitchen domain (horizon bound $h = 30$) presented in [33] and the classic Tag domain [23] ($h = 100$). We also validate OPCPS on a Fetch robot for the scenario shown in Fig. 1 ($h = 20$). We use Z3 [7] as our backend

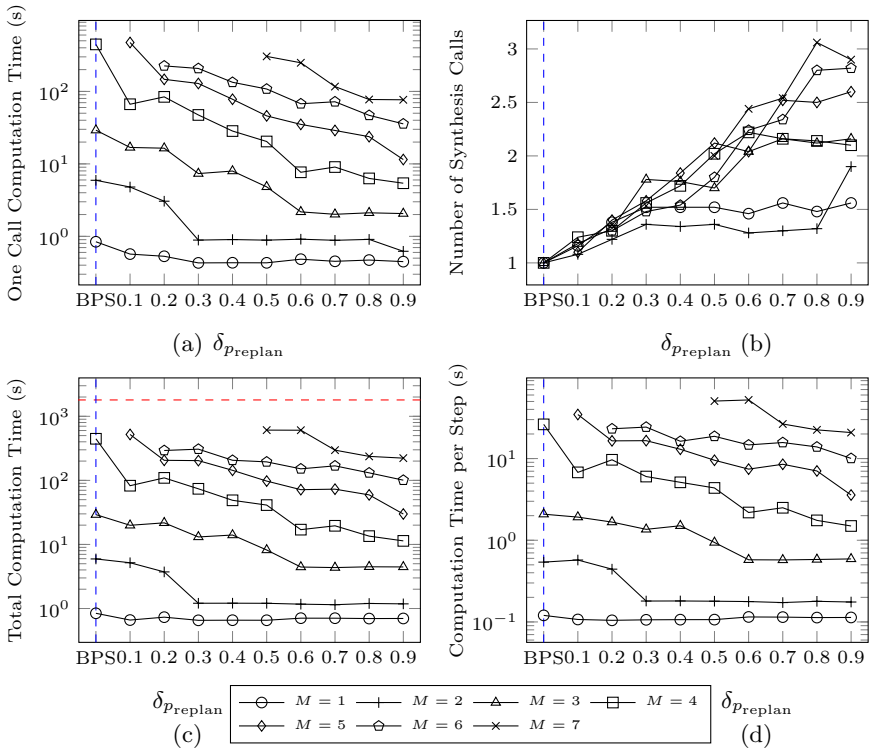


Fig. 5: Performance results for the kitchen domain as the bound $\delta_{p_{\text{replan}}}$ increases. Different plots correspond to tests with different numbers M of obstacles. Missing data points in a plot indicate time-out. The red dashed line is the plot of time = 1800 seconds (time-out). The blue dashed line passes through the data points generated by BPS. All the results are averaged over 50 independent runs.

SMT solver. All experiments were conducted on a 3.0 GHz Intel® processor with 32 GB of memory. We set the time-out to be 1800 seconds. For all the tests of the kitchen and Tag domains, the results are averaged over 50 independent runs.

In the kitchen domain [33], a robot needs to eventually pick up a cup from the storage while avoiding collisions with M uncertain obstacles. This kitchen domain is an example scenario that requires a correctness guarantee of accomplishing tasks, and POMDPs with safe-reachability objectives provide a better correctness guarantee than the quantitative POMDP formulations [33].

The kitchen environment is discretized into $N = 36$ regions. The actuation and perception of the robot are imperfect, modeled as ten uncertain robot actions: *move* and *look* in four directions, pick-up using the left or right hand. We assume that the robot starts at a known initial location. However, due to the robot's imperfect perception, the location of the robot and the locations of obstacles are all partially observable during execution. This kitchen domain has a large state space $|S| = C(N, M) \cdot N$, where $C(N, M)$ is the number of M-combinations from the set of N regions. In the largest test ($M = 7$) there are more than 10^8 states.

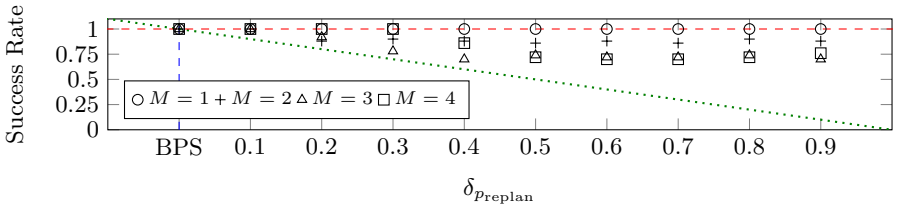


Fig. 6: Success rate as $\delta_{p_{\text{replan}}}$ increases. The green dotted line shows the plot of success rate $= 1.0 - \delta_{p_{\text{replan}}}$. The red dashed line is the plot of success rate $= 1.0$. The blue dashed line passes through the data points generated by BPS.

Performance We evaluate our previous BPS method [33] and OPCPS (with the replanning probability bound $\delta_{p_{\text{replan}}}$ ranging from 0.1 to 0.9) in the kitchen domain with various numbers of obstacles. BPS computes a full conditional plan that covers all observation branches and is equivalent to OPCPS with $\delta_{p_{\text{replan}}} = 0$.

Fig. 5a, 5b, 5c and 5d show the average computation time of one synthesis call, the average number of synthesis calls, the average total computation time and the average computation time per step as the bound $\delta_{p_{\text{replan}}}$ increases, respectively. As shown in from Fig. 5a (semi-log scale) and 5b, the computation time of one synthesis call decreases very quickly while the number of calls to partial conditional plan synthesis does not increase much as $\delta_{p_{\text{replan}}}$ increases. Therefore, the total computation time (Fig. 5c) keeps decreasing as $\delta_{p_{\text{replan}}}$ increases. Additionally, as we can see from Fig. 5c (semi-log scale), with a small bound $\delta_{p_{\text{replan}}} = 0.1$, we observe a big performance gain compared to BPS: for the test case with $M = 4$ obstacles, the speedup is around 5 times and for the test case with $M = 5$ obstacles, BPS times out while OPCPS with $\delta_{p_{\text{replan}}} = 0.1$ can solve this test in around 9 minutes. Therefore, OPCPS achieves better performance than BPS in the tests by computing *partial* conditional plans to approximate *full* conditional plans. The results of the average computation time per step (Fig. 5d) also show the same trend. These results suggest that for domains where replanning is easy, increasing the replanning probability bound usually leads to better scalability.

Success Rate For all the previous performance tests, the constructed partial conditional plans by OPCPS with different bounds $\delta_{p_{\text{replan}}}$ always achieve the safe-reachability objective (success rate $= 100\%$) because the robot can move in four directions. When the robot enters a region surrounded by obstacles in three directions, the robot can always move back to its previous position, which means replanning is always possible. However, in some domains such as autonomous driving and robot chefs, when the robot commits to an action and finds something wrong, it is difficult or impossible to reverse the action effects and replan. To evaluate how OPCPS performs in these scenarios, we test OPCPS in the kitchen domain with different numbers M of obstacles ($M \leq 4$ since BPS times out for tests with more than four obstacles), but we disable the robot's *move-north* action. Therefore, when the robot performs *move-south* and enters a region surrounded by obstacles in three directions, replanning fails. However, the robot still satisfies the safety requirement, thanks to the safety guarantee of OPCPS.

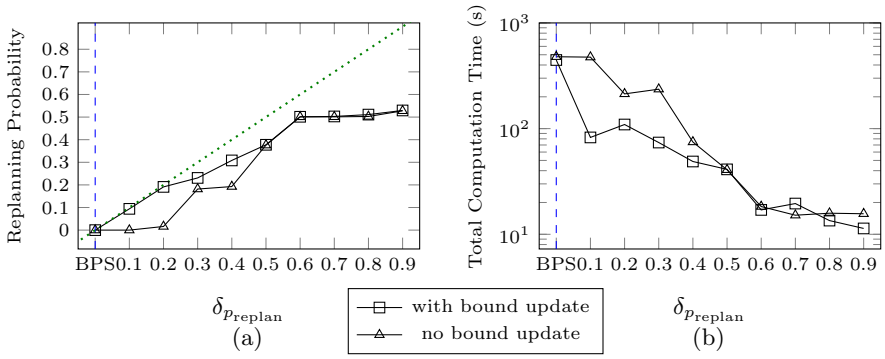


Fig. 7: Replanning probability and total computation time as the bound $\delta_{p_{\text{replan}}}$ increases ($M = 4$). The green dotted line shows the plot of replanning probability = $\delta_{p_{\text{replan}}}$. The blue dashed line passes through the data points generated by BPS.

Fig. 6 shows the success rate as the bound $\delta_{p_{\text{replan}}}$ increases. For all the tests, the success rate is always greater than $1.0 - \delta_{p_{\text{replan}}}$ (all data points are above the plot of success rate = $1.0 - \delta_{p_{\text{replan}}}$). This matches Theorem 1: the failure probability of a valid partial conditional plan is bounded by the replanning probability. Moreover, as the bound $\delta_{p_{\text{replan}}}$ decreases to 0, OPCPS produces a valid full conditional plan with 100% success rate. These results suggest that for some domains where we anticipate that replanning is difficult, users can decrease the bound $\delta_{p_{\text{replan}}}$ and allocate computational resources for a high success rate.

Note that the replanning probability bound is a conservative upper bound of the failure probability since it pessimistically assumes all the uncovered observation branches that require replanning will fail, which is a rare case in practice. As we can see from Fig. 6, even with a high replanning probability bound $\delta_{p_{\text{replan}}} = 0.9$, the success rate is still at least 70% and the failure rate is at most 30%, which is much smaller than the given bound $\delta_{p_{\text{replan}}} = 0.9$.

Gains from Updating Replanning Probability Bound As we discussed in Section 3.1, updating the replanning probability bound during partial conditional plan generation is important for avoiding unnecessary computation and improving efficiency. To evaluate the gains from this bound update step, we test OPCPS with and without bound update in the kitchen domain with $M = 4$ obstacles.

Fig. 7a and 7b (semi-log scale) show the average replanning probability of the constructed partial conditional plans and the average total computation time as the bound $\delta_{p_{\text{replan}}}$ increases, respectively. As shown in Fig. 7a, with both settings (with and without bound update) OPCPS constructs a partial conditional plan with a replanning probability smaller than $\delta_{p_{\text{replan}}}$. However, OPCPS without bound update constructs a partial conditional plan with a lower replanning probability than that constructed by OPCPS with bound update. Therefore, OPCPS without bound update performs unnecessary computation and constructs a partial conditional plan with more branches and thus spends more time than OPCPS with bound update, as shown in Fig. 7b. For the tests with

(a) (b) (c) (d) (e)

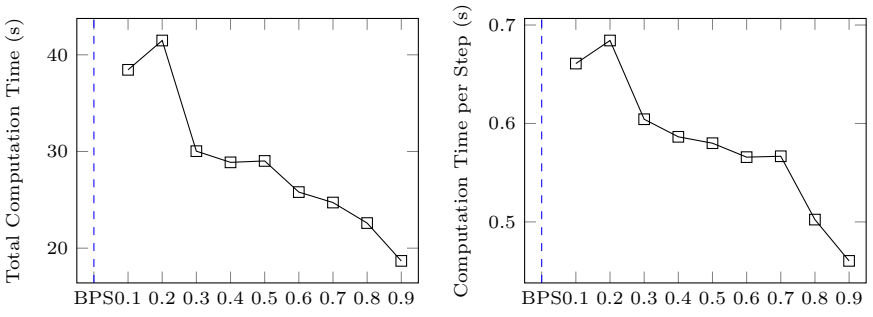
Fig. 8: Physical validation of OPCPS for the domain shown in Fig. 1.

$\delta_{p_{\text{replan}}} = 0.1, 0.2, 0.3$ that take more time to solve than those with $\delta_{p_{\text{replan}}} > 0.3$, OPCPS with bound update achieves a 2-5 times speedup.

Physical Validation We validate OPCPS on a Fetch robot for the domain shown in Fig. 1. The setup of this domain is similar to the kitchen domain. The Fetch needs to pick up a target object (the blue can on the table) while avoiding collisions with uncertain obstacles such as floor signs and file cabinets, which can be placed in different locations. The POMDP's state space consists of robot locations and object locations. We use a Vicon system to detect object locations, which is usually accurate but can still produce false negative and false positive due to occlusion or inappropriate Vicon marker configurations on objects. We estimate the false negative and false positive probabilities by counting the false negative and false positive events during 100 Vicon detections. The POMDP's probabilistic observation function is defined based on the false negative and false positive probabilities. To test the effects of different replanning probability bounds, we only allow the Fetch to move in three directions (west, east and south), similar to the setup in the previous success rate tests. Sometimes the Fetch may fail to move its base when given a *move* action command and stay in the same place. We estimate the failure probability of these move actions by counting the failure events during 100 *move* action executions. The POMDP's probabilistic transition function is defined based on this failure probability. Fig. 8a shows the initial state. There are two uncertain obstacles (a wet-floor sign and a file cabinet). We test OPCPS with two bounds $\delta_{p_{\text{replan}}} = 0.9$ and $\delta_{p_{\text{replan}}} = 0.1$.

With $\delta_{p_{\text{replan}}} = 0.9$, after observing no obstacle in the south direction, the Fetch decides to move south (Fig. 8b) because the partial conditional plan constructed with a high replanning probability bound does not cover the case where the Fetch is surrounded by obstacles and the wall. Then replanning fails but the Fetch still satisfies the safety requirement as shown in Fig. 8b, thanks to the safety guarantee provided by OPCPS.

However, with $\delta_{p_{\text{replan}}} = 0.1$, after observing no obstacles in the south direction, the Fetch decides to move west (Fig. 8c) because the partial conditional plan constructed with a low replanning probability bound covers the case where the robot is surrounded by obstacles. In order to avoid this situation, the Fetch needs to move west and gather more information. Then the Fetch observes an obstacle in the south direction and decides to move west again (Fig. 8d). Next, the Fetch observes no obstacle in the south direction, and now it can move south. Unlike the case shown in Fig. 8b where the robot is surrounded by two obstacles and



(a) $\delta_{p_{\text{replan}}}$ (b) $\delta_{p_{\text{replan}}}$

Fig. 9: Performance results for the Tag domain as the replanning probability bound $\delta_{p_{\text{replan}}}$ increases. All the results are averaged over 50 independent runs.

the wall, in the situation shown in Fig. 8d, if there is another obstacle in the south direction, the Fetch can still move west since there are only two obstacles. Finally, the Fetch moves to the table and picks up the target object (Fig. 8e).

Tag Domain To further demonstrate the advantage of OPCPS over our previous BPS method, we evaluate OPCPS on a classic POMDP domain [23]. The task for the robot is to search for and tag a moving agent in a grid with 29 locations. The agent follows a fixed strategy that intentionally moves away from the robot. Both the robot and the agent can move in four directions or stay. The robot's location is fully observable while the agent's location is unobservable unless the robot and the agent are in the same location.

This Tag domain is challenging for BPS because of a large number of observations ($|\mathcal{O}| = 30$) and more importantly, a huge planning horizon for computing a full conditional plan. However, computing a full conditional plan is unnecessary since replanning is easy in this domain. Fig. 9a and 9b show the average total computation time and the average computation time per step for the Tag domain as the bound $\delta_{p_{\text{replan}}}$ increases. These results show a similar trend to the previous kitchen domain tests: with a small bound $\delta_{p_{\text{replan}}} = 0.1$, we observe a big performance gain compared to BPS. BPS cannot solve this test within the 1800-second time limit while OPCPS with $\delta_{p_{\text{replan}}} = 0.1$ can solve this test in around 40 seconds and the computation time per step is less than 1 second.

5 Discussion

We presented a new approach, called OPCPS, to policy synthesis for POMDPs with safe-reachability objectives. We introduce the notion of a *partial conditional plan* to improve computational efficiency. Rather than explicitly enumerating all possible observations to construct a *full* conditional plan, OPCPS samples a subset of all observations to ensure bounded replanning probability. Our theoretical and empirical results show that the failure probability of a valid partial conditional

plan is bounded by the replanning probability. Moreover, OPCPS guarantees that the robot still satisfies the safety requirement when replanning fails. Compared to our previous BPS method [33], OPCPS with a proper replanning probability bound scales better in the tested domains and can solve problems that are beyond the capabilities of BPS within the time limit. The results also suggest that for domains where replanning is easy, increasing the replanning probability bound usually leads to better scalability, and for domains where replanning is difficult or impossible in some states, we can decrease the replanning probability bound and allocate more computation time to achieve a higher success rate. Our results also indicate that by updating the replanning probability bound during partial conditional plan generation, we can quickly detect if the current partial conditional plan satisfies the bound and avoid unnecessary computation.

In this work, we focus on discrete POMDPs. While many robot applications can be modeled using this discrete representation, discretization often suffers from the curse of dimensionality. Investigating how to deal with continuous POMDPs [1, 9, 24, 28] directly is a promising future direction. The current implementation of OPCPS constructs partial conditional plans by sampling observations according to the probability of occurrence (Algorithm 1, line 11), which does not consider the importance of observations [17]. How to extend OPCPS to handle critical observations is another important ongoing question.

Acknowledgments This work was supported in part by NSF CCF 1139011, NSF CCF 1514372, NSF CCF 1162076 and NSF IIS 1317849. We thank the reviewers for their insightful comments, and Juan David Hernández, Bryce Willey and Constantinos Chamzas for their assistance in the physical experiments.

References

1. Bai, H., Hsu, D., Lee, W.S.: Integrated perception and planning in the continuous space: A POMDP approach. International Journal of Robotics Research 33(9), 1288–1302 (2014)
2. Biere, A., Cimatti, A., Clarke, E.M., Strichman, O., Zhu, Y.: Bounded model checking. Advances in Computers 58, 117–148 (2003)
3. Cai, P., Luo, Y., Hsu, D., Lee, W.S.: HyP-DESPOT: A hybrid parallel algorithm for online planning under uncertainty. In: RSS (2018)
4. Chatterjee, K., Chmelík, M., Gupta, R., Kanodia, A.: Qualitative analysis of POMDPs with temporal logic specifications for robotics applications. In: ICRA. pp. 325–330 (2015)
5. Chatterjee, K., Chmelík, M., Gupta, R., Kanodia, A.: Optimal cost almost-sure reachability in POMDPs. Artificial Intelligence 234(C), 26–48 (2016)
6. Chatterjee, K., Chmelík, M., Tracol, M.: What is decidable about partially observable Markov decision processes with ω-regular objectives. Journal of Computer and System Sciences 82(5), 878–911 (2016)
7. De Moura, L., Bjørner, N.: Z3: An efficient SMT solver. In: TACAS. pp. 337–340 (2008)
8. Hadfield-Menell, D., Groshev, E., Chitnis, R., Abbeel, P.: Modular task and motion planning in belief space. In: IROS. pp. 4991–4998 (2015)

9. Hoey, J., Poupart, P.: Solving POMDPs with continuous or large discrete observation spaces. In: IJCAI. pp. 1332–1338 (2005)
10. Hou, P., Yeoh, W., Varakantham, P.: Solving risk-sensitive POMDPs with and without cost observations. In: AAAI. pp. 3138–3144 (2016)
11. Isom, J.D., Meyn, S.P., Braatz, R.D.: Piecewise linear dynamic programming for constrained POMDPs. In: AAAI. pp. 291–296 (2008)
12. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artificial Intelligence 101(1-2), 99–134 (1998)
13. Kaelbling, L.P., Lozano-Pérez, T.: Integrated task and motion planning in belief space. International Journal of Robotics Research 32(9-10), 1194–1227 (2013)
14. Kaelbling, L.P., Lozano-Pérez, T.: Implicit belief-space pre-images for hierarchical planning and execution. In: ICRA. pp. 5455–5462 (2016)
15. Kim, D., Lee, J., Kim, K.E., Poupart, P.: Point-based value iteration for constrained POMDPs. In: IJCAI. pp. 1968–1974 (2011)
16. Kurniawati, H., Hsu, D., Lee, W.S.: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: RSS (2008)
17. Luo, Y., Bai, H., Hsu, D., Lee, W.S.: Importance sampling for online planning under uncertainty. WAFR (2016)
18. Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. Artificial Intelligence 147(1-2), 5–34 (2003)
19. Marecki, J., Varakantham, P.: Risk-sensitive planning in partially observable environments. In: AAMAS. pp. 1357–1368 (2010)
20. Mundhenk, M., Goldsmith, J., Lusena, C., Allender, E.: Complexity of finite-horizon Markov decision process problems. Journal of the ACM 47(4), 681–720 (2000)
21. Papadimitriou, C., Tsitsiklis, J.N.: The complexity of Markov decision processes. Mathematics of Operations Research 12(3), 441–450 (1987)
22. Paz, A.: Introduction to Probabilistic Automata. Academic Press, Inc. (1971)
23. Pineau, J., Gordon, G., Thrun, S.: Point-based value iteration: An anytime algorithm for POMDPs. In: IJCAI. pp. 1025–1030 (2003)
24. Porta, J.M., Vlassis, N., Spaan, M.T.J., Poupart, P.: Point-based value iteration for continuous POMDPs. Journal of Machine Learning Research 7, 2329–2367 (2006)
25. Poupart, P., Malhotra, A., Pei, P., Kim, K.E., Goh, B., Bowling, M.: Approximate linear programming for constrained partially observable Markov decision processes. In: AAAI. pp. 3342–3348 (2015)
26. Ross, S., Pineau, J., Paquet, S., Chaib-draa, B.: Online planning algorithms for POMDPs. Journal of Artificial Intelligence Research 32(1), 663–704 (2008)
27. Santana, P., Thiébaux, S., Williams, B.: RAO*: An algorithm for chance-constrained POMDP's. In: AAAI. pp. 3308–3314 (2016)
28. Seiler, K.M., Kurniawati, H., Singh, S.P.N.: An online and approximate solver for POMDPs with continuous action space. In: ICRA. pp. 2290–2297 (2015)
29. Smallwood, R.D., Sondik, E.J.: The optimal control of partially observable Markov processes over a finite horizon. Operations Research 21(5), 1071–1088 (1973)
30. Somani, A., Ye, N., Hsu, D., Lee, W.S.: DESPOT: Online POMDP planning with regularization. In: NIPS. pp. 1772–1780 (2013)
31. Svoreňová, M., Chmelík, M., Leahy, K., Eniser, H.F., Chatterjee, K., Černá, I., Belta, C.: Temporal logic motion planning using POMDPs with parity objectives: Case study paper. In: HSCC. pp. 233–238 (2015)
32. Undurti, A., How, J.P.: An online algorithm for constrained POMDPs. In: ICRA. pp. 3966–3973 (2010)
33. Wang, Y., Chaudhuri, S., Kavraki, L.E.: Bounded policy synthesis for POMDPs with safe-reachability objectives. In: AAMAS. pp. 238–246 (2018)

A Proof of Theorem 1

Proof. We prove Theorem 1 by induction. First we define $\delta_{\text{fail}}(b) : \mathcal{B} \mapsto \{0, 1\}$ as an indicator and when $\delta_{\text{fail}}(b) = 1$, there are no valid partial conditional plans for belief b and exeution fails.

- Base case ($k = 1$): Since $\gamma_1^p = (b, a, \mathcal{O}_1^p, \emptyset)$ is valid, for every covered observation $o \in \mathcal{O}_1^p$, $b' = \mathcal{T}_{\mathcal{B}}(b, a, o) \in \mathit{Dest}$ is the terminal goal belief and thus $\delta_{\text{fail}}(b') = 0$. Therefore,

$$\begin{aligned} p_{\text{fail}}(\gamma_1^p) &= \sum_{o \in \mathcal{O} - \mathcal{O}_1^p} \Pr(o|b, a) \delta_{\text{fail}}(b') \\ &\leq \sum_{o \in \mathcal{O} - \mathcal{O}_1^p} \Pr(o|b, a) = p_{\text{replan}}(\gamma_1^p) \end{aligned}$$

  since $\delta_{\text{fail}}(b') \leq 1$ where $b' = \mathcal{T}_{\mathcal{B}}(b, a, o)$ is the successor belief for the uncovered observation $o \in \mathcal{O} - \mathcal{O}_1^p$.
- Inductive case ($k > 1$): Since $\gamma_k^p = (b, a, \mathcal{O}_k^p, \nu_k^p)$ is valid, for every covered observation $o \in \mathcal{O}_k^p$, the corresponding $(k-1)$-step partial conditional plan $\nu_k^p(o)$ is also valid. Assume $p_{\text{fail}}(\nu_k^p(o)) \leq p_{\text{replan}}(\nu_k^p(o))$, then

$$\begin{aligned} p_{\text{fail}}(\gamma_k^p) &= \sum_{o \in \mathcal{O}_k^p} \Pr(o|b, a) p_{\text{fail}}(\nu_k^p(o)) + \sum_{o \in \mathcal{O} - \mathcal{O}_k^p} \Pr(o|b, a) \delta_{\text{fail}}(b') \\ &\leq \sum_{o \in \mathcal{O}_k^p} \Pr(o|b, a) p_{\text{replan}}(\nu_k^p(o)) + \sum_{o \in \mathcal{O} - \mathcal{O}_k^p} \Pr(o|b, a) \\ &= p_{\text{replan}}(\gamma_k^p) \end{aligned}$$

  since $\delta_{\text{fail}}(b') \leq 1$ where $b' = \mathcal{T}_{\mathcal{B}}(b, a, o)$ is the successor belief for the uncovered observation $o \in \mathcal{O} - \mathcal{O}_k^p$.

Therefore, For any k-step valid partial conditional plan $\gamma_k^p = (b, a, \mathcal{O}_k^p, \nu_k^p)$, $p_{\text{fail}}(\gamma_k^p) \leq p_{\text{replan}}(\gamma_k^p)$.

□