Symbolic Control for Stochastic Systems via Finite Parity Games

Rupak Majumdar, Kaushik Mallik, Anne-Kathrin Schmuck, and Sadegh Soudjani

Abstract

We consider the problem of computing the maximal probability of satisfying an ω-regular specification for stochastic, continuous-state, nonlinear systems evolving in discrete time. The problem reduces, after automata-theoretic constructions, to finding the maximal probability of satisfying a parity condition on a (possibly hybrid) state space. While characterizing the exact satisfaction probability is open, we show that a lower bound on this probability can be obtained by **(I)** computing an under-approximation of the *qualitative winning region*, i.e., states from which the parity condition can be enforced almost surely, and **(II)** computing the maximal probability of reaching this qualitative winning region.

The heart of our approach is a technique to *symbolically* compute the under-approximation of the qualitative winning region in step **(I)** via a *finite-state abstraction* of the original system as a $2\frac{1}{2}$*-player parity game*. Our abstraction procedure uses only the support of the probabilistic evolution; it does not use precise numerical transition probabilities. We prove that the winning set in the abstract $2\frac{1}{2}$-player game induces an under-approximation of the qualitative winning region in the original synthesis problem, along with a policy to solve it. By combining these contributions with (a) existing symbolic fixpoint algorithms to solve $2\frac{1}{2}$-player games and (b) existing techniques for reachability policy synthesis in stochastic nonlinear systems, we get an *abstraction-based symbolic algorithm* for finding a lower bound on the maximal satisfaction probability.

We have implemented step **(I)** of our approach and evaluated it on the nonlinear model of a perturbed bistable switch from the literature. We show empirically that the lower bound on the winning region computed by our approach is precise, by comparing against an over-approximation of the qualitative winning region. Moreover, our symbolic implementation outperforms a recently proposed tool for solving this problem by a large margin. In fact, in many cases the existing tool crashed after consuming too much memory on a standard laptop, whereas our tool consumed small amount of memory and produced results within reasonable amount of time.

Index Terms

Abstraction-based policy synthesis, Approximate model checking, Controller synthesis, Discrete-time stochastic systems, Finite games, Formal specifications, Symbolic control

I. Introduction

Controlled Markov processes (CMPs) over continuous state spaces and evolving in discrete time form a general model for temporal decision making under stochastic uncertainty. In recent years, the problem of finding or approximating optimal policies in CMPs for specifications given in temporal logics or automata has received a lot of attention. While there is a steady progression towards more expressive models and properties [11], [12], [17], [30], [37], [41], a satisfactory solution that can handle *nonlinear* models for general *ω-regular specifications* in a *symbolic* way is still open. In this paper, we make progress toward a solution to this general problem.

For *finite-state* Markov decision processes (MDP), one can find optimal policies for ω-regular specifications by decomposing the problem into two parts [2], [3], [8], [9]. **(I)** Using graph-theoretic techniques that ignore the actual transition probabilities, one can find the set of states that ensures the satisfaction of the specification almost surely. Further, for any state in this *almost sure winning region*, an optimal policy for *almost sure* satisfaction of the specification can be derived. **(II)** One finds an optimal policy to *reach* the almost sure winning region using linear programming or traditional dynamic programming approaches. Combining both policies returns an optimal policy for the overall synthesis problem.

Unfortunately, this two-step solution approach does not carry over to optimal policy synthesis for *all* ω-regular specifications given a *continuous-state* CMP. First, we do not have characterizations of optimal policies for almost sure satisfaction in this case—such as whether randomization or memory is necessary. Second, in contrast to finite-sate MDPs, it is possible that the almost sure winning region of a CMP is empty, even if there is a policy that satisfies the specification with positive probability [30].

However, as we show in this paper, the same decomposition can be used to compute an *under-approximation* for the optimal policy instead: that is, the resulting policy gives a lower bound on the probability of satisfying a given ω-regular specification from every state. While existing techniques [17], [35], [41] can be used in step **(II)** to compute the reachability probability with any given precision, we provide a new technique to under-approximate the set of states of a CMP that almost surely satisfies

R. Majumdar, K. Mallik, and A.-K. Schmuck are with the Max Planck Institute for Software Systems, Kaiserslautern, 67663, Germany (e-mails: {rupak,kmallik,akschmuck}@mpi-sws.org).

S. Soudjani is with the School of Computing, Newcastle University, NE4 5TG United Kingdom (e-mail: sadegh.soudjani@ncl.ac.uk).

a *parity* specification in step **(I)** of the decomposition. A parity specification is a canonical representation for all ω-regular properties [13], [39]; thus, our approach provides a way to under-approximate any ω-regular specification.

The main contribution of our paper is to show that the approximate solution to step **(I)** can be computed by a *symbolic algorithm* over a finite state *abstraction* of the underlying CMP that is using only the support of the probabilistic evolution of the system. This *abstraction-based* policy synthesis technique is inspired by abstraction-based controller design (ABCD) for non-stochastic systems [31], [33], [38]. In ABCD, a nonlinear dynamical system is abstracted into a discrete two-player game over a finite discrete state space obtained by partitioning the continuous state space into a finite set of cells. The resulting abstract two-player game is then used to synthesize a discrete controller which is then refined into a continuous controller for the original system.

In ABCD, the abstract two-player game models the interplay between the controller ($Player\ 0$) and the dynamics ($Player\ 1$) such that the resulting abstract controller (i.e., the winning strategy of $Player\ 0$) can be correctly refined to the original system. This requires a very powerful $Player\ 1$; in every instance of the play (corresponding to the system being in one particular abstract cell $\widehat{s}$) and for any input u chosen by $Player\ 0$, $Player\ 1$ can adversarially choose both (a) the actual continuous state s within $\widehat{s}$ to which u is applied and (b) any continuous disturbance affecting the system at this state s.

The key insight in our work is that the stochastic nature of the underlying CMP does not require a fully adversarial treatment of continuous disturbances by $Player\ 1$ in the abstract game to allow for controller refinement. Intuitively, disturbances need to be handled in a *fair* way: In the long run, all transitions with positive probability will eventually occur. We show, that the resulting fairness assumption on the behavior of $Player\ 1$ can be modeled by an additional *random* player (also called $1/2$ player) resulting in a so called $2^1/_2$-player game [4]–[6] as the abstraction.

This provides a conceptually very appealing result of our paper. Using $2^1/_2$-player games as abstractions of CMPs allows to utilize the machinery of symbolic game solving, analogous to ABCD techniques for non-stochastic systems, while capturing the intuitive differences between the problem instances by the use of a random player in the abstract game. Most interestingly, the stochastic nature of the resulting abstract game *eases* the abstract synthesis problem compared to standard ABCD where disturbances are non-stochastic. In conclusion, we obtain a *symbolic algorithm* to compute an under-approximation of the almost sure winning region in a *continuous-state* CMP for all ω-regular specifications. Moreover, similar to the results for finite-state MDPs, this shows that the (approximate) solution to step **(I)** does not need to handle the actual transition probabilities. They are only needed in step **(II)**, where existing techniques can be used.

A preliminary version of our paper has been accepted for publication in an upcoming conference [29]. In this manuscript we have made substantial improvement over the conference version; the main additional contributions are as follows:

1) We have *addressed the quantitative aspect* of the optimal policy synthesis problem in Thm. 3.2 and Thm. 3.3, making our exposition complete. In contrast, the conference version of the paper only dealt with the qualitative aspect of the problem.
2) We have *added the proof* of Thm. 4.5, which is the main theorem of this paper and which was omitted in the conference version.
3) We have substantially *improved the experimental evaluation* (see Sec. VI) of our approach by considering benchmark examples proposed by other authors, and comparing the performance of our tool against the available tool from the literature.
4) We have provided examples and extensive explanation of the required steps throughout the manuscript.

Related Work. Our paper extends the recent results of Majumdar et al. [30] from Büchi specifications to parity specifications. Seen through the lens of $2^1/_2$-player games, the algorithm of Majumdar et al. [30] can be seen as directly solving a Büchi game symbolically on a non-probabilistic abstraction, by implicitly reducing the $2^1/_2$-player games to two player games on graphs with extreme fairness assumptions [18]. While it may be possible to present a similar "direct" symbolic algorithm for parity games, the details of handling fairness symbolically get difficult. Our exposition in this paper helps separate out the different combinatorial aspects: the representation of the abstraction and the solution of the game on the abstraction, leading to a clean proof of correctness.

$2^1/_2$-player games have been used as abstractions of probabilistic systems, both in the finite case [24] and for stochastic *linear* systems [37]. Our paper subsumes the result of both these cases by showing a computational procedure to abstract a general, nonlinear CMP into a finite-state $2^1/_2$-player game. Further the existing approach for stochastic linear systems [37] only considers specifications in the GR(1) fragment of linear temporal logic, whereas we can handle any ω-regular specification. Another difference is that the linearity assumption enables the use of symbolic algorithms based on polyhedral manipulations, when the specification is given using polytopic predicates on the state space. Instead, our abstractions are based on gridding the state space, as in ABCD for non-stochastic nonlinear systems.

Stochastic nonlinear systems were abstracted to finite-state bounded-parameter Markov decision processes (BMDP) by Dutreix et al. [12] for the purpose of controller synthesis. By using algorithms for finding controllers on BMDPs against deterministic Rabin automata, they provide an alternative approach for approximating the optimal controller against ω-regular specifications. Their method is conceptually very different. It explicitly computes lower and upper bounds of all involved transition probabilities for constructing the BMDP, and computes winning regions by an enumerative algorithm taking these probability bounds into account. On the other hand, our approach shows a clean separation between step **(II)**, which requires

knowing explicit transition probabilities but can be solved by existing techniques, and step **(I)**, for which this knowledge is not needed. This allows us to provide a conceptually simpler and computationally superior *symbolic algorithm* approximately solving **(I)** via abstract $2^1/_2$-player games.

In principle, one can alternatively obtain a similar $2^1/_2$-player game, like us, by first reducing a CMP to a BMDP using the method of Dutreix et al. [12] and then reducing the BMDP to a $2^1/_2$-player game using the results of Weininger et al. [44]. However, this would still require going through the expensive construction of the BMDP, whereas we present a direct and computationally efficient way to obtain a $2^1/_2$-player game through reachable set computations for the CMP. The computational benefit of our approach over the BMDP based method is enormous: On a couple of benchmark examples taken from the paper of Dutreix et al. [12] itself, our uniform-grid based implementation was up to around 150 times faster and used up to around 150 times smaller amount of memory than their adaptive refinement-based implementation. The details of the experiments can be found in Sec. VI.

II. Stochastic Nonlinear Systems

A. Preliminaries

For any set A, a sigma-algebra on A comprises subsets of A as events that includes A itself and is closed under complement and countable unions. We consider a probability space $(\Omega, \mathcal{F}_\Omega, P_\Omega)$, where Ω is the sample space, $\mathcal{F}_\Omega$ is a sigma-algebra on Ω, and P_Ω is a probability measure that assigns probabilities to events. An ($(S, \mathcal{F}_S)$-valued) random variable X is a measurable function of the form $X : (\Omega, \mathcal{F}_\Omega) \to (S, \mathcal{F}_S)$, where S is the codomain of X and $\mathcal{F}_S$ is a sigma-algebra on S. Any random variable X induces a probability measure on its space $(S, \mathcal{F}_S)$ as $P(\{A\}) = P_\Omega\{X^{-1}(A)\}$ for any $A \in \mathcal{F}_S$. We often directly discuss the probability measure on $(S, \mathcal{F}_S)$ without explicitly mentioning the underlying probability space $(\Omega, \mathcal{F}_\Omega, P_\Omega)$ and the function X itself.

A topological space S is called a Borel space if it is homeomorphic to a Borel subset of a Polish space (i.e., a separable and completely metrizable space). Examples of a Borel space are the Euclidean spaces $\mathbb{R}^n$, its Borel subsets endowed with a subspace topology, as well as hybrid spaces. Any Borel space S is assumed to be endowed with a Borel sigma-algebra (i.e., the one generated by the open sets in the topology), which is denoted by $\mathcal{B}(S)$. We say that a map $f : S \to Y$ is measurable whenever it is Borel measurable.

Given an alphabet A, we use the notation A^* and A^ω to denote respectively the set of all finite words, the set of all infinite words formed using the letters of the alphabet A, and use A^∞ to denote the set $A^* \cup A^\omega$. Let X be a set and $R \subseteq X \times X$ be a relation. For simplicity, let us assume that $\mathsf{dom}\ R \coloneqq \{x \in X \mid \exists y \in X\ .\ (x, y) \in R\} = X$. For any given $x \in X$, we use the notation $R(x)$ to denote the set $\{y \in X \mid (x, y) \in R\}$. We extend this notation to sets: For any given $Z \subseteq X$, we use the notation $R(Z)$ to denote the set $\cup_{z \in Z} R(z)$. Given a set A, we use the notation $Dist(A)$ to denote the set of all probability distributions over A.

We denote the set of nonnegative integers by $\mathbb{N} \coloneqq \{0, 1, 2, \ldots\}$ and the set of integers in an interval by $[a; b] \coloneqq \{a + k \mid k \in \mathbb{N},\ k \leq b - a\}$. We also use the symbols "$\in_{even}$" and "$\in_{odd}$" to denote memberships in the set of even and odd integers within a given set of integers: For example, for a given set of natural numbers $M \subseteq \mathbb{N}$, the notation $n \in_{even} M$ is equivalent to $n \in M \cap \{0, 2, 4, \ldots\}$, and the notation $n \in_{odd} M$ is equivalent to $n \in M \cap \{1, 3, 5, \ldots\}$.

B. Controlled Markov Processes

A *controlled Markov process (CMP)* is a tuple $\mathfrak{S} = (\mathcal{S}, \mathcal{U}, T_\mathfrak{s})$, where $\mathcal{S}$ is a Borel space called the *state space*, $\mathcal{U}$ is a finite set called the *input space*, and $T_\mathfrak{s}$ is a conditional stochastic kernel $T_\mathfrak{s} \colon \mathcal{B}(\mathcal{S}) \times \mathcal{S} \times \mathcal{U} \to [0, 1]$ with $\mathcal{B}(\mathcal{S})$ being the Borel sigma-algebra on the state space and $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ being the corresponding measurable space. The kernel $T_\mathfrak{s}$ assigns to any $s \in \mathcal{S}$ and $u \in \mathcal{U}$ a probability measure $T_\mathfrak{s}(\cdot|s, u)$ on the measurable space $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ so that for any set $A \in \mathcal{B}(\mathcal{S})$, $P_{s,u}(A) = \int_A T_\mathfrak{s}(ds|s, u)$, where $P_{s,u}$ denotes the conditional probability $P(\cdot|s, u)$.

In general, the input space $\mathcal{U}$ can be any Borel space and the set of valid inputs can be state dependent. While our results can be extended to this setting, for ease of exposition, we consider the special case where $\mathcal{U}$ is a finite set and any input can be taken at any state. This choice is motivated by the digital implementation of control policies with a finite number of possible actuations.

The evolution of a CMP is as follows. For $k \in \mathbb{N}$, let X^k denote the state at the k-th time step and A^k the input chosen at that time. If $X^k = s \in \mathcal{S}$ and $A^k = u \in \mathcal{U}$, then the system moves to the next state X^{k+1}, according to the probability distribution $P_{s,u}$. Once the transition into the next state has occurred, a new input is chosen, and the process is repeated.

Given a CMP $\mathfrak{S}$, a *finite path* of length $n + 1$ is a sequence

$$w^n = (s^0, s^1, \ldots, s^n), \quad n \in \mathbb{N},$$

where $s^i \in \mathcal{S}$ are state coordinates of the path. The space of all paths of length $n + 1$ is denoted $\mathcal{S}^{n+1}$. An *infinite path* of the CMP $\mathfrak{S}$ is the sequence $w = (s^0, s^1, s^2, \ldots)$, where $s^i \in \mathcal{S}$ for all $i \in \mathbb{N}$. The space of all infinite paths is denoted by $\mathcal{S}^\omega$. The spaces $\mathcal{S}^{n+1}$ and $\mathcal{S}^\omega$ are endowed with their respective product topologies and are Borel spaces.

A *stationary control policy* is a universally measurable function $\rho : \mathcal{S} \to \mathcal{U}$ such that at any time step $n \in \mathbb{N}$, the input u^n is taken to be $\rho(s^n) \in \mathcal{U}$. As we only deal with stationary control policies in this paper, we simply refer to them as *policies* for short. We denote the class of all such policies by Π. The function ρ is also called *state feedback controller* in control theory.

For a CMP $\mathfrak{S}$, any policy $\rho \in \Pi$ together with an initial probability measure $\alpha : \mathcal{B}(\mathcal{S}) \to [0,1]$ on the states induces a unique probability measure on the canonical sample space of paths [19], denoted by P^ρ_α with the expectation $\mathbb{E}^\rho_\alpha$. In the case when the initial probability measure is supported on a single state $s \in \mathcal{S}$, i.e., $\alpha(s) = 1$, we write P^ρ_s and $\mathbb{E}^\rho_s$ in place of P^ρ_α and $\mathbb{E}^\rho_\alpha$, respectively. We denote the set of probability measures on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ by $\mathfrak{D}$.

Given any ω-regular specification φ defined using a set of predicates over the state space $\mathcal{S}$ of $\mathfrak{S}$, we use the notation $\mathfrak{S} \models \varphi$ to denote the set of all paths of $\mathfrak{S}$ which satisfy φ. Thus, $P^\rho_\alpha(\mathfrak{S} \models \varphi)$ denotes the probability of satisfaction of φ by $\mathfrak{S}$ under the effect of the control policy ρ, when the initial probability measure is given by α. Often we will use Linear Temporal Logic (LTL) notation to express ω-regular properties. The syntax and semantics of LTL can be found in standard literature [2].

A *stochastic dynamical system* Σ is described by a state evolution

$$s^{k+1} = f(s^k, u^k, \varsigma^k), \quad k \in \mathbb{N}, \tag{1}$$

where $s^k \in \mathcal{S}$ and $u^k \in \mathcal{U}$ are states and inputs for each $k \in \mathbb{N}$, and $(\varsigma^0, \varsigma^1, \ldots)$ is assumed to be a sequence of independent and identically distributed (i.i.d.) random variables representing a stochastic disturbance. The map f gives the next state as a function of current state, current input, and the disturbance. One can construct a CMP over states $\mathcal{S}$ and inputs $\mathcal{U}$ from (1) by noticing that for any given state s^k and input u^k at time k, the next state is a random variable defined as a function of ς^k. Thus, $T_{\mathfrak{s}}(\cdot | s^k, u^k)$ is exactly the distribution of the random variable $f(s^k, u^k, \varsigma^k)$ and can be computed based on the distribution of ς^k and the map f itself [22].

C. Parity Specifications

Let $\mathfrak{S} = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$ be a CMP and suppose $\mathcal{P} = \langle B_0, B_1, \ldots, B_\ell \rangle$ is a partition of $\mathcal{S}$ with measurable sets $B_0, \ldots, B_\ell$; that is, $B_i \cap B_j = \emptyset$ for $i \neq j$ and $\cup_{i=1}^{\ell} B_i = \mathcal{S}$. We allow some B_i's to be empty. For each B_i, we call the integer i its *priority*.

Intuitively, an infinite path $w \in \mathcal{S}^\omega$ satisfies the *parity specification* with respect to $\mathcal{P}$ if the highest subset B_i visited infinitely often has even priority. This specification is formalized in Linear Temporal Logic (LTL) notation [2] using the following formula:

$$Parity(\mathcal{P}) \coloneqq \bigwedge_{i \in_{odd}[1;\ell]} \left(\square\lozenge B_i \to \bigvee_{j \in_{even}[i+1;\ell]} \square\lozenge B_j \right), \tag{2}$$

which requires that infinitely many visits to an odd priority subset ($\square\lozenge B_i$) must imply infinitely many visits to a *higher* even priority subset ($\square\lozenge B_j$). We indicate the set of all infinite paths $w \in \mathcal{S}^\omega$ of a CMP $\mathfrak{S}$ that satisfy the property $Parity(\mathcal{P})$ by $\mathfrak{S} \models Parity(\mathcal{P})$. The proof of measurability of the event $\mathfrak{S} \models Parity(\mathcal{P})$ goes back to the work by Vardi [42] that provides the proof for probabilistic finite state programs. The proof for a CMP follows similar principles, using the observation that $\mathfrak{S} \models Parity(\mathcal{P})$ can be written as a Boolean combination of events $\mathfrak{S} \models \square\lozenge A$, where A is a measurable set, and $\square\lozenge A$ is a canonical G_δ set in the Borel hierarchy.

It is well-known that every ω-regular specification whose propositions range over measurable subsets of the state space of a CMP can be modeled as a deterministic parity automaton [15, Thm. 1.19]. By taking a synchronized product of this parity automaton with the CMP, we can obtain a product CMP and a parity specification over the product state space such that every path satisfying the parity specification also satisfies the original ω-regular specification and vice versa. Moreover, a stationary policy for the parity objective gives a (possibly history-dependent) policy for the original specification. Thus, without loss of generality, we assume that an ω-regular objective is already given as a parity condition using a partition of the state space of the system.

III. Problem Definition

We are interested in computing the maximal probability that a given parity specification can be satisfied by paths of a CMP $\mathfrak{S}$ starting from a particular state $s \in \mathcal{S}$ under stationary policies. Given a control policy $\rho \in \Pi$ and an initial state $s \in \mathcal{S}$, we define the satisfaction probability and the supremum satisfaction probability as

$$f(s, \rho) \coloneqq P^\rho_s(\mathfrak{S} \models Parity(\mathcal{P})) \text{ and} \tag{3}$$

$$f^*(s) \coloneqq \sup_{\rho \in \Pi} P^\rho_s(\mathfrak{S} \models Parity(\mathcal{P})), \tag{4}$$

respectively. An *optimal control policy* for the parity condition is a policy ρ^* such that $f^*(s) = f(s, \rho^*)$ for all $s \in \mathcal{S}$. Note that an optimal policy need not exist, since the supremum may not be achieved by any policy. Our goal is to study the following *optimal policy synthesis* problem.

Problem 1 (Optimal Policy Synthesis): Given $\mathfrak{S}$ and a parity specification $Parity(\mathcal{P})$, find an optimal control policy ρ^*, if it exists, together with $f^*(s)$ such that $P_s^{\rho^*}(\mathfrak{S} \models Parity(\mathcal{P})) = f^*(s)$.

While the satisfaction probability (3) and the supremum satisfaction probability (4) are both well-defined, we are not aware of any work characterizing necessary or sufficient conditions for existence of optimal control policies on continuous-space CMPs for parity specifications. Additionally, we restrict attention to *stationary* policies. While it is possible to define more general classes of policies, that depend on the entire history and use randomization over $\mathcal{U}$, we are again unaware of any work that characterizes the class of policies that are sufficient for optimal control of CMPs for parity specifications. For finite-state systems, stationary policies are sufficient and we restrict attention to them.

Since we cannot prove existence or computability of optimal policies, in this paper, we focus on providing an approximation procedure to compute a possibly sub-optimal control policy and guaranteed lower bounds on the optimal satisfaction probability. Our procedure relies on first approximating almost sure winning regions (i.e., where the specification can be satisfied with probability one), and then solving a reachability problem, as formalized next.

Definition 3.1 (Almost sure winning region): Given a CMP $\mathfrak{S}$, a policy ρ, and a parity specification $Parity(\mathcal{P})$, the state $s \in \mathcal{S}$ satisfies the specification *almost surely* if $f(s, \rho) = 1$. The almost sure *winning region*—or simply the winning region—of the policy ρ is defined as

$$\mathsf{WinDom}(\mathfrak{S}, \rho) \coloneqq \{s \in \mathcal{S} \mid f(s, \rho) = 1\}. \tag{5}$$

We also define the *maximal almost sure winning region*—or simply the maximal winning region—as

$$\mathsf{WinDom}^*(\mathfrak{S}) \coloneqq \{s \in \mathcal{S} \mid f^*(s) = 1\}. \tag{6}$$

Note that $\mathsf{WinDom}(\mathfrak{S}, \rho) \subseteq \mathsf{WinDom}^*(\mathfrak{S})$ for any control policy $\rho \in \Pi$. It is clear by definition of the winning region that for any policy ρ, the satisfaction probability $P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P}))$ is equal to 1 for any s in the winning region $W \coloneqq \mathsf{WinDom}(\mathfrak{S}, \rho)$. The next theorem states that this satisfaction probability is lower bounded by the probability of reaching the winning region W for any $s \notin W$. We denote such a reachability by $(\mathfrak{S} \models \Diamond W) \coloneqq \{w = (s^0, s^1, s^2, \ldots) \mid \exists n \in \mathbb{N} \,.\, s^n \in W\}$.

Theorem 3.2: For any control policy $\rho \in \Pi$ on CMP $\mathfrak{S}$, and $W \coloneqq \mathsf{WinDom}(\mathfrak{S}, \rho)$, we have

$$\begin{aligned} &P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P})) = 1 && \text{if } s \in W \text{ and} \\ &P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P})) \geq P_s^\rho(\mathfrak{S} \models \Diamond W) && \text{if } s \notin W. \end{aligned} \tag{7}$$

The proof can be found in the appendix. The inequality in the second part of (7) is because the $Parity(\mathcal{P})$ specification may be satisfied with positive probability even though the path always stays outside of W. When the state space is finite (i.e., for finite Markov decision processes), equality holds [2]. However, equality need not hold for general CMPs: Majumdar et al. [30] showed an example where the maximal almost sure winning region is empty even though a Büchi specification is satisfied with positive probability.

The next theorem establishes that for any policy ρ, the winning region is an absorbing set, i.e., paths starting from this set will stay in the set almost surely.

Theorem 3.3: For any control policy ρ, The set $W = \mathsf{WinDom}(\mathfrak{S}, \rho)$ is an absorbing set, i.e., $T_{\mathfrak{s}}(W|s, \rho(s)) = 1$ for all $s \in W$. This implies $P_s^\rho(\mathfrak{S} \models \Diamond \mathcal{S} \backslash W) = 0$ for all $s \in W$.

The proof of this theorem utilizes the fact that $Parity(\mathcal{P})$ is a long-run property and its satisfaction is independent of the prefix of a path. The proof is provided in the appendix.

Notice that Thm. 3.2 and Thm. 3.3 are stated for any fixed control policy ρ, but these theorems enable us to decompose the maximization of $P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P}))$ with respect to ρ into two sub-problems. First, find a policy that gives the largest winning region W and employ that policy when the current state is in W. Then, find a policy that maximizes the reachability probability $P_s^\rho(\mathfrak{S} \models \Diamond W)$ and employ that policy as long as the current state is not in W.

Computation of the reachability probability has been studied extensively in the literature for both infinite horizon [16], [17], [40], [41] and finite horizon [21], [23], [25]–[27], [34]–[36], [43] using different abstract models and computational methods. Together with an algorithm that underapproximates the region of almost sure satisfaction, these approaches can be used to provide a lower bound on the probability of satisfaction of the parity condition. In the rest of the paper, we focus on the following problem (the first half of (7)).

Problem 2 (Approximate Maximal Winning Region): Given $\mathfrak{S}$ and a parity specification $Parity(\mathcal{P})$, find a (sub-optimal) control policy $\rho \in \Pi$, its winning region $\mathsf{WinDom}(\mathfrak{S}, \rho) \neq \emptyset$, and a bound on the absolute volume of the set difference $\mathsf{WinDom}^*(\mathfrak{S}) \backslash \mathsf{WinDom}(\mathfrak{S}, \rho)$, which we call the *approximation error*.

In Sec. IV-V, we provide a solution for Prob. 2 via the paradigm of abstraction-based controller design. Not surprisingly, we get a tighter (i.e., larger) approximation of $\mathsf{WinDom}^*(\mathfrak{S})$ if we use a finer discretization of the state space during the abstraction step. We also provide an over-approximation of $\mathsf{WinDom}^*(\mathfrak{S})$, and show closeness of the under- and over-approximation of $\mathsf{WinDom}^*(\mathfrak{S})$ in the numerical example provided in Sec. VI.

IV. Abstraction-Based Policy Synthesis

The main result of our paper is a solution to Prob. 2 via a *symbolic algorithm* over abstract $2\frac{1}{2}$-player games in the spirit of abstraction-based controller design (ABCD). ABCD is typically used to compute temporal-logic controllers for *non-stochastic*

nonlinear dynamical systems [31], [33], [38] in two steps. First, the system is abstracted into an abstract finite-state two-player game. This game is then used to synthesize a discrete controller which is then refined into a continuous controller for the original system. In standard ABCD techniques, the abstract game has two players: *Player* 0 simulating the controller and choosing the next control input u based on the currently observed abstract state $\widehat{s}$, and *Player* 1 simulating the adversarial effect of (a) choosing any continuous state s in $\widehat{s}$ to which u is applied and (b) choosing any continuous disturbance d that effects the resulting transition.

The key insight in our abstraction step is that the stochastic nature of the underlying CMP allows choosing disturbances in a *fair* random way instead of purely adversarially. We model this by introducing an additional *random* player (also called $\frac{1}{2}$ player) resulting in a so called $2^1/_2$-player game [4]–[6]. In the resulting abstract game, only the effect of the discretization is handled by *Player* 1 in an adversarial manner. The random player picks the applied disturbance uniformly at random.

After introducing necessary preliminaries on $2^1/_2$-player games in Sec. IV-A, we show how a CMP can be abstracted into a $2^1/_2$-player game in Sec. IV-B. We then recall in Sec. IV-C a symbolic procedure to find winning regions in $2^1/_2$-player games for parity specifications. Finally, we state in Sec. IV-D how an almost-sure winning strategy in the abstract $2^1/_2$-player game is refined, and that the resulting control policy is almost sure winning for the original CMP and its associated parity specification. This establishes soundness of our ABCD technique to solve Problem 2.

A. Preliminaries: $2^1/_2$-Player Parity Games

A $2^1/_2$-player *game graph* is a tuple $\mathcal{G} = \langle V, E, \langle V_0, V_1, V_r\rangle\rangle$, where V is a finite set of vertices, E is a set of directed edges $E \subseteq V \times V$, and the sets V_0, V_1, V_r form a partition of the set V. A $2^1/_2$-player *parity game* is a pair $\langle \mathcal{G}, \mathcal{P}\rangle$, where $\mathcal{G}$ is a $2^1/_2$-player game graph, and $\mathcal{P} = \langle B_0, B_1, \dots, B_\ell\rangle$ is a tuple of ℓ disjoint subsets of V, some of which can possibly be empty. The tuple $\mathcal{P}$ induces the parity specification $Parity(\mathcal{P})$ over the set of vertices V in the natural way. In order to ensure that $Parity(\mathcal{P})$ is well defined, we impose the restriction that every infinite run must have infinitely many occurrences of vertices from at least one of the nonempty sets in $\mathcal{P}$. In other words, we require that every set of vertices $U \subseteq V$ for which there is no $i \in [1; \ell]$ with $U \cap B_i \neq \emptyset$ must be "transient" vertices.

The players and their strategies. We assume that there are two players *Player* 0 and *Player* 1, who are playing a game by moving a token along the edges of the game graph $\mathcal{G}$. In every step, if the token is located in a vertex in V_0 or V_1, *Player* 0 or *Player* 1 respectively moves the token to one of the successors according to the edge relation E. On the other hand, if the token is located in a vertex $v \in V_r$, then in the next step the token moves to a vertex v' which is chosen uniformly at random from the set $E(v)$. Strategies of *Player* 0 and *Player* 1 are respectively the functions $\pi_0 \colon V^*V_0 \to Dist(V)$ and $\pi_1 \colon V^*V_1 \to Dist(V)$ such that for all $w \in V^*$, $v_0 \in V_0$ and $v_1 \in V_1$, we have $\mathsf{supp}\ \pi_0(wv_0) \subseteq E(v_0)$ and $\mathsf{supp}\ \pi_1(wv_1) \subseteq E(v_1)$. We use the notation Π_0 and Π_1 to denote the set of all strategies of *Player* 0 and *Player* 1 respectively. A strategy π_i of *Player* i, for $i \in \{0, 1\}$, is *deterministic memoryless* if for every $w_1, w_2 \in V^*$ and for every $v \in V_i$, $\pi_i(w_1 v) = \pi_i(w_2 v)$ holds; we simply write $\pi_i(v)$ in this case. We use the notation Π_i^{DM} to denote the set of all deterministic memoryless strategies of *Player* i. Observe that $\Pi_i^{\mathrm{DM}} \subseteq \Pi_i$.

Runs and winning conditions. An infinite (finite) run of the game graph $\mathcal{G}$, compatible with the strategies $\pi_0 \in \Pi_0$ and $\pi_1 \in \Pi_1$, is an infinite (a finite) sequence of vertices $r = v^0 v^1 v^2 \dots$ ($r = v^0 \dots v^n$ for some $n \in \mathbb{N}$) such that for every $k \in \mathbb{N}$, (a) $v^k \in V_0$ implies $v^{k+1} \in \mathsf{supp}\ \pi_0(v^0 \dots v^k)$, (b) $v^k \in V_1$ implies $v^{k+1} \in \mathsf{supp}\ \pi_1(v^0 \dots v^k)$, and (c) $v^k \in V_r$ implies $v^{k+1} \in E(v^k)$. Given an initial vertex v^0 and a fixed pair of strategies $\pi_0 \in \Pi_0$ and $\pi_1 \in \Pi_1$, we obtain a probability distribution over the set of infinite runs of the system. For a measurable set of runs $R \subseteq V^\omega$, we use the notation $P_{v^0}^{\pi_0,\pi_1}(R)$ to denote the probability of obtaining the set of runs R when the initial vertex is v^0 and the strategies of *Player* 0 and *Player* 1 are fixed to respectively π_0 and π_1. For an ω-regular specification φ, defined using a predicate over the set of vertices of $\mathcal{G}$, we write $(\mathcal{G} \models \varphi)$ to denote the set of all infinite runs for all possible strategies of *Player* 0 and *Player* 1 which satisfy φ. For example, $(\mathcal{G} \models Parity(\mathcal{P}))$ denotes the set of all infinite runs for all possible strategies of *Player* 0 and *Player* 1 which satisfy the parity condition $Parity(\mathcal{P})$. We say that *Player* 0 wins $Parity(\mathcal{P})$ almost surely from a vertex $v \in V$ (or v is almost sure winning for *Player* 0) if *Player* 0 has a strategy $\pi_0 \in \Pi_0$ such that for all $\pi_1 \in \Pi_1$ we have $P_v^{\pi_0,\pi_1}(\mathcal{G} \models Parity(\mathcal{P})) = 1$. We collect all vertices for which this is true in the almost sure winning region $\mathcal{W}(\mathcal{G} \models Parity(\mathcal{P}))$.

B. Abstraction: CMPs to $2^1/_2$-Player Games

Given a CMP $\mathfrak{S} = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$ and a parity specification $Parity(\mathcal{P})$ for a partition $\mathcal{P}$ of the state space $\mathcal{S}$ we construct an abstract $2^1/_2$-player game.

State-space abstraction. We introduce a finite partition $\widehat{\mathcal{S}} := \{\widehat{s}_i\}_{i \in I}$ such that $\cup_{i \in I} \widehat{s}_i = \mathcal{S}$, $\widehat{s}_i \neq \emptyset$ and $\widehat{s}_i \cap \widehat{s}_j = \emptyset$ for every $\widehat{s}_i, \widehat{s}_j \in \widehat{\mathcal{S}}$ with $i \neq j$. Furthermore, we assume that the partition $\widehat{\mathcal{S}}$ is consistent with the given priorities $\mathcal{P}$, i.e., for every partition element $\widehat{s} \in \widehat{\mathcal{S}}$, and for every $x, y \in \widehat{s}$, x and y belong to the same partition element in $\mathcal{P}$ (i.e., x and y are assigned the same priority). We call the set $\widehat{\mathcal{S}}$ the *abstract state space* and each element $\widehat{s} \in \widehat{\mathcal{S}}$ an *abstract state*.

We introduce the abstraction function $Q \colon \mathcal{S} \to \widehat{\mathcal{S}}$ as a mapping from the continuous to the abstract states: For every $s \in \mathcal{S}$, $Q \colon s \mapsto \widehat{s}$ such that $s \in \widehat{s}$. We define the concretization function as the inverse of the abstraction function: $Q^{-1} \colon \widehat{\mathcal{S}} \to 2^{\mathcal{S}}$,

$Q^{-1}\colon \widehat{s} \mapsto \{s \in \mathcal{S} \mid s \in \widehat{s}\}$. We generalize the use of Q and Q^{-1} to sets of states: For every $U \subseteq \mathcal{S}$, $Q(U) = \bigcup_{s\in U} Q(s)$, and for every $\widehat{U} \subseteq \widehat{\mathcal{S}}$, $Q^{-1}(\widehat{U}) = \bigcup_{\widehat{s}\in\widehat{U}} Q^{-1}(\widehat{s})$.

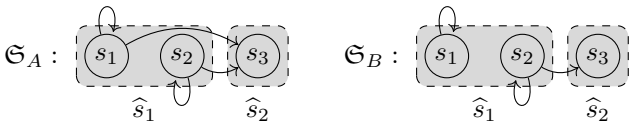
Transition abstraction. We also introduce an over- and an under-approximation of the probabilistic transitions of the CMP $\mathfrak{S}$ using the non-deterministic abstract transition functions $\overline{F}\colon \widehat{\mathcal{S}} \times \mathcal{U} \to 2^{\widehat{\mathcal{S}}}$ and $\underline{F}\colon \widehat{\mathcal{S}} \times \mathcal{U} \to 2^{\widehat{\mathcal{S}}}$ with the following properties:

$$\overline{F}(\widehat{s}, u) \supseteq \{\widehat{s}' \in \widehat{\mathcal{S}} \mid \exists s \in \widehat{s}\,.\, T_{\mathfrak{s}}(\widehat{s}' \mid s, u) > 0\}, \tag{8a}$$

$$\underline{F}(\widehat{s}, u) \subseteq \{\widehat{s}' \in \widehat{\mathcal{S}} \mid \exists \varepsilon > 0\,.\, \forall s \in \widehat{s}\,.\, T_{\mathfrak{s}}(\widehat{s}' \mid s, u) \geq \varepsilon\}. \tag{8b}$$

To understand the need for both $\overline{F}$ and $\underline{F}$ and the way they are constructed, consider the following example. Intuitively, given an abstract state $\widehat{s}$ and an input u, the set $\overline{F}$ over-approximates the set of abstract states reachable by probabilistic transitions from $\widehat{s}$ on input u. On the other hand, $\underline{F}$ under-approximates those abstract states which can be reached by *every* state in $\widehat{s}$ with probability bounded away from zero.

Example 4.1: Consider the two CMPs, $\mathfrak{S}_A$ and $\mathfrak{S}_B$:



The circles are concrete states s_i, the dashed boxes denote abstract states $\widehat{s}_i$, and the edges denote transitions with positive probability between concrete states s_i. Consider the left abstract state $\widehat{s}_1$. Here, the adversary decides which concrete state (i.e., s_1 or s_2) the game is in. In both $\mathfrak{S}_A$ and $\mathfrak{S}_B$, $\overline{F}$ says that both $\widehat{s}_1$ and $\widehat{s}_2$ are reachable from $\widehat{s}_1$. In $\mathfrak{S}_A$, $\underline{F}$ contains both $\widehat{s}_1$ and $\widehat{s}_2$, in $\mathfrak{S}_B$, $\underline{F}$ is empty. An adversary that plays according to $\overline{F}$ is too strong: it can keep playing the self loop in s_2, while the stochastic nature of the CMP ensures that eventually s_2 will transition to s_3. In order to follow the probabilistic semantics, we must ensure the adversary picks a distribution whose support contains both abstract states.

In $\mathfrak{S}_B$, the probabilistic behavior of the two concrete states s_1 and s_2 are very different: s_1 stays in $\widehat{s}_1$ with probability one and s_2 stays in $\widehat{s}_1$ or moves probabilistically to $\widehat{s}_2$. To ensure correct behavior, we look at possible supports of distributions induced by the dynamics: these are the possible subsets of abstract states between $\underline{F}$ and $\overline{F}$. Here, the game either stays in $\widehat{s}_1$ or (eventually) moves to $\widehat{s}_2$ and, in our reduction, we force the adversary to commit to one of the two options. ■

The parameter ε states that there is a uniform lower bound on transition probabilities for all states in an abstract state. This ensures that, provided $\widehat{s}$ is visited infinitely often and u is applied infinitely often from $\widehat{s}$, then $\widehat{s}'$ will be reached almost surely from $\widehat{s}$. In the absence of a uniform lower bound, this property need not hold for infinite state systems; for example, if the probability converges to zero, the probability of escaping $\widehat{s}$ can be strictly less than one.

Algorithmic computation of $\overline{F}$ and $\underline{F}$. While it is difficult to compute $\overline{F}$ and $\underline{F}$ in general, they can be approximated for the important subclass of stochastic nonlinear systems with *affine* disturbances

$$s^{k+1} = f(s^k, u^k) + \varsigma^k, \quad k \in \mathbb{N},$$

where $\varsigma^0, \varsigma^1, \ldots$ are independent and identically distributed random variables from a distribution with a bounded support D, and we assume we are only interested in a compact region $\widehat{\mathcal{S}}'$ of the state space. In this case, for any abstract state $\widehat{s}$ and any $u \in U$, one can compute an approximation $\mathit{ReachSet}(\widehat{s}, u)$ with $\mathit{ReachSet}(\widehat{s}, u) \supseteq \{s' \in \mathcal{S} \mid \exists s \in \widehat{s}\,.\, f(s, u) = s'\}$ using standard techniques [1], [7], [33]. Define $S_1, S_2\colon 2^{\widehat{\mathcal{S}}} \times \mathcal{U} \to 2^{\mathcal{S}}$ such that

$$S_1\colon (\widehat{s}, u) \mapsto D \oplus \mathit{ReachSet}(\widehat{s}, u),$$
$$S_2\colon (\widehat{s}, u) \mapsto D \ominus (-\mathit{ReachSet}(\widehat{s}, u)),$$

where the minus sign $(-\mathit{ReachSet}(\widehat{s}, u))$ is applied to each individual element of $\mathit{ReachSet}(\widehat{s}, u)$ and $\oplus$ and $\ominus$ are Minkowski sum and difference, respectively. Using S_1 and S_2, the functions $\overline{F}(\cdot,\cdot)$ and $\underline{F}(\cdot,\cdot)$ can be computed as [30, Thm. 6.1]: (1) $\widehat{s}' \in \overline{F}(\widehat{s}, u)$ iff either $S_1(\widehat{s}, u) \subseteq \mathcal{S}'$ and $\widehat{s}' \cap S_1(\widehat{s}, u) \neq \emptyset$ or $S_1(\widehat{s}, u) \not\subseteq \mathcal{S}'$ and $\widehat{s}'$ is a special sink state; and (2) $\widehat{s}' \in \underline{F}(\widehat{s}, u)$ iff either $\lambda(\widehat{s}' \cap S_2(\widehat{s}, u)) > 0$ or $\lambda(S_2(\widehat{s}, u) \setminus \mathcal{S}') > 0$ and $\widehat{s}'$ is a special sink state, and where $\lambda(\cdot)$ denotes the Lebesgue measure (generalized volume) of a set.

Remark 1: We remark that standard algorithms [7], [33] for computing $\mathit{ReachSet}(\cdot,\cdot)$, including the one that is used in our implementation, have the following *monotonicity* property: For every $\widehat{s}, \widehat{s}' \in \widehat{\mathcal{S}}$ and for every $u \in \mathcal{U}$, if $\widehat{s} \subseteq \widehat{s}'$ then

$$\mathit{ReachSet}(\widehat{s}, u) \subseteq \mathit{ReachSet}(\widehat{s}', u).$$

Monotonicity of $\mathit{ReachSet}(\cdot,\cdot)$ implies $\overline{F}(\cdot,\cdot)$ and $\underline{F}(\cdot,\cdot)$ are monotone: For every $\widehat{s}, \widehat{s}' \in \widehat{\mathcal{S}}$ and for every $u \in \mathcal{U}$, if $\widehat{s} \subseteq \widehat{s}'$ then

$$\overline{F}(\widehat{s}, u) \subseteq \overline{F}(\widehat{s}', u) \qquad \text{and} \qquad \underline{F}(\widehat{s}, u) \supseteq \underline{F}(\widehat{s}', u). \tag{9}$$

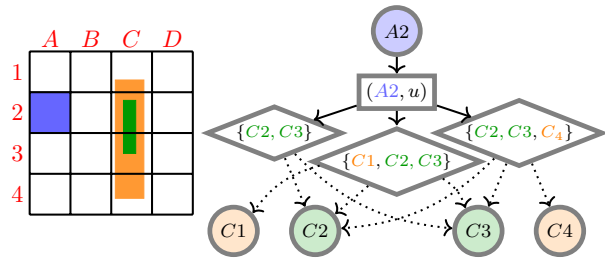


Fig. 1: Illustration of the construction of the abstract $2^1/_2$-player game (right) from a continuous-state CMP (left). The state space of the CMP is discretized into rectangular abstract states $A1, \ldots, D3$; $\underline{F}(A2, u) = \{C2, C3\}$ (intersecting the green region), and $\overline{F}(A2, u) = \{C1, C2, C3, C4\}$ (intersecting orange region). V_0, V_1 and V_r are indicated by circle, rectangular and diamond-shaped nodes. Random vertices are dashed.

This property ensures that we get a better (i.e., larger) approximation of the optimal almost sure winning domain $\mathsf{WinDom}^*(\mathfrak{S})$ in problem 2 if we use a finer discretization of the state space during the abstraction step. This observation can be empirically confirmed from the experiments in Sec. VI.

Abstract $2^1/_2$-player game graph. Given the abstract state space $\widehat{\mathcal{S}}$ and the over and under-approximations of the transition functions $\underline{F}$ and $\overline{F}$, we are ready to construct the abstract $2^1/_2$-player game graph induced by a CMP.

Definition 4.2: Let $\mathfrak{S}$ be a given CMP. Then its induced abstract $2^1/_2$-player game graph is given by $\mathcal{G} = \langle V, E, \langle V_0, V_1, V_r \rangle \rangle$ such that

- $V_0 = \widehat{\mathcal{S}}$ and $V_1 = \widehat{\mathcal{S}} \times \mathcal{U}$;
- $V_r = \bigcup_{v_1 \in V_1} V_r(v_1)$, where
  $V_r(v_1) := \{v_r \subseteq \widehat{\mathcal{S}} \mid \underline{F}(v_1) \subseteq v_r \subseteq \overline{F}(v_1), 1 \leq |v_r| \leq |\underline{F}(v_1)|+1\}$;
- and it holds that
  - for all $v_0 \in V_0$, $E(v_0) = \{(v_0, u) \mid u \in \mathcal{U}\}$
  - for all $v_1 \in V_1$, $E(v_1) = V_r(v_1)$, and
  - for all $v_r \in V_r$, $E(v_r) = \{v_0 \in V_0 \mid v_0 \in v_r\}$.

Note that $V_r(v_1)$ contains non-empty subsets of $\widehat{\mathcal{S}}$ that includes all the abstract states in $\underline{F}(v_1)$ and possibly include only one additional element from $\overline{F}(v_1)$. The construction is illustrated in Fig. 1.

In the reduced game, *Player* 0 models the controller, *Player* 1 models the effect of discretization of the state space of $\mathfrak{S}$, and the random edges from the states in V_r model the stochastic nature of the transitions of $\mathfrak{S}$. Intuitively, the game graph in Def. 4.2 captures the following interplay which is illustrated in Fig. 1: At every time step, the control policy for $\mathfrak{S}$ has to choose a control input $u \in \mathcal{U}$ based on the current vertex $\widehat{s}$ of $\mathcal{G}$. Since the control policy is oblivious to the precise continuous state $s \in \mathcal{S}$ of $\mathfrak{S}$, hence u is required to be an optimal choice for *every* continuous state $s \in \widehat{s}$. This requirement is materialized by introducing a fictitious adversary (i.e. *Player* 1) who, given $\widehat{s}$ and u, picks a continuous state $s \in \widehat{s}$ from which the control input u is to be applied. Now, we know that no matter what continuous s is chosen by *Player* 1, $T_{\mathfrak{s}}(\underline{F}(\widehat{s}, u) \mid s, u) > \varepsilon$ holds for some $\varepsilon > 0$. This explains why every successor of the $(\widehat{s}, u) \in V_1$ states contains the set of vertices $\underline{F}(\widehat{s}, u)$. Moreover, depending on which exact $s \in \widehat{s}$ *Player* 1 chooses, with positive probability the system might go to some state in $\overline{F}(\widehat{s}, u) \setminus \underline{F}(\widehat{s}, u)$. This is materialized by adding every state in $\overline{F}(\widehat{s}, u) \setminus \underline{F}(\widehat{s}, u)$ at a time to the successors of the states in V_1 (see Def. 4.2). Finally, we assume that the successor from every state in V_r is chosen uniformly at random (indicated by dotted edges in Def. 4.2). Later, it will be evident that the exact probability values are never used for obtaining the almost sure winning region, and so we could have chosen any other probability distribution.

Abstract parity specification. To conclude the abstraction of a given CMP $\mathfrak{S}$ and its parity specification $\mathcal{P} = \{B_1, \ldots, B_k\}$, we have to formally translate the priority sets B_i defined over subsets of states of the CMP into a partition of the vertices of the abstract $2^1/_2$-player game graph $\mathcal{G}$ induced by $\mathfrak{S}$. To this end, recall that we have assumed that the state space abstraction $\widehat{\mathcal{S}}$ respects the priority set $\mathcal{P}$.

Definition 4.3: Let $\mathfrak{S}$ be a CMP with parity specification $Parity(\mathcal{P})$ and $\mathcal{G}$ the abstract $2^1/_2$-player game graph induced by $\mathfrak{S}$. Then the induced abstract parity specification $\widehat{\mathcal{P}} = \{\widehat{B}_0, \ldots, \widehat{B}_\ell\}$ is defined such that $\widehat{B}_i = \{v_0 \in V_0 \mid Q^{-1}(v_0) \subseteq B_i\}$ for all $i \in [0; \ell]$. We denote the resulting $2^1/_2$-player parity game by the tuple $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$.

We note that the choice of the abstract parity set $\widehat{\mathcal{P}}$ does not partition the state space. Indeed, we implicitly assign an "undefined" color "$-$" to all nodes $V_1 \cup V_r$. Thereby, we only interpret the given parity specification over a projection of a run to its player 0 nodes. Formally, a run r over the abstract game graph $\mathcal{G}$ starting from a vertex $s^0 \in V_0$ is of the form:

$$r = s^0, (s^0, u^0), (\{s^{0,0}, \ldots, s^{0,i_0}\}), s^1, (s^1, u^1), (\{s^{1,0}, \ldots, s^{1,i_1}\}), \ldots$$

where $s^k \in \{s^{k,0}, \ldots, s^{k,i_k}\}$ for all $k \in \mathbb{N}$. The projection of the run r to the *Player* 0 states is defined as $\mathrm{Proj}_{V_0}(r) = s^0, s^1, \ldots$. Let φ be an ω-regular specification defined using a set of predicates over V_0. We use the convention that $(\mathcal{G} \models \varphi)$ will denote the set of every infinite run r of $\mathcal{G}$, for any arbitrary pair of strategies of *Player* 0 and *Player* 1, such that

$\mathrm{Proj}_{V_0}(r)$ satisfies φ. This convention is well-defined because every infinite run of $\mathcal{G}$ will have infinitely many occurrences of vertices from V_0 in it: This follows from the strict alternation of the vertices in V_0, V_1, and V_r, as per Def. 4.2.

C. Abstract Controller Synthesis

Once the $2^1/_2$-player parity game $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ is constructed from the CMP $\mathfrak{S}$ according to Def. 4.2, one can use existing techniques to compute the almost-sure winning states of *Player* 0 along with an associated almost-sure *Player* 0 winning policy ρ over $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$. The best-known algorithm is due to Chatterjee, Jurdzinski and Henzinger [5]. It first converts a $2^1/_2$-player parity game $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ into a *two-player* parity game and then uses well-known symbolic fixed-point algorithms [13], [28] to solve the latter game. The resulting *Player* 0 winning strategy ρ for the two-player game is known to be memoryless. Further, the same strategy constitutes a deterministic memoryless almost-sure *Player* 0 winning policy in the original $2^1/_2$-player game [5]. This implies that $2^1/_2$-player parity games are determined (from 2-player games being determined); that is, from every vertex, either *Player* 0 has a deterministic memoryless strategy to win almost surely, or *Player* 1 has a deterministic memoryless strategy to win with positive proability bounded away from zero [5, Thm. 2].

D. Controller Refinement

Now consider that the abstract $2^1/_2$-player parity game $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ constructed from the CMP $\mathfrak{S}$ via Def. 4.2 and Def. 4.3 has been solved as discussed in Sec. IV-C. Hence, we know the almost-sure *Player* 0 winning region $\mathcal{W}(\mathcal{G} \models \mathit{Parity}(\widehat{\mathcal{P}}))$ and the associated deterministic memoryless almost-sure *Player* 0 winning policy $\pi_0 \in \Pi^{\mathrm{DM}}$. Then we refine π_0 to a control policy $\rho \in \Pi$ for the CMP $\mathfrak{S}$ under parity condition $\mathit{Parity}(\widehat{\mathcal{P}})$ as follows.

Definition 4.4: Let $\mathfrak{S}$ be a CMP with parity specification $\mathit{Parity}(\mathcal{P})$ and $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ its induced finite $2^1/_2$-player parity game with deterministic memoryless almost sure *Player* 0 winning strategy $\pi_0 \in \Pi^{\mathrm{DM}}$. Then the control policy $\rho \in \Pi$ is called *the refinement of* π_0 if and only if for every $s \in \mathcal{S}$, if $s \in \widehat{s}$ for some $\widehat{s} \in \widehat{\mathcal{S}}$, and if $\pi_0(\widehat{s}) = (\widehat{s}, u) \in V_1$ for some $u \in \mathcal{U}$, then $\rho(s) := u$.

With the completion of this last step of our ABCD method for stochastic nonlinear systems we can finally state our main theorem providing a solution to Problem 2, which we prove in Sec. V.

Theorem 4.5 (Solution of Problem 2): Let $\mathfrak{S}$ be a CMP and $\mathit{Parity}(\mathcal{P})$ be a given parity specification. Let $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ be the abstract $2^1/_2$-player game defined in Def. 4.2. Suppose, a vertex $\widehat{s} \in V_0$ is almost sure winning for *Player* 0 in the game $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$, and $\pi_0 \in \Pi^{\mathrm{DM}}$ is the corresponding *Player* 0 winning strategy. Then the refinement ρ of π_0 ensures that $\widehat{s} \subseteq \mathsf{WinDom}(\mathfrak{S}, \rho)$.

Remark 2: An *over*-approximation of the optimal almost sure winning domain $\mathsf{WinDom}^*(\mathfrak{S})$ of $\mathfrak{S}$ w.r.t. $\mathit{Parity}(\mathcal{P})$ can be computed via $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ as well. To obtain an *over*-approximation, we solve this abstract game *cooperatively*. That is, we let player *Player* 0 choose both its own moves and the moves of player p_1 to win almost surely with respect to $\mathit{Parity}(\widehat{\mathcal{P}})$. Then the approximation error of Problem 2 can be upper-bounded as the Lebesgue measure of the set difference of the over- and the under-approximation, as will be done for the experiments in Sec. VI.

V. Proof of Theorem 4.5

Proof outline. To prove Thm. 4.5, we first decompose both the original and the abstract parity specifications $\mathit{Parity}(\mathcal{P})$ and $\mathit{Parity}(\widehat{\mathcal{P}})$ into a combination of more manageable safety and reachability sub-parts. That is, for every state reachable by a finite run in $\mathfrak{S}$, we consider a local safety specification ψ_S and a local reachability specification ψ_R defined by

$$\psi_S := \Box \neg B_i \text{ and } \psi_R := \Diamond \left(\psi_S \vee \bigvee\nolimits_{j \in_{even}[i+1;k]} B_j \right). \tag{10}$$

Intuitively, ψ_R requires that every time an odd priority—say B_i—is visited in $\mathfrak{S}$, eventually either B_i should never occur or an even priority B_j with $j > i$ should occur, almost surely. Similarly, for the abstract $2^1/_2$-player parity game $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ we consider the local safety specification $\widehat{\psi}_S$ and a local reachability specification $\widehat{\psi}_R$ defined by

$$\widehat{\psi}_S := \Box \neg \widehat{B}_i \text{ and } \widehat{\psi}_R := \Diamond \left(\widehat{\psi}_S \vee \bigvee\nolimits_{j \in_{even}[i+1;k]} \widehat{B}_j \right). \tag{11}$$

While the above decomposition needs to be established both for $\mathcal{G}$ and for $\mathfrak{S}$, the directions of the respective proof differ. For $\mathfrak{S}$ we show that if ψ_R holds for a state reachable by a finite run over $\mathfrak{S}$, then the original specification $\mathit{Parity}(\mathcal{P})$ is satisfied by a continuation of the run using the refined policy ρ (**Step 1**). For $\mathcal{G}$ we show that if $\mathit{Parity}(\widehat{\mathcal{P}})$ is satisfied, then $\widehat{\psi}_R$ holds for every state visited by a run compatible with the almost-sure winning strategy π_0 in $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$ (**Step 2**). Further, we show that satisfaction of $\widehat{\psi}_S$ (resp. $\widehat{\psi}_R$) in $\mathcal{G}$ implies satisfaction of ψ_S (resp. ψ_R) in $\mathfrak{S}$ (**Step 3-5**). With this, we have all ingredients to prove Thm. 4.5 (**Step 6**).

Step 1: Decomposition of $\mathit{Parity}(\mathcal{P})$**.** We prove a *sufficient* condition for satisfaction of $\mathit{Parity}(\mathcal{P})$ in $\mathfrak{S}$ if ψ_R holds.

Lemma 5.1: Let $\mathfrak{S}$ be a CMP, $\mathit{Parity}(\mathcal{P})$ be a parity specification, $s^0 \in \mathcal{S}$ be a given initial state, and ρ be a control policy. Suppose the following holds for every finite path $(s^0, \ldots, s^n) \in \mathcal{S}^{n+1}$ of $\mathfrak{S}$ and every $i \in_{odd} [1;k]$:

$$s^n \in B_i \Rightarrow P^{\rho}_{s^n}(\mathfrak{S} \models \psi_R) = 1. \tag{12}$$

Then $P^{\rho}_{s^0}(\mathfrak{S} \models Parity(\mathcal{P})) = 1$.

Proof: [Proof of Lem. 5.1] Define for any arbitrary $i \in_{odd} [1;k]$ the event $E_i \coloneqq (\mathfrak{S} \models \psi_i)$ with the specification $\psi_i \coloneqq \left(\square\lozenge B_i \rightarrow \bigvee_{j\in_{even}[i+1;k]} \square\lozenge B_j\right)$. We want to show that $P^{\rho}_{s^0}(\mathfrak{S} \models Parity(\mathcal{P})) = P^{\rho}_{s^0}\left(\bigcap_i E_i\right) = 1$ where $i \in_{odd} [1;k]$. We prove this by showing $P^{\rho}_{s^0}(\overline{E_i}) = 0$ for every $i \in_{odd} [1;k]$. Once we show this, the result follows according to the standard inequalities:

$$P^{\rho}_{s^0}\left(\bigcap_i E_i\right) = 1 - P^{\rho}_{s^0}\left(\bigcup_i \overline{E_i}\right) \geq 1 - \sum_i P^{\rho}_{s^0}\left(\overline{E_i}\right) = 1$$
$$\text{where } P^{\rho}_{s^0}(\overline{E_i}) = P^{\rho}_{s^0}\left((\mathfrak{S} \models \square\lozenge B_i) \cap (\mathfrak{S} \models \wedge_j \lozenge\square\neg B_j)\right),$$

with $i \in_{odd} [1;k]$ and $j \in_{even} [i+1;k]$. Define the random variable τ to be the largest time instance when the trajectory visits one of the sets B_j. Also define $\tau' > \tau$ to be the first time instance after τ when the trajectory visits B_i again. Note that for any trajectory satisfying $\square\lozenge B_i$ and $\wedge_j\lozenge\square\neg B_j$, both τ and τ' are well-defined and bounded. According to the assumption (12), we have

$$P^{\rho}_{s^0}(\overline{E_i} \,|\, \tau' = n, s^0, s^1, \cdots, s^n)$$
$$= P^{\rho}_{s^n}\left((\mathfrak{S} \models \square\lozenge B_i) \cap (\mathfrak{S} \models \wedge_j\lozenge\square\neg B_j)\right) = 0.$$

By taking the expectation with respect to the condition $(\tau', s^0, \cdots, s^n)$, we conclude that

$$P^{\rho}_{s^0}(\overline{E_i}) = \mathbb{E}^{\rho}_{s^0}\left[P^{\rho}_{s^0}(\overline{E_i} \,|\, \tau' = n, s^0, s^1, \cdots, s^n)\right] = \mathbb{E}^{\rho}_{s^0}[0] = 0,$$

i.e., $\overline{E_i}$ has a zero probability. ∎

Step 2: Decomposition of $Parity(\widehat{\mathcal{P}})$. We present a *necessary* condition for satisfaction of $Parity(\widehat{\mathcal{P}})$ in $\mathcal{G}$ if $\widehat{\psi}_R$ holds.

Lemma 5.2: Let $\langle \mathcal{G}, \widehat{\mathcal{P}}\rangle$ be a $2^1/_2$-player parity game, and v^0 be a given vertex of $\mathcal{G}$. Suppose $\pi^*_0 \in \Pi^{\mathrm{DM}}_0$ is a *Player* 0 strategy such that $\inf_{\pi_1\in\Pi_1} P^{\pi^*_0,\pi_1}_{v^0}(\mathcal{G} \models Parity(\widehat{\mathcal{P}})) = 1$. Then given every finite path $v^0 \dots v^n \in V^*$ such that there exists a *Player* 1 strategy $\pi_1 \in \Pi_1$ with $P^{\pi^*_0,\pi_1}_{v^0}(\mathcal{G} \models v^0 \dots v^n) > 0$, the following holds for every $i \in_{odd} [1;k]$:

$$v^n \in \widehat{B}_i \Rightarrow \inf_{\pi_1\in\Pi_1} P^{\pi^*_0,\pi_1}_{v^n}\left(\mathcal{G} \models \lozenge\widehat{\psi}_R\right) = 1. \tag{13}$$

The only new factor in Eq. (13) is the presence of the adversarial effect of the *Player* 1 strategies.

Proof: It follows from the definition of the parity specification in (2) that a vertex $v^0 \in V$ is almost sure winning using the strategy π^*_0 if the following condition is fulfilled:

$$\inf_{\pi_1\in\Pi_1} P^{\pi^*_0,\pi_1}_{v^0}\left(\mathcal{G} \models \bigwedge_{i\in_{odd}[1;k]} \square\left(\widehat{B}_i \Rightarrow \lozenge\psi_R\right)\right) = 1. \tag{14}$$

From the semantics of LTL, (14) implies:

$$\inf_{\pi_1\in\Pi_1} P^{\pi^*_0,\pi_1}_{v^0}\left(\mathcal{G} \models \bigwedge_i \forall m \in \mathbb{N}\,.\,(v^m \in \widehat{B}_i) \Rightarrow \lozenge\widehat{\psi}_R\right) = 1, \tag{15}$$

with $i \in_{odd} [1;k]$. We show that (15) implies for every finite run $v^0 \dots v^n \in V^*$, occurring with a positive probability $p_1 > 0$ for some strategy of *Player* 1, (13) holds. Suppose, for contradiction's sake, there exists some $i \in_{odd} [1;k]$ such that $v^n \in \widehat{B}_i$ and (13) does not hold, i.e., $\inf_{\pi_1\in\Pi_1} P^{\pi^*_0,\pi_1}_{v^n}\left(\mathcal{G} \models \lozenge\widehat{\psi}_R\right) < 1$, implying existence of some $0 < p_2 \leq 1$ with $\sup_{\pi_1\in\Pi_1} P^{\pi^*_0,\pi_1}_{v^n}\left(\mathcal{G} \not\models \lozenge\widehat{\psi}_R\right) = p_2$. This results in satisfaction of the parity specification with a probability of *at most* $(1 - p_1 \cdot p_2) < 1$, contradicting (15). ∎

Step 3: Refinement of $\widehat{\psi}_S$ to ψ_S. We show that almost sure safety with respect to a given set U in $\mathcal{G}$ implies the same with respect to the set $Q^{-1}(U)$ in $\mathfrak{S}$; this will later be used to infer $\widehat{\psi}_S \Rightarrow \psi_S$.

Proposition 5.3: Let $\mathfrak{S}$ be a CMP and $\mathcal{G}$ be a finite $2^1/_2$-player game graph as defined in Def. 4.2. Suppose $U \subseteq V_0$ is a given set of vertices of $\mathcal{G}$, and assume that there is a *Player* 0 vertex $v \in U$ for which there is a strategy $\pi_0 \in \Pi^{\mathrm{DM}}_0$ of *Player* 0 such that $\inf_{\pi_1\in\Pi_1} P^{\pi_0,\pi_1}_{v}(\mathcal{G} \models \square U) = 1$. Then the refinement $\rho \in \Pi$ of π_0 ensures that for every state $s \in v$, $P^{\rho}_{s}(\mathfrak{S} \models \square Q^{-1}(U)) = 1$.

Proof: It is known that for safety properties, almost sure satisfaction coincides with sure satisfaction, i.e., $\inf_{\pi_1\in\Pi_1} P^{\pi_0,\pi_1}_{v}(\mathcal{G} \models \square U) = 1$ if and only if for every strategy $\pi_1 \in \Pi_1$, every infinite run of $\mathcal{G}$ stays inside U at all time [10]. In other words, there must be a controlled invariant set W inside U for the strategy π_0, and $v \in W$. This controlled invariant set can be obtained by considering the 2-player game, obtained from $\mathcal{G}$ by removing all the random vertices, and redirecting the outgoing transitions of a given *Player* 1 vertex $v' \in V_1$ to the *Player* 0 vertices within the set $\overline{F}(v', \pi_0(v')) \subseteq V_0$. Since $\overline{F}(v', \pi_0(v'))$ overapproximates the set of all the continuous states reachable from v' using the input $\pi_0(v')$, hence if *Player* 0 can fulfill $\square U$ using the strategy $\pi_0(v')$, then ρ can fulfill $\square Q^{-1}(U)$ from every state $s \in v'$ in $\mathfrak{S}$. (This follows from the standard arguments in abstraction-based control using over-approximation based abstractions [33].) ∎

Step 4: Refinement of $\widehat{\psi}_R$ to ψ_R. We show that almost sure reachability with respect to a given set U in $\mathcal{G}$ implies the same with respect to the set $Q^{-1}(U)$ in $\mathfrak{S}$; this will be used to infer $\widehat{\psi}_R \Rightarrow \psi_R$.

Let $\mathfrak{S}$ be a CMP. Let us consider a reachability specification $\Diamond U$, for a set $U \subseteq V_0$, in the game $\mathcal{G}$ defined in Def. 4.2. Suppose $\pi_0 \in \Pi_0^{\mathrm{DM}}$ is some strategy of *Player* 0.

We introduce a ranking function $r\colon V_0 \to \mathbb{N} \cup \{\infty\}$ as a certificate for almost sure satisfaction of the specification $\Diamond U$. The ranking function r is defined inductively as follows:

$$r(v) = \begin{cases} 0 & v \in U, \\ \infty & \inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \models \Diamond U) < 1, \\ i+1 & \min\{n \in \mathbb{N} | \inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \models \bigcirc r^{-1}(n)) > 0\} \\ & \quad = i \wedge \inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \not\models \bigcirc r^{-1}(\infty)) = 1. \end{cases} \tag{16}$$

Note that every vertex $v \in V$ gets a rank: If $r(v) \neq \infty$, then $\inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \models \Diamond U) = 1$ by definition of r. In this case, there must exist some path to U, i.e., $\inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \not\models \bigcirc r^{-1}(\infty)) = 1$ must be true, and moreover $\inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \models \bigcirc r^{-1}(n)) > 0$ will be true for some n. Thus, $r(v) = n + 1$.

From the ranking function $r(\cdot)$ defined in (16), it is clear that $\inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \models \Diamond U) = 1$ implies $r(v) \neq \infty$. We first identify some local structural properties of the abstract transition functions $\overline{F}$ and $\underline{F}$ evaluated on some abstract states with finite ranking.

Lemma 5.4: Suppose $\pi_0 \in \Pi_0^{\mathrm{DM}}$ is some strategy of *Player* 0. For every $v \in V_0$ with $r(v) = i \neq \infty$, $i > 0$, both $\overline{F}(v, \pi_0(v)) \cap r^{-1}(\infty) = \emptyset$ and either of the following holds:

1) $\underline{F}(v, \pi_0(v)) \cap r^{-1}(i-1) \neq \emptyset$, or
2) $\underline{F}(v, \pi_0(v)) = \emptyset$ and $\overline{F}(v, \pi_0(v)) \subseteq r^{-1}(i-1)$.

Proof: Firstly, $\overline{F}(v, \pi_0(v)) \cap r^{-1}(\infty) = \emptyset$ should always hold as otherwise *Player* 1 would have a strategy to reach a state in $r^{-1}(\infty)$ with nonzero probability in the next step.

Suppose (2) does not hold, implying either (a) $\underline{F}(v, \pi_0(v)) \neq \emptyset$, or (b) the existence of a vertex $v' \in \overline{F}(v, \pi_0(v))$ with $r(v') \neq i-1$. Then $\underline{F}(v, \pi_0(v)) \cap r^{-1}(i-1) \neq \emptyset$ must hold, as otherwise, for case (a) and (b) *Player* 1 would have strategies π_1 with $\pi_1(v, \pi_0(v)) = (\underline{F}(v, \pi_0(v)))$ and $\pi_1(v, \pi_0(v)) = (\underline{F}(v, \pi_0(v)) \cup \{v'\})$ respectively, such that $P_v^{\pi_0,\pi_1}(\mathcal{G} \models \bigcirc r^{-1}(i-1)) = 0$.

On the other hand, suppose (1) does not hold. Then $\underline{F}(v, \pi_0(v)) = \emptyset$ must be true, as otherwise *Player* 1 would have a strategy π_1 with $\pi_1(v, \pi_0(v)) = (\underline{F}(v, \pi_0(v)))$ such that $P_v^{\pi_0,\pi_1}(\mathcal{G} \models \bigcirc r^{-1}(i-1)) = 0$. Moreover, $\overline{F}(v, \pi_0(v)) \subseteq r^{-1}(i-1)$ must also be true, as otherwise there would exist a vertex $v' \in \overline{F}(v, \pi_0(v))$ with $r(v') \neq i-1$, and *Player* 1 would have a strategy π_1 with $\pi_1(v, \pi_0(v)) = (\underline{F}(v, \pi_0(v)) \cup \{v'\}) = (\{v'\})$ such that $P_v^{\pi_0,\pi_1}(\mathcal{G} \models \bigcirc r^{-1}(i-1)) = 0$. ∎

The following lemma establishes soundness of the reduction with respect to reachability specifications.

Proposition 5.5: Let $\mathfrak{S}$ be a CMP and $\mathcal{G}$ be a finite $2\frac{1}{2}$-player game graph as defined in Def. 4.2. Suppose there is a *Player* 0 vertex $v \in V_0$ in $\mathcal{G}$ and a set of vertices $U \subseteq V_0$ for which there is a strategy $\pi_0 \in \Pi_0^{\mathrm{DM}}$ of *Player* 0 such that $\inf_{\pi_1 \in \Pi_1} P_v^{\pi_0,\pi_1}(\mathcal{G} \models \Diamond U) = 1$. Then the refinement $\rho \in \Pi$ of π_0 ensures that for every state $s \in v$, $P_s^{\rho}(\mathfrak{S} \models \Diamond Q^{-1}(U)) = 1$.

Proof: It follows from the definition of the ranking function in (16) that the set of almost sure winning vertices for the specification $\Diamond U$ is given by all the vertices with finite rank. We show that for every vertex v with a finite rank, the refinement $\rho \in \Pi$ of π_0 ensures that from every state $s \in v$, $P_s^{\rho}(\mathfrak{S} \models \Diamond Q^{-1}(U)) = 1$.

First, trajectories starting from any state $s \in v$ with $r(v) \neq \infty$ never go to the region $Q^{-1}(r^{-1}(\infty))$. This follows from the identity $\overline{F}(v, \pi_0(v)) \cap r^{-1}(\infty) = \emptyset$ in Lem. 5.4 and because $\overline{F}(v, \pi_0(v))$ is an overapproximation of the one step reachable set from the states within vertex v. Hence, every infinite trajectory of $\mathfrak{S}$ starting from s will visit the states in $\mathcal{S} \setminus Q^{-1}(r^{-1}(\infty))$ infinitely often.

The rest of the proof shows that if a trajectory visits the states $\mathcal{S} \setminus Q^{-1}(r^{-1}(\infty) \cup r^{-1}(0))$ infinitely often, then the trajectory will almost surely satisfy $\Diamond Q^{-1}(r^{-1}(0)) = \Diamond Q^{-1}(U)$. The is by induction over the largest rank assigned by r. For the base case, let the largest rank assigned by r be 2. We show that every state $s \in \mathcal{S}$ starting from inside a vertex v with $r(v) = 1$ or $r(v) = 2$ will almost surely reach $Q^{-1}(U)$, i.e., $P_s^{\rho}(\mathfrak{S} \models \Box Q^{-1}(r^{-1}(1) \cup r^{-1}(2))) = 0$. Note that the events $\{\Diamond \Box Q^{-1}(r^{-1}(2))\}$ and $\{\Box \Diamond Q^{-1}(r^{-1}(1))\}$ form a partition of the event of $\{\Box Q^{-1}(r^{-1}(1) \cup r^{-1}(2))\}$. Therefore,

$$\begin{aligned} & P_s^{\rho}(\mathfrak{S} \models \Box Q^{-1}(r^{-1}(1) \cup r^{-1}(2))) \\ & \quad = P_s^{\rho}(\mathfrak{S} \models \Box Q^{-1}(r^{-1}(1) \cup r^{-1}(2)) \wedge \Diamond \Box Q^{-1}(r^{-1}(2))) \\ & \quad + P_s^{\rho}(\mathfrak{S} \models \Box Q^{-1}(r^{-1}(1) \cup r^{-1}(2)) \wedge \Box \Diamond Q^{-1}(r^{-1}(1))). \end{aligned}$$

The first term is upper bounded by $P_s^{\rho}(\mathfrak{S} \models \Diamond \Box Q^{-1}(r^{-1}(2)))$ which is zero, because $P_s^{\rho}(\mathfrak{S} \models \Box Q^{-1}(r^{-1}(2))) = \prod_{n=1}^{\infty}(1-\varepsilon)^n = 0$. The second term is also zero because the event requires the number of transitions from $Q^{-1}(r^{-1}(1))$ to be infinite.

To see this, let $\boldsymbol{i}_n = (i_0, i_1, \ldots, i_n)$ be the first $(n+1)$ time instances that a trajectory visits $Q^{-1}(r^{-1}(1))$. Then,

$$\begin{aligned}
&P_s^\rho(\mathfrak{S} \models \Box Q^{-1}(r^{-1}(1) \cup r^{-1}(2)) \wedge \Box\Diamond Q^{-1}(r^{-1}(1))) = \\
&\sum_{\boldsymbol{i}_n} P_s^\rho(\mathfrak{S} \models \Box Q^{-1}(r^{-1}(1) \cup r^{-1}(2)) \wedge \Box\Diamond Q^{-1}(r^{-1}(1)) \,|\, \boldsymbol{i}_n) P_s^\rho(\boldsymbol{i}_n) \\
&\leq \sum_{\boldsymbol{i}_n} (1-\varepsilon)^n P_s^\rho(\boldsymbol{i}_n) = (1-\varepsilon)^n.
\end{aligned}$$

The last inequality is due to either Cond. (1) or Cond. (2) of Lem. 5.4 applied to the vertices in $v \in r^{-1}(1)$. Note that this inequality holds for any n. By taking the limit when n goes to infinity, we have that this second term is also zero. Hence the base case is established.

For the induction hypothesis, assume that the claim holds when the maximum rank assigned by the function r is i. Then for the induction step, i.e., when the maximum rank is $i+1$, we can follow same argument, as we did for the states with rank 2 in the base case, to show that every infinite trajectory inside $\mathcal{S} \setminus Q^{-1}(r^{-1}(\infty) \cup r^{-1}(0))$ will never get trapped inside $Q^{-1}(r^{-1}(i+1))$, which will mean that the trajectory will visit the states in $\mathcal{S} \setminus Q^{-1}(r^{-1}(\infty) \cup r^{-1}(0) \cup r^{-1}(i+1))$ infinitely often. Then it follows from the induction hypothesis that the trajectories will reach $Q^{-1}(U)$ almost surely. ∎

Step 5: Refinement of runs. We show that every finite path in $\mathfrak{S}$ can be mapped to a positive probability finite run in $\mathcal{G}$; this will be used establish a bridge from the universal quantification over finite paths in $\mathfrak{S}$ to the universal quantification over finite runs in $\mathcal{G}$.

Lemma 5.6: Let $\mathfrak{S}$ be a CMP, $\mathcal{G}$ be the abstract game graph as defined in Def. 4.2, $\pi_0 \in \Pi_0^{\mathrm{DM}}$ be an arbitrary *Player* 0 strategy in the game $\mathcal{G}$, and $s \in \mathcal{S}$ be a state of $\mathfrak{S}$. Suppose $\rho \in \Pi$ is the refinement of π_0. Then for every finite trajectory $s^0 \ldots s^n \in \mathcal{S}^*$ of $\mathfrak{S}$ in the support of the distribution $P_{s^0}^\rho$, there exists a *Player* 1 strategy $\pi_1 \in \Pi_1$ such that $P_{\widehat{s}^0}^{\pi_0,\pi_1}(\mathcal{G} \models \widehat{s}^0 \ldots \widehat{s}^n) > 0$, where $\widehat{s}^i = Q(s^i)$ for every $i \in [0; n]$.

Proof: The initial state $s^0 \in \widehat{s}^0$, and for every $0 \leq i < n$, from the definition of $\overline{F}$ it follows that $\widehat{s}^{i+1} \in \overline{F}(\widehat{s}^i, \pi_0(\widehat{s}^i))$. Thus, from every *Player* 1 vertex $(\widehat{s}^i, \pi_0(\widehat{s}^i))$, there is a successor vertex in V_r whose successor is $\widehat{s}^{i+1}$. Hence, for every $0 \leq i < n$, there is some move of *Player* 1 which causes a transition to $\widehat{s}^{i+1}$ with some positive probability p^i. Then $P_{\widehat{s}^0}^{\pi_0,\pi_1}(\mathcal{G} \models \widehat{s}^0 \ldots \widehat{s}^n) = \prod_{i=0}^{n-1} p^i > 0$. ∎

Step 6: The final assembly of the proof. Finally, we finish the proof of Thm. 4.5 by stitching everything together. It is known that memoryless strategies suffice for winning almost surely in $2^1/_2$-player parity games [45]. Let $\pi_0^* \in \Pi_0^{\mathrm{DM}}$ be the witness strategy of *Player* 0 to almost surely win from the vertex $\widehat{s}^*$ in the game $\langle \mathcal{G}, \widehat{\mathcal{P}} \rangle$, and ρ^* be the refinement of π_0^*. We claim that $\widehat{s}^* \subseteq \mathsf{WinDom}(\mathfrak{S}, \rho^*)$.

We will show that for *every* finite path of $\mathfrak{S}$ starting within $\widehat{s}^*$ and ending in some odd priority state B_i, eventually either B_i will not be visited any more, or a state of higher even color will be visited. Then the claim will follow from Lem. 5.1. We know from Lem. 5.6 that existence of a finite path $s^0 \ldots s^n$ of $\mathfrak{S}$ implies existence of an abstract run $\widehat{s}^0 \ldots \widehat{s}^n$ such that $\sup_{\pi_1 \in \Pi_1} P_{\widehat{s}^0}^{\pi_0,\pi_1}(\mathcal{G} \models \widehat{s}^0 \ldots \widehat{s}^n) > 0$. Moreover, due to the priority preserving partitions we have $\widehat{s}^n \in \widehat{B}_i$, where i is odd. Since π_0^* is an almost sure winning strategy, hence, by using Lem. 5.2, we know that the following holds:

$$\inf_{\pi_1 \in \Pi_1} P_{\widehat{s}^n}^{\pi_0^*,\pi_1}\left(\mathcal{G} \models \Diamond\left(\Box\neg\widehat{B}_i \vee \cup_{j \in even[i+1;k]} \widehat{B}_j\right)\right) = 1. \tag{17}$$

From Prop. 5.3, we know that the set of abstract states from which the specification $\Box\neg\widehat{B}_i$ is satisfied (almost) surely using the strategy π_0^* are also the set of continuous states from which the specification $\Box\neg B_i$ is satisfied almost surely using the controller ρ^*. Together with Prop. 5.5 and Lem. 5.1, we can infer Thm. 4.5 from (17).

VI. Numerical Example

We consider the controller synthesis problems for a two-dimensional stochastic bistable switch with a couple of different parity specifications; the examples have been adopted, mutatis mutandis, from the work of Dutreix et al. [12]. The dynamics of the system is modeled by the following difference equations:

$$s_1^{k+1} = s_1^k + \left(-a \cdot s_1^k + s_2^k\right) \cdot \tau + u_1^k + \varsigma_1^k \tag{18}$$

$$s_2^{k+1} = s_2^k + \left(\frac{\left(s_1^k\right)^2}{\left(s_1^k\right)^2 + 1} - b \cdot s_2^k\right) \cdot \tau + u_2^k + \varsigma_2^k, \tag{19}$$

where s_1, s_2 are the state variables, u_1, u_2 are the control inputs, a, b are constant parameters, τ is the sampling time, and ς_1, ς_2 are stochastic noises. We assume that the domain of the state variables is $[0.0, 4.0] \times [0.0, 4.0]$, and we saturate the state trajectories at the boundary of the domain. We consider a finite set of values for both of the control inputs: for every k, $u_1^k, u_2^k \in \{-0.05, 0.0, 0.05\}$. The values of the constants are given by: $a = 1.3$, $b = 0.25$, and $\tau = 0.05$. Finally, we

assume that the stochastic noise samples $(\varsigma_1^k, \varsigma_2^k)$ are drawn from a piecewise continuous density function with the support $D = [-0.4, -0.2] \times [-0.4, -0.2]$.[1]

Let A, B, C, D be sets of states, as shown in Fig. 2a. We are interested in synthesizing the almost sure winning controllers for the above system for the following two LTL specifications:

$$\varphi_1 := \square\left((\neg A \wedge \bigcirc A) \rightarrow (\bigcirc\bigcirc A \wedge \bigcirc\bigcirc\bigcirc A)\right),$$
$$\varphi_2 := (\square\lozenge B \rightarrow \lozenge C) \wedge (\lozenge D \rightarrow \square\neg C).$$

The specifications φ_1 and φ_2 can be represented using the parity automata [2] shown in Fig. 2b and 2c. Firstly, for each of the two specifications, we compute a product with the system model. Secondly, we apply the algorithm from Sec. IV on the product system to solve the synthesis problem. Both of these steps have been implemented on the open-source tool Mascot-SDS [30]. By symbolically encoding the abstract $2^1/_2$-player game using BDD-s, and by using sophisticated acceleration techniques for solving symbolic fixpoint algorithms from the literature [28], [32], we achieve significant improvement in performance, in comparison with the implementation of the enumerative algorithm of Dutreix et al. [12] in the tool called StochasticSynthesis.[3]

The tool StochasticSynthesis has a couple of additional features compared to our implementation in Mascot-SDS. Firstly, it performs an adaptive abstraction refinement procedure for achieving better computational efficiency over uniform abstractions. This is an orthogonal optimization tool that is known to be effective in discretization-based approaches for controller synthesis [14], [20], [31], [35], and we expect that our uniform abstraction-based synthesis procedure will benefit further from this in the future. Secondly, StochasticSynthesis also addresses the quantitative aspect of the synthesis problem, which is to maximize the probability of satisfying the given specification. In all our experiments, we disabled this second feature of StochasticSynthesis, since the quantitative part is not the main algorithmic contribution of our paper.

We performed all the experiments on a Macbook Pro (2015) laptop equipped with a 2.7 GHz Intel Core i5 processor and a 16 GB RAM.

We performed two sets of experiments to compare the performance of Mascot-SDS and StochasticSynthesis, one with the adaptive refinement feature of StochasticSynthesis disabled and one with the same enabled. The results have been summarized in Tab. I, Fig. 3, and Fig. 4. All the experiments empirically show that the approximation error reduces with finer discretization, as already mentioned in Rem. 2. We highlight the other main findings in the following: (A) When the abstraction-refinement feature of StochasticSynthesis was disabled, Mascot-SDS outperformed StochasticSynthesis by a large margin. In fact, for some levels of discretization, StochasticSynthesis crashed due to memory limitation, whereas Mascot-SDS consumed quite manageable amount of memory and synthesized controllers within reasonable amount of time. (B) Even when the abstraction-refinement feature of StochasticSynthesis was enabled, for achieving the same level of approximation error, Mascot-SDS was significantly faster for φ_1 and was competitive for φ_2, and consumed much less memory for both φ_1 and φ_2: For φ_1, at one point Mascot-SDS was more than 150 times faster and consumed around 150 times lesser memory. These findings demonstrate the superior capabilities of our symbolic solution approach, which can be potentially further improved by using adaptive refinement techniques.

VII. Appendix

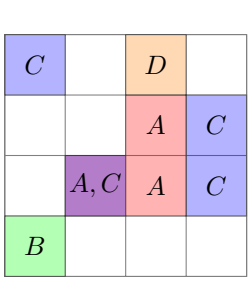
A. Proof of Statements

Proof: [Proof of Thm. 3.2] By the definition of the winning set, we already know that $P_s^{\rho}(\mathfrak{S} \models Parity(\mathcal{P})) = 1$ for all $s \in \mathsf{WinDom}(\mathfrak{S}, \rho)$. Take any $s \notin W := \mathsf{WinDom}(\mathfrak{S}, \rho)$. Define τ to be the first time step when the path visits the set W. Note that τ is a random variable taking values in $\mathbb{N} \cup \{\infty\}$. We use the law of total probability by making the event

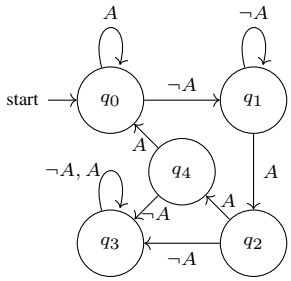
[1] Dutreix et al. [12] considered the density function to be given by truncated Gaussian distribution with support D. In our work, we disregard the shape of the distribution because we restrict the focus to only the qualitative satisfaction of the specification.

[2] Dutreix et al. [12] modeled φ_1 and φ_2 using Rabin automata, and we transformed them into (language-) equivalent parity automata to match our setup.

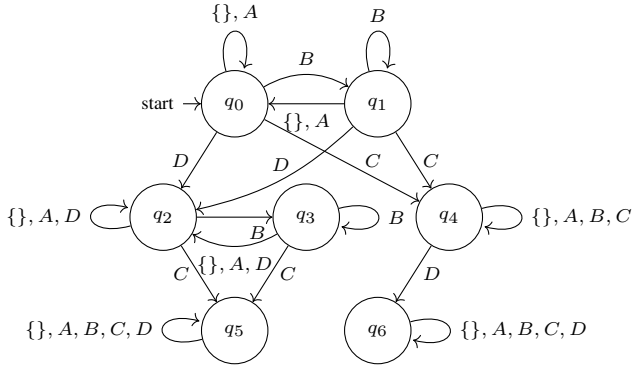
[3] Repository: `https://github.com/gtfactslab/StochasticSynthesis`,
commit nr.: `888b9dcf67369a732b8c225d790bd3343e4442e5`



(a) The state predicates.



(b) Parity automaton for φ_1; the prioritized partition is $\mathcal{P} = \langle \emptyset, \{q_2, q_3, q_4\}, \{q_0, q_1\} \rangle$. $\neg A$ stands for any element in $\{\{\}, B, C, D\}$.

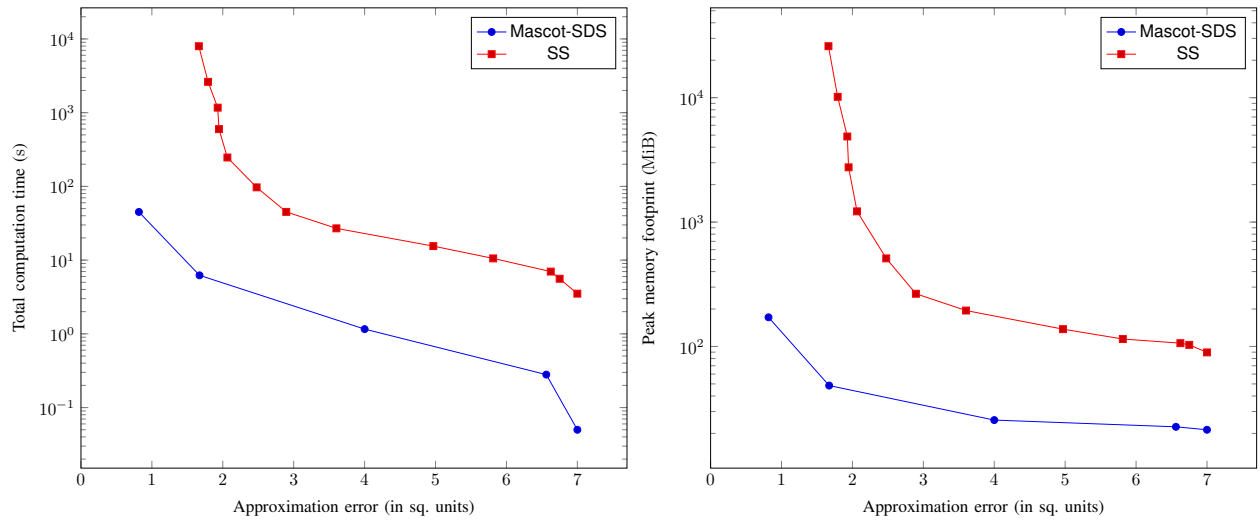


(c) Parity automaton for φ_2; the prioritized partition is $\mathcal{P} = \langle \{q_0, q_2, q_4\}, \{q_1, q_3, q_5, q_6\} \rangle$.

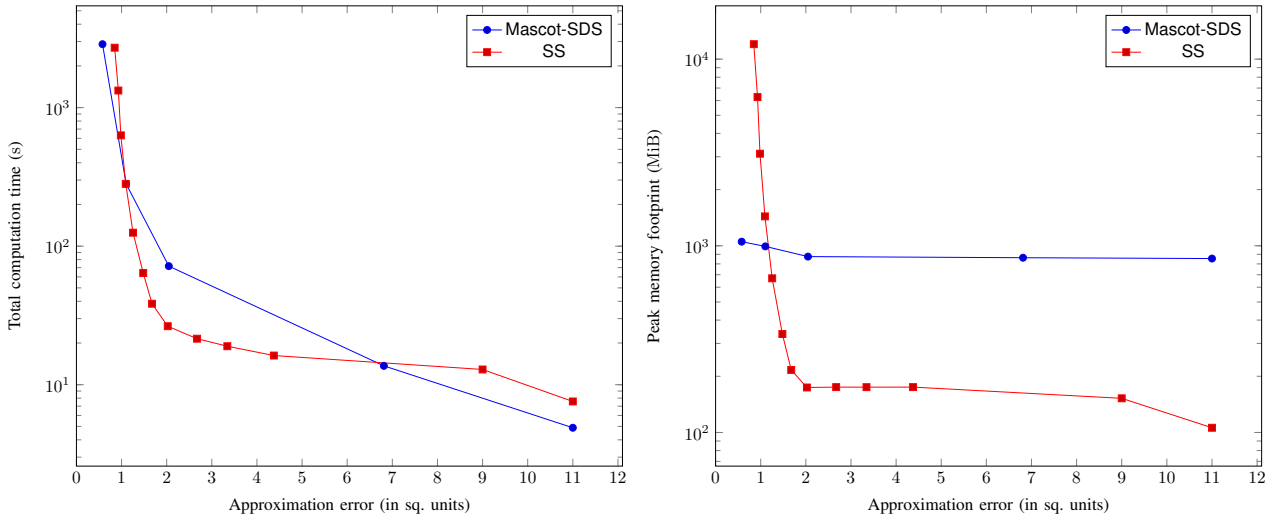
Fig. 2: The state predicates and the parity specifications.

Specification	Size of abstract states	Approximation error		Abstraction time		Total synthesis time		Peak memory footprint	
		Mascot-SDS	SS [12]	Mascot-SDS	SS [12]	Mascot-SDS	SS [12]	Mascot-SDS	SS [12]
φ_1	$1/2 \times 1/2$	7.0	9.0	0.003 s	0.4 s	0.02 s	8 s	21 MiB	125 MiB
	$1/4 \times 1/4$	6.6	5.9	0.04 s	13 s	0.2 s	18 s	23 MiB	1 GiB
	$1/8 \times 1/8$	4.0	3.0	0.2 s	35 min 5 s	0.7 s	9 min 18 s	26 MiB	81 GiB
	$1/16 \times 1/16$	1.7	OoM	0.9 s	OoM	5 s	OoM	50 MiB	127 GiB
	$1/32 \times 1/32$	0.8	OoM	5 s	OoM	37 s	OoM	171 MiB	127 GiB
φ_2	$1/2 \times 1/2$	11.0	11.8	0.004 s	0.6 s	5 s	30 s	864 MiB	156 MiB
	$1/4 \times 1/4$	6.8	3.4	0.02 s	15 s	13 s	55 s	866 MiB	1 GiB
	$1/8 \times 1/8$	2.0	1.8	0.2 s	34 min 20 s	1 min 10 s	16 min 1 s	890 MiB	81 GiB
	$1/16 \times 1/16$	1.1	OoM	0.9 s	OoM	4 min 37 s	OoM	994 MiB	126 GiB
	$1/32 \times 1/32$	0.6	OoM	5 s	OoM	45 min 39 s	OoM	1 GiB	127 GiB

TABLE I: Performance comparison between Mascot-SDS and StochasticSynthesis (abbreviated as SS) [12] when both tools are restricted to use *uniform grid-based abstraction* only. Col. 1 shows the specification considered, Col. 2 shows the size of each individual abstract state, Col. 3 and 4 compare the approximation errors, Col. 5 and 6 compare the abstraction computation times, Col. 7 and 8 compare the total synthesis times (combined time for computing the over- and the under-approximations of the almost sure winning regions), and Col. 9 and 10 compare the peak memory footprint (as measured using the "time" command) for both tools. "OoM" stands for out-of-memory. The length of the sides of the abstract states is measured in units, and the approximation error is measured in sq. units (the area covered by the abstract states which are in the over-approximation but not in the under-approximation).

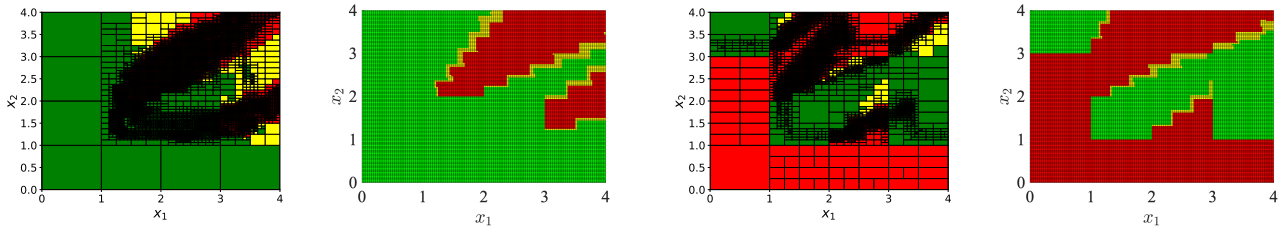


(a) Specification φ_1



(b) Specification φ_2

Fig. 3: Performance comparison between Mascot-SDS and StochasticSynthesis (abbreviated as SS) [7] when the latter *was allowed to use its inbuilt abstraction refinement process for better performance*. The different points on the plots were obtained by running Mascot-SDS using different sizes of abstract states, and running SS using different numbers of refinement stages.



(a) φ_1 using StochasticSynthesis (b) φ_1 using Mascot-SDS. (c) φ_2 using StochasticSynthesis. (d) φ_2 using Mascot-SDS.

Fig. 4: Visualization of the under-approximations—in green—and the over-approximations—in yellow and green, combined—of the almost sure winning regions for the specifications φ_1 and φ_2 as computed using the tools Mascot-SDS and StochasticSynthesis; in red are the complements of the over-approximations. For Mascot-SDS, the abstract states were chosen of the size $1/32 \times 1/32$. For StochasticSynthesis, initially the abstracts states were chosen of the size 1×1, and then the tool was made to execute 14 refinement steps to improve the approximation of the solution.

$(\mathfrak{S} \models Parity(\mathcal{P}))$ conditional on τ. Then we have

$$\begin{aligned}
&P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P})) \\
&\quad = \sum_{n=0}^{\infty} P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P}) \,|\, \tau = n) P_s^\rho(\tau = n) \\
&\quad + P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P}) \,|\, \tau = \infty) P_s^\rho(\tau = \infty) \\
&\quad = \mathbb{E}_s^\rho \left[P_{s^n}^\rho(\mathfrak{S} \models Parity(\mathcal{P}) \,|\, s^1, s^2, \ldots, s^n, \tau = n) \right] \\
&\quad + P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P}) \wedge \tau = \infty) \\
&\quad =^* \sum_{n=0}^{\infty} P_s^\rho(s^1, s^2, \ldots, s^{n-1} \in \mathcal{S}\backslash W, s^n \in W) \\
&\quad + P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P}) \wedge \mathfrak{S} \models \Box \mathcal{S}\backslash W) \\
&\quad \geq P_s^\rho(\mathfrak{S} \models \Diamond W).
\end{aligned}$$

The equality (*) holds due to $s^n \in W$ and $P_{s^n}^\rho(\mathfrak{S} \models Parity(\mathcal{P})) = 1$. ■

Proof: [Proof of Thm. 3.3] For any $s \in W$, we have

$$\begin{aligned}
&P_s^\rho(\mathfrak{S} \models Parity(\mathcal{P})) \\
&= \int_{\mathcal{S}} P_{s_1}^\rho(\mathfrak{S} \models Parity(\mathcal{P})) T_{\mathfrak{s}}(ds_1 | s, \rho(s)) \\
&= \int_{W} T_{\mathfrak{s}}(ds_1 | s, \rho(s)) + \int_{\mathcal{S}\backslash W} P_{s_1}^\rho(\mathfrak{S} \models Parity(\mathcal{P})) T_{\mathfrak{s}}(ds_1 | s, \rho(s)).
\end{aligned}$$

This means

$$\begin{aligned}
&\int_{\mathcal{S}\backslash W} (1 - P_{s_1}^\rho(\mathfrak{S} \models Parity(\mathcal{P}))) T_{\mathfrak{s}}(ds_1 | s, \rho(s)) = 0 \Rightarrow \\
&\forall \epsilon > 0, P_s^\rho \left[(1 - P_{s_1}^\rho(\mathfrak{S} \models Parity(\mathcal{P}))) \mathbf{1}_{\mathcal{S}\backslash W}(s_1) \geq \epsilon \right] \leq \frac{0}{\epsilon} = 0,
\end{aligned}$$

where the last inequality is a consequence of Markov's inequality for non-negative random variables. By taking the union over a monotone positive sequence $\{\epsilon_n \to 0\}$, we get

$$\begin{aligned}
&P_s^\rho \left[(1 - P_{s_1}^\rho(\mathfrak{S} \models Parity(\mathcal{P}))) \mathbf{1}_{\mathcal{S}\backslash W}(s_1) > 0 \right] = 0, \\
&P_s^\rho \left[s_1 \in \mathcal{S}\backslash W \text{ and } P_{s_1}^\rho(\mathfrak{S} \models Parity(\mathcal{P})) < 1 \right] = 0, \\
&P_s^\rho \left[s_1 \in \mathcal{S}\backslash W \right] = 0.
\end{aligned}$$

■

REFERENCES

[1] M. Althoff and B. H. Krogh. Reachability analysis of nonlinear differential-algebraic systems. *IEEE Transactions on Automatic Control*, 59(2):371–383, 2013.

[2] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[3] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In P. S. Thiagarajan, editor, *Foundations of Software Technology and Theoretical Computer Science*, pages 499–513, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[4] K. Chatterjee and T. A. Henzinger. A survey of stochastic ω-regular games. *J. Comput. Syst. Sci.*, 78(2):394–413, 2012.

[5] K. Chatterjee, M. Jurdzinski, and T. A. Henzinger. Simple stochastic parity games. In *Computer Science Logic, 17th International Workshop, CSL 2003, 12th Annual Conference of the EACSL, and 8th Kurt Gödel Colloquium, KGC 2003, Vienna, Austria, August 25-30, 2003, Proceedings*, volume 2803 of *Lecture Notes in Computer Science*, pages 100–113. Springer, 2003.

[6] A. Condon. The complexity of stochastic games. *Inf. Comput.*, 96(2):203–224, 1992.

[7] S. Coogan and M. Arcak. Efficient finite abstraction of mixed monotone systems. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, pages 58–67. ACM, 2015.

[8] C. Courcoubetis and M. Yannakakis. Markov decision processes and regular events. In *Automata, Languages and Programming*, pages 336–349, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg.

[9] L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Department of Computer Science, Stanford University, 1998.

[10] L. de Alfaro and T. A. Henzinger. Concurrent omega-regular games. In *Proceedings Fifteenth Annual IEEE Symposium on Logic in Computer Science (Cat. No. 99CB36332)*, pages 141–154. IEEE, 2000.

[11] M. Dutreix and S. Coogan. Specification-guided verification and abstraction refinement of mixed-monotone stochastic systems, 2019.

[12] M. Dutreix, J. Huh, and S. Coogan. Abstraction-based synthesis for stochastic systems with omega-regular objectives. *arXiv preprint*, 2020. arXiv:2001.09236.

[13] E. A. Emerson and C. S. Jutla. Tree automata, Mu-calculus and determinacy. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, SFCS '91, page 368?377, USA, 1991. IEEE Computer Society.

[14] A. Girard, G. Gößler, and S. Mouelhi. Safety controller synthesis for incrementally stable switched systems using multiscale symbolic models. *IEEE Trans. Autom. Control.*, 61(6):1537–1549, 2016.

[15] E. Gradel and W. Thomas. *Automata, logics, and infinite games: a guide to current research*, volume 2500. Springer Science & Business Media, 2002.

[16] S. Haesaert, P. Nilsson, and S. Soudjani. Formal multi-objective synthesis of continuous-state MDPs. *IEEE Control Systems Letters*, pages 1–1, 2020.

[17] S. Haesaert and S. Soudjani. Robust dynamic programming for temporal logic control of stochastic systems. *IEEE Transactions on Automatic Control*, pages 1–1, 2020.

[18] S. Hart, M. Sharir, and A. Pnueli. Termination of probabilistic concurrent program. *ACM Trans. Program. Lang. Syst.*, 5(3):356–380, July 1983.

[19] O. Hernández-Lerma and J. B. Lasserre. *Discrete-time Markov control processes*, volume 30 of *Applications of Mathematics*. Springer, 1996.

[20] K. Hsu, R. Majumdar, K. Mallik, and A.-K. Schmuck. Multi-layered abstraction-based controller synthesis for continuous-time systems. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (part of CPS Week)*, pages 120–129, 2018.

[21] P. Jagtap, S. Soudjani, and M. Zamani. Formal synthesis of stochastic systems via control barrier certificates. *IEEE Transactions on Automatic Control*, pages 1–1, 2020.

[22] O. Kallenberg. *Foundations of modern probability*. Probability and its Applications. Springer Verlag, New York, 2002.

[23] N. Kariotoglou, M. Kamgarpour, T. H. Summers, and J. Lygeros. The linear programming approach to reach-avoid problems for Markov decision processes. *J. Artif. Int. Res.*, 60(1):263–285, Sept. 2017.

[24] M. Kwiatkowska, G. Norman, D. Parker, and G. Santos. PRISM-games 3.0: Stochastic game verification with concurrency, equilibria and time. In *Proc. 32nd International Conference on Computer Aided Verification (CAV'20)*, volume 12225 of *LNCS*, pages 475–487. Springer, 2020.

[25] M. Lahijanian, S. B. Andersson, and C. Belta. Formal verification and synthesis for discrete-time stochastic systems. *IEEE Transactions on Automatic Control*, 60(8):2031–2045, Aug 2015.

[26] A. Lavaei, S. Soudjani, and M. Zamani. Compositional construction of infinite abstractions for networks of stochastic control systems. *Automatica*, 107:125 – 137, 2019.

[27] K. Lesser and M. Oishi. Approximate safety verification and control of partially observable stochastic hybrid systems. *IEEE Transactions on Automatic Control*, 62(1):81–96, Jan 2017.

[28] D. E. Long, A. Browne, E. M. Clarke, S. Jha, and W. R. Marrero. An improved algorithm for the evaluation of fixpoint expressions. In *International Conference on Computer Aided Verification*, pages 338–350. Springer, 1994.

[29] R. Majumdar, K. Mallik, A.-K. Schmuck, and S. Soudjani. Symbolic qualitative control for stochastic systems via finite parity games. In *ADHS*, 2021. To appear.

[30] R. Majumdar, K. Mallik, and S. Soudjani. Symbolic controller synthesis for Büchi specifications on stochastic systems. In *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*, pages 1–11, 2020.

[31] P. Nilsson, N. Ozay, and J. Liu. Augmented finite transition systems as abstractions for control synthesis. *Discrete Event Dynamic Systems*, 27(2):301–340, 2017.

[32] N. Piterman and A. Pnueli. Faster solutions of rabin and streett games. In *21st Annual IEEE Symposium on Logic in Computer Science (LICS'06)*, pages 275–284. IEEE, 2006.

[33] G. Reissig, A. Weber, and M. Rungger. Feedback refinement relations for the synthesis of symbolic controllers. *TAC*, 62(4):1781–1796, 2017.

[34] S. Soudjani and A. Abate. Higher-order approximations for verification of stochastic hybrid systems. In *Automated Technology for Verification and Analysis*, volume 7561 of *LNCS*, pages 416–434. Springer Verlag, Berlin Heidelberg, 2012.

[35] S. Soudjani and A. Abate. Adaptive and sequential gridding procedures for the abstraction and verification of stochastic processes. *SIAM Journal on Applied Dynamical Systems*, 12(2):921–956, 2013.

[36] S. Soudjani, A. Abate, and R. Majumdar. Dynamic Bayesian networks for formal verification of structured stochastic processes. *Acta Informatica*, 54(2):217–242, Mar 2017.

[37] M. Svoreňová, J. Křetínskỳ, M. Chmelík, K. Chatterjee, I. Černá, and C. Belta. Temporal logic control for stochastic linear systems using abstraction refinement of probabilistic games. *Nonlinear Analysis: Hybrid Systems*, 23:230–253, 2017.

[38] P. Tabuada. *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer, 2009.

[39] W. Thomas. On the synthesis of strategies in infinite games. In *STACS 95, 12th Annual Symposium on Theoretical Aspects of Computer Science, Munich, Germany, March 2-4, 1995, Proceedings*, volume 900 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 1995.

[40] I. Tkachev and A. Abate. Regularization of Bellman equations for infinite-horizon probabilistic properties. In *Proceedings of the 15th ACM international conference on Hybrid Systems: computation and control*, pages 227–236, Beijing, PRC, April 2012.

[41] I. Tkachev, A. Mereacre, J.-P. Katoen, and A. Abate. Quantitative model-checking of controlled discrete-time Markov processes. *Information and Computation*, 253:1 – 35, 2017.

[42] M. Y. Vardi. Automatic verification of probabilistic concurrent finite state programs. In *26th Annual Symposium on Foundations of Computer Science*, pages 327–338. IEEE, 1985.

[43] A. P. Vinod and M. M. K. Oishi. Scalable underapproximative verification of stochastic LTI systems using convexity and compactness. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (Part of CPS Week)*, HSCC '18, pages 1–10, New York, NY, USA, 2018. ACM.

[44] M. Weininger, T. Meggendorfer, and J. Křetínskỳ. Satisfiability bounds for ω-regular properties in bounded-parameter markov decision processes. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2284–2291. IEEE, 2019.

[45] W. Zielonka. Perfect-information stochastic parity games. In *International Conference on Foundations of Software Science and Computation Structures*, pages 499–513. Springer, 2004.

Rupak Majumdar is a Scientific Director at the Max Planck Institute for Software Systems. His research interests are in the verification and control of reactive, real-time, hybrid, and probabilistic systems, software verification and programming languages, logic, and automata theory. Dr. Majumdar received the President's Gold Medal from IIT, Kanpur, the Leon O. Chua award from UC Berkeley, an NSF CAREER award, a Sloan Foundation Fellowship, an ERC Synergy award, "Most Influential Paper" awards from PLDI and POPL, and several best paper awards (including from SIGBED, EAPLS, and SIGDA). He received the B.Tech. degree in Computer Science from the Indian Institute of Technology at Kanpur and the Ph.D. degree in Computer Science from the University of California at Berkeley.

Kaushik Mallik is a PhD student at the Max Planck Institute for Software Systems (MPI-SWS) in Kaiserslautern, Germany. His research interests are in formal verification and control of dynamical systems, which are either continuous or discrete, stochastic or non-stochastic, and with a distributed or a monolithic structure. Prior to joining MPI-SWS, he received M.Tech in System and Control in 2015 from the Indian Institute of Technology Roorkee, India, and B.Tech in Electrical Engineering in 2012 from Meghnad Saha Institute of Technology, Kolkata, India. Mr. Mallik received a Sandwich Scholarship from DAAD in 2014 for pursuing his master dissertation in the Technical University Berlin, and received several other awards and scholarships during his B.Tech. and M.Tech.

Anne-Kathrin Schmuck is an independent research group leader at the Max Planck Institute for Software Systems (MPI-SWS) in Kaiserslautern, Germany, funded by the Emmy Noether Programme of the German Science Foundation (DFG). She received the Dipl.-Ing. (M.Sc.) degree in engineering cybernetics from OvGU Magdeburg, Germany, in 2009 and the Dr.-Ing. (Ph.D.) degree in electrical engineering from TU Berlin, Germany, in 2015. Between 2015 and 2020 she was a Postdoctoral researcher at MPI-SWS. Her current research interests include abstraction based controller design, reactive synthesis, supervisory control theory, hierarchical control and contract-based distributed synthesis.

Sadegh Soudjani is a Lecturer (Assistant Professor) in the School of Computing at Newcastle University, United Kingdom. He received the B.Sc. degrees in Pure Mathematics and Electrical Engineering, and the M.Sc. degree in control engineering from the University of Tehran, Tehran, Iran, in 2007 and 2009, respectively. He received the Ph.D. degree in Systems and Control in November 2014 from the Delft Center for Systems and Control at the Delft University of Technology, Delft, the Netherlands. Before joining Newcastle University, he was a postdoctoral researcher at the department of Computer Science, University of Oxford, United Kingdom, and at the Max Planck Institute for Software Systems, Germany. His research interests are formal synthesis, abstraction, and verification of complex dynamical systems with application in Cyber-Physical Systems, particularly involving power and energy networks.