Bounded Policy Synthesis for POMDPs with Safe-Reachability Objectives

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 robot applications. A key algorithmic problem for POMDPs is policy synthesis. While this problem has traditionally been posed w.r.t. optimality objectives, many robot applications are better modeled by POMDPs where the objective is a boolean requirement. In this paper, we study the latter problem in a setting where the requirement is a safe-reachability property, which states that with a probability above a certain threshold, it is possible to eventually reach a goal state while satisfying a safety requirement.

The central challenge in our problem is that it requires reasoning over a vast space of probability distributions. What’s more, it has been shown that policy synthesis of POMDPs with reachability objectives is undecidable in general. To address these challenges, we introduce the notion of a goal-constrained belief space, which only contains beliefs (probability distributions over states) reachable from the initial belief under desired executions. This constrained space is generally much smaller than the original belief space. Our approach 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 robotics domain with uncertain obstacles. Our results suggest that it is possible to synthesize policies over large belief spaces with a small number of SMT solver calls by focusing on goal-constrained belief space, and our method offers a stronger guarantee of both safety and reachability than alternative unconstrained/constrained POMDP formulations.

1 INTRODUCTION
Partially Observable Markov Decision Processes (POMDPs) [37] 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 robotics domain can be modeled as POMDP problems [5, 7, 15, 22]. Perhaps the central algorithmic problem for POMDPs is the synthesis of policies [37]: 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 w.r.t. a quantitative objective such as discounted reward [1, 12, 15–17, 25, 26, 38, 39]. While this purely quantitative formulation of the problem is suitable for many applications, there are, however, 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, formulating boolean requirements as quantitative objectives leads to policies that are overly conservative or overly risky [42], depending on the particular reward function chosen. Therefore, new models and algorithms are required for handling POMDPs with boolean requirements explicitly.

Recently, there has been a large body of work that extends the traditional POMDP model with notions of risk and cost that handles constraints explicitly, including constrained POMDPs (C-POMDPs) [21, 24, 35, 42], chance-constrained POMDPs (CC-POMDPs) [36] and risk-sensitive POMDPs (RS-POMDPs) [20, 28]. There are two major differences between their models and our formulation of POMDPs with boolean requirements. 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 boolean requirement in all possible executions including the worst case (hard constraints), providing a stronger safety guarantee than the formulation of expected cost/risk threshold constraints (soft 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 probability threshold (reach a goal state with a probability greater than some probability threshold $\lambda$), while our formulation can directly encode this probability threshold constraint as a boolean requirement, providing stronger reachability guarantee than the quantitative formulation of C/RS/CC-POMDPs. In Section 4, we will discuss an example to show our formulation can provide stronger guarantee of both safety and reachability than C/RS/CC-POMDPs.

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 [7]. Here, 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 a threshold [7]. In this work, we study this problem for the special case of safe-reachability properties, which require that with a probability at least $\lambda \in (0, 1]$, a goal state is eventually visited while satisfying a safety requirement. Many robot tasks such as the one in Figure 1 can be formulated using a safe-reachability objective.

Previous results [6, 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 horizon bound $h$ such that $h$ is sufficiently large to prove the existence of a valid policy, or the user is not interested in plans beyond the horizon bound $h$. Such assumption is particularly reasonable for robotics domains, where it is often the case that a task is required to be accomplished in finite time. 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 finite time 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 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, 38] for POMDPs with discounted reward objectives. These POMDP algorithms exploit the notion of the reachable belief space $R(b_{\text{init}})$ from an initial belief $b_{\text{init}}$ and compute an approximately optimal policy over $R(b_{\text{init}})$ 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, 43] and many of them 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.

We show experimentally that BPS can synthesize policies for robots in an uncertain mobile manipulation domain (see Figure 1). The results demonstrate that BPS can scale up to huge belief spaces by focusing on the goal-constrained belief space.

In summary, the contributions of the paper are:

- We present a novel approach called BPS for policy synthesis of POMDPs with safe-reachability objectives.
- We introduce the notion of a goal-constrained belief space to address the scalability challenge of solving POMDPs with safe-reachability objectives.
- We demonstrate the scalability of BPS using a case study involving a partially observable robotics domain with uncertain obstacles (see Figure 1). Compared to unconstrained POMDPs and C/RS/CC-POMDPs, our method offers stronger safe-reachability guarantee for the tested benchmark.

2 RELATED WORK

POMDPs [37] provide a principle mathematical framework for modeling a variety of robotics problems in the face of uncertainty. Many POMDP algorithms [1, 15, 25, 26, 38, 39] for robot applications focus on discounted reward objectives. Recent work [5, 7, 41] 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 be able to formulate a safe-reachability objective as an optimization problem, this formulation does not always yield good policies [42].

Recently, there has been a growing interest in C-POMDPs, CC-POMDPs, and RS-POMDPs that handle cost/risk constraints explicitly [20, 21, 24, 28, 35, 36, 42]. Their objective of maximizing reward while keeping cost/risk below some threshold is different from the safe-reachability objective considered in this paper. To
ensure reachability, C/RS/CC-POMDPs typically need to assign a proper positive reward for goal states and do not have direct control over probability threshold (reach a goal state with a probability greater than some threshold $\lambda$), while our safe-reachability objective can directly encode this probability threshold constraint. While C/RS/CC-POMDPs are suitable for many applications, there are, however, settings that demand synthesis of policies that can provide such strong guarantee of reaching goal states safely.

Our method computes a valid policy by iteratively searching for a candidate plan that is likely to succeed with determined 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 temporal logic specifications. 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.

Task and Motion Planning (TMP) [2, 8, 13, 14, 18, 19, 23, 40, 43] 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 [34] assuming the most likely observation will be obtained, while our approach synthesizes a valid policy offline contingent on all possible events.

3 PROBLEM FORMULATION

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

Definition 3.1 (POMDP).
A Partially Observable Markov Decision Processes (POMDP) is a tuple $P = (S, A, T, O, Z)$:
- $S$ is a finite set of states.
- $A$ is a finite set of actions.
- $T$ is a probabilistic transition function $T(s, a, s') = p(s'|s, a)$, which defines the probability of moving to state $s' \in S$ after taking an action $a \in A$ in state $s \in S$.
- $O$ is a finite set of observations.
- $Z$ is the probabilistic observation function $Z(s', a, o) = p(o|s', a)$, which defines the probability of observing $o \in O$ after taking an action $a \in A$ and reaching state $s' \in S$.

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 $b : S \mapsto [0, 1]$ with $\sum_{s \in S} b(s) = 1$. The set of beliefs $B = \{ b : S \mapsto [0, 1] \}$ $\sum_{s \in S} b(s) = 1$ is known as belief space. Note that a transition $T_B$ in belief space is deterministic, i.e., given an action $a \in A$ and an observation $o \in O$, the updates to beliefs are deterministic:

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

where $\alpha$ is a normalization constant.

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

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

3.1 Safe-Reachability Objective

In this work, we consider POMDPs with safe-reachability objectives, defined as follows:

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

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

Note that safe-reachability objectives are defined using sets of beliefs (probability distributions). The quantitative analysis problem of POMDPs with safe-reachability objectives, which requires with a probability at least $\lambda \in (0, 1]$ a goal state is eventually visited while satisfying a safety requirement, can be easily formulated as a safe-reachability objective with the set Dest of goal beliefs and the set Safe of safe beliefs defined as follows:

$$\text{Dest} = \{ b \in B \mid \exists s, s \text{ is a goal state and } b(s) \geq \lambda \}$$  \hspace{1cm} (2)

$$\text{Safe} = \{ b \in B \mid \forall s, s \text{ violates safety and } b(s) \leq \lambda \}$$  \hspace{1cm} (3)

Moreover, a safe-reachability objective $G$ compactly represents the desired set $\Omega_G$ of plans in belief space, i.e., for every plan $\sigma \in \Omega_G$, $\sigma$ satisfies the safe-reachability objective $G$. 
Thus, computing policies over the goal-constrained belief space is usually much smaller than the reachable belief space. Given a POMDP \( \pi \) with safe-reachability objective \( G \) over the corresponding goal-constrained belief space \( \Omega \), it is intractable to compute a full policy that satisfies a given safe-reachability objective \( G \). Policy \( \pi_{\Omega} \) is a subset of the set \( \Omega \) defined by the safe-reachability objective \( G \).

### 4 Relation to Unconstrained POMDPs and C/RS/CC-POMDPs

Two distinct approaches to solving the above policy synthesis problem of POMDPs with safe-reachability objectives exist 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 [42] have shown a counter example that demonstrates formulating constraints as unconstrained POMDPs with quantitative objectives does not always yield good policies.

Both approaches that handle constraints explicitly have been presented in [20, 21, 24, 28, 35, 36, 42], where the authors investigate C/RS/CC-POMDPs. In this section, we show the differences between our formulation of POMDPs with safe-reachability objectives and unconstrained C/RS/CC-POMDPs through an example (see Figure 1).

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 observation 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 at least 0.8 (reachability) while keeping the

### 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 finite horizon bound, due to the curse of dimensionality [32]; the belief space \( B \) is a high-dimensional, continuous space that contains infinite beliefs.

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

### 3.3 Problem Statement

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

**Definition 3.6 (Valid Policy).**

A valid policy \( \pi_{\Omega} : \mathcal{R}^*(b_{\text{init}}, G) \rightarrow A \) over a goal-constrained belief space \( \mathcal{R}^*(b_{\text{init}}, G) \) is a function that maps a belief \( b \in \mathcal{R}^*(b_{\text{init}}, G) \) to an action \( a \in A \). Therefore, the set of plans \( \Omega_{\pi_{\Omega}} \) defined by the
probability of visiting unsafe state $s_{\text{unsafe}}$ below the threshold 0.1 (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 negative penalty $-P$ ($P > 0$) for unsafe state $s_{\text{unsafe}}$ and positive reward $R$ 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 $(0.28, 0.72)$ violates the original safety-reachability objective.

If we model this problem as a C/RS/CC-POMDP by assigning positive reward $R$ for goal state $s_{\text{goal}}$ and 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 $\leq 0.1$) 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 our formulation provide stronger guarantee of both safety and reachability than unconstrained POMDPs and C/RS/CC-POMDPs. 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 $R^*(b_{\text{init}}, G)$ 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 $R^*(b_{\text{init}}, G)$ (the black box in Figure 5) w.r.t. the given POMDP $P = (S, \mathcal{A}, T, O, Z)$, the initial belief $b_{\text{init}}$ and the safe-reachability objective $G$ over a bounded horizon $k$ as a logical formula $\Phi_k$ (Algorithm 1, lines 2, 6, 8). 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 can efficiently generate alternate candidate plans by maintaining a stack of scopes for the “knowledge” learned from constraints [9] to enable 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.

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 $R^*(b_{\text{init}}, G)$ w.r.t. the POMDP $P = \ldots$
(S, A, T, O, Z), the initial belief b_{init} and the safe-reachability objective G over the bounded horizon k. The idea behind BMC is to find a plan of increasing horizon that satisfies the given safe-reachability objective.

The constraint Φ_k contains three parts:

1. Starting from the initial belief (line 2): b_k = b_{init}.
2. Unfolding of the transition up to horizon k (line 6): \( \text{G}(\sigma_k, G, k) = \bigcup_{i=s}^{k} (b_i \in \text{Dest}) \). 
3. Satisfying the safe-reachability objective (line 8).

We can translate safe-reachability objectives to the constraint \( \text{G}(\sigma_k, G, k) \) on bounded plans \( \sigma_k = (b_s, a_{s+1}, b_{s+1}, \ldots, \sigma_k, b_k) \) using the rules provided by BMC [3] as follows:

\[
\text{G}(\sigma_k, G, k) = \bigcup_{i=s}^{k} (b_i \in \text{Dest} \land \bigwedge_{j=s}^{i-1} (b_j \in \text{Safe}))
\]  

**Algorithm 1: BPS**

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

**Output:** A Valid Policy \( \pi \)

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

For a safe-reachability objective \( G \) 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 \( G \). Therefore, we only need to specify that a bounded plan with length \( k \) eventually visits a belief \( b_j \in \text{Dest} \) while staying in the safe region (\( \bigwedge_{j=s}^{i-1} (b_j \in \text{Safe}) \)), as shown in Formula 4.

**Algorithm 2: PolicyGeneration**

**Input:**
- POMDP \( P = (S, A, T, O, Z) \)
- Safe-Reachability Objective \( G \)
- Constraint \( \Phi_k \)
- Candidate Plan \( \sigma_k = (b_s, a_{s+1}, b_{s+1}, \ldots, a_k, b_k) \)
- 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 \( i = s \) downto \( s \) do
25. /* Try observation \( o \) */
26. \( b'_i \leftarrow T(h_{i-1}, a_i, b_i, o) \); /* Call BPS to construct the branch */
27. \( \pi' \leftarrow \text{BPS}(P, b'_i, G, \Phi_k, i, h) \);
28. if \( \emptyset = \pi' \) then /* Construction failed */
29. Construct \( \phi \) using Formula 5
30. return \( \emptyset, \phi \);
31. /* Combine policy */
32. \( \pi(b'_i) \leftarrow \phi \);
33. return \( \pi, \emptyset \);

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 \( R^*(b_{init}, G) \) 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 \( b'_{i+1} = a'_{i+1} \) for the belief \( b_{i+1} \), action \( a_{i+1} \) and observation \( o_{i+1} \) at each step \( i \) respectively, which can be used to construct the candidate plan \( \sigma_k = (b'_s, a'_s, b'_s, \ldots, a'_k, b'_k) \).

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 \( G \). This candidate plan is a single path that only covers a particular observation \( o'_{i+k} \) at each step \( i \). To construct a valid policy, we should also consider other possible observations \( o'_{i} \neq o'_{i+k} \), 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 \( a_i^m \neq a_i^{\sigma_k} \) (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 \( a_i^m \), we need to compute a valid policy for the branch starting from \( b'_i \), which is another BPS problem with additional information (i.e., the current stack of constraints), 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 \( \sigma_k \) cannot be an element of a valid policy \( \sigma_k \notin \Omega_t \). In this case, we know that the prefix of the candidate plan \( (b_1^{\sigma_k}, a_1^{\sigma_k}, a_2^{\sigma_k}, \ldots, b_{h-1}^{\sigma_k}, a_h^{\sigma_k}) \) is invalid for current horizon \( h \) and we can add additional constraints \( \phi \) to block all invalid plans that have this prefix (line 29): \[
\phi = \neg ((b_2 = b_2^{\sigma_k}) \land (a_1 = a_1^{\sigma_k}) \land \bigwedge_{m=s+1}^{i-1} (a_m = a_m^{\sigma_k}) \land (a_m = a_m^{\sigma_k}) \land (b_m = b_m^{\sigma_k})) \quad (5)
\]

Note that \( \phi \) is only valid for current horizon \( h \) 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 \( b_i^{\sigma_k-1} \) is valid for all possible observations. Then we record this choice for belief \( b_i^{\sigma_k-1} \) 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 \( R(b_{init}) \) can be seen as a tree where the root node is the initial belief \( b_{init} \) 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 \( R_h(b_{init}) \) of height \( h \) contains \( O(|A|^h|O|^h) \) plans, where \(|A|\) and \(|O|\) are the size of action set \( A \) and observation set \( O \) respectively.

To synthesize a valid policy, a naive approach that checks every plan in the reachable belief space \( R_h(b_{init}) \) requires \( O(|A|^h|O|^h) \) calls to the SMT solver. This exponential growth of the reachable belief space \( R_h(b_{init}) \) 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 \( R^g(b_{init}, G) \) and efficiently explores the goal-constrained belief space \( R^g(b_{init}, G) \) 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 \( O(|O|^h) \) 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 \( O(|A|^h|O|^h) \) SMT solver calls. In general, \( I \) is often much smaller than \(|A|^h|O|^h \), 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 this yellow “shadow” region into \( N \) small cells. We assume that the locations of the obstacles are uniformly distributed among those cells and there is at most one obstacle in each cell. Therefore, a positive observation of one obstacle in one cell will decrease the probabilities of other obstacles in this cell. We also assume that the robot knows its exact location and the exact locations of cups to be picked up, but the robot does not know the locations of obstacles.

In this domain, the actuation and perception of the robot is 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 \( p_{fail} \), 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-up, look-down, look-forward, look-backward (look at the adjacent cell in the corresponding direction). When the robot calls look on a particular cell, \( i \), it may either make an observation \( o = o_{obstacle} \) representing obstacle observed \((1 \leq k \leq M)\) or \( o = o_{neg} \) representing nothing observed, depending on the probabilities defined by the probabilistic observation function \( Z(s', a, o) \). The robot needs to identify the specific obstacle in a cell to update the beliefs on the locations of other obstacles. Thus for a problem with \( M \) uncertain obstacles, the size of the observation set \( O \) is \(|O| = M + 1\).
3. Two pick-up actions that pick up an object from the black storage area: pick-up using the left hand \( a_{lh} \) and pick-up using the right hand \( a_{rh} \). 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:

\[
\text{Safe} = \{ b \in B \mid \forall i, k. b(\text{robot and obstacle}_k \in \text{cell}_i) < \delta_1 \} \\
\text{Dest} = \{ b \in B \mid \exists j. b(\text{cup}_j \text{ in robot’s hand}) > 1 - \delta_2 \}
\] (6)

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

Our goal is to generate a valid policy for the robot that is guaranteed to achieve the safe-reachability objective, even with uncertain actuation and perception.
6.1 Results

We use Z3 [9] as our backend incremental SMT solver. All experiments were conducted on a 2.50GHz Intel® processor with 32GB memory. We evaluate the performance of BPS for the kitchen domain in test cases with various numbers of obstacles. For all the tests, the horizon bound is $h = 30$ and the number of cells in the the yellow “shadow” region is $N = 20$. 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 Z3 requires solving the SMT constraints from scratch, rather than reusing the results from the previous Z3 calls. Since the BMC encoding used in BPS is particularly suitable for incremental solving (increasing horizon and blocking invalid plans corresponds to pushing/popping constraints), we expect enabling incremental solving leads to performance improvement in policy synthesis.

Figure 6 shows the performance results of BPS with and without incremental solving. Figure 7 shows the number of plans checked during policy synthesis. As we can see from Figure 6, enabling incremental solving leads to a significant performance improvement in policy synthesis (around 3 times speedup). In the largest test, the number of obstacles is $M = 4$ and the size of the observation set is $|O| = 5$. Therefore, there are $O(|A|^h |O|^h)$ (approximately $10^{50}$) plans in the reachable belief space. However, as we can see from Figure 7, for the largest test, the number of plans checked (around 3000) in BPS is very small compared to the total 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.

Figure 6 and Figure 7 also show that the synthesis time and the number of plans checked grow exponentially as the number of obstacles increases, which matches our complexity analysis in Section 5.4. This is because the current implementation operates on an exact tree representations 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 observation set (the branching factor) increase, which leads to an exponential growth of plans in the policy tree and makes the policy synthesis problem much harder.

6.2 Comparison with Unconstrained POMDPs and C/RS/CC-POMDPs

As we discussed in Section 4, if we model this benchmark as unconstrained POMDPs with quantitative objectives or C/RS/CC-POMDPs, and apply the corresponding solvers (e.g., point-based POMDP solvers for unconstrained POMDPs [25] or RAO [36] for CC-POMDPs) to construct a policy, the resulting policy chooses left-hand pick-up action $a_l$ to pick up a cup, which is not the best action in the worst case. On the other hand, using our formulation of POMDPs with safe-reachability objectives, the policy returned by our method chooses the best action $a_R$ (pick-up using right hand), which satisfies the safe-reachability objective in all possible executions. Therefore, our method provides stronger guarantee of both safety and reachability than those based on unconstrained POMDPs or C/RS-CC-POMDPs for the tested benchmark.

7 DISCUSSION

We presented a novel approach for the quantitative analysis problem of POMDPs with safe-reachability objectives. Our method introduces the notion of a goal-constrained belief space to improve computational efficiency. We construct constraints in a way similar to BMC [3] that compactly represent the goal-constrained belief space, which we efficiently explore through an incremental SMT solver [9]. We evaluate BPS in an uncertain robotics domain and the experimental results show that our method can synthesize policies for large problems with safe-reachability objectives.

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
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 M. C. Lin (Eds.) Springer, Berlin, Heidelberg, 175–191. https://doi.org/10.1007/978-3-642-17452-0_11
[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. http://dl.acm.org/citation.cfm?id=3038594.3038603
[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. http://dl.acm.org/citation.cfm?id=3016100.3016355
[6] Krishnendu Chatterjee, Martin Chmelík, and Mathieu Truclol. 2016. What is Decidable About Partially Observable Markov Decision Processes with Omega-regular Objectives. J. Comput. System Sci 82, 5 (Aug. 2016), 878–911. https://doi.org/10.1016/j.tcs.2016.02.009
[7] K. Chatterjee, M. ChmelÃ­k, R. Gupta, and A. Kanodia. 2015. Qualitative analysis of the POMDP solution space and continuous observations directly without discretization is another promising future direction for this work and its application in robotics.

Bounded Policy Synthesis for POMDPs with Safe-Reachability Objectives
AAMAS’18, July 2018, Stockholm, Sweden

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.

[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 M. C. Lin (Eds.) Springer, Berlin, Heidelberg, 175–191. https://doi.org/10.1007/978-3-642-17452-0_11
[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. http://dl.acm.org/citation.cfm?id=3038594.3038603
[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. http://dl.acm.org/citation.cfm?id=3016100.3016355
[6] Krishnendu Chatterjee, Martin Chmelík, and Mathieu Truclol. 2016. What is Decidable About Partially Observable Markov Decision Processes with Omega-regular Objectives. J. Comput. System Sci 82, 5 (Aug. 2016), 878–911. https://doi.org/10.1016/j.tcs.2016.02.009
[7] K. Chatterjee, M. ChmelÃ­k, R. Gupta, and A. Kanodia. 2015. Qualitative analysis of the POMDP solution space and continuous observations directly without discretization is another promising future direction for this work and its application in robotics.

Bounded Policy Synthesis for POMDPs with Safe-Reachability Objectives
AAMAS’18, July 2018, Stockholm, Sweden

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.

[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 M. C. Lin (Eds.) Springer, Berlin, Heidelberg, 175–191. https://doi.org/10.1007/978-3-642-17452-0_11
[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. http://dl.acm.org/citation.cfm?id=3038594.3038603
[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. http://dl.acm.org/citation.cfm?id=3016100.3016355
[6] Krishnendu Chatterjee, Martin Chmelík, and Mathieu Truclol. 2016. What is Decidable About Partially Observable Markov Decision Processes with Omega-regular Objectives. J. Comput. System Sci 82, 5 (Aug. 2016), 878–911. https://doi.org/10.1016/j.tcs.2016.02.009
[7] K. Chatterjee, M. ChmelÃ­k, R. Gupta, and A. Kanodia. 2015. Qualitative analysis of the POMDP solution space and continuous observations directly without discretization is another promising future direction for this work and its application in robotics.
[36] Pedro Santana, Sylvie Thiebaux, 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. http://dl.acm.org/citation.cfm?id=3016100.3016366

[37] 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.

[38] Adhiraj Somani, Nan Ye, David Hsu, and Wee Sun Lee. 2013. DESPOT: Online POMDP planning with regularization. In Advances in neural information processing systems. pp. 1772–1780.

[39] M. T. J. Spaan and N. Spaan. 2004. A point-based POMDP algorithm for robot planning. In Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004 IEEE International Conference on, Vol. 3. 2399–2404 Vol.3. https://doi.org/10.1109/ROBOT.2004.1307420

[40] 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). pp. 639–646. https://doi.org/10.1109/ICRA.2014.6906922

[41] Maria Svořková, Martin Chmelík, Kevin Leeb, 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. https://doi.org/10.1145/2728606.2728617

[42] A. Undurti and J. P. How. 2010. An online algorithm for constrained POMDPs. In 2010 IEEE International Conference on Robotics and Automation. 3966–3973. https://doi.org/10.1109/ROBOT.2010.5509743

[43] 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. http://dl.acm.org/citation.cfm?id=3038594.3038663