Sensor Synthesis for POMDPs with Reachability Objectives *

Krishnendu Chatterjee
IST Austria
krishnendu.chatterjee@ist.ac.at

Martin Chmelík
TTTech Computertechnik AG
martin.chmelik@tttech.com

Ufuk Topcu
University of Texas at Austin
utopcu@utexas.edu

Abstract

Partially observable Markov decision processes (POMDPs) are widely used in probabilistic planning problems in which an agent interacts with an environment using noisy and imprecise sensors. We study a setting in which the sensors are only partially defined and the goal is to synthesize "weakest" additional sensors, such that in the resulting POMDP, there is a small-memory policy for the agent that almost-surely (with probability 1) satisfies a reachability objective. We show that the problem is NP-complete, and present a symbolic algorithm by encoding the problem into SAT instances. We illustrate trade-offs between the amount of memory of the policy and the number of additional sensors on a simple example. We have implemented our approach and consider three classical POMDP examples from the literature, and show that in all the examples the number of sensors can be significantly decreased (as compared to the existing solutions in the literature) without increasing the complexity of the policies.

1 Introduction

In this work we study synthesis of sensor requirements for partially defined POMDPs, i.e., required precision of sensors, need for additional sensors, minimal set of necessary sensors, etc.

POMDPs. Markov decision processes (MDPs) are a standard model for systems that have both probabilistic and nondeterministic behaviors (Howard 1960), and they provide a framework to model and solve control and probabilistic planning problems (Filar and Vrieze 1997; Puterman 1994). The various choices of control actions for the controller (or planner) are modeled as *nondeterminism* while the stochastic response to the control actions is represented by the probabilistic behavior. In *partially observable MDPs (POMDPs)* to resolve the nondeterministic choices in control actions the controller observes the state space according to observations, i.e., the controller can only view the observation of the current state, but not the precise state (Papadimitriou and Tsitsiklis 1987). POMDPs are a widely used model for several applications and research fields, such as in computational biology (Durbin et al. 1998), speech processing (Mohri 1997), image processing (Culik and Kari 1997), software verification (Cerný et al. 2011), robot planning (Kaelbling, Littman, and Cassandra 1998), cyber-physical systems (Bagnato et al. 2017), reinforcement learning (Kaelbling, Littman, and Moore 1996), to name a few.

Reachability objectives. One of the most basic objectives is the *reachability objective*, where given a set of target states, the objective requires that some state in the target set is visited at least once. The classical computational questions for POMDPs with reachability objectives are as follows: (a) the *quantitative* question asks for the existence of a policy (that resolves the choice of control actions) that ensures the reachability objective with probability at least $0 < \lambda \leq 1$; and (b) the *qualitative* question is the special case of the quantitative question with $\lambda = 1$ (i.e., it asks that the objective is satisfied almost-surely).

Previous results. The quantitative question for POMDPs with reachability objectives is undecidable (Paz 1971) (and the undecidability result even holds for any approximation (Madani, Hanks, and Condon 2003)). In contrast, the qualitative question is EXPTIME-complete (Chatterjee, Doyen, and Henzinger 2010; Baier, Größer, and Bertrand 2012). The main algorithmic idea to solve the qualitative question (that originates from (Chatterjee et al. 2006)) is as follows: first construct the belief-support MDP explicitly (which is an exponential-size perfect-information MDP where every state is the support of a belief), and then solve the qualitative analysis on the perfect-information MDP (which is in polynomial time (Chatterjee, Jurdziński, and Henzinger 2003; Chatterjee and Henzinger 2014; 2011)). This gives an EXPTIME upper bound for the qualitative analysis of POMDPs, and a matching EXPTIME lower bound has been established in (Chatterjee, Doyen, and Henzinger 2010).

Modeling and analysis. In the design of systems there are two crucial phases, namely, the *modeling* phase, where a formal model of the system is constructed, and the *analy-*

*The research was partly supported by Vienna Science and Technology Fund (WWTF) Project ICT15-003, Austrian Science Fund (FWF) NFN Grant No S11407-N23 (RiSE/SHiNE), ERC Starting grant (279307: Graph Games), ONR N000141310778, ONR ONR N000141712623, and the European Union Horizon 2020 research and innovation programme under grant agreement No 731946.

sis phase, where the model is analyzed for correctness. Currently POMDPs are typically used in the analysis phase, where in the modeling phase a *fully specified* POMDP for the system is constructed, which is analyzed (in the model-checking terminology this is called *a posteriori* analysis or verification). However, POMDPs are seldom used in the modeling phase, where the model is not yet fully specified.

Partially specified POMDPs. In this work we consider the problem in which a POMDP is *partially specified* and can be used also in the modeling phase (i.e., *a priori* verification). To motivate our problem consider the standard applications in robotics or planning, where the state space of the POMDP is obtained from valuations of the variables of the system, and the sensors are designed to obtain the observations. We consider a partially specified POMDP where the state space and the transitions are completely specified, but the observations are not. This corresponds to scenarios where (i) the state space of the system is designed but the sensors have not yet been designed (Censi 2015; Mehta, DelPreto, and Rus 2015) or (ii) the sensors are designed and there is a possibility to augment and annotate the state space, in order to make the task for the agent simpler. In both scenarios the goal is to synthesize the observations (that is from the partially specified POMDP obtain a fully specified POMDP) such that in the resulting POMDP there is a policy that satisfies the reachability objective almost-surely. Since additional sensors increase complexity, one goal is to obtain as few additional observations as possible; and since policies represent controllers another goal is to ensure that the resulting policies are not too complex (Amato, Bernstein, and Zilberstein 2010). Concretely, we consider the following problem: given a partially specified POMDP (where the observations are not completely specified), the problem asks to synthesize at most ν additional observations such that in the resulting POMDP there is a policy with memory size at most μ to ensure that the reachability objective is satisfied almost-surely. Note that the problem we consider provides trade-offs between the additional observations (i.e., ν) and the memory of the policy (i.e., μ).

Significance of qualitative question. The qualitative question is of great importance as in several applications it is required that the correct behavior happens with probability 1. For example, in the analysis of randomized embedded schedulers, the important question is whether every thread progresses with probability 1. Moreover, though it might be sufficient that the correct behavior arises with probability at least $\lambda < 1$, the correct choice of the threshold λ is still challenging, due to simplifications and imprecisions introduced during modeling. Importantly it has been shown recently (Chatterjee et al. 2016) that for the fundamental problem of minimizing the total expected cost to reach the target set (Bertsekas 1995; Bonet and Geffner 2009; Kolobov et al. 2011; Kolobov, Mausam, and Weld 2012) under positive cost functions (or the stochastic shortest path problem), it suffices to first compute the almost-sure winning set, and then apply any finite-horizon algorithm for approximation. Moreover, the qualitative analysis problem has also a close connection with planning: while the qualitative analysis problem is different as compared to strong or contingent planning (Bonet 2010; Meuleau and Smith 2003; Maliah et al. 2014; Cimatti et al. 2003; Albore, Palacios, and Geffner 2009), it is equivalent to the qualitative contingent planning and the strong cyclic planning problem (Cimatti et al. 2003; Bertoli, Cimatti, and Pistore 2006). Thus results for qualitative analysis of POMDPs carry over to strong cyclic planning. Finally, besides the practical relevance, almost-sure convergence, like convergence in expectation, is a fundamental concept in probability theory, and provides the strongest probabilistic guarantee (Durrett 1996).

Our contributions. Our main contributions are as follows. First, we show that when ν and μ are constants, then the problem we consider is NP-complete. Note that the unrestricted problem (without restrictions on ν and μ) is EXPTIME-complete, because we can use observations of at most the size of the state space, and the general qualitative analysis problem of fully specified POMDPs is EXPTIME-complete. Second, we present an efficient reduction of our problem to SAT instances. This results in a practical, symbolic algorithm for the problem we consider and state-of-the-art SAT solvers, from artificial intelligence as well as many other fields (Biere 2013; Rintanen 2011; Biere et al. 1999), can be used for our problem. Then, we illustrate the trade-offs between the amount of memory of the policy and the number of additional sensors on a simple example. Finally, we present experimental results. We consider three classical POMDP examples from the literature, and show that in these examples the number of observations (hence the number of sensors in practice) can be significantly decreased as compared to the existing models in the literature, without increasing the memory size of the policies. We report scalability results on three examples showing that our implementation can handle POMDPs with ten thousand states.

2 Preliminaries

A probability distribution f on a finite set X is a function $f : X \to [0, 1]$ such that $\sum_{x \in X} f(x) = 1$, we denote by $\mathcal{D}(X)$ the set of all probability distributions on X and by $\mathsf{Uniform}(X)$ the uniform distribution over a finite set X. For a distribution $f \in \mathcal{D}(X)$ we denote by $\mathsf{Supp}(f) = \{x \in X \mid f(x) > 0\}$ the support of f.

POMDPs. A *Partially Observable Markov Decision Process (POMDP)* is defined as a tuple $P = (S, \mathcal{A}, \delta, \mathcal{Z}, \mathcal{O}, I)$ where

- (i) S is a finite set of states;
- (ii) $\mathcal{A}$ is a finite alphabet of *actions*;
- (iii) $\delta : S \times \mathcal{A} \to \mathcal{D}(S)$ is a *probabilistic transition function* that given a state s and an action $a \in \mathcal{A}$ gives the probability distribution over the successor states, i.e., $\delta(s, a)(s')$ denotes the transition probability from s to s' given action a;
- (iv) $\mathcal{Z}$ is a finite set of *observations*;
- (v) $I \in S$ is the unique initial state;
- (vi) $\mathcal{O} : S \to \mathcal{D}(\mathcal{Z})$ is a probabilistic *observation function* that maps every state to a probability distribution over observations.

Plays. A *play* (or a path) in a POMDP is an infinite sequence $(s_0, a_0, s_1, a_1, s_2, a_2, \ldots)$ of states and actions such that $s_0 = I$ and, for all $i \geq 0$, we have $\delta(s_i, a_i)(s_{i+1}) > 0$. We write Ω for the set of all plays.

Policies. A *policy (or a strategy)* is a recipe to extend prefixes of plays. That is, a policy is a function $\sigma : (\mathcal{Z} \cdot \mathcal{A})^* \cdot \mathcal{Z} \to \mathcal{D}(\mathcal{A})$ that, given a finite history of observations and actions, selects a probability distribution over the actions to be played next. We present an alternative definition of policies with finite memory for POMDPs.

Policies with Memory. A policy with memory is a tuple $\sigma = (\sigma_u, \sigma_n, M, m_0)$ with the following elements:

- M is a finite set of *memory elements*.
- The function $\sigma_n : M \to \mathcal{D}(\mathcal{A})$ is the *action selection function* that maps the current memory element to a probability distribution over actions.
- The function $\sigma_u : M \times \mathcal{Z} \times \mathcal{A} \to \mathcal{D}(M)$ is the *memory update function* that, given the current memory element, the current observation and action, updates the memory element probabilistically.
- The element $m_0 \in M$ is the *initial memory element*.

We will say a policy has memory size n if the number of memory elements is n, i.e., $|M| = n$.

Probability Measure. Given a policy σ and a starting state I, the unique probability measure obtained given σ is denoted as $\mathbb{P}_I^\sigma(\cdot)$ (Billingsley 1995; Littman 1996).

Reachability Objectives. Given a set $T \subseteq S$ of *target* states, a *reachability objective* in a POMDP P is a measurable set $\varphi \subseteq \Omega$ of plays defined as follows: $\mathsf{Reach}(T) = \{(s_0, a_0, s_1, a_1, s_2 \ldots) \in \Omega \mid \exists i \geq 0 : s_i \in T\}$, i.e., the set of plays, such that a state from the set of *target* states T is visited at least once.

In the remainder of the paper, we assume that the set of target states consists of a single *goal* state, i.e., $T = \{G\} \subseteq S$. This assumption is w.l.o.g. because it is always possible to add a state G with transitions from all target states in T. Note, that there are no costs or rewards associated with transitions.

Almost-Sure Winning. A policy σ is *almost-sure winning* for a POMDP P with a reachability objective $\mathsf{Reach}(T)$ iff $\mathbb{P}_I^\sigma(\mathsf{Reach}(T)) = 1$. In the sequel, whenever we refer to a winning policy, we mean an almost-sure winning policy.

3 Partially Defined Observation Functions

Traditionally, POMDPs are equipped with a fully defined observation function $\mathcal{O} : S \to \mathcal{D}(\mathcal{Z})$ that assigns to every state of the POMDP a probability distribution over observations. In order to model the *partially defined* observation function, we assume the input POMDP P is given with *partially defined* observation function $\mathcal{O}_\perp : S \to \mathcal{D}(\mathcal{Z} \cup \{\perp\})$. The probability distributions in the range of the function $\mathcal{O}_\perp$ contain an additional symbol $\perp$, and whenever for a state $s \in S$ we have $\perp \in \mathsf{Supp}(\mathcal{O}_\perp(s))$, we will say that the state s has observations only partially defined.

Observation function completions. We say a fully defined observation function $\mathcal{O}$ is a *completion* of a partially defined observation function $\mathcal{O}_\perp : S \to \mathcal{D}(\mathcal{Z} \cup \{\perp\})$ (and write $\mathcal{O}_\perp \prec \mathcal{O}$) if all of the following conditions are met:

1. There exists a set $\mathcal{Z}_A$ of additional observations and the observation function $\mathcal{O} : S \to \mathcal{D}(\mathcal{Z} \cup \mathcal{Z}_A)$ maps the states only to the set of old observations $\mathcal{Z}$ and the newly added observations $\mathcal{Z}_A$, i.e., the observations are defined for all states.
2. The function $\mathcal{O}$ agrees on assigned observations with $\mathcal{O}_\perp$, i.e., for all states $s \in S$ and observations $z \in \mathcal{Z}$, we have $\mathcal{O}_\perp(s)(z) = \mathcal{O}(s)(z)$.

Intuitively, given a POMDP with a reachability objective and a partially defined observation function $\mathcal{O}_\perp$, Problem 1 asks, whether there exists a completion using no more than ν additional observations such that in the resulting POMDP there exists an almost-sure winning policy using no more than μ memory elements. More formally we study:

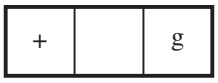

Figure 1: Grid POMDP

Problem 1 *Given a POMDP $P = (S, \mathcal{A}, \delta, \mathcal{Z}, \mathcal{O}_\perp, I)$ with a reachability objective $\mathsf{Reach}(T)$, and two integer parameters $\mu > 0$ and $\nu \geq 0$, decide whether there exists a completion $\mathcal{O}_\perp \prec \mathcal{O}$ using additional observations $\mathcal{Z}_A$ and an almost-sure winning policy $\sigma = (\sigma_u, \sigma_n, M, m_0)$ for the $\mathsf{Reach}(T)$ objective in the POMDP $P' = (S, \mathcal{A}, \delta, \mathcal{Z} \cup \mathcal{Z}_A, \mathcal{O}, I)$, with $|\mathcal{Z}_A| \leq \nu$ and $|M| \leq \mu$.*

Example 1 *Consider a deterministic POMDP depicted in Figure 1. There are three states corresponding to the position of the agent on the grid. The agent starts in the leftmost grid cell, and tries to move to the rightmost grid cell, where a treasure is hidden. There are three deterministic actions available to the agent:* move-left*,* move-right*, and* grab-treasure*. When the action* grab-treasure *is played in the rightmost cell, the agents wins, if it is played in any other cell the agent loses. The remaining two movement actions move the agent in the corresponding directions, if the wall is hit the agent loses.*

- *In the setting where $\mu = 3$ and $\nu = 1$, the problem is satisfiable by a policy that plays actions in the following sequence* move-right*,* move-right*, and* grab-treasure*.*
- *In the setting where $\mu = 2$ and $\nu = 2$, the problem is satisfiable by an observation function that assigns the rightmost grid cell an observation different from the two remaining grid cells. The policy plays action* move-right *in the first memory element until an observation corresponding to the rightmost cell is observed. After that it switches to the second memory element, where it plays action* grab-treasure*.*
- *In the setting where $\mu = 2$ and $\nu = 1$, the problem is not satisfiable, i.e., there is no two-memory almost-sure winning policy if all the states have the same observation.*

4 Complexity and SAT Encoding

In this section we consider properties of almost-sure winning policies, the complexity of Problem 1, and its encoding to SAT instances.

Complexity

Theorem 1 *Deciding Problem 1 given constant parameters μ and ν is NP-complete.*

Main ideas. We remark that Theorem 1 holds even if parameter μ is polynomial and ν is sublinear (such as logarithmic, or square-root) in the size of the POMDP.

- *Inclusion in NP.* Note that for polynomial μ and ν, a guess of the observation completion and the policy (if they exist) is polynomial. Thus we have polynomial-sized witnesses. Given a policy and an observation function, we obtain a Markov chain where qualitative analysis is polynomial time using standard discrete graph algorithms (Chatterjee, Jurdziński, and Henzinger 2003; Chatterjee and Henzinger 2014; 2011). Hence inclusion in NP follows.
- *NP-hardness.* An NP-hardness result was established for a similar problem, namely, for no memory policies in fully specified two-player games with partial-observation, in (Chatterjee, Kößler, and Schmid 2013, Lemma 1). The reduction constructed a game that is a DAG (directed acyclic graph), and replacing the adversarial player with a uniform distribution over choices shows that Problem 1 is NP-hard even with $\mu = 1$ (no memory policies) and $\nu = 0$ (fully specified observation).

SAT Encoding

In this section we present SAT encoding for Problem 1, which generalizes the results of (Chatterjee, Chmelik, and Davies 2016), where only a special case of fully specified observation function was considered.

Standard Results. We now present two basic lemmas. The following lemma presents a standard result for qualitative analysis of POMDPs, and it basically follows from the fact that in a Markov chain for qualitative analysis, the exact probability distributions are not important, and the supports of the distributions completely characterize almost-sure winning.

Lemma 1 *Given an almost-sure winning policy $\sigma = (\sigma_u, \sigma_n, M, m_0)$ for a $\mathsf{Reach}(T)$ objective, the policy $\sigma' = (\sigma'_u, \sigma'_n, M, m_0)$, where for $m \in M$ the action selection function σ'_n is defined as $\sigma'_n(m) = \mathsf{Uniform}(\mathsf{Supp}(\sigma_n(m)))$, and for $m \in M, a \in \mathcal{A}$, and $z \in \mathcal{Z}$ the memory update function σ'_u is defined as $\sigma'_u(m, z, a) = \mathsf{Uniform}(\mathsf{Supp}(\sigma_u(m, z, a)))$, is also an almost-sure winning policy for $\mathsf{Reach}(T)$.*

Given a policy σ and a POMDP $P = (S, \mathcal{A}, \delta, \mathcal{Z}, \mathcal{O}, I)$ and two state-memory pairs $(s, m), (s', m') \in S \times M$ we define a predicate $\mathsf{Path}_{k,\sigma,P}((s, m), (s', m'))$ to be True iff there exists a sequence of state-memory pairs $((s_1, m_1), (s_2, m_2), \ldots, (s_j, m_j))$ of length j where $0 < j \leq k$, such that $s = s_1, m = m_1, s' = s_j$, and $m' = m_j$, and for all $1 \leq i < j$ there exists an action $a_i \in \mathcal{A}$, observation $z_i \in \mathcal{Z}$, such that $\sigma_n(m_i)(a_i) > 0$, $\delta(s_i, a_i)(s_{i+1}) > 0$, $\mathcal{O}(s_{i+1})(z_i) > 0$, and $\sigma_u(m_i, z_i, a_i)(m_{i+1}) > 0$. Let $R_{P,\sigma}$ be the set of all pairs $(s, m) \in S \times M$ such that $\mathsf{Path}_{k,\sigma,P}((I, m_0), (s, m))$ is True for some $k \in \mathbb{N}$. The following lemma states that almost-sure winning policies are characterized by paths of bounded length to the goal state.

Lemma 2 *A policy σ is almost-sure winning in a POMDP P iff for every state-memory pair $(s, m) \in R_{P,\sigma}$ the predicate $\mathsf{Path}_{k,\sigma,P}((s, m), (G, m'))$ for some $m' \in M$ and $k = |S| \cdot |M|$ is True.*

Consequences for the SAT encoding. The consequences of the presented lemmas for the SAT encoding are as follows: Lemma 1 allows to encode only the supports of the probability distributions of the policy σ, i.e., a boolean property whether an action (resp. a memory element) is present in the support of the distribution σ_n (resp. σ_u). Lemma 2 allows to characterize state-memory pairs (s, m) that are almost-sure winning by encoding the boolean predicate $\mathsf{Path}_{k,\sigma,P}$ that represents existence of paths to the goal state.

Notations. Given a POMDP P, reachability objective $\mathsf{Reach}(T)$, a bound on the number of memory elements μ, a bound on the number of additional observations ν, and a path length $k \leq |S| \cdot |M|$ which is a parameter related to the length of paths in the POMDP, we will define a formula in conjunctive normal form (CNF) $\Phi_{k,\mu,\nu}$ that for a sufficiently large parameter k, e.g., $k = |S| \cdot |M|$, will be satisfiable if and only if there exists completion of the observation function using no more than ν additional observations and an almost-sure winning policy with no more than μ memory elements, i.e., the associated instance of Problem 1 is true. We define the set $\mathcal{Z}_A = \{z_1, z_2, \ldots, z_\nu\}$ of additional observations, and denote by $\mathcal{Z}' = \mathcal{Z} \cup \mathcal{Z}_A$ the disjoint union of the old observations in $\mathcal{Z}$ and the newly added observations in $\mathcal{Z}_A$. We describe the CNF formula $\Phi_{k,\mu,\nu}$ by defining all of its Boolean variables, followed by the clausal constraints over those variables.

Boolean Variables. We first introduce the variables.

- We begin by encoding the action selection function σ_n of the policy σ. We introduce a Boolean variable $A_{m,a}$ for each memory-state $m \in M$ and action $a \in \mathcal{A}$ to represent that action a is played with positive probability in memory state m, i.e., that $\sigma_n(m)(a) > 0$ (see Lemma 1).
- Next, we encode the memory update function σ_u. We introduce a Boolean variable $M_{m,z,a,m'}$ for each pair of memory-states $m, m' \in M$, observation $z \in \mathcal{Z}'$ and action $a \in \mathcal{A}$. If such a variable is assigned to True, it indicates that, if the current memory-state is m, the current observation is z, and action a is played, then it is possible that the new memory-state is m', i.e., $\sigma_u(m, z, a)(m') > 0$ (see Lemma 1).
- We encode the completion $\mathcal{O}$ of the partially defined observation function $\mathcal{O}_\perp$. We introduce a variable $O_{s,z}$ for every state $s \in S$ and observation $z \in \mathcal{Z}'$. The intuitive

meaning is that the observation function completion $\mathcal{O}$ assigns to state s observation z with positive probability.

- Boolean variables $C_{i,m}$ for each state $i \in S$ and memory state $m \in M$ indicate which (state, memory-state) pairs are reachable by the policy.
- The variables $P_{i,m,j}$ for all $i \in S$, $m \in M$, and $0 \leq j \leq k$, correspond to the proposition that there is a path of length at most j from (i, m) to the goal state, that is compatible with the policy.

Logical Constraints. We introduce the following clause for each $m \in M$ to ensure that at least one action is chosen with positive probability for each memory state (see Lemma 1):

$$\bigvee_{j \in \mathcal{A}} A_{m,j}.$$

To ensure that the memory update function is well-defined, we introduce the following clause for each $m \in M$, $a \in \mathcal{A}$ and $z \in \mathcal{Z}'$ (see Lemma 1):

$$\bigvee_{m' \in M} M_{m,z,a,m'}.$$

To ensure that every state i has at least one observation z in the support of the observation function, we introduce the following clause for every state $i \in S$:

$$\bigvee_{z \in \mathcal{Z}'} O_{i,z}.$$

For every state $i \in S$ and every $z \in \mathsf{Supp}(\mathcal{O}(i))$, we enforce the consistency by adding the clause:

$$O_{i,z}.$$

For every state $i \in S$, with observations fully defined, i.e., $\perp \notin \mathsf{Supp}(\mathcal{O}(i))$, for every additional observation $z \in \mathcal{Z}_A$ we add the following clause:

$$\neg O_{i,z}.$$

The following clauses ensure that the variables $C_{i,m}$ will be assigned True for all pairs (i, m) that are reachable using the policy:

$$(C_{i,m} \wedge A_{m,a} \wedge O_{j,z} \wedge M_{m,z,a,m'}) \Rightarrow C_{j,m'}.$$

Such a clause is defined for each pair $m, m' \in M$ of memory-states, each pair $i, j \in S$ of states, each observation $z \in \mathcal{Z}$ and each action $a \in \mathcal{A}$ such that $\delta(i, a)(j) > 0$.

Therefore, the fact that the initial state and initial memory element is reachable is enforced by adding the single clause

$$C_{I,m_0}.$$

We introduce the following unit clause for each $m \in M$ and $0 \leq j \leq k$, which says that the goal state with any memory element is reachable from the goal state and that memory element using a path of length at most 0:

$$P_{G,m,0}.$$

Next, we define the following binary clause for each $i \in S$ and $m \in M$ so that, if the pair (i, m) of a state and a memory element is reachable, then the existence of a path from (i, m) to the goal state is enforced (see Lemma 2):

$$C_{i,m} \Rightarrow P_{i,m,k}.$$

Finally, we use the following constraints to define the value of variables $P_{i,m,j}$ for all $i \in S$, $m \in M$, and $0 \leq j \leq k$ in terms of the chosen policy (see Lemma 1 and definition of predicate Path).

$$P_{i,m,j} \iff \bigvee_{a \in \mathcal{A}} \left[A_{m,a} \wedge \left(\bigvee_{\substack{m' \in M, z \in \mathcal{Z}, \\ i' \in S : \delta(i,a)(i') > 0}} [O_{i',z} \wedge M_{m,z,a,m'} \wedge P_{i',m',j-1}] \right) \right].$$

The conjunction of all clauses defined above forms the CNF formula $\Phi_{k,\mu,\nu}$.

Theorem 2 *The formula $\Phi_{k,\mu,\nu}$ for $k \geq |S| \cdot \mu$ is satisfiable, iff there exists a completion of the observation function using no more than ν additional observations and an almost-sure winning policy using no more than μ memory elements.*

Proof [Proof sketch.] *Satisfiable formula $\Rightarrow$ completion and a policy:* If the formula $\Phi_{k,\mu,\nu}$ is satisfiable, the SAT solver outputs a valuation v of the variables. The boolean variables $O_{s,z}$ that are true according to v encode the completion of the observation function, variables $A_{m,a}$ encode the action selection function σ_n, and $M_{m,z,a,m'}$ encode the memory update function σ_u. The fact that the encoded policy is almost-sure winning follows from the clauses and lemmas 1 and 2.

Completion and a policy $\Rightarrow$ satisfiable formula: Given a completion of the observation function and an almost-sure winning policy σ we show how to construct a satisfying valuation v for the formula $\Phi_{k,\mu,\nu}$. The completion of the observation function $\mathcal{O}_\perp$ gives the valuation for the $O_{s,z}$ variables, the action selection function σ_n for the $A_{m,a}$ variables, and the memory update function σ_u for the $M_{m,z,a,m'}$ variables. The valuation for the $C_{i,m}$ and $P_{i,m,j}$ is obtained by constructing $R_{P,\sigma}$ and examining which state-memory pairs are reachable and the shortest path to a goal state. □

Partial Specification with Constraints

In the previous section we have presented a SAT encoding for POMDPs with partially specified observation functions. In this section we discuss additional constraints that might be desirable and our encoding can be easily extended to handle these constraints.

Non-distinguishable states. In many scenarios it might be the case that there are states that cannot be distinguished by any available sensors, i.e., the observation assigned to these states must necessarily be the same. This can be enforced by adding the following clause for any pair $j, j' \in S$ of non-distinguishable states and all the observations $z \in \mathcal{Z}'$.

$$O_{j,z} \Leftrightarrow O_{j',z}.$$

Distinguishable states. In some scenarios it might be the case that two states cannot have the same observation. This can be enforced by adding the following clause for any pair $j, j' \in S$ of states and all the observations $z \in \mathcal{Z}'$.

$$(O_{j,z} \wedge \neg O_{j',z}) \vee (\neg O_{j,z} \wedge O_{j',z}).$$

Dependencies among observations. Various dependencies among observations can be expressed. For example in a state i whenever an observation z is observed with positive probability also observation z' is observed with positive probability can be expressed by the following clause:

$$O_{i,z} \Rightarrow O_{i,z'}.$$

Adding sensor variables. Let P be a POMDP with the set of observations $\mathcal{Z} = \{z_1, z_2, \ldots, z_n\}$ and observation function $\mathcal{O}$. By adding a new sensor C, that receives values $\mathsf{Val}(C) = \{c_1, c_2, \ldots, c_l\}$, the new set of observations in the modified POMDP $\widetilde{P}$ is $\widetilde{\mathcal{Z}} = \mathcal{Z} \times \mathsf{Val}(C)$ with observation function $\widetilde{\mathcal{O}}$. This corresponds to increasing the observation dimensionality, rather than increasing cardinality. Our approach allows to synthesise observations in POMDP $\widetilde{P}$ as follows:

- We set the observations of all states to be undefined.
- We add constrains to the resulting formula as follows: In POMDP $\widetilde{P}$ an observation $(z, c) \in \widetilde{\mathcal{Z}}$ for some $c \in \mathsf{Val}(C)$ is received with positive probability in state i, i.e., $\widetilde{\mathcal{O}}(i)((z, c)) > 0$ if and only if the observation $z \in \mathcal{Z}$ is received in state i in the original POMDP P with positive probability, i.e., $\mathcal{O}(i)(z) > 0$.

  1. For every state $i \in S$ and observation $z \in \mathsf{Supp}(\mathcal{O}(i))$ we add the following clause:

$$\bigvee_{c \in \mathsf{Val}(C)} O_{i,(z,c)}$$

  2. For every state $i \in S$ and observation $z \notin \mathsf{Supp}(\mathcal{O}(i))$ we add the following constraint:

$$\bigwedge_{c \in \mathsf{Val}(C)} \neg O_{i,(z,c)}$$

Deterministic observations function. For every state $s \in S$ we introduce the following clause:

$$\sum_{z \in \mathcal{Z}'} O_{i,z} = 1 \ \ (\text{ exactly one of } O_{i,z} \text{ for } z \in \mathcal{Z}' \text{ is true}).$$

Remark 1 *Deterministic observation functions are a special case of probabilistic observation functions. Therefore, the number of observations in the deterministic case is an upper bound for the probabilistic case. However, a probabilistic observation function might require less observations.*

5 Experimental Results

In this section we present experimental results and evaluate our approach on several POMDP examples that were published in the literature. We have implemented the encoding presented in Section 4 as a program in Python and use the MiniSAT solver (Eén and Sörensson 2003) on an Intel(R) Xeon(R)@ 3.50GHz CPU.

Name	Grid	# States	μ	ν	Time (s)	SAT
Escape2	2×2	19	5	5	0.18	$\surd$
Escape3	3×3	84	5	5	1.22	$\surd$
Escape4	4×4	259	5	5	5.69	$\surd$
Escape5	5×5	628	5	5	19.31	$\surd$
Escape6	6×6	1299	5	5	52.65	$\surd$
Escape7	7×7	2404	5	5	131.77	$\surd$
Escape8	8×8	4099	5	5	280.19	$\surd$
Escape9	9×9	6564	5	5	674.42	$\surd$
Escape10	10×10	10003	5	5	1519.48	$\surd$

Table 1: Escape instances.

Remark 2 *In our experimental results we consider the synthesis of deterministic observation functions. As mentioned in Remark 1, deterministic observation functions provide upper bound for the number of observations required by probabilistic observation functions. Thus synthesizing deterministic observation functions with fewer observations is the more challenging problem, which we consider to illustrate the effectiveness of our approach.*

We present our results first on a small, simple example to illustrate how various selections of the memory bounds μ and additional observation bounds ν affect the computed policies and discuss the possible trade-offs between the memory vs. observation budgets in Problem 1.

Name	Grid	# States	μ	ν	Time (s)	SAT
Hallway1	7×5	38	2	2	0.22	$\times$
Hallway1	7×5	38	3	2	0.55	$\surd$
Hallway2	11×9	190	3	2	5.95	$\times$
Hallway2	11×9	190	3	3	5.28	$\times$
Hallway2	11×9	190	4	2	20.82	$\surd$
Hallway3	11×10	226	3	2	6.53	$\times$
Hallway3	11×10	226	3	3	7.33	$\times$
Hallway3	11×10	226	4	2	28.98	$\surd$

Table 2: Hallway instances.

Deterministic Hallway

We consider a simplification of the well-known Hallway problem (Littman, Cassandra, and Kaelbling 1995), where an agent navigates itself on a rectangular grid (see Figure 2a). There are four actions N, E, S, and W available to the agent. For simplicity, all the movement on the grid is deterministic (probabilistic movement is considered in the Hallway problem later in the scalability evaluation). There are multiple initial states (depicted as $+$ in Figure 2a) and the agent starts in any of them with uniform probability. The objective of the agent is to reach any of the goal states (depicted as g in Figure 2a). Whenever an agent hits a wall or enters a trap state (depicted as x in Figure 2a) an absorbing

Name	# States	μ	ν	Time (s)	SAT
RockSample4	351	2	2	2.43	$\surd$
RockSample5	909	2	2	18.14	$\surd$
RockSample6	2187	2	2	95.28	$\times$
RockSample6	2187	2	3	165.87	$\times$
RockSample6	2187	3	2	519.21	$\surd$
RockSample7	5049	2	2	565.49	$\times$
RockSample7	5049	3	2	565.43	$\times$
RockSample7	5049	3	3	5196.40	$\surd$

Table 3: RockSample instances.

state is reached, from which it is no longer possible to reach the desired goal states. We consider that there are no observations defined in the POMDP, i.e., for all states $s \in S$ we have $\mathcal{O}_\bot(s) = \bot$.

4 **memory elements and** 2 **observations.** In the setting, where $\mu = 4$ and $\nu = 2$ the SAT solver reports that there exists a completion of $\mathcal{O}_\bot \prec \mathcal{O}$ and an almost-sure winning policy σ. We depict the synthesized observation function $\mathcal{O}$ in Figure 2b, where the red color corresponds to the new synthesized observation z_1 and green color corresponds to new synthesized observation z_2. The synthesized policy σ uses four memory elements $M = \{m_1, m_2, m_3, m_4\}$. The synthesized action selection function is defined as $\sigma_n(m_1) = E, \sigma_n(m_2) = W, \sigma_n(m_3) = S, \sigma_n(m_4) = S$. The computed policy initially updates its memory element to m_3 in case the first observation is z_1 (red) and to m_4 if the observation is z_2 (green), i.e., the information whether the agent starts in the left or right start state is stored in the memory element. Then action S is played until the bottom row is reached (this is detected by changed observations, from z_2 to z_1 in the left part, and from z_1 to z_2 in the right part). Finally, in case the agent is in the left part, memory element m_3 is updated to m_2 and by action W the goal state is reached. Similarly, in the right part, memory element m_4 is updated to m_1 and by action E the goal state is reached.

3 **memory elements and** 3 **observations.** In the setting, where $\mu = 3$ and $\nu = 3$ the SAT solver reports there exists a completion of $\mathcal{O}_\bot \prec \mathcal{O}$ and an almost-sure winning policy σ. This allows to reduce the number of memory elements needed, provided we add one more observation to the POMDP. We depict the synthesized observation function $\mathcal{O}$ in Figure 2c, where red color corresponds to the new synthesized observation z_1, green color to the new observation z_2, and blue color corresponds to the new observation z_3. The synthesized policy σ uses three memory elements $M = \{m_1, m_2, m_3\}$. The synthesized next-action selection function is defined as $\sigma_n(m_1) = W, \sigma_n(m_2) = E, \sigma_n(m_3) = S$. The computed policy starts with the initial memory element m_3 and plays actions S until either observation z_1 or z_3 is received. In case z_1 (red) is observed, the policy is updated to memory element m_2 and reaches the goal state with action E. In case z_3 (blue) is observed, the policy is updated to memory element m_1 and reaches the goal state with action W.

3 **memory elements and** 2 **observations.** In the setting, where $\mu = 3$ and $\nu = 2$ the SAT solver reports there does not exist a completion of $\mathcal{O}_\bot \prec \mathcal{O}$ that would allow for a two-memory almost-sure winning policy. This follows from the fact that at least three memory elements are necessary for actions S, E, W, i.e., with the restriction $\mu = 3$, there are no available memory elements to store additional information. As the agent needs to avoid hitting into walls, a randomized action selection function cannot be used. It follows that there can be at most one memory element m such that $\sigma_n(m) = S$. It follows easily that, with only two observations and one memory element for action S, it is not possible to detect that the agent is already present in the bottom row.

Scalability Evaluation

In this part we demonstrate the scalability of our approach on three well-known POMDP examples of varying sizes. Our results show that in all cases the observations considered in these examples from the literature are unnecessarily refined and significantly less precise observations suffice even without making the policies more complicated.

Escape POMDPs. The problem is originally based on a case study published in (Svorenova et al. 2015), where the goal is to compute a policy to control a robot in an uncertain environment. A robot navigates on a square grid. There is an agent moving over the grid, and the robot must avoid being captured by the agent forever. The robot has four actions: move north, south, east, or west. These actions have deterministic effects, i.e., they always succeed. In the original POMDP instance, there are 179 different observations.

The memory and observation trade-offs for the smallest instance *Escape2* are depicted on Figure 3, which shows that for $\mu = 5$ and $\nu = 179$ there exists an almost-sure policy. However, it is possible to significantly decrease the number of observations to $\nu = 5$ and there is still an almost-sure winning policy with $\mu = 5$. If μ is increased to 8, it is possible to decrease ν to 4. If the memory size μ is further increased to 12, it is possible to reduce the number of observations ν to 3. We illustrate the scalability results in Table 1, where we report the number of states, the parameters μ, ν, the running time of the SAT solver, and whether the formula is satisfiable. In all the cases with $\mu = 5$ and $\nu = 5$ there exists an almost-sure policy and the sizes of the instances go up to 10000 states. There are approx. 1.6×10^8 clauses in the largest instance.

Hallway POMDPs. Hallway POMDP instances are inspired by the Hallway problem introduced in (Littman, Cassandra, and Kaelbling 1995) and used later in (Spaan 2004; Smith and Simmons 2004; Bonet and Geffner 2009; Chatterjee et al. 2015). In the Hallway POMDPs, a robot navigates on a rectangular grid. The grid has barriers through which the robot cannot move, as well as trap locations that destroy the robot. The robot must reach a specified goal location. The robot has three actions: move forward, turn left, and turn right. The actions may all fail with positive probability, in which case the robot's state remains unchanged. The state is therefore comprised of the robot's location in the grid, and its orientation. Initially, the robot is randomly positioned among multiple start locations. Originally, the POMDP in-

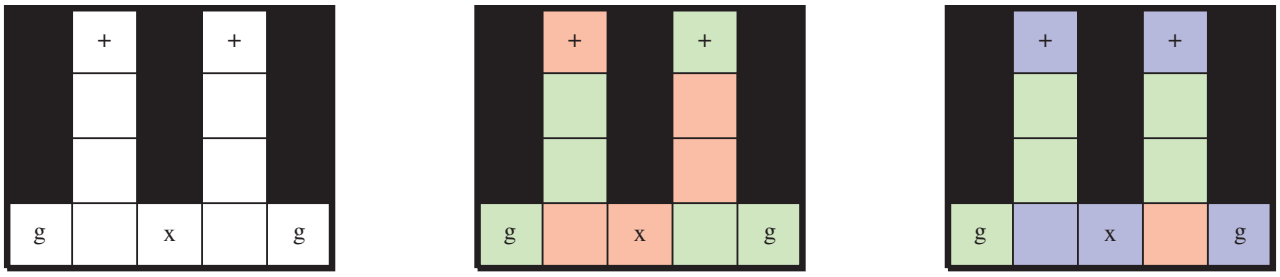

Figure 2: (a) Deterministic Hallway POMDP. (b) Synthesized $\mathcal{O}$ for Hallway $\mu = 4$ and $\nu = 2$. (c) Synthesized $\mathcal{O}$ for Hallway $\mu = 3$ and $\nu = 3$.

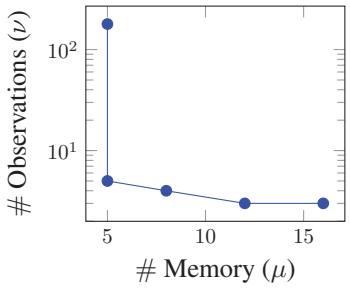

Figure 3: Memory vs. observation trade-off for Escape2

stances contain 16 different observations. The results are reported in Table 2, where we consider three different Hallway instances and vary the parameters μ for the number of memory elements and ν for the number of additional observations. For every entry we report the number of states, the parameters μ, ν, the running time of the SAT solver, and whether the formula is satisfiable. The results show that in all cases two observations are sufficient. In the smallest instance memory of size 3 is sufficient. For larger instances, memory size needs to be increased to 4. There are approx. 1.1×10^7 clauses in the largest instance.

RockSample POMDPs. We consider a variant of the RockSample problem introduced in (Smith and Simmons 2004) and used later in (Bonet and Geffner 2009; Chatterjee et al. 2015). The RockSample instances model rover science exploration. Only some of the rocks have a scientific value, and we will call these rocks "good". Whenever a bad rock is sampled the rover is destroyed and a losing absorbing state is reached. If a rock is sampled for the second time, then with probability 0.5 the action has no effect. With the remaining probability the sample is destroyed and the rock needs to be sampled one more time. An instance of the RockSample problem is parametrized with a parameter $[n]$: n is the number of rocks on a grid of size 3×3. The goal of the rover is to obtain two samples of good rocks. The results are presented in Table 3. Originally, the POMDP instances contain 15 different observations. The results show that, with increasing sizes of the POMDP instances, either increasing the memory size or increasing the number of additional observations is enough to obtain an almost-sure winning policy. There are approx. 9.8×10^7 clauses in the largest instance.

6 Conclusion

In this work we consider POMDPs with partially specified observations, and the problem to synthesize additional observations along with small-memory almost-sure winning policies. Interesting directions of future work would be to consider (a) the usage of incremental SAT solvers (b) other aspects of partial specifications (such as transitions), and (c) other objectives, such as discounted-sum.

References

Albore, A.; Palacios, H.; and Geffner, H. 2009. A translation-based approach to contingent planning. In *IJCAI*, 1623–1628.

Amato, C.; Bernstein, D.; and Zilberstein, S. 2010. Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs. *AAMAS* 21(3):293–320.

Bagnato, A.; Biro, R. K.; Bonino, D.; Pastrone, C.; Elmenreich, W.; Reiners, R.; Schranz, M.; and Arnautovic, E. 2017. Designing Swarms of Cyber-Physical Systems: the H2020 CPSwarm Project. In *ACM International Conference on Computing Frontiers*.

Baier, C.; Größer, M.; and Bertrand, N. 2012. Probabilistic omega-automata. *J. ACM* 59(1).

Bertoli, P.; Cimatti, A.; and Pistore, M. 2006. Strong cyclic planning under partial observability. *ICAPS* 141:580.

Bertsekas, D. 1995. *Dynamic Programming and Optimal Control*. Athena Scientific. Volumes I and II.

Biere, A.; Cimatti, A.; Clarke, E.; Fujita, M.; and Zhu, Y. 1999. Symbolic model-checking using SAT procedures instead of BDDs. In *DAC*, 317–320.

Biere, A. 2013. Lingeling, plingeling and treengeling entering the SAT competition 2013. In *SAT Comp.*

Billingsley, P., ed. 1995. *Probability and Measure*. Wiley-Interscience.

Bonet, B., and Geffner, H. 2009. Solving POMDPs: RTDP-Bel vs. point-based algorithms. In *IJCAI*, 1641–1646.

Bonet, B. 2010. Conformant plans and beyond: Principles and complexity. *Artif. Intell.* 174(3-4):245–269.

Censi, A. 2015. A Mathematical Theory of Co-Design. *arXiv preprint arXiv:1512.08055*.

Cerný, P.; Chatterjee, K.; Henzinger, T. A.; Radhakrishna, A.; and Singh, R. 2011. Quantitative synthesis for concurrent programs. In *Proc. of CAV*, LNCS 6806, 243–259. Springer.

Chatterjee, K., and Henzinger, M. 2011. Faster and dynamic algorithms for maximal end-component decomposition and related graph problems in probabilistic verification. In *SODA*. ACM-SIAM.

Chatterjee, K., and Henzinger, M. 2014. Efficient and dynamic algorithms for alternating Büchi games and maximal end-component decomposition. *J. ACM* 61(3):15.

Chatterjee, K.; Doyen, L.; Henzinger, T.; and Raskin, J. 2006. Algorithms for omega-regular games with imperfect information. In *CSL'06*, 287–302. LNCS 4207, Springer.

Chatterjee, K.; Chmelik, M.; Gupta, R.; and Kanodia, A. 2015. Qualitative Analysis of POMDPs with Temporal Logic Specifications for Robotics Applications. *ICRA*.

Chatterjee, K.; Chmelik, M.; Gupta, R.; and Kanodia, A. 2016. Optimal Cost Almost-sure Reachability in POMDPs. In *AI*.

Chatterjee, K.; Chmelik, M.; and Davies, J. 2016. A Symbolic SAT-based Algorithm for Almost-sure Reachability with Small Strategies in POMDPs. *AAAI* 3225–3232.

Chatterjee, K.; Doyen, L.; and Henzinger, T. A. 2010. Qualitative analysis of partially-observable Markov decision processes. In *MFCS*, 258–269.

Chatterjee, K.; Jurdziński, M.; and Henzinger, T. 2003. Simple stochastic parity games. In *CSL'03*, LNCS 2803, 100–113. Springer.

Chatterjee, K.; Kößler, A.; and Schmid, U. 2013. Automated analysis of real-time scheduling using graph games. In *HSCC'13*, 163–172.

Cimatti, A.; Pistore, M.; Roveri, M.; and Traverso, P. 2003. Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence* 147(1):35–84.

Culik, K., and Kari, J. 1997. Digital images and formal languages. *Handbook of formal languages* 599–616.

Durbin, R.; Eddy, S.; Krogh, A.; and Mitchison, G. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge Univ. Press.

Durrett, R. 1996. *Probability: Theory and Examples (Second Edition)*. Duxbury Press.

Eén, N., and Sörensson, N. 2003. An extensible SAT-solver. In *Theory and Applications of Satisfiability Testing*, 502–518.

Filar, J., and Vrieze, K. 1997. *Competitive Markov Decision Processes*. Springer-Verlag.

Howard, H. 1960. *Dynamic Programming and Markov Processes*. MIT Press.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artif. Intell.* 101(1):99–134.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement learning: A survey. *JAIR* 4:237–285.

Kolobov, A.; Mausam; Weld, D.; and Geffner, H. 2011. Heuristic search for generalized stochastic shortest path MDPs. In *ICAPS*.

Kolobov, A.; Mausam; and Weld, D. 2012. A theory of goal-oriented MDPs with dead ends. In *UAI*, 438–447.

Littman, M. L.; Cassandra, A. R.; and Kaelbling, L. P. 1995. Learning policies for partially observable environments: Scaling up. In *ICML*, 362–370.

Littman, M. L. 1996. *Algorithms for Sequential Decision Making*. Ph.D. Dissertation, Brown University.

Madani, O.; Hanks, S.; and Condon, A. 2003. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artif. Intell.* 147(1-2):5–34.

Maliah, S.; Brafman, R.; Karpas, E.; and Shani, G. 2014. Partially observable online contingent planning using landmark heuristics. In *ICAPS*.

Mehta, A.; DelPreto, J.; and Rus, D. 2015. Integrated codesign of printable robots. *Journal of Mechanisms and Robotics* 7(2):021015.

Meuleau, N., and Smith, D. E. 2003. Optimal Limited Contingency Planning. In *UAI*, 417–426.

Mohri, M. 1997. Finite-state transducers in language and speech processing. *Comp. Linguistics* 23(2):269–311.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov decision processes. *Mathematics of Operations Research* 12:441–450.

Paz, A. 1971. *Introduction to probabilistic automata (Computer science and applied mathematics)*. Academic Press.

Puterman, M. L. 1994. *Markov Decision Processes*. John Wiley and Sons.

Rintanen, J. 2011. Planning with SAT, admissible heuristics and A*. In *IJCAI*, 2015–2020.

Smith, T., and Simmons, R. 2004. Heuristic search value iteration for POMDPs. In *UAI*, 520–527. AUAI Press.

Spaan, M. 2004. A point-based POMDP algorithm for robot planning. In *ICRA*, volume 3, 2399–2404. IEEE.

Svorenova, M.; Chmelik, M.; Leahy, K.; Eniser, H. F.; Chatterjee, K.; Cerna, I.; and Belta, C. 2015. Temporal Logic Motion Planning using POMDPs with Parity Objectives. In *HSCC*.