# Bounded Policy Synthesis for POMDPs with Safe-Reachability Objectives

Robotics Track

Yue Wang
Department of Computer Science
Rice University
Houston, TX 77005, USA
yw27@rice.edu

Swarat Chaudhuri
Department of Computer Science
Rice University
Houston, TX 77005, USA
swarat@rice.edu

Lydia E. Kavraki
Department of Computer Science
Rice University
Houston, TX 77005, USA
kavraki@rice.edu

## ABSTRACT

Planning robust executions under uncertainty is a fundamental challenge for building autonomous robots. Partially Observable Markov Decision Processes (POMDPs) provide a standard framework for modeling uncertainty in many applications. In this work, we study POMDPs with *safe-reachability* objectives, which require that with a probability above some threshold, a goal state is eventually reached while keeping the probability of visiting unsafe states below some threshold. This POMDP formulation is different from the traditional POMDP models with optimality objectives and we show that in some cases, POMDPs with safe-reachability objectives can provide a better guarantee of both safety and reachability than the existing POMDP models through an example. A key algorithmic problem for POMDPs is *policy synthesis*, which requires reasoning over a vast space of beliefs (probability distributions). To address this challenge, we introduce the notion of a *goal-constrained belief space*, which only contains beliefs reachable from the initial belief under desired executions that can achieve the given safe-reachability objective. Our method compactly represents this space over a *bounded* horizon using symbolic constraints, and employs an *incremental* Satisfiability Modulo Theories (SMT) solver to efficiently search for a valid policy over it. We evaluate our method using a case study involving a partially observable robotic domain with uncertain obstacles. The results show that our method can synthesize policies over large belief spaces with a small number of SMT solver calls by focusing on the goal-constrained belief space.

## KEYWORDS

Planning under Uncertainty; Policies; Robotics; Formal Methods

**ACM Reference Format:**
Yue Wang, Swarat Chaudhuri, and Lydia E. Kavraki. 2018. Bounded Policy Synthesis for POMDPs with Safe-Reachability Objectives. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 9 pages.

## 1 INTRODUCTION

Partially Observable Markov Decision Processes (POMDPs) [38] provide a principled mathematical framework for modeling a variety of problems in the face of uncertainty [5, 11, 22, 29]. As an example, in robotics, accounting for uncertainty is a fundamental challenge for deploying autonomous robots in the physical world. Many applications in uncertain robotic domains can be modeled as POMDP problems [5, 6, 15, 22].

*Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.* 

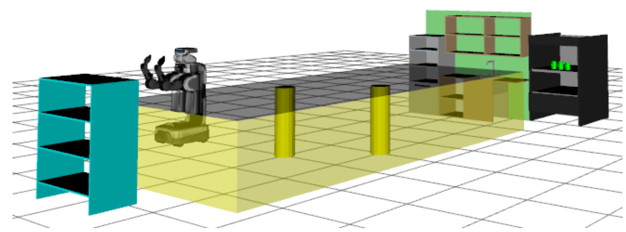

Figure 1: An example of a safe-reachability objective: a robot with uncertain actuation and perception needs to navigate through the kitchen and pick up a green cup from the black storage area (reachability), while avoiding collisions with uncertain obstacles (e.g., chairs) modeled as cylinders in the yellow "shadow" region (safety).

A key algorithmic problem for POMDPs is the synthesis of *policies* [38]: recipes that specify the actions to take under *all possible* events in the environment. Typically, the goal in policy synthesis is to find optimal solutions with respect to quantitative objectives such as maximizing discounted reward [1, 12, 15–17, 25, 26, 39, 40]. While the purely quantitative formulations of the problem are suitable for many applications, there are, settings that demand synthesis with respect to *boolean* requirements. For example, consider the scenario shown in Figure 1 where we want to guarantee that a robot can accomplish a task *safely* in an uncertain domain. This goal is naturally formulated as policy synthesis from a high-level requirement written in a temporal logic. Moreover, in some cases, formulating boolean requirements as quantitative objectives by assigning negative rewards for states that violate the boolean requirements and positive rewards for states that satisfy the boolean requirements, leads to policies that are overly conservative or overly risky [43], depending on the particular reward function chosen. Therefore, new models and algorithms are required for handling POMDPs with boolean requirements explicitly. In Section 4, we discuss an example that shows in some scenarios, handling boolean requirements explicitly in POMDPs provides a better guarantee of both safety and reachability than the traditional quantitative POMDP formulations.

Policy synthesis in POMDPs with respect to boolean requirements has been studied before. Specifically, inspired by applications in robotics, the *qualitative analysis* problem of *almost-sure satisfaction* of POMDPs with temporal logic specifications was first introduced in [6]. In their work, the goal is to find policies that satisfy a temporal property with probability 1.

A more general *quantitative analysis* problem of POMDPs with temporal logic specifications is to synthesize policies that satisfy a temporal property with a probability above some threshold. In this work, we study this problem for the special case of *safe-reachability* properties, which require that with a probability above some threshold, a goal state is eventually reached while keeping the probability of visiting unsafe states below some threshold. Many robot tasks such as the one in Figure 1 can be formulated using a safe-reachability objective.

Previous results [7, 27, 33] have shown that the quantitative analysis problem of POMDPs with reachability objectives is undecidable. To make the problem tractable, we assume there exists a *bounded horizon* $h$ such that $h$ is sufficiently large to prove the existence of a valid policy, or the user is not interested in plans beyond the bounded horizon $h$. This assumption is particularly reasonable for robotic domains because robots are often required to accomplish a task in bounded steps due to some resource constraints such as energy/time constraints. Figure 1 shows an example of such a scenario: a robot with uncertain actuation and perception needs to navigate through a kitchen to pick up an object in bounded steps while avoiding collisions with uncertain obstacles.

To the best of our knowledge, the quantitative analysis problem of POMDPs with safe-reachability objectives has not been considered before. In this work, we present a practical policy synthesis approach for this problem. Like most other algorithms for policy synthesis, our approach is based on reasoning about the space of *beliefs*, or probability distributions over possible states of the POMDP. Our primary algorithmic challenge is that the belief space is a vast, high-dimensional space of probability distributions.

Our approach to this challenge is based on the new notion of a *goal-constrained belief space*. This notion takes inspiration from recent advances in point-based algorithms [25, 26, 34, 39] for POMDPs with discounted reward objectives. These POMDP algorithms exploit the notion of the *reachable belief space* $\mathcal{R}(b_{init})$ from an initial belief $b_{init}$ and compute an approximately optimal policy over $\mathcal{R}(b_{init})$ rather than the entire belief space. Similarly, we compute a valid policy over a *goal-constrained belief space*, which contains beliefs visited by desired executions that can achieve the safe-reachability objective. The goal-constrained belief space is generally much smaller than the original belief space.

Our synthesis algorithm, *bounded policy synthesis* (BPS), computes a valid policy by iteratively searching for a candidate plan in the goal-constrained belief space and constructing a policy from this candidate plan. We compactly represent the goal-constrained belief space over a *bounded* horizon using symbolic constraints. The applicability of constraint-based methods has been already advocated in several robotics planning algorithms [8, 23, 31, 44]. Many of these algorithms take advantage of a modern, incremental SMT solver [9] for efficiency. Inspired by this, we apply the SMT solver to efficiently explore the symbolic goal-constrained belief space to generate candidate plans. Note that a candidate plan is a single path that only covers a particular observation at each step, while a valid policy is contingent on all possible observations. Therefore, once a candidate plan is found, BPS tries to generate a valid policy from the candidate plan by considering all possible events at each step. If this policy generation fails, BPS adds additional constraints that *block* invalid plans and force the SMT solver to generate other better plans. The incremental capability of the SMT solver allows BPS to efficiently generate alternate candidate plans when we update the constraints. If there is no new candidate plan for the current horizon, BPS increases the horizon and repeats the above steps until it finds a valid policy or reaches a given horizon bound.

In summary, the contributions of the paper are:

- We show that in some domains, our formulation of POMDPs with safe-reachability objectives offers a better guarantee of both safety and reachability than the existing POMDP models through an example (Section 4).
- We introduce the notion of a goal-constrained belief space to address the scalability challenge of solving POMDPs with safe-reachability objectives. Based on this notion, we present a novel approach called BPS for policy synthesis of POMDPs with safe-reachability objectives.
- We evaluate the scalability of BPS using a case study involving a partially observable robotic domain with uncertain obstacles (Figure 1). The experimental results demonstrate that BPS can scale up to huge belief spaces by focusing on the goal-constrained belief space.

## 2 RELATED WORK

POMDPs [38] provide a principled mathematical framework for modeling a variety of robotics problems in the face of uncertainty. Many POMDP algorithms [1, 15, 25, 26, 39, 40] for robot applications focus on discounted reward objectives. Recent work [5, 6, 42] has investigated *almost-sure satisfaction* of POMDPs with temporal logic specifications, where the goal is to check whether a temporal logic objective can be ensured with probability 1. Our approach can be seen as synthesizing policies for large POMDP problems with basic temporal logic objectives (safe-reachability), but not limited to almost-sure satisfaction analysis. Though we may formulate a safe-reachability objective as an optimization problem by assigning negative rewards for unsafe states and positive rewards for goal states, this formulation does not always yield good policies [43].

Recently, there has been a large body of work that extends the traditional POMDP model with notions of risk and cost, including constrained POMDPs (C-POMDPs) [21, 24, 36, 43], risk-sensitive POMDPs (RS-POMDPs) [20, 28] and chance-constrained POMDPs (CC-POMDPs) [37]. There are two major differences between their models and our formulation of POMDPs with safe-reachability objectives. First, the objective of these models is to maximize the cumulative expected reward while keeping the expected cost/risk below some threshold, while in our case, the objective is to satisfy a safe-reachability objective in all possible executions including the worst case, providing a better safety guarantee than the formulation of expected cost/risk threshold constraints. Second, C/RS/CC-POMDPs typically need to assign a proper positive reward for goal states to ensure reachability and do not have direct control over the probability of reaching goal states (e.g., reach a goal state with a probability greater than some threshold), while our safe-reachability objective can directly encode this probability threshold constraint as a boolean requirement, providing a better reachability guarantee than the quantitative formulation of C/RS/CC-POMDPs. While C/RS/CC-POMDPs are suitable for many applications, there are domains in robotics such as autonomous driving and disaster rescue that demand synthesis of policies that can provide such strong guarantee of reaching goal states safely.

Task and Motion Planning (TMP) [2, 8, 13, 14, 18, 19, 23, 41, 44] describes a class of challenging problems that combine low-level motion planning and high-level task reasoning. Most of these TMP approaches focus on deterministic domains, while several of them apply to uncertain domains with uncertainty in perception [18, 23]. The main difference is that, the above works perform *online* planning with a determinized approximation of belief space dynamics [35] assuming the most likely observation will be obtained, while our approach synthesizes a valid policy *offline* contingent on all possible events.

Our method computes a valid policy by iteratively searching for a candidate plan that is likely to succeed with determinized observations in the goal-constrained belief space, and then constructing a policy from this candidate plan by considering other possible observations. This idea has been shown to improve the scalability of algorithms for a variety of uncertain domains [4, 10, 30]. The scalability of our approach also relies on exploiting the notion of a *goal-constrained belief space*. This idea resembles efficient point-based POMDP algorithms [25, 26] based on (optimally) reachable belief space.

We apply techniques from Bounded Model Checking (BMC) [3] to compactly represent the goal-constrained belief space over a bounded horizon. BMC verifies whether a finite state system satisfies a given temporal logic specification. Thanks to the tremendous increase in the reasoning power of practical SMT (SAT) solvers, BMC can scale up to large systems with hundreds of thousands of states. Our approach efficiently explores the goal-constrained belief space by leveraging a modern, incremental SMT solver [9]. It has been shown that the incremental capability of the SMT solver leads to an efficient planning algorithm for TMP [8]. Inspired by this result, we now leverage incremental SMT solvers for belief space policy synthesis.

# 3 PROBLEM FORMULATION

In this work, we consider the problem of policy synthesis for POMDPs:

*Definition 3.1 (POMDP).*
A *Partially Observable Markov Decision Process* (POMDP) is a tuple $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$:

- $\mathcal{S}$ is a *finite* set of states.
- $\mathcal{A}$ is a *finite* set of actions.
- $\mathcal{T}$ is a probabilistic transition function $\mathcal{T}(s, a, s') = p(s'|s, a)$, which defines the probability of moving to state $s' \in \mathcal{S}$ after taking an action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$.
- $\mathcal{O}$ is a *finite* set of observations.
- $\mathcal{Z}$ is the probabilistic observation function $\mathcal{Z}(s', a, o) = p(o|s', a)$, which defines the probability of observing $o \in \mathcal{O}$ after taking an action $a \in \mathcal{A}$ and reaching state $s' \in \mathcal{S}$.

Due to uncertainty in transition and observation, the actual state is partially observable and typically we maintain a *belief*, which is a probability distribution over all possible states $b : \mathcal{S} \to [0, 1]$ with $\sum_{s \in \mathcal{S}} b(s) = 1$. The set of beliefs $\mathcal{B} = \{b : \mathcal{S} \to [0, 1] \mid \sum_{s \in \mathcal{S}} b(s) = 1\}$ is known as *belief space*. Note that a transition $\mathcal{T}_B$ in belief space is a *deterministic* function $b' = \mathcal{T}_B(b, a, o)$, i.e., given an action $a \in \mathcal{A}$ and an observation $o \in \mathcal{O}$, the updates to beliefs are deterministic based on the formula:

$$b'(s') = \alpha \mathcal{Z}(s', a, o) \sum_{s \in \mathcal{S}} \mathcal{T}(s, a, s') b(s) \tag{1}$$

where $\alpha$ is a normalization constant.

*Definition 3.2 (Plan).*
A *plan* in belief space is a sequence $\sigma = (b_0, a_1, o_1, b_1, a_2, o_2, b_2, \dots)$ such that for all $i > 0$, the belief updates satisfy the transition function $\mathcal{T}_B$, i.e., $b_i = \mathcal{T}_B(b_{i-1}, a_i, o_i)$, where $a_i \in \mathcal{A}$ is an action and $o_i \in \mathcal{O}$ is an observation.

*Definition 3.3 (Policy).*
A *policy* $\pi : \mathcal{B} \to \mathcal{A}$ is a function that maps a belief $b \in \mathcal{B}$ to an action $a \in \mathcal{A}$. A policy $\pi$ defines a set of plans in belief space: $\Omega_\pi = \{\sigma = (b_0, a_1, o_1, \dots) \mid \forall i > 0, a_i = \pi(b_{i-1}) \textit{ and } o_i \in \mathcal{O}\}$. For each plan $\sigma \in \Omega_\pi$, the action $a_i$ at each step $i$ is chosen by the policy $\pi$.

## 3.1 Safe-Reachability Objective

In this work, we consider POMDPs with *safe-reachability* objectives:

*Definition 3.4 (Safe-Reachability Objective).*
A *safe-reachability objective* is a tuple $\mathcal{G} = (Dest, Safe)$:

- *Safe* is a set of safe beliefs
- *Dest* is a set of goal beliefs. In general, goal beliefs are safe beliefs, i.e., $Dest \subseteq Safe$.

A safe-reachability objective $\mathcal{G}$ compactly represents the set $\Omega_\mathcal{G}$ of satisfiable plans in belief space:

*Definition 3.5 (Satisfiable Plan).*
A plan $\sigma = (b_0, a_1, o_1, \dots)$ *satisfies* a safe-reachability objective $\mathcal{G} = (Dest, Safe)$ if there exists a belief $b_k$ at step $k$ in the plan $\sigma$ that is a goal belief $b_k \in Dest$ and all the beliefs $b_i$ $(i < k)$ visited before step $k$ are safe beliefs $b_i \in Safe$.

Note that safe-reachability objectives are defined using sets of beliefs (probability distributions). The quantitative analysis problem of POMDPs with requirements of a goal state is eventually reached with a probability above some threshold while keeping the probability of visiting unsafe states below some threshold, can be easily formulated as a safe-reachability objective $\mathcal{G} = (Dest, Safe)$ defined as follows:

$$Dest = \{b \in \mathcal{B} \mid \left( \sum_{s \text{ is a goal state}} b(s) \right) > 1 - \delta_1\} \tag{2}$$

$$Safe = \{b \in \mathcal{B} \mid \left( \sum_{s \text{ violates safety}} b(s) \right) < \delta_2\} \tag{3}$$

Where $\delta_1$ and $\delta_2$ are a small values that represents tolerance.

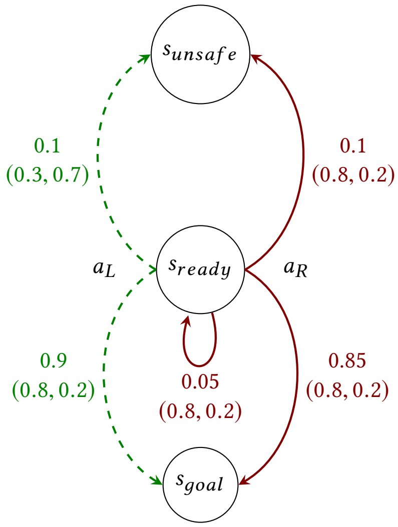


**Figure 2: An example to show the difference between our formulation of POMDPs with safe-reachability objectives and unconstrained/C/RS/CC-POMDPs. There are 3 states: start state $s_{\text{ready}}$, unsafe state $s_{\text{unsafe}}$ and goal state $s_{\text{goal}}$. Dashed green edges represent transitions of executing left-hand pick-up action $a_L$ in state $s_{\text{ready}}$ and solid red edges represent transitions of executing right-hand pick-up action $a_R$ in state $s_{\text{ready}}$. For each edge, the first line is the transition probability and the second line is the tuple of observation probabilities $(p_{o_{\text{pos}}}, p_{o_{\text{neg}}})$.**

## 3.2 Goal-Constrained Belief Space

It is intractable to compute a full policy that satisfies a given safe-reachability objective for POMDPs, even under the assumption of bounded horizon, due to the curse of dimensionality [32]: the belief space $\mathcal{B}$ is a high-dimensional, continuous space that contains an infinite number of beliefs.

However, the *reachable belief space* [25] $\mathcal{R}(b_{init})$ that contains beliefs reachable from the given initial belief $b_{init}$, is much smaller than $\mathcal{B}$ in general. Moreover, the safe-reachability objective $\mathcal{G}$ defines a set $\Omega_{\mathcal{G}}$ of plans that satisfy $\mathcal{G}$. Combining $\mathcal{R}(b_{init})$ and $\Omega_{\mathcal{G}}$, we can construct a *goal-constrained belief space* $\mathcal{R}^*(b_{init}, \mathcal{G})$ that contains beliefs reachable from the initial belief $b_{init}$ under satisfiable plans $\sigma \in \Omega_{\mathcal{G}}$. The *goal-constrained belief space* $\mathcal{R}^*(b_{init}, \mathcal{G})$ is usually much smaller than the reachable belief space $\mathcal{R}(b_{init})$. Thus, computing policies over the goal-constrained belief space $\mathcal{R}^*(b_{init}, \mathcal{G})$ can lead to a substantial gain in efficiency.

## 3.3 Problem Statement

Given a POMDP $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$, an initial belief $b_{init}$ and a safe-reachability objective $\mathcal{G}$, our goal is to synthesize a *valid* policy $\pi_{\mathcal{R}^*}$ over the corresponding goal-constrained belief space $\mathcal{R}^*(b_{init}, \mathcal{G})$:

*Definition 3.6 (Valid Policy).*
A *valid* policy $\pi_{\mathcal{R}^*} : \mathcal{R}^*(b_{init}, \mathcal{G}) \mapsto \mathcal{A}$ over a goal-constrained belief space is a function that maps a belief $b \in \mathcal{R}^*(b_{init}, \mathcal{G})$ to an action $a \in \mathcal{A}$. Therefore, the set $\Omega_{\pi_{\mathcal{R}^*}}$ of plans defined by the policy $\pi_{\mathcal{R}^*}$ is a subset of the set $\Omega_{\mathcal{G}}$ defined by the safe-reachability objective $\mathcal{G}$. i.e., $\Omega_{\pi_{\mathcal{R}^*}} \subseteq \Omega_{\mathcal{G}}$.

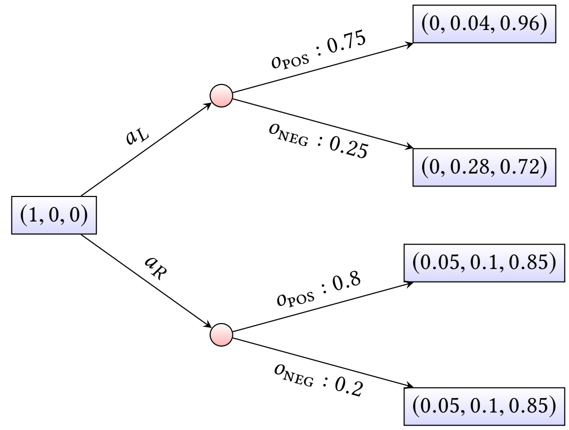


**Figure 3: The belief space transition for the POMDP in Figure 2. Blue nodes $(p_{s_{\text{ready}}}, p_{s_{\text{unsafe}}}, p_{s_{\text{goal}}})$ represent beliefs (probability distribution over states) and red nodes represent observation. The edges from blue nodes to red nodes represent actions and the edges from red nodes to blue nodes represent observations and the corresponding probabilities.**

# 4 RELATION TO UNCONSTRAINED POMDPS AND C/RS/CC-POMDPS

There are two distinct approaches that can model safe-reachability objectives *implicitly* using the existing POMDP models in the literature. The first approach is to incorporate safety and reachability constraints as negative penalties for unsafe states and positive rewards for goal states in *unconstrained* POMDPs with quantitative objectives. However, the authors of [43] have shown a counterexample that demonstrates formulating constraints as unconstrained POMDPs with quantitative objectives does not always yield good policies. The second approach is to encode safe-reachability objectives implicitly as C/RS/CC-POMDPs that extend unconstrained POMDPs with notions of risk and cost [20, 21, 24, 28, 36, 37, 43]. In this section, we show the differences between POMDPs with safe-reachability objectives and unconstrained/C/RS/CC-POMDPs through an example.

In Figure 1, after the robot passes the yellow "shadow" region and moves to the position where it is ready to pick up a green cup from the black storage area (start state $s_{\text{ready}}$), it needs to decide how to pick up the object. There are two action choices: pick-up using the left hand (action $a_L$) and pick-up using the right hand (action $a_R$). Both $a_L$ and $a_R$ are uncertain, and the robot may hit the storage while executing $a_L$ or $a_R$, which results in an unsafe collision state $s_{\text{unsafe}}$. There are two possible observations after executing $a_L$ or $a_R$: observation $o_{\text{pos}}$ representing the robot observes a cup in its hand and observation $o_{\text{neg}}$ representing the robot observes no cup in its hand (Note that the actual state may be different from the observation due to uncertainty). The task objective is to reach a goal state $s_{\text{goal}}$ where the robot holds a cup in its hand with a probability greater than 0.8 (reachability) while

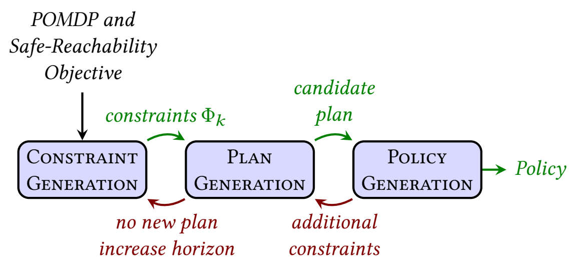


**Figure 4: The core steps of the BPS algorithm.**

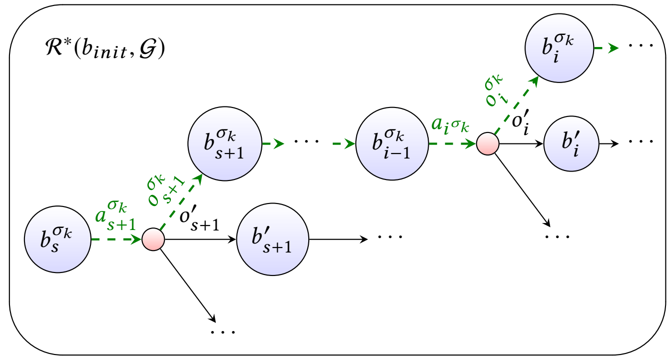


**Figure 5: An example run of BPS. The black box represents the goal-constrained belief space $\mathcal{R}^*(b_{init}, \mathcal{G})$ over the bounded horizon $k$, blue nodes represent beliefs, red nodes represent observations, and the dashed green path represents one candidate plan $\sigma_k$ found by the incremental SMT solver. BPS constructs a policy tree from this candidate plan by considering other branches following the red observation node for each step.**

keeping the probability of visiting unsafe state $s_{\text{unsafe}}$ below the threshold 0.2 (safety). The probability transition and observation functions are shown in Figure 2. Based on Formula 1, we can get the transition in the corresponding belief space (see Figure 3).

If we model this problem as an unconstrained POMDP by assigning a negative penalty $-P$ ($P > 0$) for unsafe state $s_{\text{unsafe}}$ and a positive reward $R$ ($R > 0$) for goal state $s_{\text{goal}}$, the optimal action for $s_{\text{ready}}$ that achieves the maximum reward is always $a_L$, no matter what values of $P$ and $R$ are. This is because the expected reward of action $a_L$ ($0.9R - 0.1P$) is greater than the expected reward of $a_R$ ($0.85R - 0.1P$). However, action $a_L$ does not satisfy the original safe-reachability objective in the worst case where the robot observing $o_{\text{neg}}$ after executing action $a_L$ and the resulting belief state $(0, 0.28, 0.72)$ violates the original safety-reachability objective.

If we model this problem as a C/RS/CC-POMDP by assigning a positive reward $R$ for goal state $s_{\text{goal}}$ and a cost 1 for visiting unsafe state $s_{\text{unsafe}}$, the best action for $s_{\text{ready}}$ will be $a_L$ since both $a_L$ and $a_R$ satisfies the cost/risk constraint (expected cost/risk $0.1 < 0.2$) and the expected reward of $a_L$ ($0.9R$) is greater than the expected reward of $a_R$ ($0.85R$). However, action $a_L$ violates the original safe-reachability objective for the same reason explained above.

On the other hand, using our formulation of POMDPs with safe-reachability objectives, the best action for $s_{\text{ready}}$ will be $a_R$. This is because, as shown in Definition 3.6, a valid policy in our formulation should satisfy the safe-reachability objective in all possible executions and only $a_R$ satisfies the safe-reachability objective in every possible execution.

The intent of this simple example is to illustrate that in some domains where we want the robot to safely accomplish the task, our formulation of POMDPs with safe-reachability objectives can provide a better guarantee of both safety and reachability than the existing POMDP models. While the formulations of cost/risk as negative penalties in unconstrained POMDPs and expected cost/risk threshold constraints in C/RS/CC-POMDPs are suitable for many applications, there are domains such as autonomous driving and disaster rescue that demand synthesis of policies that can provide such strong guarantee of reaching goal states safely as in our formulation, especially when violating safety requirements results in irreversible damage to robots.

## 5 BOUNDED POLICY SYNTHESIS

The core steps of BPS (Algorithm 1) are shown in Figure 4. BPS computes a valid policy by iteratively searching for a candidate plan in the goal-constrained belief space $\mathcal{R}^*(b_{init}, \mathcal{G})$ and constructing a valid policy from this candidate plan. Figure 5 graphically depicts one example run of BPS.

First BPS compactly encodes the goal-constrained belief space $\mathcal{R}^*(b_{\text{init}}, \mathcal{G})$ (the black box in Figure 5) w.r.t. the given POMDP $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$, the initial belief $b_{\text{init}}$ and the safe-reachability objective $\mathcal{G}$ over a bounded *horizon* $k$ as a logical formula $\Phi_k$ (Algorithm 1, lines 2, 6, 8). We describe the details of the constraints that encode the goal-constrained belief space in Section 5.1.

Then BPS computes a candidate plan by checking the satisfiability of the constraint $\Phi_k$ (line 10) through a modern, incremental SMT solver [9]. Note that the horizon $k$ restricts the length of the plan and thus the robot can only execute $k$ actions.

If $\Phi_k$ is satisfiable, the SMT solver returns a candidate plan (the dashed green path in Figure 5) and BPS tries to generate a valid policy from the candidate plan by considering all possible observations, i.e., other branches following the red observation node at each step (line 14). If this policy generation succeeds, we find a valid policy. Otherwise, BPS adds additional constraints that block this invalid plan (line 16) and forces the SMT solver to generate another better candidate.

If $\Phi_k$ is unsatisfiable and thus there is no new plan for the current horizon, BPS increases the horizon by one (line 21) and repeats the above steps until a *valid* policy is found (line 18) or a given horizon bound $h$ is reached (line 3).

This incremental SMT solver [9] can efficiently generate alternate candidate plans by maintaining a *stack* of *scopes*, where each scope is a container for a set of constraints and the corresponding "knowledge" learned from this set of constraints. For fast repeated satisfiability checks, when we update constraints (lines 2, 6, 8, 16), rather than rebuilding the "knowledge" from scratch, the incremental SMT solver only changes the "knowledge" related to the updates by pushing (line 7) and popping (line 20) scopes. Thus the "knowledge" learned from previous satisfiability checks can be reused.

**Algorithm 1:** BPS

**Input:**
POMDP $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$
Initial Belief $b_{init}$
Safe-Reachability Objective $\mathcal{G}$
Start Step $s$
Horizon Bound $h$
**Output:** A *Valid* Policy $\pi$

1 $k \leftarrow s$; /* Initial horizon */
2 $\Phi_k \leftarrow (b_s = b_{init})$ ; /* Initial belief */
3 **while** $k \leq h$ **do**
4 $\quad \sigma_k \leftarrow \emptyset$; /* $\sigma_k$: Candidate plan */
$\quad$ /* Add transition at step $k$ if $k > s$ */
5 $\quad$ **if** $k > s$ **then**
6 $\quad\quad \Phi_k \leftarrow \Phi_k \wedge (b_k = \mathcal{T}_B(b_{k-1}, a_k, o_k))$;
7 $\quad$ push($\Phi_k$); /* Push scope */
$\quad$ /* Add goal constraints at step $k$ (Formula 4) */
8 $\quad \Phi_k \leftarrow \Phi_k \wedge G(\sigma_k, \mathcal{G}, k)$;
9 $\quad$ **while** $\emptyset = \sigma_k$ **do**
$\quad\quad$ /* Candidate generation */
10 $\quad\quad \sigma_k \leftarrow$ IncrementalSMT($\Phi_k$);
11 $\quad\quad$ **if** $\emptyset = \sigma_k$ **then** /* No new plan */
12 $\quad\quad\quad$ **break;**
13 $\quad\quad$ **else**
$\quad\quad\quad$ /* $\phi$: constraints for blocking invalid plans */
14 $\quad\quad\quad \pi, \phi =$ PolicyGeneration($P, \mathcal{G}, \sigma_k, s+1, k$);
15 $\quad\quad\quad$ **if** $\emptyset \neq \phi$ **then** /* Generation failed */
16 $\quad\quad\quad\quad \Phi_k \leftarrow \Phi_k \wedge \phi$;
17 $\quad\quad\quad$ **else**
18 $\quad\quad\quad\quad$ **return** $\pi$;
19 $\quad\quad\quad \sigma_k \leftarrow \emptyset$;
$\quad$ /* Pop scope: pop goal and $\phi$ at step $k$ */
20 $\quad$ pop($\Phi_k$);
21 $\quad k \leftarrow k + 1$ ; /* Increase horizon */
22 **return** $\emptyset$;

**Algorithm 2:** PolicyGeneration

**Input:**
POMDP $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$
Safe-Reachability Objective $\mathcal{G}$
Candidate Plan $\sigma_k = (b_s^{\sigma_k}, a_{s+1}^{\sigma_k}, o_{s+1}^{\sigma_k}, b_{s+1}^{\sigma_k} \dots)$
Start Step $s$
Horizon Bound $h$
**Output:** A *Valid* Policy $\pi$ and Constraints $\phi$ for blocking invalid plans if the input candidate plan is invalid

23 $\pi \leftarrow \emptyset$;
24 **for** $i = h$ **downto** $s$ **do**
25 $\quad$ **foreach** *observation* $o \in \mathcal{O} - \{o_i^{\sigma_k}\}$ **do**
$\quad\quad$ /* Try observation $o$ */
26 $\quad\quad b_i' \leftarrow \mathcal{T}_B(b_{i-1}^{\sigma_k}, a_i^{\sigma_k}, o)$;
$\quad\quad$ /* Call BPS to construct the branch */
27 $\quad\quad \pi' \leftarrow$ BPS($P, b_i', \mathcal{G}, i, h$);
28 $\quad\quad$ **if** $\emptyset = \pi'$ **then** /* Construction failed */
29 $\quad\quad\quad$ Construct $\phi$ using Formula 5
30 $\quad\quad\quad$ **return** $\emptyset$, $\phi$;
31 $\quad\quad \pi \leftarrow \pi \cup \pi'$; /* Combine policy */
$\quad$ /* Record action choice for belief $b_{i-1}^{\sigma_k}$ */
32 $\quad \pi(b_{i-1}^{\sigma_k}) \leftarrow a_i^{\sigma_k}$;
33 **return** $\pi$, $\emptyset$;

## 5.1 Constraint Generation

In the first step, we use an encoding from Bounded Model Checking (BMC) [3] to construct the constraint $\Phi_k$ representing the goal-constrained belief space $\mathcal{R}^*(b_{\text{init}}, \mathcal{G})$ w.r.t. the POMDP $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z})$, the initial belief $b_{\text{init}}$ and the safe-reachability objective $\mathcal{G}$ over the bounded horizon $k$. The idea behind BMC is to find a finite plan with increasing horizon that satisfies the given safe-reachability objective.

The constraint $\Phi_k$ contains three parts:

(1) Starting from the initial belief (line 2): $b_s = b_{\text{init}}$.
(2) Unfolding of the transition up to the horizon $k$ (line 6): $\bigwedge_{i=s+1}^{k} (b_i = \mathcal{T}_B(b_{i-1}, a_i, o_i))$.
(3) Satisfying the safe-reachability objective $\mathcal{G}$ (line 8).

We can translate a safe-reachability objective to the constraint $G(\sigma_k, \mathcal{G}, k)$ on bounded plans $\sigma_k = (b_s, a_{s+1}, o_{s+1}, \dots, a_k, o_k, b_k)$ using the rules provided by BMC [3] as follows:

$$G(\sigma_k, \mathcal{G}, k) = \bigvee_{i=s}^{k} (b_i \in \mathit{Dest} \wedge (\bigwedge_{j=s}^{i-1} (b_j \in \mathit{Safe}))) \quad (4)$$

For a safe-reachability objective $\mathcal{G}$ with a set *Dest* of goal beliefs and a set *Safe* of safe beliefs, a finite plan that visits a goal belief while staying in the safe region is sufficient to satisfy $\mathcal{G}$. Therefore, we only need to specify that a bounded plan with length $k$ eventually visits a belief $b_i \in \mathit{Dest}$ while staying in the safe region $(\bigwedge_{j=s}^{i-1} (b_j \in \mathit{Safe}))$, as shown in Formula 4.

## 5.2 Plan Generation

The next step is to generate a candidate plan $\sigma_k$ of length $k$ that satisfies the constraint $\Phi_k$. We apply an incremental SMT solver to efficiently search for such a candidate in the *goal-constrained belief space* $\mathcal{R}^*(b_{\text{init}}, \mathcal{G})$ defined by $\Phi_k$ (line 10). If $\Phi_k$ is unsatisfiable, there is no bounded plan $\sigma_k$ for the current horizon. In this case, we need to increase the horizon (line 21). If $\Phi_k$ is satisfiable, the SMT solver will return a satisfying model that assigns concrete values $b_i^{\sigma_k}$, $a_{i+1}^{\sigma_k}$ and $o_{i+1}^{\sigma_k}$ for the belief $b_i$, action $a_{i+1}$ and observation $o_{i+1}$ at each step $i$ respectively, which can be used to construct the candidate plan $\sigma_k = (b_s^{\sigma_k}, a_{s+1}^{\sigma_k}, o_{s+1}^{\sigma_k}, b_{s+1}^{\sigma_k} \dots, a_k^{\sigma_k}, o_k^{\sigma_k}, b_k^{\sigma_k})$.

## 5.3 Policy Generation

After *plan generation*, we get a candidate plan $\sigma_k$ (the dashed green path in Figure 5) that satisfies the safe-reachability objective $\mathcal{G}$. This candidate plan is a single path that only covers a particular observation $o_i^{\sigma_k}$ at each step $i$. To construct a *valid* policy, we should also consider other possible observations $o_i' \neq o_i^{\sigma_k}$, i.e., other branches following the red observation node for each step $i$. *Policy generation* (Algorithm 2) tries to construct a valid policy from a candidate plan by considering all possible observations at each step.

For a candidate plan $\sigma_k$, we process each step of $\sigma_k$, starting from the last step (Algorithm 2, line 24). For each step $i$, since the set of observations $O$ is finite, we can enumerate every possible observation $o'_i \neq o_i^{\sigma_k}$ (line 25) and compute the next belief $b'_i$ using the transition function (line 26). To ensure the action $a_i^{\sigma_k}$ also works for this different observation $o'_i$, we need to compute a *valid* policy for the branch starting from $b'_i$, which is another BPS problem and can be solved using Algorithm 1 (line 27).

If we successfully construct the valid policy $\pi'$ for this branch, we can add $\pi'$ to the policy $\pi$ for the original synthesis problem (line 31). Otherwise, this candidate plan $\sigma_k$ can not be an element of a *valid* policy $\sigma_k \notin \Omega_\pi$. In this case, we know that the prefix of the candidate plan $(b_s^{\sigma_k}, a_{s+1}^{\sigma_k}, o_{s+1}^{\sigma_k}, \ldots, b_{i-1}^{\sigma_k}, a_i^{\sigma_k})$ is invalid for current horizon $k$ and we can add additional constraints $\phi$ to block all invalid plans that have this prefix (line 29):

$$\phi = \neg \left((b_s = b_s^{\sigma_k}) \wedge (a_i = a_i^{\sigma_k}) \wedge \left(\bigwedge_{m=s+1}^{i-1} (a_m = a_m^{\sigma_k}) \wedge (o_m = o_m^{\sigma_k}) \wedge (b_m = b_m^{\sigma_k})\right)\right) \quad (5)$$

Note that $\phi$ is only valid for current horizon $k$ and when we increase the horizon, we should *pop* the scope related to the additional constraints $\phi$ from the stack of the SMT solver (line 20) so that we can *revisit* this prefix with the increased horizon. If we successfully construct policies for all other branches at step $i$, we know that the choice of action $a_i^{\sigma_k}$ for belief $b_{i-1}^{\sigma_k}$ is valid for all possible observations. Then we record this choice for belief $b_{i-1}^{\sigma_k}$ in the policy (line 32). This policy generation terminates when it reaches the start step $s$ as stated in the for-loop (line 24) or it fails to construct the valid policy $\pi'$ for a branch (line 28).

## 5.4 Algorithm Complexity

The *reachable belief space* $\mathcal{R}(b_{\text{init}})$ can be seen as a tree where the root node is the initial belief $b_{\text{init}}$ and at each node, the tree branches on every action and observation. The given horizon bound $h$ limits the height of the tree. Therefore, the reachable belief space $\mathcal{R}_h(b_{\text{init}})$ of height $h$ contains $O(|\mathcal{A}|^h|O|^h)$ plans, where $|\mathcal{A}|$ and $|O|$ are the size of action set $\mathcal{A}$ and the size of observation set $O$ respectively. To synthesize a *valid* policy, a naive approach that checks every plan in the reachable belief space $\mathcal{R}_h(b_{\text{init}})$ requires $O(|\mathcal{A}|^h|O|^h)$ calls to the SMT solver. This exponential growth of the reachable belief space $\mathcal{R}_h(b_{\text{init}})$ due to branches on both action and observations is a major challenge for synthesizing a *valid* policy.

In our case, BPS exploits the notion of *goal-constrained belief space* $\mathcal{R}^*(b_{\text{init}}, \mathcal{G})$ and efficiently explores the *goal-constrained belief space* $\mathcal{R}^*(b_{\text{init}}, \mathcal{G})$ by leveraging an incremental SMT solver to generate a candidate plan $\sigma$ of length at most $h$. This candidate plan fixes the choice of actions at each step and thus the policy generation process only needs to consider the branches on observations for each step, as shown in Figure 5. Therefore, BPS requires $O(I|O|^h)$ calls to the SMT solver, where $I$ is the number of interactions between plan generation and policy generation, while the naive approach described above requires $O(|\mathcal{A}|^h|O|^h)$ SMT solver calls. In general, $I$ is often much smaller than $|\mathcal{A}|^h$, which leads to much faster policy synthesis. Therefore, we expect our method to be effective for POMDPs with a high-dimensional action space and a restricted partially observable component, but would not scale well for POMDPs with high-dimensional/continuous observation space.

# 6 EXPERIMENTS

We evaluate BPS in a partially observable kitchen domain (Figure 1) with a PR2 robot and $M$ uncertain obstacles placed in the yellow "shadow" region. The task for the robot is to safely pass the yellow "shadow" region avoiding collisions with uncertain obstacles and eventually pick up a green cup from the black storage area.

We first discretize the kitchen environment into $N$ *cells*. We assume that the locations of the obstacles are uniformly distributed among the cells in the yellow "shadow" region and there is at most one obstacle in each cell. We also assume the robot starts at a known initial location. However, due to the robot's imperfect perception, the locations of the robot, the locations of uncertain obstacles, and the location of the target cups are all partially observable during execution.

In this domain, the actuation and perception of the robot are imperfect. There are ten uncertain robot actions ($|\mathcal{A}| = 10$):

(1) Four *move* actions that move the robot to an adjacent cell in four directions: including *move-north*, *move-south*, *move-west* and *move-east*. *Move* actions could fail with a probability $p_{\text{fail}}$, resulting in no change in the state.
(2) Four *look* actions that observe a cell to see whether there is an obstacle in that cell, including *look-north*, *look-south*, *look-west*, *look-east* (look at the adjacent cell in the corresponding direction). When the robot calls *look* to observe a particular cell$_i$, it may either make an observation $o = o_{\text{pos}}$ representing the robot observes an obstacle in cell$_i$ or $o = o_{\text{neg}}$ representing the robot observers no obstacle in cell$_i$. The probabilistic observation function $\mathcal{Z}(s', a, o)$ for *look* actions is defined based on the false positive probability $p_{\text{fp}}$ and the false negative probability $p_{\text{fn}}$.
(3) Two *pick-up* actions that pick up an object from the black storage area: pick-up using the left hand $a_L$ and pick-up using the right hand $a_R$. The model of *pick-up* actions is the same as what we discussed in Section 4 (see Figure 2).

The task shown in Figure 1 can be specified as a safe-reachability objective with a set *Dest* of goal beliefs and a set *Safe* of safe beliefs, defined as follows:

$$\begin{aligned} Dest &= \{b \in \mathcal{B} \mid \left(\sum b(\text{target cup in robot's hand})\right) > 1 - \delta_1\} \\ Safe &= \{b \in \mathcal{B} \mid \left(\sum b(\text{robot in collision})\right) < \delta_2\} \end{aligned} \quad (6)$$

where $\delta_1$ and $\delta_2$ are small values that represent tolerance. The reachability objective specifies that in a goal belief, the probability of having the target cup in the robot's hand should be greater than the threshold $1 - \delta_1$. The safety objective specifies that in a safe belief, the probability of the robot in collision (the robot and one obstacle in the same cell) should be less than the tolerance $\delta_2$.

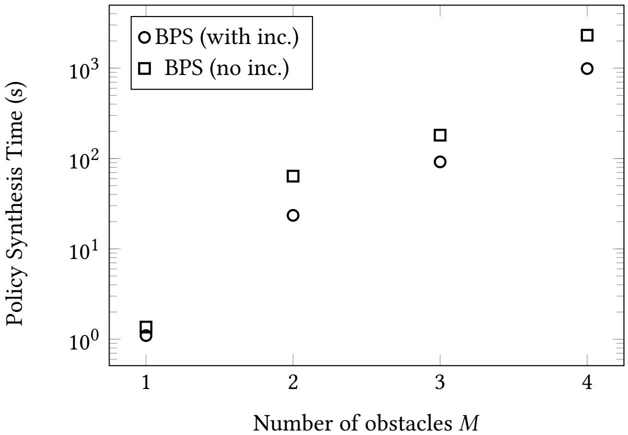


**Figure 6: Performance of BPS as the number of obstacles $M$ varies. The plot of circles shows the performance of BPS with incremental solving and the plot of squares shows the performance of BPS without incremental solving.**

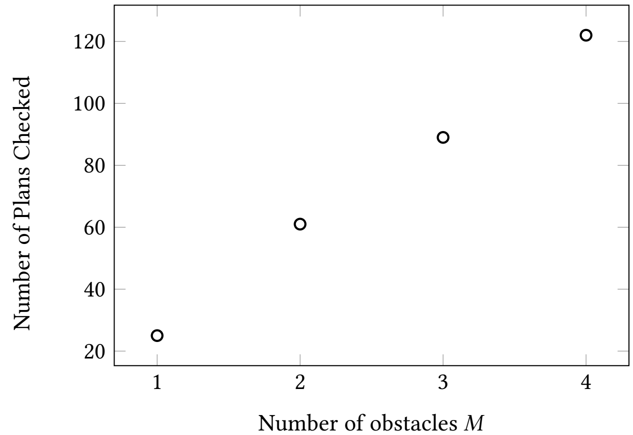


**Figure 7: The number of plans checked (i.e, the number of SMT calls) by BPS during policy synthesis as the number of obstacles $M$ varies.**

We evaluate the performance of BPS using test cases of the kitchen domain with various numbers of obstacles. We use Z3 [9] as our backend incremental SMT solver. All experiments were conducted on a 3.0GHz Intel® processor with 32GB memory. For all the tests, the horizon bound is $h = 20$ and the number of cells in the kitchen environment is $N = 24$.

To evaluate the gains from incremental solving, we test BPS in two settings: with and without incremental solving. Note that when incremental solving is disabled, each call to the SMT solver requires solving the SMT constraints from scratch, rather than reusing the results from the previous SMT solver calls. Figure 6 shows the performance results of BPS with and without incremental solving. As we can see from Figure 6, enabling incremental solving in BPS leads to a performance improvement in policy synthesis. This is because the BMC encoding [3] used in BPS is particularly suitable for incremental solving since increasing horizon and blocking invalid plans correspond to pushing/popping constraints.

To demonstrate the gains from utilizing the goal-constrained belief space compared to a naive exhaustive search in the reachable belief space, we first estimate the number of plans in the reachable belief space. There is no observation branching for the four *move* actions and there are two observation branches for the four *look* actions. We ignore the two *pick-up* actions since these two actions are not available in every step and can only be performed when the robot is fairly confident that it is in the position where it is ready to pick up a cup from the black storage area. Therefore, the approximate lower bound of the number of plans in the reachable belief space with at most $h = 20$ steps is $(4 + 4 \times 2)^{20} \approx 10^{21}$. However, as we can see from Figure 7 where we show the number of plans checked by BPS during policy synthesis, for the largest test, the number of plans checked (around 120) in BPS is very small compared to the number of plans in the reachable belief space. These results show that BPS can solve problems in huge reachable belief spaces with a small number of SMT solver calls by focusing on the goal-constrained belief space.

However, Figure 6 also shows that the synthesis time grows exponentially as the number of obstacles increases, which matches our complexity analysis in Section 5.4. This is because the current implementation of BPS operates on an *exact* tree representation of policies with all the observation branches. As the number of obstacles increases, both the horizon bound (the height of the policy tree) and the size of the state space (the belief space dimension) increase, which leads to an exponential growth of plans in the policy tree and makes the policy synthesis problem much harder.

## 7 CONCLUSION AND DISCUSSION

We present a novel policy synthesis method called BPS for POMDPs with safe-reachability objectives. We exploit the notion of a goal-constrained belief space to improve computational efficiency. We construct constraints in a way similar to Bounded Model Checking [3] to compactly represent the goal-constrained belief space, which we efficiently explore through an incremental Satisfiability Modulo Theories solver [9]. We evaluate BPS in an uncertain robotic domain and the results show that our method can synthesize policies for large problems by focusing on the goal-constrained belief space.

The current implementation of BPS operates on an *exact* representation of the policy (the tree structure shown in Figure 5). As a result, BPS suffers from the exponential growth as the horizon increases. An important ongoing question is how to approximately represent the policy while preserving correctness. Another issue arises from the discrete representations (discrete POMDPs) used in our approach. While many robot tasks can be modeled using these representations, discretization often suffers from the "curse of dimensionality". Investigating how to deal with continuous state spaces and continuous observations directly without discretization is another promising future direction for this work and its application in robotics.

## 8 ACKNOWLEDGMENTS

This work was supported in part by NSF CCF 1139011, NSF CCF 1514372, NSF CCF 1162076 and NSF IIS 1317849.

## REFERENCES


[1] Haoyu Bai, David Hsu, Wee Sun Lee, and Vien A. Ngo. 2011. Monte Carlo Value Iteration for Continuous-State POMDPs. In *Algorithmic Foundations of Robotics IX: Selected Contributions of the Ninth International Workshop on the Algorithmic Foundations of Robotics*, David Hsu, Volkan Isler, Jean-Claude Latombe, and Ming C. Lin (Eds.). Springer, Berlin, Heidelberg, 175–191.

[2] Julien Bidot, Lars Karlsson, Fabien Lagriffoul, and Alessandro Saffiotti. 2017. Geometric backtracking for combined task and motion planning in robotic systems. *Artificial Intelligence* 247 (2017), 229–265.

[3] Armin Biere, Alessandro Cimatti, Edmund M Clarke, Ofer Strichman, and Yunshan Zhu. 2003. Bounded model checking. *Advances in Computers* 58 (2003), 117–148.

[4] Alberto Camacho, Christian Muise, and Sheila A. McIlraith. 2016. From FOND to Robust Probabilistic Planning: Computing Compact Policies That Bypass Avoidable Deadends. In *Proceedings of the Twenty-Sixth International Conference on International Conference on Automated Planning and Scheduling (ICAPS'16)*. AAAI Press, 65–69.

[5] Krishnendu Chatterjee, Martin Chmelík, and Jessica Davies. 2016. A Symbolic SAT-Based Algorithm for Almost-Sure Reachability with Small Strategies in POMDPs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 3225–3232.

[6] K. Chatterjee, M. Chmelík, R. Gupta, and A. Kanodia. 2015. Qualitative analysis of POMDPs with temporal logic specifications for robotics applications. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 325–330.

[7] Krishnendu Chatterjee, Martin Chmelík, and Mathieu Tracol. 2016. What is Decidable about Partially Observable Markov Decision Processes with $\omega$-regular Objectives. *J. Comput. System Sci.* 82, 5 (Aug. 2016), 878–911.

[8] Neil T. Dantam, Zachary K. Kingston, Swarat Chaudhuri, and Lydia E. Kavraki. 2016. Incremental Task and Motion Planning: A Constraint-Based Approach. In *Robotics: Science and Systems*.

[9] Leonardo De Moura and Nikolaj Bjørner. 2008. Z3: An Efficient SMT Solver. In *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'08/ETAPS'08)*. Springer-Verlag, Berlin, Heidelberg, 337–340.

[10] Thomas Dean, Leslie Pack Kaelbling, Jak Kirman, and Ann Nicholson. 1995. Planning under time constraints in stochastic domains. *Artificial Intelligence* 76, 1-2 (1995), 35–74.

[11] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press.

[12] Adam Eck and Leen-Kiat Soh. 2014. Online Heuristic Planning for Highly Uncertain Domains. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems (AAMAS '14)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 741–748.

[13] E. Erdem, K. Haspalamutgil, C. Palaz, V. Patoglu, and T. Uras. 2011. Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*. 4575–4581.

[14] M. Gharbi, R. Lallement, and R. Alami. 2015. Combining symbolic and geometric planning to synthesize human-aware plans: toward more efficient combined search.. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 6360–6365.

[15] D. K. Grady, M. Moll, and L. E. Kavraki. 2015. Extending the Applicability of POMDP Solutions to Robotic Tasks. *IEEE Transactions on Robotics* 31, 4 (Aug 2015), 948–961.

[16] Marek Grzes and Pascal Poupart. 2014. POMDP Planning and Execution in an Augmented Space. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems (AAMAS '14)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 757–764.

[17] Marek Grzes and Pascal Poupart. 2015. Incremental Policy Iteration with Guaranteed Escape from Local Optima in POMDP Planning. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1249–1257.

[18] D. Hadfield-Menell, E. Groshev, R. Chitnis, and P. Abbeel. 2015. Modular task and motion planning in belief space. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 4991–4998.

[19] K. He, M. Lahijanian, L. E. Kavraki, and M. Y. Vardi. 2015. Towards manipulation planning with temporal logic specifications. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 346–352.

[20] Ping Hou, William Yeoh, and Pradeep Varakantham. 2016. Solving Risk-Sensitive POMDPs with and without Cost Observations. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 3138–3144.

[21] Joshua D. Isom, Sean P. Meyn, and Richard D. Braatz. 2008. Piecewise Linear Dynamic Programming for Constrained POMDPs. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI'08)*. AAAI Press, 291–296.

[22] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence* 101, 1-2 (May 1998), 99–134.

[23] Leslie Pack Kaelbling and Tomás Lozano-Pérez. 2016. Implicit belief-space pre-images for hierarchical planning and execution. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 5455–5462.

[24] Dongho Kim, Jaesong Lee, Kee-Eung Kim, and Pascal Poupart. 2011. Point-based Value Iteration for Constrained POMDPs. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI'11)*. AAAI Press, 1968–1974.

[25] Hanna Kurniawati, David Hsu, and Wee Sun Lee. 2008. SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. In *Robotics: Science and Systems*.

[26] Yuanfu Luo, Haoyu Bai, David Hsu, and Wee Sun Lee. 2016. Importance Sampling for Online Planning under Uncertainty. In *Workshop on Algorithmic Foundations of Robotics*.

[27] Omid Madani, Steve Hanks, and Anne Condon. 2003. On the Undecidability of Probabilistic Planning and Related Stochastic Optimization Problems. *Artificial Intelligence* 147, 1-2 (July 2003), 5–34.

[28] Janusz Marecki and Pradeep Varakantham. 2010. Risk-sensitive Planning in Partially Observable Environments. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '10)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1357–1368.

[29] Mehryar Mohri. 1997. Finite-state Transducers in Language and Speech Processing. *Computational Linguistics* 23, 2 (June 1997), 269–311.

[30] Christian Muise, Vaishak Belle, and Sheila A. McIlraith. 2014. Computing Contingent Plans via Fully Observable Non-deterministic Planning. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI'14)*. AAAI Press, 2322–2329.

[31] S. Nedunuri, S. Prabhu, M. Moll, S. Chaudhuri, and L. E. Kavraki. 2014. SMT-based synthesis of integrated task and motion plans from plan outlines. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 655–662.

[32] Christos Papadimitriou and John N. Tsitsiklis. 1987. The Complexity of Markov Decision Processes. *Mathematics of Operations Research* 12, 3 (1987), 441–450.

[33] Azaria Paz. 1971. *Introduction to Probabilistic Automata (Computer Science and Applied Mathematics)*. Academic Press, Inc., Orlando, FL, USA.

[34] Joelle Pineau, Geoff Gordon, and Sebastian Thrun. 2003. Point-based Value Iteration: An Anytime Algorithm for POMDPs. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1025–1030.

[35] Robert Platt Jr, Russ Tedrake, Leslie Kaelbling, and Tomás Lozano-Pérez. 2010. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems*.

[36] Pascal Poupart, Aarti Malhotra, Pei Pei, Kee-Eung Kim, Bongseok Goh, and Michael Bowling. 2015. Approximate Linear Programming for Constrained Partially Observable Markov Decision Processes. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 3342–3348.

[37] Pedro Santana, Sylvie Thiébaux, and Brian Williams. 2016. RAO*: An Algorithm for Chance-constrained POMDP's. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 3308–3314.

[38] Richard D. Smallwood and Edward J. Sondik. 1973. The Optimal Control of Partially Observable Markov Processes Over a Finite Horizon. *Operations Research* 21, 5 (1973), 1071–1088.

[39] Adhiraj Somani, Nan Ye, David Hsu, and Wee Sun Lee. 2013. DESPOT: Online POMDP planning with regularization. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*. Curran Associates Inc., USA, 1772–1780.

[40] Matthijs T. J. Spaan and Nikos Vlassis. 2004. A point-based POMDP algorithm for robot planning. In *2004 IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 3. 2399–2404 Vol.3.

[41] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel. 2014. Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 639–646.

[42] Mária Svoreňová, Martin Chmelík, Kevin Leahy, Hasan Ferit Eniser, Krishnendu Chatterjee, Ivana Černá, and Calin Belta. 2015. 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)*. ACM, New York, NY, USA, 233–238.

[43] A. Undurti and J. P. How. 2010. An online algorithm for constrained POMDPs. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*. 3966–3973.

[44] Yue Wang, Neil T. Dantam, Swarat Chaudhuri, and Lydia E. Kavraki. 2016. Task and Motion Policy Synthesis As Liveness Games. In *Proceedings of the Twenty-Sixth International Conference on International Conference on Automated Planning and Scheduling (ICAPS'16)*. AAAI Press, 536–540.