Temporal Logic Control for Stochastic Linear Systems using Abstraction Refinement of Probabilistic Games

Mária Svoreňová
Faculty of Informatics
Masaryk University
Brno, Czech republic
svorenova@mail.muni.cz

Jan Křetínský
IST Austria
Klosterneuburg, Austria
jan.kretinsky@ist.ac.at

Martin Chmelík
IST Austria
Klosterneuburg, Austria
martin.chmelik@ist.ac.at

Krishnendu Chatterjee
IST Austria
Klosterneuburg, Austria
kchatterjee@ist.ac.at

Ivana Černá
Faculty of Informatics
Masaryk University
Brno, Czech republic
cerna@muni.cz

Calin Belta
Dep. of Mechanical Eng.
Boston University
Boston, MA, USA
cbelta@bu.edu

ABSTRACT

We consider the problem of computing the set of initial states of a dynamical system such that there exists a control strategy to ensure that the trajectories satisfy a temporal logic specification with probability 1 (almost-surely). We focus on discrete-time, stochastic linear dynamics and specifications given as formulas of the Generalized Reactivity(1) fragment of Linear Temporal Logic over linear predicates in the states of the system. We propose a solution based on iterative abstraction-refinement, and turn-based 2-player probabilistic games. While the theoretical guarantee of our algorithm after any finite number of iterations is only a partial solution, we show that if our algorithm terminates, then the result is the set of satisfying initial states. Moreover, for any (partial) solution our algorithm synthesizes witness control strategies to ensure almost-sure satisfaction of the temporal logic specification. We demonstrate our approach on an illustrative case study.

1. INTRODUCTION

The formal verification problem, in which the goal is to check whether behaviors of a finite model satisfy a correctness specification, received a lot of attention during the past thirty years [10, 2]. In contrast, in the synthesis problem the goal is to synthesize or control a finite system from a temporal logic specification. While the synthesis problem also has a long tradition [9, 3, 22], it has gained significant attention in formal methods more recently. These techniques are being deployed in control and path planning in particular: model checking techniques can be adapted to synthesize (optimal) controllers for deterministic finite systems [24, 14], Büchi and Rabin games can be reformulated as control strategies for nondeterministic systems [29, 27], and probabilistic games can be used to control finite probabilistic systems such as Markov decision processes [23, 18].

With the widespread integration of physical and digital components in cyber physical systems, and the safety and security requirements in such systems, there is an increased need for the development of formal methods techniques for systems with infinite state spaces, normally modeled as differential or differential equations. Most of the works in the area use partitions and simulation / bisimulation relations to construct a finite abstraction of the system, followed by verification or control of the abstraction. Existing results showed that such approaches are feasible for discrete and continuous time linear systems [26, 12]. With some added conservativeness, more complicated dynamics and stochastic dynamics can also be handled [11, 17].

In this work, we focus on the problem of finding the set of initial states of a dynamic system from which a given constraint can be satisfied, and synthesizing the corresponding witness control strategies. In particular, we consider discrete-time continuous-domain linear stochastic dynamics with the constraints given as formulas of the Generalized Reactivity(1) (GR(1)) [21] fragment of Linear Temporal Logic (LTL) over linear predicates in the states of the system. The GR(1) fragment offers polynomial computational complexity as compared to the doubly exponential one of general LTL, while being expressive enough to describe most of the usually considered temporal properties [21]. We require the formula to be satisfied almost-surely, i.e., with probability 1. The almost-sure satisfaction is the strongest probability guarantee one can achieve while accounting for the stochasticity of the dynamics.

In our proposed approach, we iteratively construct and refine a discrete abstraction of the system and solve the synthesis problem for the abstract model. The discrete model considered in this work is a turn-based 2-player probabilistic game, also called 2\(/1\)-player game [7]. Every iteration of our algorithm produces a partial solution given as a par-
tion of the state space into three categories. The first is a set of satisfying initial states together with corresponding witness strategies. The second is a set of non-satisfying initial states, i.e., those from which the system cannot be controlled to satisfy the specification with probability 1. Finally, some parts of the state space may remain undecided due to coarse abstraction. As the abstraction gets more precise, more states are being decided with every iteration of the algorithm. The designed solution is partially correct. That means, we guarantee soundness, i.e., almost sure satisfaction of the formula by all controlled trajectories starting in the satisfying initial set and non-existence of a satisfying control strategy for non-satisfying initial states. On the other hand, completeness is only ensured if the algorithm terminates. If a weaker abstraction model, such as 2 player games, was used, there would be no soundness guarantee on the non-satisfying initial states and no completeness guarantees. We provide a practical implementation of the algorithm that ends after a predefined number of iterations.

The main novelty of our work is the abstraction-refinement of a dynamic system using a 2/2-player game. While abstraction-refinement exists for discrete systems such as non-deterministic and probabilistic systems [16, 5, 18, 8], and some classes of hybrid systems [15, 20], to the best of our knowledge, the approach that we present in this paper is the first attempt to construct abstraction-refinement of stochastic systems with continuous state and control spaces in the form of 2/2-player games. The game theoretic solutions are necessary to determine what needs to be refined, and the dynamics of the linear-stochastic systems determine the refinement steps. Thus both game theoretic aspects and the dynamics of the system play a crucial role in the refinement step, see Rem. 1.

This paper is closely related to [29, 13, 19, 1, 28]. Our computation of the abstraction is inspired from [29], which, however, does not consider stochastic dynamics and does not perform refinement. The latter issue is addressed in [13] for non-stochastic dynamics and specifications with finite-time semantics in the form of syntactically co-safe LTL formulas. The exact problem that we formulate in this paper was also considered in [19], but for finite-time specifications in the form of probabilistic Computation Tree Logic (PCTL) formulas and for the particular case when the control space is finite. Also, in [19], the abstraction is constructed in the form of an interval-valued MDP, which is less expressive than the game considered here. An uncontrolled version of the abstraction problem for a stochastic system was considered in [1], where the finite system was in form of a Markov set chain. In [28], the authors consider the problem of controlling uncertain MDPs from LTL specification. When restricted to almost sure satisfaction, uncertain MDPs have the same expressivity as the games considered here. To obtain a control strategy, the authors of [28] use dynamic programming (value iteration), as opposed to games.

The paper is organized as follows. We give some preliminaries in Sec. 2 before we formulate the problem and outline the approach in Sec. 3. The abstraction, game, and refinement algorithms are presented in Sec. 4. A case study is included in Sec. 5. We conclude with final remarks and directions for future work in Sec. 6. Due to space constraints, detailed descriptions of involved technical computations together with additional simulations can be found in [25].

2. NOTATION AND PRELIMINARIES

For a non-empty set $S$, let $S^\omega$, $S^*$, and $S^+$ denote the set of all infinite, finite and non-empty finite sequences of elements of $S$, respectively. For $\sigma \in S^\omega$ and $\rho \in S^*$, we use $|\sigma|$ to denote the length of $\sigma$, and $\sigma(n)$ and $\rho(n)$ to denote the $n$-th element, for $1 \leq n \leq |\sigma|$ and $n \geq 1$, respectively. For two sets $S_1 \subseteq S^\omega$, $S_2 \subseteq S^* \cup S^\omega$, we use $S_1 \cdot S_2 = \{s_1 \cdot s_2 \mid s_1 \in S_1, s_2 \in S_2\}$ to denote their concatenation. Finally, for a finite set $S$, $|S|$ is the cardinality of $S$, $D(S)$ is the set of all probability distributions over $S$ and $\{s \in S \mid d(s) > 0\}$ is the support set of $d \in D(S)$.

2.1 Polytopes

A (convex) polytope $P \subset \mathbb{R}^N$ is defined as the convex hull of a finite set $X = \{x_i\}_{i \in I} \subset \mathbb{R}^N$, i.e.,

$$P = \text{hull}(X) = \{\sum_{i \in I} \lambda_i x_i \mid \forall i : \lambda_i \in [0,1], \sum_{i \in I} \lambda_i = 1\}. \quad (1)$$

We use $V(P)$ to denote the vertices of $P$ that is the minimum set of vectors in $\mathbb{R}^N$ for which $P = \text{hull}(V(P))$. Alternatively, a polytope can be defined as an intersection of a finite number of half-spaces in $\mathbb{R}^N$, i.e.,

$$P = \{x \in \mathbb{R}^N \mid H_P^T x \leq K_P\}, \quad (2)$$

where $H_P$, $K_P$ are matrices of appropriate sizes. Forms in Eq. (1) and (2) are referred to as the V-representation and H-representation of polytope $P$, respectively. A polytope $P \subset \mathbb{R}^N$ is called full-dimensional if it has at least $N + 1$ vertices. In this work, we consider all polytopes to be full-dimensional, i.e., if a polytope is not full-dimensional, we consider it empty.

2.2 Automata and Specifications

**Definition 1 (\(\omega\)-automata).** A deterministic \(\omega\)-automaton with Büchi implication (aka one-pair Streett) acceptance condition is a tuple $A = (Q, \Sigma, \delta, \nu, (E, F))$, where $Q$ is a non-empty finite set of states, $\Sigma$ is a finite alphabet, $\delta : Q \times \Sigma \rightarrow Q$ is a deterministic transition function, $\nu \in Q$ is an initial state, and $(E, F) \subseteq Q \times Q$ defines an acceptance condition.

Given an automaton, every word $w \in \Sigma^\omega$ over the alphabet $\Sigma$ induces a run which is an infinite sequence of states $q_0, q_1, \ldots \in Q^\omega$, such that $q_{i+1} = \delta(q_i, w(i))$ for all $i \geq 0$. Given a run $r$, let $\inf(r)$ denote the set of states that appear infinitely often in $r$. Given a Büchi implication acceptance condition $(E, F)$, a run $r$ is accepting, if $\inf(r) \cap E \neq \emptyset$ implies $\inf(r) \cap F \neq \emptyset$, i.e., if the set $E$ is visited infinitely often, then the set $F$ is visited infinitely often. The Büchi acceptance condition is a special case of Büchi implication acceptance condition where $E = Q$, i.e., we require $F$ to be visited infinitely often. The language of an automaton is the set of words that induce an accepting run.

**Definition 2 (GR(1) formulae).** A GR(1) formula $\varphi$ is a particular type of an LTL formula over alphabet $\Sigma$ of the form

$$\varphi = (\bigwedge_{i=1}^{m} \varphi_i) \implies (\bigwedge_{j=1}^{n} \varphi_j), \quad (3)$$

where each $\varphi_i, \varphi_j$ is an LTL formula that can be represented by a deterministic $\omega$-automaton with Büchi acceptance condition.

The above definition of GR(1) is the extended version of the standard General Reactivity(1) fragment introduced
in [21]. The advantage of using GR(1) instead of full LTL as specification language is that realizability for LTL is 2-EXPTIME-complete [22], whereas for GR(1) it is only cubic in the size of the formula [21]. Given a finite number of deterministic ω-automata with Büchi acceptance conditions, we can construct a deterministic ω-automaton with Büchi acceptance condition that accepts the intersection of the languages of the given automata [2]. Thus a GR(1) formula can be converted to a deterministic ω-automaton with a Büchi implication acceptance condition.

2.3 Games

In this work, we consider the following probabilistic games that generalize Markov decision processes (MDPs).

**Definition 3 (2/1-player Games).** A two-player turn-based probabilistic game, or 2/1-player game, is a tuple $G = (S_1, S_2, Act, \delta)$, where $S_1, S_2$ are disjoint finite sets of states for Player 1 and Player 2, respectively. Act is a finite set of actions, and $\delta: (S_1 \cup S_2) \times Act \to D(S_1 \cup S_2)$, is a probabilistic transition function.

Let $S = S_1 \cup S_2$. A play of a 2/1-player game $G$ is a sequence $s = s^T$ such that for all $n \geq 1$ there exists a $a \in Act$ such that $\delta(s(n), a)(s(n+1)) > 0$. A finite play is a finite prefix of a play of $G$. A Player 1 strategy for $G$ is a function $C_1^G: S^* \to Act$ that determines the Player 1 action to be applied after any finite prefix of a play ending in a 1-player state, and strategies for Player 2 are defined analogously. If there exists an implementation of a strategy that uses finite memory, e.g., a finite-state transducer, the strategy is called finite-memory. If there exists an implementation that uses only one memory element, it is called memoryless. Given a Player 1 and Player 2 strategy, and a starting state, there exists a unique probability measure over sets of plays.

Given a game $G$, an acceptance condition defines the set of accepting plays. We consider GR(1) formulae and Büchi implication over $S$ as accepting conditions for $G$. The almost-sure winning set, denoted as $\text{Almost}^G(\varphi)$ for a GR(1) formula and $\text{Almost}^G((E, F))$ for a Büchi implication condition, is the set of states such that Player 1 has a strategy to ensure the objective with probability 1 irrespective of the strategy of Player 2. Formally, $\text{Almost}^G(\varphi) = \{s \in S \mid 3C_1^G \forall C_2^G \text{ s.t. } \varphi\}$, where $\varphi$ is a formula using the two strategies and starting from state $s$ is 1, and $\text{Almost}^G((E, F))$ is defined similarly. The almost-sure winning set $\text{Almost}^G((E, F))$ for Büchi implication acceptance condition can be solved in quadratic time [4, 7]. In this work, we use more intuitive, cubic time algorithm described in detail in [25]. Moreover, in the states of the set $\text{Almost}^G((E, F))$, there always exist witness strategies, called almost-sure winning strategies, that are memoryless and indeed pure, i.e., not randomized, as defined above. This follows from the fact that the Büchi implication condition can be seen as a special case of a more general parity condition [7]. In Sec. 4, we show how to compute the almost-sure winning set $\text{Almost}^G(\varphi)$ for a GR(1) formula $\varphi$.

In this work, we also consider the following cooperative interpretation of 2/1-player games which is an MDP or so called 1/1-player game.

**Definition 4 (1/1-player Games).** An MDP or 1/1-player game, is a tuple $G = (S, Act, \delta)$, where $S$, Act are non-empty finite sets of states and actions, and $\delta: S \times Act \to D(S)$ is a probabilistic transition function.

Given a 2/1-player game $G$, the 1/1-player interpretation, where the players cooperate, is called $G^\text{coop}$ with $S = S_1 \cup S_2$.

3. Problem Formulation

In this work, we assume we are given a linear stochastic system $T$ defined as

$$T : x_{t+1} = Ax_t + Bu_t + w_t,$$

where $x_t \in X \subset \mathbb{R}^n, w_t \in U \subset \mathbb{R}^m$. $X, U$ are polytopes in the corresponding Euclidean spaces called the state space and control space, respectively, $w_t \in W \subset \mathbb{R}^m$ is the value at time $t$ of a random vector with values in polytope $W$. The random vector has positive density on all values in $W$. Finally, $A$ and $B$ are matrices of appropriate sizes.

The system $T$ evolves in traces. A trace of a linear stochastic system $T$ is an infinite sequence $\rho \in X^\omega$ such that for every $n \geq 1$, we have $\rho(n+1) = A\rho(n) + Bu_n + w_n$ for some $u_n \in U, w_n \in W$. A finite trace $\sigma \in X^+$ of $T$ is then a finite prefix of a trace. A linear stochastic system $T$ can be controlled using control strategies, where a control strategy is a function $C_T: X^+ \to U$.

To formulate specifications over the linear stochastic system $T$, we assume we are given a set $\Pi$ of linear predicates over the state space $X$ of $T$.

$$\Pi = \{\pi_k | \pi_k : x \leq d_k, c_k \in \mathbb{R}^n, d_k \in \mathbb{R}, k \in K\},$$

where $K$ is a finite index set. Every trace of the system generates a word over $2^H$, and every GR(1) specification formulated over $\Pi$ can be interpreted over these words.

**Problem 1.** Given a linear stochastic system $T$ (Eq. (4)), a finite set of linear predicates $\Pi$ (Eq. (5)) and a GR(1) formula $\varphi$ over alphabet $\Pi$, find the set $X_{\text{init}}$ of states $x \in X$ for which there exists a control strategy $C_T$ such that the probability that a trace starting in state $x$ using $C_T$ satisfies $\varphi$ is 1, and find the corresponding strategies for $x \in X_{\text{init}}$.

**Approach overview.** The solution we propose for Problem 1 can be summarized as follows. First, we abstract the linear stochastic system $T$ using a 2/1-player game based on the partition of the state space $X$ given by linear predicates $\Pi$. The game is built only using polytopic operations on the state space and control space. We analyze the game.
Table 1: Definitions of polytopic operators Post (posterior), Pre (predecessor), PreR (robust predecessor), PreP (precise predecessor), Attr (attractor) and AttrR (robust attractor), where $X' \subseteq X, U' \subseteq U$ are polytopes, and $\{X_j\}_{j \in J}$ is a set of polytopes in $X$. The algorithms to compute all the operators are listed in [25].

<table>
<thead>
<tr>
<th>Operator</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Post($X', U'$)</td>
<td>${ x \in \mathbb{R}^N \mid \exists x' \in X', \exists u \in U', \exists w \in W : x = Ax' + Bu + w }$</td>
</tr>
<tr>
<td>Pre($X', U', {X_j}_{j \in J}$)</td>
<td>${ x \in X' \mid \exists u \in U' : \text{Post}(x, u) \cap \bigcup_{j \in J} X_j \text{ is non-empty} }$</td>
</tr>
<tr>
<td>PreR($X', U', {X_j}_{j \in J}$)</td>
<td>${ x \in X' \mid \exists u \in U' : \text{Post}(x, u) \subseteq \bigcup_{j \in J} X_j }$</td>
</tr>
<tr>
<td>PreP($X', U', {X_j}_{j \in J}$)</td>
<td>${ x \in X' \mid \exists u \in U' : \text{Post}(x, u) \subseteq \bigcup_{j \in J} X_j \text{ and } \forall j \in J : \text{Post}(x, u) \cap X_j \text{ is non-empty} }$</td>
</tr>
<tr>
<td>Attr($X', U', {X_j}_{j \in J}$)</td>
<td>${ x \in X' \mid \exists u \in U' : \text{Post}(x, u) \cap \bigcup_{j \in J} X_j \text{ is non-empty} }$</td>
</tr>
<tr>
<td>AttrR($X', U', {X_j}_{j \in J}$)</td>
<td>${ x \in X' \mid \exists u \in U' : \text{Post}(x, u) \cap \bigcup_{j \in J} X_j }$</td>
</tr>
</tbody>
</table>

and identify those partition elements of the state space $X$ that probably belong to the solution set $X_{\text{init}}$, as well as those that do not contain any state from $X_{\text{init}}$. The remaining parts of the state space have the potential to contribute to the set $X_{\text{init}}$ but are not decided yet due to coarse abstraction. In the next step, the partition of state space $X$ is refined using deep analysis of the game. Given the new partition, we build a new game and repeat the analysis. The approach can be graphically represented as shown in Fig. 1.

We prove that the result of every iteration is a partial solution to Problem 1. In other words, the computed set of satisfying initial states as well as the set of non-satisfying initial states are correct. Moreover, they are improved or maintained with every iteration as the abstraction gets more precise. This allows us to efficiently use the proposed algorithm for a fixed number of iterations. Finally, we prove that if the algorithm terminates then the result is indeed the solution to Problem 1.

The main difficulty of the approach is the abstraction-refinement of 2/2-player game. Abstraction-refinement has been considered for discrete systems [18, 16, 5, 8], and also for some classes of hybrid systems [15, 20]. However, in all of these approaches, even if the original system is considered to be probabilistic, the distributions are assumed to be discrete and given, and are not abstracted away during the refinement. The key challenge is the extension of abstraction-refinement approach to continuous stochastic systems, where the transition probabilities in the abstract discrete model need to be abstracted. We show that by exploiting the nature of the considered dynamic systems we can develop an abstraction-refinement approach for our problem, see Rem. 1.

4. SOLUTION

In this section, we describe the proposed solution in detail and present necessary proofs. We start with the abstraction procedure that consists of two steps. The linear stochastic system $T$ is first abstracted using a non-deterministic transition system which is then extended to a 2/2-player game.

**Definition 5 (NTS).** A non-deterministic transition system (NTS) is a tuple $\mathcal{N} = (S, \text{Act}, \delta)$, where $S$ is a non-empty finite set of states, $\text{Act}$ is an empty finite set of actions, and $\delta: S \times \text{Act} \to 2^S$ is a non-deterministic transition function.

**NTS construction.** In order to build an NTS abstraction for $T$, we assume we are given a partition $\{X_i\}_{i \in I}$, or $\{X_i\}$ for short, of the state space $X$. Initially, the partition is given by the set of linear predicates $P$, i.e., it is the partition given by the equivalence relation $\sim_P$ defined as

$$x \sim_P x' \iff \forall k \in K: (c_k^T x \leq d_k \Leftrightarrow c_k^T x' \leq d_k).$$

In the later iterations of the algorithm, the partition is given by the refinement procedure. The construction below builds on the approach from [29].

We use $\mathcal{N}(X_i) = (S_N, \text{Act}_N, \delta_N)$ to denote the NTS corresponding to partition $\{X_i\}$ defined as follows. The states of $\mathcal{N}(X_i)$ are given by the partition of the state space $X$ and the outer set $\mathcal{X}_{\text{out}}$, i.e., $S_N = \{X_i\} \cup \{\mathcal{X}_{\text{out}}\}$. Let $X_i \in \{X_i\} \subset S_N$ be a state of the NTS, a polytope in $X$. We use $\sim$ to denote the equivalence relation on $\mathcal{U}$ such that $u \sim u'$ if for every state $X_i \in \{X_i\} \cup \{X_{\text{out}}\}$, it holds that $\text{Post}(X_i, u) \cap X_i$ is non-empty if and only if $\text{Post}(X_i, u') \cap X_i$ is non-empty. Intuitively, two control inputs are equivalent with respect to $X_i$, if from $X_i$ the system $T$ can transit to the same set of partition elements of $X$ and $\mathcal{X}_{\text{out}}$. The partition $\mathcal{U}/\sim$ is then the set of all actions of the NTS $\mathcal{N}(X_i)$ that are allowed in state $X_i$. We use $U'_{\sim}$ to denote the union of those partition elements from $\mathcal{U}/\sim$ that contain control inputs that lead the system from $X_i$ to polytopes $X_j, j \in J \subseteq I \cup I_{\text{out}}$, i.e.,

$$U'_{\sim} = \{ u \in \mathcal{U} : \forall j \in J : \text{Post}(X_i, u) \cap X_j \text{ is non-empty and } \forall j \notin J : \text{Post}(X_i, u) \cap X_j \text{ is empty} \}. \quad (6)$$

The set $U'_{\sim}$ can be computed using only polytopic computations as described in [25]. For a state $X_i \in \{X_i\} \subset S_N$ and action $U'_0 \subset \text{Act}_N$, we let

$$\delta_N(X_i, U'_0) = \begin{cases} \{X_j \mid j \in J\} & \text{if } i = i', \\ \emptyset & \text{otherwise.} \end{cases}$$

that are used in this section are formally defined in Tab. 1 and their computation is described in detail in [25].

4.1 Abstraction

The abstraction consists of two steps. The linear stochastic system is first abstracted using a non-deterministic transition system which is then extended to a 2/2-player game.

The algorithm for a fixed number of iterations. Finally, we prove that if the algorithm terminates then the result is indeed the solution to Problem 1.

The main difficulty of the approach is the abstraction-refinement of 2/2-player game. Abstraction-refinement has been considered for discrete systems [18, 16, 5, 8], and also for some classes of hybrid systems [15, 20]. However, in all of these approaches, even if the original system is considered to be probabilistic, the distributions are assumed to be discrete and given, and are not abstracted away during the refinement. The key challenge is the extension of abstraction-refinement approach to continuous stochastic systems, where the transition probabilities in the abstract discrete model need to be abstracted. We show that by exploiting the nature of the considered dynamic systems we can develop an abstraction-refinement approach for our problem, see Rem. 1.

4. SOLUTION

In this section, we describe the proposed solution in detail and present necessary proofs. We start with the abstraction procedure that consists of two steps. The linear stochastic system $T$ is first abstracted using a non-deterministic transition system which is then extended to a 2/2-player game. The game analysis section then describes how to identify parts of the solution to Problem 1. The procedure for refinement is presented last. Finally, we prove all properties of the proposed solution.

Let $\mathcal{X}_{\text{out}}$ be the set of all states outside of the state space $X$ that can be reached within one step in system $T$, i.e., $\mathcal{X}_{\text{out}}$ is the set $\text{Post}(X, U) \setminus X$, where $\text{Post}$ is the posterior operator defined in Tab. 1. Note that $\mathcal{X}_{\text{out}}$ is generally not a polytope, but it can be represented as a finite set of polytopes $\{\mathcal{X}_{\text{out}, j}\}_{j \in J_{\text{out}}}$ or $\{\mathcal{X}_{\text{out}}\}$ for short. All polytopic operators
For states $X_{\text{init}} \in \{X_{i}\} \subset S_{N}$, no actions or transitions are defined.

**From NTS to game.** Since the NTS does not capture the probabilistic aspect of the linear stochastic system, we build a 2/2-player game on top of the NTS. Let $X_{i}$ be a polytope within the state space $X$ of $T$, a state of $\mathcal{N}(X_{i})$. When $T$ is in a particular state $x \in X_{i}$ and a control input $u \in \mathcal{U}_{i}$ is to be applied, we can compute the probability distribution over the set $\{X_{j}\}_{j \in J}$ that determines the probability of the next state of $T$ being in $X_{j}, j \in J$, using the distribution of the random vector for uncertainty. The evolution of the polytope distribution that will be used to make a transition. For a in our case we do not need to consider all possible probabil-

**exact probabilities in admissible probability distributions of**

**and in the theory of finite discrete probabilistic models, it is**

**of actions. On the other hand, in Problem 1 we are inter-

**struction implies that Player 2 has a possibly infinite number**

**to which a transition in**

**Player 1 state,**

**and Player 2 determines the exact state within the polytope**

**and thus chooses the probability distribution according**

**to which a transition in $T$ is made. This intuitive game con-

**struction implies that Player 2 has a possibly infinite number**

**of actions. On the other hand, in Problem 1 we are inter-

**ested in satisfying the GR(1) specification with probability 1**

**and in the theory of finite discrete probabilistic models, it is**

**a well-studied phenomenon that in almost-sure analysis, the**

**exact probabilities in admissible probability distributions of**

**the model are not relevant. It is only important to know sup-

**ports of such distributions, see e.g., [2]. That means that**

**in our case we do not need to consider all possible probabil-

**bility distributions as actions for Player 2, but it is enough to**

**consider that Player 2 chooses support for the probability dis-

**tribution that will be used to make a transition. For a**

**polytope $X_{i} \in S_{N}$ and $\mathcal{U}_{i} \in \mathcal{U}_{N}$, we use $\text{Supp}(X_{i}, \mathcal{U}_{i})$ to denote the set of all subsets $J \subset J$ for which there exist $x \in X_{i}, u \in \mathcal{U}_{i}$ such that the next state $x' = Ax + Bu + w$ of $T$ belongs to $X_{j}, j \in J'$ with non-zero probability and with probability zero to $X_{j}, j \notin J'$, i.e.,

$$\text{Supp}(X_{i}, \mathcal{U}_{i}) = \{J' \subset J \mid \text{PrP}(X_{i}, \mathcal{U}_{i}, \{X_{j}\}_{j \in J'})$$

**is non-empty}, (7)

where PrP is the precise predecessor operator from Tab. 1.

**Game construction.** Given the NTS $\mathcal{N}(X_{i})$, the 2/2-player game $\mathcal{G}(X_{i}) = (S_{1}, S_{2}, \mathcal{A}, \delta)$ is defined as follows. Player 1 states $S_{1} = \{X_{i}\} \times \{X_{\text{init}}\}$ are the states $S_{N}$ of the NTS and Player 1 actions are the actions $\mathcal{A}_{N}$ of $\mathcal{N}(X_{i})$. Player 2 states are given by the choice of an action in a Player 1 state, i.e., $S_{2} = \{X_{i}\} \times \{\mathcal{U}_{i}\}$. The Player 2 actions available in a state $(X_{i}, \mathcal{U}_{i})$ are the elements of the set $\text{Supp}(X_{i}, \mathcal{U}_{i})$ defined in Eq. (7). For Player 1, the transition probability function $\delta$ defines non-zero probability transitions only for triples of the form $(X_{i}, \mathcal{U}_{i}), (X_{i}, \mathcal{U}_{i})$ and for such it holds $\delta((X_{i}, \mathcal{U}_{i}),(X_{i}, \mathcal{U}_{i})) = 1$. For Player 2, the function $\delta$ defines the following transitions:

$$\delta_{\mathcal{U}}((X_{i}, \mathcal{U}_{i}), J') = \left\{ \begin{array}{ll} \frac{1}{|J'|} & \text{if } J' \in \text{Supp}(X_{i}, \mathcal{U}_{i}) \\
0 & \text{otherwise} \end{array} \right. \quad \text{if } j \in J'$$

The definition reflects the fact that once Player 2 chooses the support, the exact transition probabilities are irrelevant and without loss of generality, we can consider them to be uniform.

**Example 1.** (Illustrative example, part I) Let $T$ be a linear stochastic system of the form given in Eq. (4), where

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

**Figure 2: Partition of state space $X$ of system $T$ in Ex. 1 given by linear predicates $\Pi$. Polytopes $X_{1}, \ldots, X_{6}$ form the set $X_{\text{init}}$.**

**Figure 3: Part of the transition function of the game $\mathcal{G}(X_{i})$ constructed in Ex. 1. The state space is $X = \{x \in \mathbb{R}^{2} \mid 0 \leq x_{1} \leq 4.0, 0 \leq x_{2} \leq 2\}$, the control space is $\mathcal{U} = \{u \in \mathbb{R}^{2} \mid -1 \leq u(1), u(2) \leq 1\}$, and the random vector takes values in polytope $W = \{w \in \mathbb{R}^{2} \mid -0.1 \leq w(1), w(2) \leq 0.1\}$. Let $\Pi$ contain a single linear predicate $\pi_{1}: x(1) \leq 2$. In Fig. 2, polytopes $X_{1}$ and $X_{2}$ form the partition of $X$ given by $\Pi$, and polytopes $X_{3}, X_{4}, X_{5}, X_{6}$ form the rest of the one step reachable set of system $T$, i.e., $X_{\text{init}}$. The game $\mathcal{G}(X_{i})$ given by this partition has 6 states and 18 actions. In Fig. 3, we visualize part of the transition function as follows. In Player 1 state $X_{1}$, if Player 1 chooses, e.g., action $\mathcal{U}_{1}^{1,2,5}$ that leads from $X_{1}$ to polytopes $X_{1}, X_{2}, X_{5}$, the game is in Player 2 state $(X_{1}, \mathcal{U}_{1}^{1,2,5})$ with probability 1. Actions of Player 2 are the available supports of the action over the set $\{X_{1}, X_{2}, X_{5}\}$, which are in this case all non-empty subsets. If Player 2 chooses, e.g., support $\{X_{1}, X_{2}\}$, the game is in Player 1 state $X_{1}$ or $X_{2}$ with equal probability 0.5.

The following proposition proves that the game $\mathcal{G}(X_{i})$ simulates the linear stochastic system $T$.

**Proposition 1.** Let $\rho$ be a trace of the linear stochastic system $T$. Then there exists a play $\gamma$ of the game $\mathcal{G}(X_{i})$ such that $\rho(n) = g(2n - 1)$ for every $n \geq 1$.

**Proof.** The play $\gamma$ is defined as follows. The states $g(2n - 1) = X_{i}$ such that $\rho(n) = X_{i}$. The states $g(2n) = (X_{i}, \mathcal{U}_{i})$ such that there exist $u \in \mathcal{U}_{i}, w \in W$ for which $\rho(n + 1) = Ap(n) + Bu + w$. On the other hand, since $\mathcal{G}(X_{i})$ is only an abstraction of the system $T$, it may contain plays that do not correspond to any trace of the system.

**4.2 Game analysis**

Let $\mathcal{G}(X_{i})$ be the 2/2-player game constructed for the linear stochastic system $T$ and partition $\{X_{i}\}$ of its state space using the procedure from Sec. 4.1. In this section, we identify partition elements from $\{X_{i}\}$ which are part of the solution set of initial states $X_{\text{init}}$ as well as those that do not contain any satisfying initial states from $X$. **HSCC'15, April 14-16, 2015, Seattle, Washington**
Computing satisfying states. First, we compute the almost-sure winning set \( S_{ps} \) in game \( G_{(X_1)} \) with respect to the GR(1) formula \( \varphi \) from Problem 1, i.e.,
\[
S_{ps} = \text{Almost}^P_{(X_1)}(\varphi).
\]
We proceed as follows. Let \( A = (Q, 2^Q, \delta, q_0, (E, F)) \) be a deterministic \( \omega \)-automaton with Büchi implication acceptance condition for the GR(1) formula \( \varphi \) constructed as described in Sec. 2.2. We consider the 2/\( \omega \)-player game \( P_{(X_1)} = (S^P, S^P, \text{Act}, \delta^P) \) that is the synchronous product of \( G_{(X_1)} \) and \( A \), i.e., \( S^P = S_1 \times Q \) and for every \((X_1, q) \in S^P_1 \) and \( U^i \in \text{Act} \) we have
\[
\delta^P((X_1, q), U^i) (X_1', q') = \begin{cases} \delta(X_1, U^i) ((X_1', U^i)) & \text{if } \delta^A(q, \Pi(X_1')) = q', \\ 0 & \text{otherwise}, \end{cases}
\]
where \( \Pi(X_1) \) is the set of all linear predicates from \( \Pi \) that are true on polytope \( X_1 \), and similarly, for all \((X_1, U^i), q) \in S^P_2 \) and \( J^i \in \text{Act} \) we have
\[
\delta^P((X_1, U^i), J^i) (X_1', q') = \begin{cases} \delta(X_1, U^i, J^i) (X_1) & \text{if } q = q', \\ 0 & \text{otherwise}. \end{cases}
\]
When constructing the product game, we only consider those states from \( S_1 \times Q \) and \( S_2 \times Q \) that are reachable from some \((X_1, q_0) \in Q_0 \), where \( Q_0 \) is the initial state of the automaton \( A \). Finally, we consider Büchi implication acceptance condition \( (E^P, F^P) \), where \( E^P = (S^P \cup S^P_2) \times E \) and \( F^P = (S^P \cup S^P_2) \times F \).

**Proposition 2.** The set \( S_{ps} \) defined in Eq. (8) consists of all \( X_i \in S_1 \) for which \((X_i, q_0) \in S^P_{ps} \), where the set
\[
S^P_{ps} = \text{Almost}^P_{(X_1)}((E^P, F^P))
\]
can be computed using algorithm in [25].

**Proof.** Follows directly from the construction of the game \( P_{(X_1)} \) above and the results of [6].

The next proposition proves that the polytopes from \( S_{ps} \) are part of the solution to Problem 1.

**Proposition 3.** For every \( X_i \in S_{ps} \), there exists a finite-memory strategy \( C_T \) for \( T \) such that every trace of \( T \) under strategy \( C_T \) that starts in any \( x \in X_i \) satisfies \( \varphi \) with probability 1.

**Proof.** Let \( X_i \in S_{ps} \) and let \( C_{G_{(X_1)}}(X_1) \) be a finite-memory almost-sure winning strategy for Player 1 from state \( X_i \) in game \( G_{(X_1)} \), see Sec. 2.3. Let \( C_T \) be a strategy for \( T \) defined as follows. For a finite trace \( \sigma_T \), let \( C_T(\sigma_T) = u \), where \( u \in C_{G_{(X_1)}}(\sigma_{G_{(X_1)}}(X_1)) \), where \( \sigma_{G_{(X_1)}}(X_1) \) is fix point such that \( \sigma_T(n) \in \sigma_{G_{(X_1)}}(2n) \) for every \( 1 \leq n \leq |\sigma_T| \). Since \( C_{G_{(X_1)}}(X_1) \) for game \( G_{(X_1)} \) is almost-sure winning from \( X_i \) w.r.t. \( \varphi \), i.e., every play that starts in \( X_i \) almost-surely satisfies \( \varphi \), the analogous property holds for \( C_T \) and traces in \( T \).

Computing non-satisfying states. Next, we consider the set \( S_{no} \) of Player 1 states in game \( G_{(X_1)} \) defined as follows:
\[
S_{no} = S_1 \setminus \text{Almost}^P_{(X_1)}((E^P, F^P)).
\]
Intuitively, \( S_{no} \) is the set of states, where even if Player 2 cooperates with Player 1, \( \varphi \) can still not be satisfied with probability 1.

**Proposition 4.** The set \( S_{no} \) defined in Eq. (10) consists of all \( X_i \in S_1 \) for which \((X_i, q_0) \in S^P_{no} \), where
\[
S^P_{no} = S^P \setminus \text{Almost}^P_{(X_1)}((E^P, F^P)).
\]

**Proof.** Follows directly from the construction of the product game \( P_{(X_1)} \).

We prove that no state \( x \in X_i \) for \( X_i \in S_{no} \) is part of the solution to Problem 1.

**Proposition 5.** For every \( X_i \in S_{no} \) and \( x \in X_i \), there does not exist a strategy \( C_T \) for \( T \) such that every trace of \( T \) under \( C_T \) starting in \( x \) satisfies \( \varphi \) with probability 1.

**Proof.** Intuitively, from the construction of the game \( G_{(X_1)} \) in Sec. 4.1, Player 2 represents the unknown precise state of the system \( T \) within in the abstraction, i.e., he makes the choice of a state inside polytope \( X_i \) at each step. Therefore, if \( \varphi \) cannot be almost-surely satisfied from \( X_i \) in the game even if the two players cooperate, in \( T \) it translates to the fact that \( \varphi \) cannot be almost-surely satisfied from any \( x \in X_i \) even if we consider strategies that can moreover change inside each \( X_i \) arbitrarily at any moment.

Undecided states. Finally, consider the set
\[
S_T = S_1 \setminus (S_{ps} \cup S_{no}).
\]
These are the polytopes within the state space of \( T \) that have not been decided as satisfying or non-satisfying due to coarse abstraction. Alternatively, from Prop. 2 and 4, and Eq. (12), we can define the set \( S_T \) as the set of all \( X_i \in S_1 \), for which \((X_i, q_0) \in S^P_{no} \), where
\[
S^P_{no} = S^P \setminus (S^P_{ps} \cup S^P_{no}).
\]

**Proposition 6.** For every \( X_i \in S_T \) it holds that the product game \( P_{(X_1)} \) can be won cooperatively starting from the Player 1 state \((X_i, q_0) \). Analogously, for every \((X, q) \in S^P_T \) it holds that the product game \( P_{(X_1)} \) can be won cooperatively starting from \((X, q) \).

**Proof.** The proposition follows directly from Eq. (12) and (13), and Prop. 2 and 4.

Example 2. (Illustrative example, part II) Recall the linear stochastic system \( T \) from Ex. 1 and consider GR(1) formula \( F \Rightarrow \pi_1 \) over the set \( P \) that requires to eventually reach a state \( x \in X \) such that \( x(1) \geq 2 \). The deterministic \( \omega \)-automaton for the formula has only two states, \( q_0 \) and \( q_1 \). The automaton remains in the initial state \( q_0 \) until polytope \( X_2 \) is visited in \( T \). Then it transits to state \( q_1 \) and remains there forever. The Büchi implication condition \( (E, F) \) is \( E = \{q_0 \}, F = \{q_1 \} \). The solution of the game \( G_{(X_1)} \) constructed in Ex. 1 with respect to the above formula is depicted in Fig. 4.

If the set \( S_{no} \) contains all Player 1 states of the game \( G_{(X_1)} \), the GR(1) formula \( \varphi \) cannot be satisfied in the system \( T \) and our algorithm terminates. If set \( S_T \) is empty, the algorithm terminates and returns the union of all polytopes from \( S_{ps} \) as the solution to Problem 1. The corresponding satisfying strategies are synthesized as described in the proof of Prop. 3. Otherwise, we continue the algorithm by computing a refined partition of the state space \( X \) as described in the next section.
Refinement is a heuristic that constructs a new partition of $\mathcal{X}$, a subpartition of $\{X_i\}$, that is used in the next iteration of the overall algorithm. We design two refinement procedures, called positive and negative, that aim to enlarge the combined volume of polytopes in the set $S_{\text{pos}}$ and $S_{\text{neg}}$, respectively, or equivalently, to reduce the combined volume of polytopes in the set $S_i$. Based on Prop. 2 and 4, both procedures are formulated over the product game $P(x_i)$ and reach their respective goals through refining polytopes $X_i$ for which $(X_i, q) \in S_i^P$ for some $q \in Q$.

In this section, we use $J_{\text{pos}}^i$ to denote the set of all indices $i \in I$ for which $(X_i, q) \in S_{\text{pos}}^i$ and $J_{\text{neg}}^i, J_{\text{neg}}^{i,j}$ are defined analogously. In the two refinement procedures, every polytope $X_i$ can be partitioned into a set of polytopes in iterative manner, as $(X_i, q) \in S_i^P$ can hold for multiple $q \in Q$. Therefore, given a partition of the $X_i$, the refinement of $X_i$ according to a polytope $B$ refers to the partition of $X_i$ that contains all intersections and differences of elements of the original partition of $X_i$ and polytope $B$.

**Positive refinement.** In the positive refinement, we explore the following property of states in $S_i^P$. In Prop. 6, we proved that the product game $P(x_i)$ can be won cooperatively from every $(X_i, q) \in S_i^P$. It follows that there exists a Player 1 action $U_i^q$ and Player 2 action $J^q$ such that after their application in $(X_i, q)$, the game is not in a losing state with probability 1. We can graphically represent this property in the following way:

$$
\begin{align*}
(\mathcal{X}, q) \xrightarrow{U_i^q} (\mathcal{X}', q') & \quad \left\{ \begin{array}{l}
(X_j, q') \quad (J_{\text{pos}}^i, q') \\
(X_j, q') \quad (J_{\text{neg}}^i, q') \\
(X_j, q') \quad (J_{\text{neg}}^{i,j}, q') \\
\end{array} \right.
\end{align*}
$$

where an arrow $a \rightarrow b$ represents the uniform probability distribution $\delta_P(a, b)$, and $\{j_1, \ldots, j_n\} = J^q \subseteq J_{\text{pos}}^i \cup J_{\text{neg}}^i$. Note that from the construction of the product game $P(x_i)$ in Sec. 2.3 it follows that $q'$ is given uniquely over all actions $U_i^q$. The following design ensures that every polytope $X_i$ is refined at least once for every its appearance $(X_i, q) \in S_i^P$, $q \in Q$.

Let $(X_i, q) \in S_i^P$. The positive refinement first refines $X_i$ according to the robust predecessor

$$
\text{PreR}(X_i, U_i, \{X_j\}_{j \in J_{\text{pos}}^i}).
$$

That means, we find all states $x \in X_i$ for which there exists any control input under which the system $T$ evolves from $x$ to a state $x' \in X_j, j \in J_{\text{pos}}^i$. Next, the positive refinement considers three cases. First, assume that from $(X_i, q)$, the two players can cooperatively reach a winning state of the product game in two steps with probability 1, and let $U_i^q$ and $J^q$ be Player 1 and Player 2 actions, respectively, that accomplish that, i.e., in Eq. (14), $\{j_1, \ldots, j_n\} = J^q \subseteq J_{\text{pos}}^i$. For every such $U_i^q, J^q$, we find an arbitrary partition $\{U_q \}_{q \in Q}$ of the polytope $U_i^{q'}$ and we partition $X_i$ according to the robust attractors

$$
\text{AttrR}(X_i, U_q, \{X_j\}_{j \in J_{\text{pos}}^i}).
$$

Intuitively, the above set contains all $x \in X_i$ such that under any control input $u \in U_q$, $T$ evolves from $x$ to a state $x' \in X_j, j \in J_{\text{pos}}^i$. Note that the robust attractor sets partition the robust predecessor set from Eq. (15), as every state $x$ that belongs to one of the robust attractor set must lie in the robust predecessor set as well. In the next iteration of the overall algorithm, the partition elements given by the robust attractor sets will belong to the set $S_{\text{pos}}^i$. In the second case, assume that the two players can reach a winning state of the product game cooperatively in two steps, but only with probability $0 < p < 1$, while the probability of reaching a losing state is 0. Let $U_i^q, J^q$ be Player 1 and Player 2 actions, respectively, that maximize $p$, i.e., in Eq. (14), there exists $m < n$ such that $\{j_1, \ldots, j_m\} = J^q \cap J_{\text{pos}}^i$, $J^q \cup J_{\text{neg}}^i$, and $p = \frac{m}{n}$ is maximal. Similarly as in the first case, we refine the polytope $X_i$ according to the robust attractor sets as in Eq. (16), but we compute the sets with respect to the set of indices $J_{\text{pos}}^i \cup \{j_{m+1}, \ldots, j_n\}$. Finally, assume that $(X_i, q)$ does not belong to any of the above two categories. As argued at the beginning of this section, there still exist Player 1 and Player 2 actions $U_i^q, J^q$, respectively, such that in Eq. (14), $\{j_1, \ldots, j_n\} = J^q$. Again, we refine the polytope $X_i$ according to the robust attractor sets as in Eq. (16), where the sets are computed with respect to the set of indices $J_{\text{pos}}^i$.

**Example 3.** (ILLUSTRATIVE EXAMPLE, PART III) We demonstrate a part of the the positive refinement for the game in Ex. 2. Consider polytope $X_1 \in S_i$. It follows from the form of the $\omega$-automaton in Ex. 2 that $X_1$ appears in $S_i^P$ only in pair with $q_0$, i.e., $(X_1, q_0) \in S_i^P$. Note that for state $(X_1, q_0)$, every successor state is of the form $((X_1, U_i^q), q_0)$, i.e., $q = q_0$. First, polytope $X_1$ is refined with respect to the robust predecessor

$$
\text{PreR}(X_1, U_i, \{X_2\}),
$$

since $J_{\text{pos}}^1 = \{X_2\}$ because $(X_2, q_0) \in S_{\text{pos}}^1$ is a winning state of the product game. The robust predecessor set is depicted in Fig. 5 in cyan. Next, we decide which of the three cases described in the positive refinement procedure above applies to state $(X_1, q_0)$. Consider for example Player 1 action $U_i^{1(1,2,5)}$ and Player 2 action (2), as shown in Fig. 3. It holds that

$$
(X_1, q_0) \xrightarrow{U_i^{1(1,2,5)}} ((X_1, U_i^{1(1,2,5)}), q_0) \xrightarrow{2} (X_2, q_0),
$$

and $(X_2, q_0) \in S_{\text{pos}}^2$ is a winning state of the product game. Therefore, the state $(X_1, q_0)$ is of the first type. To further refine polytope $X_1$, we first partition the polytope

$$
U_i^{1(1,2,5)} = \{u \in U \mid 0.1 \leq u(1), u(2) \leq 1\},
$$

e.g., into 4 parts as shown in Fig. 5 on the right. The robust attractor

$$
\text{AttrR}(X_1, U_3, \{X_2\})
$$

$\text{HSCC'15, April 14-16, 2015, Seattle, Washington}$
We consider all Player 1 states \((X_i, q) \in S^P\) such that if Player 2 does not cooperate, but rather plays against Player 1, the game is lost with non-zero probability. In other words, for every Player 1 \(U'_i\), there exists Player 2 action \(J'\) such that in Eq. (14), there exists an index \(j \in J'\) such that \((X_j, q') \in S^P_\text{no}\). In this case, we refine polytope \(X'_i\) according to the attractor set

\[\text{Attr}(X_i, U_i, \{X_j\}_{j \in J'_\text{no}})\]

Intuitively, the attractor set contains all states \(x \in X_i\) such that by applying any control \(u \in U_i\), system \(T\) evolves from \(x\) to a state in \(X_j\) for some \(j \in J'_\text{no}\) with non-zero probability. In the next iteration of the algorithm, the partition elements given by the attractor set will belong to the set \(S^P_\text{no}\).

**Remark 1.** We remark that both game theoretic aspects as well as the linear stochastic dynamics play an important role in the refinement step. The game theoretic results compute the undecided states, and thereby determine what parts of the state space need to be refined and which actions need to be considered in the refinement. The linear stochastic dynamics allow us to perform the refinement itself using polytopic operators.

### 4.4 Correctness and complexity

We prove that the algorithm presented in this section provides a partially correct solution to Problem 1.

For \(n \in \mathbb{N}\), let \(S^P_{\text{yes}}, S^P_{\text{no}}\) be the sets from Eq. (8) and (10), respectively, computed in the \(n\)-th iteration of the algorithm presented above. We use \(X^P_{\text{yes}}, X^P_{\text{no}} \subseteq X\) to denote the union of polytopes from \(S^P_{\text{yes}}\) and \(S^P_{\text{no}}\), respectively.

**Theorem 1.** (Progress) For every \(n \in \mathbb{N}\), it holds that

\[X^P_{\text{yes}} \subseteq X^P_{\text{yes}}^{n+1}\]

and

\[X^P_{\text{no}} \subseteq X^P_{\text{no}}^{n+1}\]

**Proof.** Follows from Prop. 2 and 4, and the fact that the partition of the state space \(X\) used in \(n+1\)-th iteration is a subpartition of the one used in \(n\)-th iteration.

**Theorem 2.** (Soundness) For every \(n \in \mathbb{N}\), it holds that

\[X^P_{\text{yes}} \subseteq X_{\text{init}}\]

and

\[X^P_{\text{no}} \subseteq X\backslash X_{\text{init}}\]

**Proof.** Follows directly from Prop. 3 and 5.

**Theorem 3.** (Partial Correctness) If the algorithm from Sec. 4 terminates, after \(n\)-th iteration, then \(X_{\text{init}} = X^P_{\text{yes}}\) is the solution of Problem 1 and the corresponding winning strategies for every \(x \in X_{\text{init}}\) are given by the winning strategies in the 2/1-player game from the last iteration.

**Proof.** Follows directly from the condition of the algorithm termination and from Th. 1 and 2.

It is important to note that if instead of a 2/1-player game a weaker abstraction model such as a 2 player game, i.e., the NTS from Sec. 4.1, was used, our approach would not be sound. Namely, some states of \(X\) might be wrongfully identified as non-satisfying initial states based on behavior that has zero probability in the original stochastic system. In such a case, even after termination, the resulting set would only be a subset of \(X_{\text{init}}\). Therefore, the approach with 2-player games is not complete. The 2/1-player game is needed to account for both the non-determinism introduced by the abstraction and for the stochasticity of the system to be able to recognize (non-satisfying) behavior of zero probability.

Note that there exist linear stochastic systems for which our algorithm does not terminate, i.e., there does not exist a finite partition of the systems' state space over which Problem 1 can be solved for a given GR(1) formula.

**Example 4.** (Non-termination) Let \(T\) be a linear stochastic system of the form given in Eq. (4), where

\[A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\]

state space \(X = \{x \in \mathbb{R}^2 \mid 0 \leq x_1, x_2 \leq 3\}\), control space \(U = \{u \in \mathbb{R}^2 \mid -1.5 \leq u_1, u_2 \leq 1.5\}\) and the random vector takes values in polytope \(W = \{w \in \mathbb{R}^2 \mid -0.5 \leq w_1, w_2 \leq 0.5\}\). Let \(I\) contain four linear predicates that partition the state space into a grid of three by three equally sized square polytopes. Assume that the aim is to eventually reach the polytope \(X_j\), where \(1 \leq x_1, x_2 \leq 2\). In this case, the maximal set \(X_{\text{init}}\) of states from which \(X_j\) can be reached with probability 1 is the whole state space \(X\), as for any \(x \in X\), \(x\) exists exactly one control input \(u = (1.5,1.5) - x \in U\) that leads the system \(T\) from \(x\) to a state in \(X_j\) with probability 1. Since the control input is different for every \(x \in X\), there does not exist any finite state space partition, which could be used to solve Problem 1.

**Complexity analysis.** Finally, let us analyze the computational complexity of the designed algorithm. In the abstraction part, the 2/1-player game \(G(X_j)\) requires to first compute the set of actions for every state \(X_j, i \in I\), in time in \(O(2^{|I|})\) using algorithm in [25]. For every action \(U'_i\), the set of valid supports \(J' \subseteq J\) is then computed in time in \(O(2^{|J|})\), see [25]. Overall, the abstraction runs in time in \(O(2^{|J|})\). The game is then analyzed using the algorithm described in detail in [25] in time in \(O(|J|^5)\). Finally, the refinement procedure iteratively refines every polytope \(X'_i\) at most \(|Q| \times |\{U'_i\}|\) times, where \(|\{U'_i\}|\) denotes the set of all actions of \(X_j\). For every \(q \in Q\) such that \((X_j, q) \in S^P_i\), \(X_i\) is first refined using the robust predecessor operator in time exponential in \(|J^P_{\text{yes}}|\). Then \(X_i\) is refined \(|Y|\) times using the robust attractor operator in polynomial time. Negative refinement is performed again for every \(q \in Q\) such that
Algorithm 1: Computing the set $\mathcal{X}_{\text{init}} \subseteq \mathcal{X}$ of states from which a set of polytopes $\{\mathcal{X}_j\}_{j \in J}$ in $\mathcal{X}$ can be reached with probability 1.

**Input:** linear stochastic system $\mathcal{T}$, polytopes $\{\mathcal{X}_j\}_{j \in J}$

$\mathcal{X}_{\text{init}} = \{\mathcal{X}_j\}_{j \in J}$

while $\mathcal{X}_{\text{init}}$ is not a fixed point do

$\mathcal{X}_{\text{init}} \leftarrow \operatorname{Pre}(\mathcal{X}, \mathcal{U}, \mathcal{X}_{\text{init}})$

end while

$\mathcal{X}_{\text{init}} \leftarrow X_{\mathcal{X}_{\text{init}}}$

return: $\mathcal{X}_{\text{init}}$

(\mathcal{X}_i, q) \in S^T$, using the attractor operator in polynomial time. Overall, the refinement runs in time in $O(Q \cdot 2^H)$.

As the game construction is the most expensive part of the overall algorithm, the refinement procedure is designed in a way that extends both sets $S_{\text{pre}}, S_{\text{po}}$ as much as possible and thus speed up convergence and minimize the number of iterations of the overall algorithm.

5. CASE STUDY

We demonstrate the designed framework on a discrete-time double integrator dynamics with uncertainties. Let $\mathcal{T}$ be a linear stochastic system of the form given in Eq. (4), where

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 0.5 \\ 1 \end{pmatrix}. \tag{17}$$

The state space is $\mathcal{X} = \{x \in \mathbb{R}^2 \mid -5 \leq x(1) \leq 5, -3 \leq x(2) \leq 3\}$ and the control space is $\mathcal{U} = \{u \in \mathbb{R} \mid -1 \leq u \leq 1\}$. The random vector, or uncertainty, takes values within polytope $\mathcal{W} = \{w \in \mathbb{R}^2 \mid -0.1 \leq w(1), w(2) \leq 0.1\}$. The set $\Pi$ consists of 4 linear predicates $\pi_1 : x(1) \leq -1, \pi_2 : x(1) \leq 1, \pi_3 : x(2) \leq -1, \pi_4 : x(2) \leq 1$. Consider GR(1) formula

$$\mathcal{F}(\neg \pi_1 \land \pi_2 \land \neg \pi_3 \land \pi_4)$$

that requires the system to eventually reach a state, where both variables of the system have values in interval $(-1, 1)$.

As we consider a reachability property, we can compare our approach to the algorithm shown in Alg. 1 that is an extension of the reachability algorithm for Markov decision processes [2] to linear stochastic systems. Intuitively, the algorithm finds the set $\mathcal{X}_{\text{init}}$ using two fixed-point computations. The first one computes the set of all states that can reach the given target polytopes with non-zero probability. As a result, the remaining states of the state space $\mathcal{X}$ have zero probability of reaching the target polytopes. The second fixed-point computation finds the attractor of this set, i.e., all states that have non-zero probability, under each control input from $\mathcal{U}$, of ever transiting to a state from which the target polytopes cannot be reached. Finally, the complement of the attractor is the desired set $\mathcal{X}_{\text{init}}$.

Note that Alg. 1 operates directly on the linear stochastic system. It performs polytopic operations only, and it involves neither refinement nor building a product with an automaton. Therefore, it performs considerably faster than the abstraction-refinement algorithm from Sec. 4, as shown in Tab. 2. However, it has two serious drawbacks.

Firstly, Alg. 1 computes the set of satisfying initial states of the system, but no satisfying strategy. In extreme cases, every state may use a different control input in order to reach polytopes computed during the fixed-point computations, as in Ex. 4. In order to extract a finite satisfying strategy (if there is one), these polytopes have to be further partitioned so that a fixed input can be used in all states of the new polytope. This partitioning is exactly the refinement procedure that our method performs when applied to reachability. Note that simpler refinement methods such as constructing only the NTS $\mathcal{X}(\pi_1)$, which is the first step of the abstraction in Sec. 4.1, are not sufficient, see [25]. The whole 2/2-player game abstraction presented in Sec. 4.1 is necessary for ensuring the correctness of the strategy.

Secondly, Alg. 1 cannot be used for more complex properties than reachability. For more complex formulas, the product of the game with the automaton for the formula needs to be considered, since a winning strategy may require memory and pure polytopic methods can only provide memoryless strategies. In contrast, our abstraction-refinement approach designed in Sec. 4 works for general GR(1) properties. Moreover, it could easily be extended to the whole LTL at the cost of a higher complexity.

We implemented both algorithms in Matlab, on a dual-core Intel i7 processor with 8 GB of RAM. The results are summarized in Fig. 6 and Tab. 2. For Alg. 1, the set $\mathcal{X}_{\text{init}}$ was computed fast but it is a single polytope not providing any information about the satisfying strategies. For the abstraction-refinement algorithm, we computed the initial game and all three iterations. Unlike for Alg. 1, in every iteration, satisfying strategies for states in the partial solution are built as described in the proof of Prop. 3.

6. CONCLUSION AND FUTURE WORK

In this work, we considered the problem of computing the set of initial states of a linear stochastic system such that there exists a control strategy to ensure a GR(1) specification over states of the system. The solution is based on iterative abstraction-refinement using a 2/2-player game. Every iteration of the algorithm provides a partial solution given as a set of satisfying initial states with the satisfying strategies, and a set of non-satisfying initial states.

While the algorithm guarantees progress and soundness in every iteration, it’s complexity calls for more efficient implementation. The analyzed case study with a reachability property indicates that the current design would be too complex to deal with more complex properties such as persistent surveillance. In our future work, we aim to design efficient heuristic refinements that minimize the overall computation time for both reachability and general GR(1).

Acknowledgement We thank Ebru Aydin Gol for useful...
and the next three iterations, where polytopes from sets $S_{\text{pre}}, S_{1}, S_{\text{safe}}$ are shown in green, light blue and white, respectively.

**7. REFERENCES**


