Abstraction-based Synthesis for Stochastic Systems with Omega-Regular Objectives

Maxence Dutreix[a,*], Jeongmin Huh[a], Samuel Coogan[b]

The Georgia Institute of Technology, Atlanta, GA, USA

[a]*School of Electrical and Computer Engineering*

[b]*School of Electrical and Computer Engineering and School of Civil and Environmental Engineering*

Abstract

This paper studies the synthesis of controllers for discrete-time, continuous state stochastic systems subject to omega-regular specifications using finite-state abstractions. Omega-regular properties allow specifying complex behaviors and encompass, for example, linear temporal logic. First, we present a synthesis algorithm for minimizing or maximizing the probability that a discrete-time switched stochastic system with a finite number of modes satisfies an omega-regular property. Our approach relies on a finite-state abstraction of the underlying dynamics in the form of a Bounded-parameter Markov Decision Process arising from a finite partition of the system's domain. Such Markovian abstractions allow for a range of probabilities of transition between states for each selected action representing a mode of the original system. Our method is built upon an analysis of the Cartesian product between the abstraction and a Deterministic Rabin Automaton encoding the specification of interest or its complement. Specifically, we show that synthesis can be decomposed into a qualitative problem, where the so-called greatest permanent winning components of the product automaton are created, and a quantitative problem, which

*Corresponding author

Email addresses: `maxdutreix@gatech.edu` (Maxence Dutreix), `jhuh32@gatech.edu` (Jeongmin Huh), `sam.coogan@gatech.edu` (Samuel Coogan)

This project was supported in part by the NSF under project #1749357, and by the Air Force Office of Scientific Research under project FA9550-19-1-0015.

requires maximizing the probability of reaching this component in the worst-case instantiation of the transition intervals. Additionally, we propose a quantitative metric for measuring the quality of the designed controller with respect to the continuous abstracted states and devise a specification-guided domain partition refinement heuristic with the objective of reaching a user-defined optimality target. Next, we present a method for computing control policies for stochastic systems with a continuous set of available inputs. In this case, the system is assumed to be affine in input and disturbance, and we derive a technique for solving the qualitative and quantitative problems in the resulting finite-state abstractions of such systems. For this, we introduce a new type of abstractions called Controlled Interval-valued Markov Chains. Specifically, we show that the greatest permanent winning component of such abstractions are found by appropriately partitioning the continuous input space in order to generate a bounded-parameter Markov decision process that accounts for all possible qualitative transitions between the finite set of states. Then, the problem of maximizing the probability of reaching these components is cast as a (possibly non-convex) optimization problem over the continuous set of available inputs. A metric of quality for the synthesized controller and a partition refinement scheme are described for this framework as well. Finally, we present a detailed case study.

Keywords: finite-state abstractions, formal methods, interval-valued Markov chains, bounded-parameter Markov decision processes, stochastic systems.

1. Introduction

The need for systems that are both complex and reliable is more critical than ever. Not only are the models describing these systems becoming increasingly complicated, but the tasks they are expected to perform also continue to grow in complexity. For example, the operating specification may combine an invariance and a reachability condition and require that *the system will always return to a good state while always avoiding a bad state.* Such specifications can be formally and unambiguously represented as, for instance, a *Linear Temporal Logic* (LTL) [1] specification, among other classes of symbolic languages. In this paper, we consider the class of *ω-regular properties* [2], a superset of LTL.

Recent research efforts in formal verification and synthesis have focused on the development of robust controllers to ensure that systems requirements are unequivocally met for broad classes of specifications and dynamics [3] [4] [5] [6] [7] [8] [9]. A general approach is to obtain a (non)deterministic finite abstraction of the continuous-state system, encode the specification as an appropriate transition system called an automaton, compute a product construction between the system abstraction and the automaton, and then synthesize a controller by solving graph-based problems on the product [10] [11]. The controller obtained from the finite abstraction is then mapped onto the original abstracted states. However, this basic recipe does not immediately work for stochastic systems because the random disturbances acting upon such systems add a quantitative component to the transitions between states in the form of transition probabilities, preventing the use of standard transition systems as finite abstractions for this framework. Typically, this limitation is overcome by using probabilistic finite transition systems as abstractions for stochastic systems [12] [13] [14] [15] [16]. Even though general synthesis procedures for such abstractions inherit ideas from approaches proposed in non-stochastic settings, the mathematical machinery required is quite different.

Indeed, for stochastic systems, satisfaction of a specification may never be fully guaranteed due to randomness. Therefore, the synthesis problem requires finding a control policy which maximizes or minimizes the probability of occurrence of some desired behavior from a given initial condition. In this work, we consider the problem of synthesizing a control policy for a discrete-time, continuous-state stochastic system subject to an ω-regular specification. Although the existence of optimal policies for this problem is not known, we seek to devise policies which are satisfactory with respect to a reasonable metric of quality. First, we consider the case when the control action is selected from a finite set of modes that the system can switch between at each time step. Then we consider the case when the control action is selected from a continuous set of possible inputs.

Recent literature demonstrated the effectiveness of *Bounded-Parameter Markov Decision Processes* (BMDP) as a tool for the synthesis of control policies in stochastic systems [16] [17]. Indeed, BMDPs are naturally amenable to finite-state abstractions of switched stochastic systems constructed from a finite partition of the continuous system domain. As each discrete state abstracts the behavior of an uncountably infinite number of underlying continuous states, the probabilities of transition between states are specified as intervals for each mode of the BMDP, rather than just a single number as in standard Markov Decision Processes. Solving for an optimal switching policy in the BMDP abstraction results in a near-optimal policy for the objective of maximizing or minimizing the probability of satisfying the specification with respect to the original abstracted states. The quality of this policy with respect to the original system states naturally depends on the quality and fineness of the continuous domain partition from which the abstraction is constructed.

In [16], the authors present an algorithm for computing switching policies that either minimize or maximize the probability of satisfying Probabilistic Computation Tree Logic (PCTL) specifications in a BMDP. The theory developed in [16] has been applied to linear systems with additive Gaussian noise subject to cosafe LTL specifications and was shown to be computationally efficient [18]. This BMDP-based technique was also recently implemented in the comprehensive verification and synthesis toolbox StocHy [19]. However, PCTL and cosafe LTL are strictly less expressive than the ω-regular logic and cannot articulate certain important liveness and persistence properties, such as the infinite repetition of some event [20]. A similar problem was solved in [21] for LTL specifications, but the proposed solution makes simplifying assumptions on the connectivity properties of the system's abstraction which drastically reduces its scope of applicability. The synthesis of control strategies for interval Markov decision processes with multi-objectives that include ω-regular properties was discussed in [22]; unfortunately, the qualitative structure of the transition system is again assumed to be invariant, which alleviates key difficulties associated with the problem. The authors of [23] reinterpret the switching policy synthesis problem for ω-regular properties in BMDPs as an ω-regular stochastic game for which optimal policies can be computed. However, it is unclear how the game-theoretic framework used in this work can be extended to refine the BMDP abstraction of the original system if deemed necessary after the policy has been determined.

In this paper, we implement a procedure for computing switching policies in finite-mode discrete-time stochastic systems with the objective of minimizing or maximizing the probability of occurrence of any ω-regular property. We first create a partition of the continuous domain from which a BMDP abstraction of the system is generated. We then consider the Cartesian product between the BMDP abstraction and a Deterministic Rabin Automaton (DRA) representing the ω-regular property of interest for the maximization problem, or the complement of the property for the minimization problem, which is a different approach from [23]. We prove that any such product BMDP induces a largest set of so-called *Permanent Winning Component* for a subset of all possible switching policies, and show that the probability maximization and minimization problems reduce to a reachability maximization task on these sets of states in the product BMDP. Note that our approach does not necessitate any assumption on the connectivity structure of the BMDP unlike in [21] and [22]. Furthermore, we introduce a quantitative measure capturing the quality of the switching policy designed in the BMDP abstraction when mapped onto the continuous abstracted states with respect to the objective of minimizing or maximizing the probability of fulfilling some specification in the original system. Finally, we propose a partition refinement technique inspired by our method in [24], which considered only the verification problem without inputs, in order to reach a desired level of optimality for the computed policy with respect to the continuous system states and progressively discard control actions which are guaranteed to be suboptimal. While no formal proof of the convergence of this technique is provided in this article, such refinement-based heuristics have shown to work remarkably well in practice and offer advantages in terms of scalability.

Expanding on the theory for finite-mode systems, we address the problem of synthesizing controllers for stochastic systems with ω-regular objectives from a continuous set of available inputs using finite-state abstractions. Related works discussed the synthesis of controllers for continuous input stochastic systems subject to subsets of ω-regular properties, such as Büchi objectives [25], using abstraction-based methods. Here, we specifically study the class of stochastic systems which are affine-in-disturbance and affine-in-input. We introduce *Controlled Interval-valued Markov Chains* (CIMC), which serve as abstractions for continuous input systems. We present an algorithm for constructing the largest permanent winning components in the product between a CIMC and a DRA. Then, we show that the reachability maximization step on these components can be formulated as an optimization program. The quality of the designed policy with respect to the original abstracted system and state-space refinement are discussed as well in this framework.

In brief, the novel contributions of this article over existing works, and in particular over our work on the verification of stochastic systems in [24], are as follows:

- We present a synthesis procedure for finite-mode discrete-time stochastic systems against ω-regular specifications, implemented in Algorithm 4. Our approach employs BMDP abstractions constructed from a partition of the continuous domain of the system, and, in contrast to [23], we devise an automaton-based synthesis algorithm for BMDPs against ω-regular specifications from the results of Theorem 1 in conjunction with Algorithms 1 to 2. These algorithms perform a search of specific components of a BMDP which do not exist in abstractions without control actions along with the computation of policies generating these components, and therefore are more involved than the graph search algorithms found in [24].

- We introduce a quantitative measure of the quality of the policy computed from the BMDP abstraction with respect to the original abstracted system states. The results in [24] are not concerned with the computation of switching policies and therefore do not propound such a measure. This metric is determined from the facts highlighted in Theorem 2.

- We develop a specification-guided refinement strategy on the partition of the system domain in Algorithm 3 to enhance the quality of the switching policy in refined BMDP abstractions of the dynamics. While an algorithm is presented in [24] for verification that is similar in spirit, major differences are found in the input of both algorithms, their termination criteria and the computations performed to select the states to be refined.

- We extend the techniques above to synthesize controllers for affine-in-disturbance, affine-in-input stochastic systems with a continuous set of permissible inputs. The control policy is computed by means of CIMC abstractions constructed from a partition of the system domain and mapped onto the abstracted states as detailed in Algorithm 7. To this end, we present a synthesis procedure for CIMC abstractions arising from systems with the aforementioned structure that relies on Algorithms 5 and 6.

- For such systems with continuous input sets, we propose a refinement scheme for the domain partition to improve the quality of the computed controller with respect to the original abstracted states.

The paper is organized as follows: Section 2 introduces some preliminaries; Section 3 formulates the problem to be solved; Section 4 describes our controller synthesis strategy for finite-mode stochastic systems; Section 5 presents a controller synthesis algorithm for stochastic systems with a continuous set of inputs; Section 6 shows a case study; Section 7 concludes our work.

2. Preliminaries

A *Deterministic Rabin Automaton (DRA)* [11] is a 5-tuple $\mathcal{A} = (S, \Pi, \delta, s_0, Acc)$ where:

- S is a finite set of states,
- Π is an alphabet,
- $\delta : S \times \Pi \to S$ is a transition function,
- s_0 is an initial state,
- $Acc \subseteq 2^S \times 2^S$. An element $(E_i, F_i) \in Acc$, with $E_i, F_i \subseteq S$, is called a *Rabin Pair*.

A DRA $\mathcal{A}$ reads an infinite string or *word* over alphabet Π as an input and transitions from state to state according to δ. The resulting sequence of states or *run* is an *accepting* run if some states of F_i are visited infinitely often and all states of E_i are visited finitely often for some i. A word is said to be accepted by $\mathcal{A}$ if it produces an accepting run in $\mathcal{A}$. We call a set of words a *property*. The property *accepted* by $\mathcal{A}$ is the set of all words accepted by $\mathcal{A}$.

A property over an alphabet Π is *ω-regular* if and only if it is accepted by a Rabin Automaton with alphabet Π (for more detailed definitions of ω-regular properties, see [11, Section 4.3.1]). In particular, all properties defined by a *Linear Temporal Logic* (LTL) formula are ω-regular. See [11] for a detailed description of the syntax and semantics of LTL.

3. Problem Formulation

We first consider the discrete-time, continuous-state stochastic system

$$x[k+1] = \mathcal{F}_a(x[k], w_a[k]) \tag{1}$$

where $x[k] \in D \subset \mathbb{R}^n$ is the state of the system at time k, $a \in A$ where A is a finite set of *modes*, $w_a[k] \in W_a \subset \mathbb{R}^{p_a}$ is a random disturbance (which could be mode-dependent), $\mathcal{F}_a : D \times W_a \to D$ is a continuous map. Let $L : D \to 2^{\Sigma}$ be a labeling function, where Σ is a finite alphabet, 2^{Σ} its power set, and such that, for all $\sigma \in \Sigma$, the subset $D_\sigma \subseteq D$ defined by $D_\sigma = \{x \in D :$

$\sigma \in L(x)\}$ can be written as a finite union of subsets of D, that is, $D_\sigma = \cup_{i=1}^{N} J_i$, $J_i \subseteq D$, $n \in \mathbb{N}$. In Section 5, we extend this setup to allow for an infinite set of modes, *i.e.*, a control input selected from a continuous set of inputs. An infinite random path $x[0]x[1]x[2]\ldots$ satisfying (1) generates the word $L(x[0])L(x[1])L(x[2])\ldots$ over 2^{Σ}. At each time-step k, a mode $a \in A$ is chosen and the random disturbance $w_a[k]$ is sampled from a probability distribution with probability density function $f_{w_a} : \mathbb{R}^{p_a} \to \mathbb{R}_{\geq 0}$ satisfying $f_{w_a}(z) = 0$ if $z \notin W_a$. Then, a transition from state $x[k]$ to state $x[k+1]$ takes place according to the dynamics defined by mode a. The set of all infinite paths of (1) is denoted by $Paths$. A finite sequence of states $\pi = x[0]x[1]\ldots x[n]$ produced by (1) is called a *finite path*. The set of all finite paths of (1) is denoted by $Paths_{fin}$. A function $\mu : Paths_{fin} \to A$ assigning a mode to each finite path in (1) is called a *switching policy* and the set of all switching policies of (1) is denoted by $\mathcal{U} = \{\mu \mid \mu : Paths_{fin} \to A\}$. For simplicity, we assume that all modes of A are available at each state of D. A policy $\mu \in \mathcal{U}$ induces a unique, well-defined probability measure $Prob_\mu$ on the outcome space of infinite paths of (1) [26, Ch. 2.2].

We denote by Ψ an arbitrary ω-regular property over alphabet Σ and write as $(p_\Psi^x)_\mu$ the probability that a word generated by a random path starting in x satisfies property Ψ under policy μ (for a rigorous formalization of this probability, see, e.g., [14]). Our objective is to determine switching policies $\breve{\mu}_\Psi$ and $\widehat{\mu}_\Psi$ that respectively minimize and maximize the probability of satisfying property Ψ for any path in the system and, by extension, for any initialization to x of the system.

Problem 1: *Given a system of the form* (1)*, any initial state* $x \in D$ *and an* ω*-regular property* Ψ*, find switching policies* $\breve{\mu}_\Psi \in \mathcal{U}$ *and* $\widehat{\mu}_\Psi \in \mathcal{U}$ *that respectively minimize and maximize the probability of satisfying* Ψ *from* x*,* i.e.,

$$\breve{\mu}_\Psi = \arg\min_{\mu \in \mathcal{U}} (p_\Psi^x)_\mu \quad , \quad \widehat{\mu}_\Psi = \arg\max_{\mu \in \mathcal{U}} (p_\Psi^x)_\mu \; .$$

For complex specifications and dynamics, devising these exact optimal policies is likely to be intractable or infeasible due to the uncountably infinite number of states of the system's domain. To determine a policy which is close to optimal, we consider an abstraction-based approach that consists in partitioning D into a finite collection of states P to construct a finite abstraction of the stochastic dynamics.

Definition 1 (Partition)**.** *A partition P of a domain $D \subset \mathbb{R}^n$ is a collection of discrete states $P = \{Q_j\}_{j=1}^m$, $Q_j \subset D$, satisfying*

- $\bigcup_{j=1}^m Q_j = D$,
- $\boldsymbol{int}(Q_j) \cap \boldsymbol{int}(Q_\ell) = \emptyset \;\; \forall j, \ell, \; j \neq \ell$,

*where **int** denotes the interior. For any continuous state x belonging to a state Q_j, we write $x \in Q_j$.*

For a partition P of the domain D of (1), the likelihood of transitioning from a state Q_j of P to another state Q_ℓ generally varies with the continuous state abstracted by Q_j from which the transition is actually taking place. Therefore, we cannot use partition P to exactly abstract the system into a standard finite-mode Markovian model, such as an MDP. Instead, we propose producing a *BMDP abstraction* of the system where, for each action of the BMDP abstracting the behavior of (1) under some mode, the transition probabilities between states are constrained within some bounds, as depicted in Figure 1.

Definition 2 (Bounded-parameter Markov Decision Process)**.** *A* Bounded-parameter Markov Decision Process (BMDP) *[17] is a 6-tuple $\mathcal{B} = (Q, Act, \check{T}, \widehat{T}, q_0, \Sigma, L)$ where:*

- *Q is a finite set of states,*

- *Act is a finite set of actions, and the set of actions available at state $Q_j \in Q$ is denoted by $A(Q_j) \subseteq Act$,*
- *$\breve{T} : Q \times Act \times Q \to [0,1]$ maps pairs of states and an action to a lower transition bound so that $\breve{T}_{Q_j \xrightarrow{a} Q_\ell} := \breve{T}(Q_j, a, Q_\ell)$ denotes the lower bound of the transition probability from state Q_j to state Q_ℓ under action $a \in A(Q_j)$, and*
- *$\widehat{T} : Q \times Act \times Q \to [0,1]$ maps pairs of states and an action to an upper transition bound so that $\widehat{T}_{Q_j \xrightarrow{a} Q_\ell} := \widehat{T}(Q_j, a, Q_\ell)$ denotes the upper bound of the transition probability from state Q_j to state Q_ℓ under action $a \in A(Q_j)$,*
- *$q_0 \subseteq Q$ is a set of initial states,*
- *Σ is a finite set of atomic propositions,*
- *$L : Q \to 2^\Sigma$ is a labeling function from states to the power set of Σ,*

and $\breve{T}$ and $\widehat{T}$ satisfy $\breve{T}(Q_j, a, Q_\ell) \leq \widehat{T}(Q_j, a, Q_\ell)$ for all $Q_j, Q_\ell \in Q$, all $a \in A(Q_j)$, and

$$\sum_{Q_\ell \in Q} \breve{T}(Q_j, a, Q_\ell) \leq 1 \leq \sum_{Q_\ell \in Q} \widehat{T}(Q_j, a, Q_\ell)$$

for all $Q_j \in Q$ and all $a \in A(Q_j)$.

Definition 3 (BMDP Abstraction)**.** *Given the system* (1) *evolving on a domain $D \subset \mathbb{R}^n$ and a partition $P = \{Q_j\}_{j=1}^m$ of D, a BMDP $\mathcal{B} = (Q, Act, \breve{T}, \widehat{T}, q_0, \Sigma, L)$ is an* abstraction *of* (1) *if:*

- *$Q := P$, that is, the set of states of the BMDP is the partition P,*
- *$Act := A$, that is, the set of actions of the BMDP are the modes of* (1)*,*
- *For all $Q_j, Q_\ell \in P$ and action $a \in Act$,*

$$\breve{T}_{Q_j \xrightarrow{a} Q_\ell} \leq \inf_{x \in Q_j} Pr(\mathcal{F}_a(x, w_a) \in Q_\ell), \text{ and}$$
$$\widehat{T}_{Q_j \xrightarrow{a} Q_\ell} \geq \sup_{x \in Q_j} Pr(\mathcal{F}_a(x, w_a) \in Q_\ell),$$

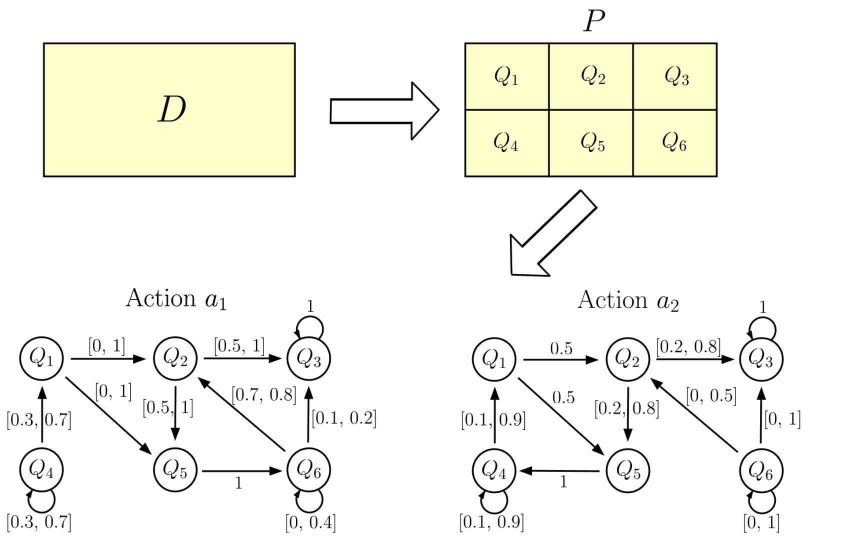


Figure 1: A finite-state BMDP abstraction $\mathcal{B}$ of system (1) with domain D. A partition P of D is generated and bounds on the transition probabilities between states are estimated for two actions a_1 and a_2 of $\mathcal{B}$.

where $Pr(\mathcal{F}_a(x, w_a) \in Q_\ell)$ *for fixed* x *denotes the probability that* (1) *transitions from* x *to some state* $x' = \mathcal{F}_a(x, w_a)$ *in* Q_ℓ *under mode* a*,*

- $P = q_0$*, i.e., the set of initial states of the BMDP is the partition* P*,*
- *For all* $Q_j \in P$ *and for any two states* $x_i, x_\ell \in Q_j$*, it holds that* $L(Q_j) := L(x_i) = L(x_\ell)$*, that is, the partition conforms to the boundaries induced by the labeling function.*

For a given action, two continuous states belonging to the same discrete state of a BMDP abstraction $\mathcal{B}$ may, in general, give rise to different transition probabilities. This fact is encoded in $\mathcal{B}$ by the upper and lower transition probabilities.

In this paper, we do not present algorithms for computing BMDP abstraction of (1), which typically rely on overapproximating reachable sets; see [27] for such an approach. Thus, we assume that BMDP abstractions are available given a partition P of D for (1). However, we will focus on the problem of refining P in order to obtain better BMDP abstractions.

Furthermore, we make the assumption that any state in Q of a BMDP can serve as an initial state. From a given initial state, a BMDP $\mathcal{B}$ generates a finite sequence of states $\pi = q_0 \dots q_k$ called a finite path by selecting an action in Act at each time step and non-deterministically resolving a probability distribution over the transition probabilities which is compatible with the bounds imposed by the chosen action and sampled to determine the state of $\mathcal{B}$ at the next time step. Denoting the set of all finite paths of a BMDP $\mathcal{B}$ by $(Paths_{fin})_{\mathcal{B}}$, a switching policy $\mu : (Paths_{fin})_{\mathcal{B}} \rightarrow Act$ for $\mathcal{B}$ is a function assigning an action to all finite paths in $\mathcal{B}$. The set of all switching policies of $\mathcal{B}$ is denoted by $\mathcal{U}_{\mathcal{B}} = \{\mu \mid \mu : (Paths_{fin})_{\mathcal{B}} \rightarrow Act\}$. Under a switching policy μ, the available actions in BMDP $\mathcal{B}$ reduce to a single possibility at each time step, namely, that prescribed by the switching policy μ, inducing a (possibly countably infinite-state) *Interval-valued Markov Chain* (IMC), defined formally next. As will be discussed further, only finite-memory policies need to be considered in this work, which induce finite-state IMCs.

Definition 4 (Interval-valued Markov Chain)**.** *An* Interval-valued Markov Chain *(IMC)* $\mathcal{I} = (Q, \check{T}, \widehat{T}, q_0, \Sigma, L)$ *is defined similarly to a BMDP with the difference that a single action (which is omitted in the defining tuple) is available.*

The IMC induced by policy μ in BMDP $\mathcal{B}$ is denoted by $\mathcal{B}[\mu]$.

The state of an IMC $\mathcal{I}$ evolves as follows: at each time step k, the environment non-deterministically chooses a transition matrix T_k compatible with the transition bound functions $\check{T}$ and $\widehat{T}$ of $\mathcal{I}$, that is, T_k is a $|Q| \times |Q|$ row stochastic matrix such that the entries satisfy the lower and upper bounds prescribed by $\check{T}$ and $\widehat{T}$, and the next transition occurs according to T_k [28] [1]. A mapping ν

[1]This is the Interval Markov Decision Process interpretation of IMCs.

from a finite path $\pi = q_0 \dots q_k$ in $\mathcal{I}$ to a transition matrix T_k is called an *adversary*. The set of all adversaries of $\mathcal{I}$ is denoted by $\nu_{\mathcal{I}}$. A unique probability measure $Prob_{\nu}$ is induced over the set of all infinite paths $Paths_{\mathcal{I}}$ of IMC $\mathcal{I}$ under adversary $\nu \in \nu_{\mathcal{I}}$ [11, Def. 10.10]. By extension, a probability measure $Prob_{\mu,\nu}$ is induced over the set of all infinite paths $Paths_{\mathcal{B}}$ of BMDP $\mathcal{B}$ under policy μ and adversary $\nu \in \nu_{\mathcal{B}[\mu]}$.

The probability of satisfying ω-regular property Ψ starting from initial state Q_j in IMC $\mathcal{I}$ under adversary ν is denoted by $\mathcal{P}_{\mathcal{I}[\nu]}(Q_j \models \Psi)$. The greatest lower bound and least upper bound on the probability of satisfying property Ψ starting from initial state Q_j in IMC $\mathcal{I}$ are denoted by $\check{\mathcal{P}}_{\mathcal{I}}(Q_j \models \Psi) = \inf_{\nu \in \nu_{\mathcal{I}}} \mathcal{P}_{\mathcal{I}[\nu]}(Q_j \models \Psi)$ and $\widehat{\mathcal{P}}_{\mathcal{I}}(Q_j \models \Psi) = \sup_{\nu \in \nu_{\mathcal{I}}} \mathcal{P}_{\mathcal{I}[\nu]}(Q_j \models \Psi)$ respectively. When these bounds are the same for all states in a set of states C of $\mathcal{I}$, we write $\check{\mathcal{P}}_{\mathcal{I}}(C \models \Psi)$ and $\widehat{\mathcal{P}}_{\mathcal{I}}(C \models \Psi)$.

To design switching policies in BMDPs, it is crucial to note that a BMDP $\mathcal{B}$ subject to a switching policy μ reduces to an IMC $\mathcal{B}[\mu]$; therefore, finding the probability of satisfying a specification Ψ from some initial state of $\mathcal{B}[\mu]$ amounts to solving a verification problem on an IMC. As discussed above, the probability of satisfying a specification Ψ in an IMC is not uniquely defined and depends on the instantiation of a non-deterministic adversary. Consequently, the verification of the IMC $\mathcal{B}[\mu]$ induced by a policy μ in a BMDP $\mathcal{B}$ does not compute, in general, a fixed probability but an interval of satisfaction probabilities $(I_j)_{\mu} = [(p^j_{min})_{\mu}, (p^j_{max})_{\mu}]$ for all initial states Q_j of $\mathcal{B}[\mu]$. The meaning of this interval is that the probability of fulfilling Ψ from state Q_j in $\mathcal{B}[\mu]$ is contained in $(I_j)_{\mu}$ for all possible adversaries of $\mathcal{B}[\mu]$, that is, $\mathcal{P}_{\mathcal{B}[\mu][\nu]}(Q_j \models \Psi) \in (I_j)_{\mu}$, $\forall \nu \in \nu_{\mathcal{B}[\mu]}$.

Because a switching policy in a BMDP returns an interval of satisfaction for all its initial states, it may not seem obvious which quantities to minimize or maximize when synthesizing policies in BMDP abstractions of continuous state systems. Note that a policy μ for a BMDP abstraction $\mathcal{B}$ of (1) maps to a policy for (1) in the natural way, *i.e.*, the control action prescribed by μ at a discrete state Q_i of $\mathcal{B}$ is applied to all continuous states $x \in Q_i$ in (1). By virtue of $\mathcal{B}$ being an abstraction of (1), it then holds that the exact probability of satisfying Ψ from any continuous initial state $x \in Q_j$ for (1) is contained within the bounds of the interval $(I_j)_\mu$ induced by policy μ for initial state Q_j in $\mathcal{B}$ [16]. Therefore, given a BMDP abstraction $\mathcal{B}$ of (1) generated from a partition P of the domain D, our approach to Problem 1 is to find policies $\widehat{\mu}_\Psi^{low}$ and $\widecheck{\mu}_\Psi^{up}$ in $\mathcal{B}$ that respectively maximize the lower bound probability (for the maximization objective) and minimize the upper bound probability (for the minimization objective) of satisfying Ψ for all initial states Q_j of $\mathcal{B}$.

Subproblem 1.1: *Given a system of the form* (1)*, a partition P of its domain D, a BMDP abstraction $\mathcal{B}$ of* (1) *arising from P, any initial state $Q_j \in Q$ of $\mathcal{B}$ and an ω-regular property Ψ, compute switching policies $\widecheck{\mu}_\Psi^{up} \in \mathcal{U}_\mathcal{B}$ and $\widehat{\mu}_\Psi^{low} \in \mathcal{U}_\mathcal{B}$ that respectively minimize the upper bound probability and maximize the lower bound probability of satisfying Ψ in $\mathcal{B}$,* i.e.,

$$\widecheck{\mu}_\Psi^{up} = \arg\min_{\mu \in \mathcal{U}_\mathcal{B}} \widehat{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) \quad , \quad \widehat{\mu}_\Psi^{low} = \arg\max_{\mu \in \mathcal{U}_\mathcal{B}} \widecheck{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) \ .$$

If $\mathcal{B}$ is a BMDP abstraction of (1), then a unique control action is assigned to all continuous states abstracted by some Q_i in $\mathcal{B}$. In this case, the quality of the policies $\widehat{\mu}_\Psi^{low}$ and $\widecheck{\mu}_\Psi^{up}$ heavily depends on the quality and fineness of the partition P of the domain D. Indeed, because these policies only accommodate the extreme behaviors of all discrete states of $\mathcal{B}$, it is reasonable to assume that the computed policies may be suboptimal for a collection of continuous states abstracted by some Q_i. In this work, we address this problem by starting with a coarse partition of the system's domain; then, we iteratively and selectively refine this partition so as to target discrete states that are at a higher risk of containing suboptimally controlled continuous states or are responsible for considerable uncertainty in the control of other states. As finer partitions result in larger abstractions to be analyzed, it is crucial to avoid performing unnecessary refinement in order to alleviate the state-space explosion phenomenon.

The procedure terminates once a precision threshold which will be defined in further sections has been reached.

Subproblem 1.2: *Given a system of the form* (1) *with a BMDP abstraction* $\mathcal{B}$ *arising from a partition* P *of the domain* D *and an* ω*-regular property* Ψ*, refine the partition* P *of* D *until the computed switching policy reaches a user-defined threshold of quality with respect to the objective of minimizing or maximizing the probability of satisfying* Ψ *in* (1)*.*

After presenting solutions to Subproblem 1.1 and 1.2 in Section 4, we next investigate stochastic systems of the form

$$x[k+1] = \mathcal{F}(x[k], u[k], w[k]) \tag{2}$$

where $x[k] \in D \subset \mathbb{R}^n$ is the state of the system at time k, $u[k] \in U$ where $U \subset \mathbb{R}^m$ is a continuous set of inputs, $w[k] \in W \subset \mathbb{R}^p$ is a random disturbance whose probability density function f_w is assumed to be independent of u, $\mathcal{F}$: $D \times U \times W \rightarrow D$ is a continuous map. Here, a *control policy* is a function $\mu : Paths_{fin} \rightarrow U$ assigning a control action to each finite path in (2). The set of all control policies of (2) is denoted by $\mathcal{U} = \{\mu \mid \mu : Paths_{fin} \rightarrow A\}$ as in the finite-mode system case.

The difficulty of establishing policies aiming to maximize or minimize the probability of satisfying a temporal property in (2) is highly dependent on the structure of the considered system. In this work, we restrict our attention to systems which are affine in input and disturbance, that is

$$x[k+1] = \mathcal{F}(x[k]) + u[k] + w[k] \ . \tag{3}$$

As in the finite-mode case, we are interested in the design of a control policy that maximizes or minimizes the probability of satisfying an ω-regular property Ψ.

Problem 2: *Given a system of the form* (3)*, any initial state* $x \in D$ *and an*

ω-regular property Ψ, find control policies $\breve{\mu}_\Psi \in \mathcal{U}$ and $\widehat{\mu}_\Psi \in \mathcal{U}$ that respectively minimize and maximize the probability of satisfying Ψ from x.

Solving this problem for an arbitrary property Ψ again involves a partition P of the domain D from which a finite-state abstraction of the system is constructed and analyzed. In this work, we introduce new abstraction tools called *Controlled Interval-valued Markov Chains* (CIMC) which differ from BMDPs in that the set of available actions is uncountably infinite. CIMCs are the abstractions of choice for systems of the form (3).

Definition 5 (Controlled Interval-valued Markov Chain)**.** *A* Controlled Interval-valued Markov Chain (CIMC) *is a 6-tuple $\mathcal{C} = (Q, U, \breve{T}, \widehat{T}, q_0, \Sigma, L)$ defined similarly to a BMDP with the difference that a continuous set of inputs $U \subseteq \mathbb{R}^m$ replaces the finite set of actions Act.*

Definition 6 (Controlled Interval-valued Markov Chain Abstraction)**.** *Given the system* (3) *evolving on a domain $D \subset \mathbb{R}^n$ and a partition $P = \{Q_j\}_{j=1}^m$ of D, a CIMC $\mathcal{C} = (Q, U, \breve{T}, \widehat{T}, q_0, \Sigma, L)$ is an* abstraction *of* (3) *if it satisfies the same conditions as a BMDP abstraction with the difference that a continuous set of inputs $U \subseteq \mathbb{R}^m$ replaces the finite set of actions Act.*

Denoting the set of all finite paths in a CIMC $\mathcal{C}$ by $(Paths_{fin})_\mathcal{C}$, a control policy $\mu : (Paths_{fin})_\mathcal{C} \rightarrow U$ for $\mathcal{C}$ is a function assigning an input to all finite paths in $\mathcal{C}$. The set of all control policies of $\mathcal{C}$ is denoted by $\mathcal{U}_\mathcal{C} = \{\mu \mid \mu : (Paths_{fin})_\mathcal{C} \rightarrow U\}$. A policy μ applied to a CIMC $\mathcal{C}$ induces an IMC denoted by $\mathcal{C}[\mu]$.

For all possible finite paths in $\mathcal{C}$, the goal is to find the input in the uncountable set U that yields the most favorable IMC abstraction with respect to the desired objective. Note that, unlike in a BMDP abstraction, this problem offers an infinite set of available inputs to select from.

Subproblem 2.1: *Given a system of the form* (3)*, a partition P of its domain D, a CIMC abstraction $\mathcal{C}$ of* (3) *arising from P, any initial state $Q_j \in Q$ of $\mathcal{C}$ and an ω-regular property Ψ, compute the control policies $\breve{\mu}_{\Psi}^{up} \in \mathcal{U}_{\mathcal{C}}$ and $\widehat{\mu}_{\Psi}^{low} \in \mathcal{U}_{\mathcal{C}}$ that respectively minimize the upper bound probability and maximize the lower bound probability of satisfying Ψ in $\mathcal{C}$,* i.e.,

$$\breve{\mu}_{\Psi}^{up} = \arg\min_{\mu \in \mathcal{U}_{\mathcal{C}}} \widehat{\mathcal{P}}_{\mathcal{C}[\mu]}(Q_j \models \Psi) \quad , \quad \widehat{\mu}_{\Psi}^{low} = \arg\max_{\mu \in \mathcal{U}_{\mathcal{C}}} \breve{\mathcal{P}}_{\mathcal{C}[\mu]}(Q_j \models \Psi) \ .$$

As our approach again relies on finite-state abstractions, finer partitions of the domain D generally yield higher-quality control policies. Therefore, partition refinement for this case is discussed as well.

Subproblem 2.2: *Given a system of the form* (3) *with a CIMC abstraction $\mathcal{C}$ arising from a partition P of the domain D and an ω-regular property Ψ, refine the partition P of D until the computed control policy reaches a user-defined threshold of quality with respect to the objective of minimizing or maximizing the probability of satisfying Ψ in* (3)*.*

In the next section, we comprehensively detail our solution to the synthesis of switching policies for finite mode systems as formalized in Problem 1. Specifically, Subsections 4.1 and 4.2 focus on the computation of controllers for BMDP abstractions as stated in Subproblem 1.1, whereas Subsection 4.3 is concerned with Subproblem 1.2 and the refinement of BMDP abstractions for the synthesis of improved policies with respect to the abstracted system.

4. CONTROLLER SYNTHESIS FOR FINITE MODE SYSTEMS

4.1. BMDP CONTROLLER SYNTHESIS

In this subsection, we present the theory for addressing Subproblem 1.1. We adopt an automaton-based approach for computing maximizing and minimizing switching policies in a BMDP $\mathcal{B}$ with respect to an ω-regular property Ψ. As discussed in Section 2, for every such property, there exists a corresponding DRA representation $\mathcal{A}$. Similar to [11, page 798] and [24] where the Cartesian product with a Markov Chain (MC) and an IMC are introduced, we define the product $\mathcal{B} \otimes \mathcal{A}$ between a BMDP and a DRA.

Definition 7 (Product Bounded-Parameter Markov Decision Process). *Let* $\mathcal{B} = (Q, Act, \check{T}, \widehat{T}, q_0, \Sigma, L)$ *be a BMDP and* $\mathcal{A} = (S, 2^{\Sigma}, \delta, s_0, Acc)$ *be a DRA. The* product $\mathcal{B} \otimes \mathcal{A} = (Q \times S, Act, \breve{T'}, \widehat{T'}, q_0^{\otimes}, Acc', L')$ *is a BMDP where:*

- $Q \times S$ *is a set of states,*
- *Act is the same set of actions of* $\mathcal{B}$*, where* $A(\langle Q_j, s_i \rangle) = A(Q_j)$ *for all* $Q_j \in Q$ *and for all* $s_i \in S$*,*
- $\breve{T'}_{\langle Q_j, s \rangle \xrightarrow{a} \langle Q_\ell, s' \rangle} = \begin{cases} \check{T}_{Q_j \xrightarrow{a} Q_\ell}, & \textit{if } s' = \delta(s, L(Q_\ell)) \\ 0, & \textit{otherwise} \end{cases}$
- $\widehat{T'}_{\langle Q_j, s \rangle \xrightarrow{a} \langle Q_\ell, s' \rangle} = \begin{cases} \widehat{T}_{Q_j \xrightarrow{a} Q_\ell}, & \textit{if } s' = \delta(s, L(Q_\ell)) \\ 0, & \textit{otherwise} \end{cases}$
- $q_0^{\otimes} = \{(Q_j, s_0) : Q_j \in Q\}$ *is a finite set of initial states,*
- $Acc' = \{E_1, E_2, \ldots, E_k, F_1, F_2, \ldots, F_k\}$ *is a set of atomic propositions, where* E_i *and* F_i *are the sets in the Rabin pairs of Acc,*
- $L' : Q \times S \to 2^{Acc'}$ *such that, for all atomic proposition* $H \in Acc'$*, for all* $Q_j \in Q$ *and for all* $s_i \in S$*,* $H \in L'(\langle Q_j, s_i \rangle)$ *if and only if* s_i *belongs to the set in the Rabin pairs of Acc corresponding to* H*.*

In this product construction, the DRA $\mathcal{A}$ is used as a finite-memory instrument that monitors all transitions occurring in $\mathcal{B}$ and assesses whether the resulting path satisfies Ψ. Indeed, any random path $\pi = q_0 q_1 \ldots$ in $\mathcal{B}$ generates a unique path $\pi_{\otimes}^{\mathcal{A}} = \langle q_0, s_0 \rangle \langle q_1, s_j \rangle \ldots$ in $\mathcal{B} \otimes \mathcal{A}$ which depends on the labels of the states of $\mathcal{B}$ as per Definition 7. It follows that a switching policy in $\mathcal{B}$ can be induced by inspecting the sequences of states generated in $\mathcal{B} \otimes \mathcal{A}$ and choosing control actions accordingly.

Definition 8 (Generated Path in Product BMDP)**.** *Consider a BMDP $\mathcal{B}$ with set of states Q and labeling function L and a DRA $\mathcal{A}$ with set of states S and transition function δ. A path $\pi_{\otimes}^{\mathcal{A}} = \langle q_0, s'_0 \rangle, \langle q_1, s'_1 \rangle \ldots,\ q_i \in Q,\ s'_i \in S$, in the product BMDP $\mathcal{B} \otimes \mathcal{A}$ is said to be* generated *by the path $\pi = q_0, q_1 \ldots$ in $\mathcal{B}$ if it holds that $s'_{i+1} = \delta(s'_i, L(q_{i+1})), \forall i = 0, 1, 2, \ldots$.*

Definition 9 (Induced Switching Policy)**.** *Consider a BMDP $\mathcal{B}$, a DRA $\mathcal{A}$ and a switching policy $\mu \in \mathcal{U}_{\mathcal{B}}$. Let $\pi \in (Paths_{fin})_{\mathcal{B}}$ be any finite path in $\mathcal{B}$. We denote by $\pi_{\otimes}^{\mathcal{A}}$ the path generated by π in the product BMDP $\mathcal{B} \otimes \mathcal{A}$. The switching policy μ is said to be* induced *by a switching policy $\mu_{\otimes}$ of $\mathcal{B} \otimes \mathcal{A}$ if, for all $\pi \in (Paths_{fin})_{\mathcal{B}}$, it holds that $\mu(\pi) = \mu_{\otimes}(\pi_{\otimes}^{\mathcal{A}})$.*

For a fixed switching policy μ of $\mathcal{B}$, the probability of satisfying Ψ in the induced IMC $\mathcal{B}[\mu]$ is equal to the probability of reaching a so-called Accepting *Bottom Strongly Connected Component* (BSCC) in the product IMC $\mathcal{B}[\mu] \otimes \mathcal{A}$ [24] defined below. The probability of reaching an accepting BSCC in $\mathcal{B}[\mu] \otimes \mathcal{A}$ is not uniquely defined and depends on the assumed transition values within the probability intervals selected by a non-deterministic adversary $\nu \in \nu_{\mathcal{B}[\mu] \otimes \mathcal{A}}$ which induces a product MC $\mathcal{B}[\mu][\nu]_{\otimes}^{\mathcal{A}}$.

Definition 10 (Product Interval-valued Markov Chain)**.** *Let $\mathcal{I} = (Q, \check{T}, \widehat{T}, q_0, \Sigma, L)$ be an IMC and $\mathcal{A} = (S, 2^{\Sigma}, \delta, s_0, Acc)$ be a DRA. The* product *$\mathcal{I} \otimes \mathcal{A} = (Q, \widecheck{T'}, \widehat{T'}, q_0^{\otimes}, Acc', L')$ is an IMC defined similarly to a product BMDP with* *the difference that a single action (which is omitted in the defining tuple) is available.*

Definition 11 (Markov Chain)**.** *A Markov Chain (MC) $\mathcal{M} = (Q, T, q_0, \Sigma, L)$ is defined similarly to an IMC with the difference that the transition probability function or transition matrix of the Markov Chain $T : Q \times Q \to [0, 1]$ satisfies $0 \leq T(Q_j, Q_\ell) \leq 1$ for all $Q_j, Q_\ell \in Q$ and $\sum_{Q_\ell \in Q} T(Q_j, Q_\ell) = 1$ for all $Q_j \in Q$.*

The probability of satisfying property Ψ in Markov Chain $\mathcal{M}$ from initial state Q_j is denoted by $P_{\mathcal{M}}(Q_j \models \Psi)$.

Definition 12 (Induced Product Markov Chain)**.** *A Product Markov Chain $\mathcal{I}[\nu]^{\mathcal{A}}_{\otimes} = (Q \times S, T, q_0^{\otimes}, Acc', L')$ is said to be* induced *by an adversary ν of a product IMC $\mathcal{I} \otimes \mathcal{A}$ if they share the same Q (for memoryless policies μ), $\mathcal{A}$, $q_0^{\otimes}$, L' and Acc', and for all q_j, $q_\ell \in Q \times S$ and all action $a = \mu(q_j)$, the transition probability function T satisfies $\breve{T}_{q_j \xrightarrow{a} q_\ell} \leq T(q_j, q_\ell) \leq \widehat{T}_{q_j \xrightarrow{a} q_\ell}$.*

Definition 13 (Bottom Strongly Connected Component)**.** *Given a Markov Chain $\mathcal{M}$ with states Q, a subset $B \subseteq Q$ is called a* Bottom Strongly Connected Component *(BSCC) of $\mathcal{M}$ if*

- *B is strongly connected: for each pair of states (q, t) in B, there exists a path $q_0 q_1 \ldots q_n$ such that $T(q_i, q_{i+1}) > 0$, $i = 0, 1, \ldots, n-1$, and $q_i \in B$ for $0 \leq i \leq n$ with $q_0 = q$, $q_n = t$,*
- *no proper superset of B is strongly connected,*
- *$\forall s \in B$, $\Sigma_{t \in B} T(s, t) = 1$.*

In words, every state in a BSCC B is reachable from any state in B, and every state in B only transitions to another state in B.

Definition 14 (Accepting and Non-Accepting Bottom Strongly Connected Component)**.** *A Bottom Strongly Connected Component B of a product Markov Chain $\mathcal{M}_{\otimes}^{\mathcal{A}}$ is said to be* accepting *if:*

$$\exists i : \left(\exists \langle Q_j, s_\ell \rangle \in B \ : \ F_i \in L'(\langle Q_j, s_\ell \rangle) \right) \wedge \left(\forall \langle Q_j, s_\ell \rangle \in B \ : \ E_i \notin L'(\langle Q_j, s_\ell \rangle) \right).$$

$\mathcal{M}_{\otimes}^{\mathcal{A}}$ *is said to be* non-accepting *if it is not accepting.*

A key observation is that, for any policy μ in $\mathcal{B}$ induced by a policy $\mu_{\otimes}$ in the product $\mathcal{B} \otimes \mathcal{A}$, the bounds on the probability of reaching an accepting BSCC from the initial states of $\mathcal{B}[\mu] \otimes \mathcal{A}$ are identical to the bounds on the probability of reaching an accepting BSCC from the initial states of $(\mathcal{B} \otimes \mathcal{A})[\mu_{\otimes}]$ according to Definitions 7 and 10 which ensure that the elements in the defining tuples of $\mathcal{B}[\mu] \otimes \mathcal{A}$ and $(\mathcal{B} \otimes \mathcal{A})[\mu_{\otimes}]$ are the same. Consequently, an analysis of the product $\mathcal{B} \otimes \mathcal{A}$ is sufficient for approaching the synthesis problem.

Because ω-regular properties are closed under complementation, the problem of minimizing the upper bound probability of satisfying property Ψ in $\mathcal{B}$ can be converted to the problem of maximizing the lower bound probability of satisfying the complement property $\overline{\Psi}$ with corresponding DRA $\overline{\mathcal{A}}$. It follows that Subproblem 1.1 is solved by applying the same tools to both $\mathcal{B} \otimes \mathcal{A}$ and $\mathcal{B} \otimes \overline{\mathcal{A}}$.

Fact 1. *Let $\mathcal{B}$ be a BMDP and Ψ be an ω-regular specification. We denote by $\overline{\Psi}$ the complement of property Ψ. For any initial state $Q_j \in Q$ of $\mathcal{B}$ and policy $\mu \in \mathcal{U}_{\mathcal{B}}$, it holds that*

$$\widehat{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) = 1 - \widecheck{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \overline{\Psi})$$
$$\widecheck{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) = 1 - \widehat{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \overline{\Psi}) \ .$$

Therefore, our objective consists in computing a policy that maximizes the lower bound probability of reaching an accepting BSCC from all initial states of the resulting product IMC $\mathcal{B}[\mu] \otimes \mathcal{A}$. We introduce the class of *memoryless* policies, which solely depend on the current state of the BMDP and will further prove optimal for our problem in the product BMDP $\mathcal{B} \otimes \mathcal{A}$. Additionally, we introduce the class of memoryless adversaries of an IMC, which will prove sufficient for achieving the extreme probabilities of satisfying the specification as established in Lemma 1 and 2, and Theorem 1 and 2.

Definition 15 (Memoryless Policy)**.** *A policy $\mu \in \mathcal{U}_{\mathcal{B}}$ of a BMDP $\mathcal{B}$ is said to be* memoryless *if, for all finite paths $\pi = q[0]q[1]\ldots q[k]$ of $\mathcal{B}$, it holds that $\mu(\pi) = \mu(q[k])$, where $q[k]$ is interpreted as a finite path of length 1 in the expression $\mu(q[k])$. That is, a policy μ is memoryless if and only if for every distinct pair of finite paths π and π' ending in the same state q, we have $\mu(\pi) = \mu(\pi')$.*

Definition 16 (Memoryless Adversary)**.** *An adversary $\nu \in \mathcal{I}_{\nu}$ of an IMC $\mathcal{I}$ is said to be* memoryless *if, for all finite paths $\pi = q[0]q[1]\ldots q[k]$ of $\mathcal{I}$, it holds that $\nu(\pi) = \nu(q[k])$, where $q[k]$ is interpreted as a finite path of length 1 in the expression $\nu(q[k])$. That is, an adversary ν is memoryless if and only if for every distinct pair of finite paths π and π' ending in the same state q, we have $\nu(\pi) = \nu(\pi')$.*

Before presenting a solution to Subproblem 1.1, we first recall some basic results established in [24] for the purpose of verification in IMCs which we then extend to compute switching policies in BMDPs.

For a given policy μ of $\mathcal{B}$ and automaton $\mathcal{A}$, the sets of accepting and non-accepting BSCCs of the resulting product IMC $\mathcal{B}[\mu] \otimes \mathcal{A}$ depend on the assumed probability values for the transitions with zero lower bound and non-zero upper bound. Specifically, whether a zero or a non-zero value is assigned to these transitions directly affects the qualitative structure of the product IMC, and therefore its sets of accepting and non-accepting BSCCs, as a zero probability implies that a transition can never occur between the corresponding states, while a non-zero probability indicates that a transition is possible. When a non-zero probability is assumed for such a transition, we describe the transition as being "on", and we say that the transition is "off" in the scenario that a probability of zero is assumed. Nonetheless, it is shown in [24] that, for any product IMC, there exists a *largest winning component* and a *largest losing component* which can be created among all combinations of "on" and "off" transitions allowed by the transition bound functions of the product IMC. A winning component of a product MC is a set of states that reach an accepting BSCC with probability 1, while a losing component is a set of states that reach a non-accepting BSCC with probability 1.

Definition 17 (Winning Component)**.** *[24] A* winning component *WC of a product MC $\mathcal{M}_{\otimes}^{\mathcal{A}}$ is a set of states satisfying $\mathcal{P}(WC \models \Diamond R) = 1$, where R is the set of states belonging to an accepting BSCC in $\mathcal{M}_{\otimes}^{\mathcal{A}}$.*

Definition 18 (Losing Component)**.** *[24] A* losing component *LC of a product MC $\mathcal{M}_{\otimes}^{\mathcal{A}}$ is a set of states satisfying $\mathcal{P}(LC \models \Diamond R) = 1$, where R is the set of states belonging to a non-accepting BSCC in $\mathcal{M}_{\otimes}^{\mathcal{A}}$.*

Definition 19 (Largest Winning/Losing Components)**.** *[24] A state $\langle Q_j, s_i\rangle \in Q \times S$ of a product IMC $\mathcal{I}$ is a member of the* Largest Winning *(respectively,* Losing*)* Component *$(WC)_L$ $\big($respectively, $(LC)_L\big)$ if there exists a product MC induced by $\mathcal{I}$ such that $\langle Q_j, s_i\rangle$ is a winning (respectively, losing) component.*

Moreover, it was shown in [24] that the upper bound probability of satisfying Ψ in the IMC $\mathcal{I}$ from state Q_j is equal to the upper bound probability of reaching the largest winning component $(WC)_L$ of the product $\mathcal{I} \otimes \mathcal{A}$ from state $\langle Q_j, s_0\rangle$. Likewise, the lower bound probability of satisfying Ψ is found by solving a reachability problem on the largest losing component $(LC)_L$. These results naturally apply to product IMCs $\mathcal{B}[\mu]\otimes\mathcal{A}$ constructed from an IMC $\mathcal{B}[\mu]$ induced by a policy μ of a BMDP $\mathcal{B}$.

Fact 2 ([24]). *Let $\mathcal{B}[\mu]$ be an IMC induced by a switching policy μ of a BMDP $\mathcal{B}$ and $\mathcal{A}$ be a DRA corresponding to the ω-regular property Ψ. Let $(WC)_L$ and $(LC)_L$ be the largest winning and losing components of $\mathcal{B}[\mu]\otimes\mathcal{A}$ respectively. It holds that, for all initial states Q_j of $\mathcal{B}[\mu]$,*

$$\widehat{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) = \widehat{\mathcal{P}}_{\mathcal{B}[\mu]\otimes\mathcal{A}}(\langle Q_j, s_0\rangle \models \Diamond(WC)_L)$$
$$\check{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) = 1 - \widehat{\mathcal{P}}_{\mathcal{B}[\mu]\otimes\mathcal{A}}(\langle Q_j, s_0\rangle \models \Diamond(LC)_L)\ .$$

The intuitive interpretation of this property is that any IMC $\mathcal{B}[\mu]$ has a "best-case" adversary and a "worst-case" adversary in the product $\mathcal{B}[\mu]\otimes\mathcal{A}$ that respectively maximizes and minimizes the probability of reaching an accepting BSCC for all initial states of $\mathcal{B}[\mu]\otimes\mathcal{A}$ simultaneously, since reachability probabilities are maximized by memoryless adversaries. These probabilities are equal to the upper and lower bound probabilities of satisfying Ψ from the initial states of $\mathcal{B}[\mu]$. In an induced product MC corresponding to the best-case scenario, the set of winning components is as large as it can possibly be; in an induced product MC corresponding to the worst-case scenario, the set of winning components is reduced to the smallest possible set of *permanent* winning components.

Definition 20 (Permanent Winning Component). *[24] A state $\langle Q_j, s_i\rangle \in Q\times S$ of a product IMC $\mathcal{I}\otimes\mathcal{A}$ is a member of the* Permanent Winning Component *$(WC)_P$ of $\mathcal{I}\otimes\mathcal{A}$ if $\langle Q_j, s_i\rangle$ is a winning component for all product MCs induced by $\mathcal{I}\otimes\mathcal{A}$.*

We further introduce the notions of permanent accepting BSCC, which is a subset of the permanent winning components of a product IMC. These sets, which are unique for a given product IMC, will prove useful in subsequent sections.

Definition 21 (Permanent Accepting Bottom Strongly Connected Component)**.** *[24] A state $\langle Q_j, s_i\rangle \in Q \times S$ of a product IMC $\mathcal{I} \otimes \mathcal{A}$ is a member of the* Permanent Accepting BSCC $(U^A)_P$ *of $\mathcal{I} \otimes \mathcal{A}$ if $\langle Q_j, s_i\rangle$ belongs to an accepting BSCC for all product MCs induced by $\mathcal{I} \otimes \mathcal{A}$.*

Recall our primary objective which is to find switching policies $\widetilde{\mu}_{\Psi}^{up}$ and $\widehat{\mu}_{\Psi}^{low}$ that respectively minimize the upper bound probability and maximize the lower bound probability of satisfying property Ψ from initial state Q_j in a BMDP $\mathcal{B}$. In light of the above facts, this amounts to enforcing the best possible worst-case scenario with respect to the probability of reaching an accepting BSCC in the product $\mathcal{B} \otimes \mathcal{A}$ for the maximization case, or in the product $\mathcal{B} \otimes \overline{\mathcal{A}}$ for the minimization case. To this end, we first state in the following lemma that there exist sets of memoryless switching policies of $\mathcal{B} \otimes \mathcal{A}$ resulting in the greatest possible set of permanent winning components in the corresponding induced product IMCs.

Lemma 1. *Let $\mathcal{B}$ be a BMDP and Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$. The set of policies of the product $\mathcal{B} \otimes \mathcal{A}$ is denoted by $\mathcal{U}_{\otimes}^{\mathcal{A}}$ and the set of memoryless policies of the product $\mathcal{B} \otimes \mathcal{A}$ is denoted by $(\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem} \subseteq \mathcal{U}_{\otimes}^{\mathcal{A}}$. There exists a set of memoryless switching policies $\mathcal{U}_{(WC)_P^G} \subseteq (\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$ generating the set $(WC)_P^G$ in $\mathcal{B} \otimes \mathcal{A}$ such that, for all $\mu \in \mathcal{U}_{\otimes}^{\mathcal{A}}$, $(WC)_P \subseteq (WC)_P^G$ where $(WC)_P$ is the permanent winning component of $(\mathcal{B} \otimes \mathcal{A})[\mu]$, and, for all $\mu \in \mathcal{U}_{(WC)_P^G}$, the permanent winning component of $(\mathcal{B} \otimes \mathcal{A})[\mu]$ is $(WC)_P^G$.*

A constructive proof of this lemma is provided in the Appendix. The set $(WC)_P^G$

is called the *Greatest Permanent Winning Component* of the product BMDP $\mathcal{B} \otimes \mathcal{A}$.

From Lemma 1, we infer in the following theorem that a maximizing policy with respect to Ψ in BMDP $\mathcal{B}$ is induced by a policy $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ in the product BMDP $\mathcal{B} \otimes \mathcal{A}$ that effectively generates the set $(WC)_P^G$ and, for all states not in $(WC)_P^G$, maximizes the lower bound probability of reaching this set; on the other hand, a minimizing policy with respect to Ψ in $\mathcal{B}$ is induced by a policy $(\breve{\mu}_{\Psi}^{up})_{\otimes}$ achieving the same thing in $\mathcal{B} \otimes \overline{\mathcal{A}}$, with $\overline{\mathcal{A}}$ denoting a DRA for the complement property of Ψ. Because optimal switching policies for reachability objectives are memoryless in BMDPs [29], it follows that the policy $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ maximizing the lower bound probability of reaching an accepting BSCC in $\mathcal{B} \otimes \mathcal{A}$ is the same for all initial states of $\mathcal{B} \otimes \mathcal{A}$. Likewise, the policy $(\breve{\mu}_{\Psi}^{up})_{\otimes}$ maximizing the lower bound probability of reaching an accepting BSCC in $\mathcal{B} \otimes \overline{\mathcal{A}}$ is the same for all initial states of $\mathcal{B} \otimes \overline{\mathcal{A}}$.

Theorem 1. *Let $\mathcal{B}$ be a BMDP, Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$, and $\overline{\Psi}$ be the complement of Ψ with corresponding DRA $\overline{\mathcal{A}}$. Let $(WC)_P^G$ and $(\overline{WC})_P^G$ be the greatest permanent winning component, respectively, of the product BMDP $\mathcal{B} \otimes \mathcal{A}$ and $\mathcal{B} \otimes \overline{\mathcal{A}}$, and $\mathcal{U}_{(WC)_P^G}$ and $\mathcal{U}_{(\overline{WC})_P^G}$ be the memoryless policies generating these sets in the corresponding product BMDP as defined in Lemma 1. A lower bound maximizing and upper bound minimizing switching policy $\widehat{\mu}_{\Psi}^{low}$ and $\breve{\mu}_{\Psi}^{up}$ in $\mathcal{B}$ with respect to Ψ are respectively induced by switching policies $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ in $\mathcal{B} \otimes \mathcal{A}$ and $(\breve{\mu}_{\Psi}^{up})_{\otimes}$ in $\mathcal{B} \otimes \overline{\mathcal{A}}$ such that*

$$(\widehat{\mu}_{\Psi}^{low})_{\otimes} = \underset{\mu \in \mathcal{U}_{(WC)_P^G}}{\arg\max}\ \breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu]}\big(\langle Q_j, s_0\rangle \models \Diamond (WC)_P^G\big) \tag{4}$$

$$(\breve{\mu}_{\Psi}^{up})_{\otimes} = \underset{\mu \in \mathcal{U}_{(\overline{WC})_P^G}}{\arg\max}\ \breve{\mathcal{P}}_{(\mathcal{B}\otimes\overline{\mathcal{A}})[\mu]}\big(\langle Q_j, s_0\rangle \models \Diamond (\overline{WC})_P^G\big) \tag{5}$$

for all initial states Q_j of $\mathcal{B}$.

Proof. We first prove equation (4). For all states belonging to $(WC)_P^G$, the lower bound probability of reaching an accepting BSCC under the defined policy

$(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ is equal to 1, since $(\widehat{\mu}_{\Psi}^{low})_{\otimes} \in \mathcal{U}_{(WC)_P^G}$, and is therefore maximized.

Next, in [24, Theorem 1], it was shown that a lower bound on the probability of reaching an accepting BSCC in a product IMC $\mathcal{I} \otimes \mathcal{A}$ is achieved in an induced product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})$ with the smallest possible set of winning components admissible by $\mathcal{I} \otimes \mathcal{A}$, which is the permanent winning component $(WC)_P$ of $\mathcal{I} \otimes \mathcal{A}$, for all states of $\mathcal{I} \otimes \mathcal{A}$. Furthermore, it was shown in [24, Lemma 9] that the probability of reaching an accepting BSCC in an induced product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})$ increases for all states of $(\mathcal{M}_{\otimes}^{\mathcal{A}})$ as more states are added to the set of winning components of $(\mathcal{M}_{\otimes}^{\mathcal{A}})$ while keeping all other transition probabilities identical. Assume the optimal policy μ^* does not belong to $\mathcal{U}_{(WC)_P^G}$ for some initial state $\langle Q_j, s_0 \rangle$ of $\mathcal{B} \otimes \mathcal{A}$ and denote by $(WC)_P^*$ the permanent winning component of $\mathcal{B} \otimes \mathcal{A}[\mu^*]$. As per the facts above, it follows that the probability of reaching an accepting BSCC from $\langle Q_j, s_0 \rangle$ in the worst-case MC of $\mathcal{B} \otimes \mathcal{A}[\mu^*]$ has to be less than the probability of reaching an accepting BSCC from $\langle Q_j, s_0 \rangle$ in the worst-case MC of $\mathcal{B} \otimes \mathcal{A}[(\mu^*)']$, where $(\mu^*)' \in \mathcal{U}_{(WC)_P^G}$ allows the states in $(WC)_P^G \backslash (WC)_P^*$ to be members of the permanent winning component and is the same as μ^* for all states outside of $(WC)_P^G$, which is a contradiction. Therefore, for all states of $\mathcal{B} \otimes \mathcal{A}$ which are not in $(WC)_P^G$, a policy μ maximizing the lower bound probability of reaching a winning component has to belong to the set $\mathcal{U}_{(WC)_P^G}$ and generates the largest possible permanent winning component in $(\mathcal{B} \otimes \mathcal{A})[\mu]$.

Due to the properties of reachability problems in BMDPs, whose optimal policies are memoryless [29], there exists a policy in $\mathcal{U}_{(WC)_P^G}$ maximizing the lower bound probability of reaching $(WC)_P^G$ simultaneously for all states which are not in $(WC)_P^G$, and, in particular, for all initial states $\langle Q_j, s_0 \rangle$ of $\mathcal{B} \otimes \mathcal{A}$ that do not belong to $(WC)_P^G$, concluding the proof of (4). Symmetric arguments combined with Fact 1 prove (5). □

This theorem shows that the desired policies are computed by solving a lower bound reachability maximization problem on a fixed set of states, which can be accomplished using the value iteration scheme presented in [16]. An algorithm for finding the sets $(WC)_P^G$ and $(\overline{WC})_P^G$ as well as their associated control actions are presented in the next subsection.

In this work, we also consider the policies $(\widehat{\mu}_{\Psi}^{up})_{\otimes}$ and $(\widecheck{\mu}_{\Psi}^{low})_{\otimes}$ that respectively maximize the upper bound and minimize the lower bound probability of reaching a winning component for all states in a product BMDP $\mathcal{B} \otimes \mathcal{A}$. While these policies are not mapped onto the original system states, they will prove useful for assessing the quality of $\widehat{\mu}_{\Psi}^{low}$ and $\widecheck{\mu}_{\Psi}^{up}$ in further sections. These are found by solving an upper bound reachability maximization problem on the *Greatest Winning Component* $(WC)_L^G$ in $\mathcal{B} \otimes \mathcal{A}$ (or $\mathcal{B} \otimes \overline{\mathcal{A}}$), whose existence is established in the lemma below.

Lemma 2. *Let $\mathcal{B}$ be a BMDP and Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$. The set of policies of the product $\mathcal{B} \otimes \mathcal{A}$ is denoted by $\mathcal{U}_{\otimes}^{\mathcal{A}}$ and the set of memoryless policies of the product $\mathcal{B} \otimes \mathcal{A}$ is denoted by $(\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem} \subseteq \mathcal{U}_{\otimes}^{\mathcal{A}}$. There exists a set of memoryless switching policies $\mathcal{U}_{(WC)_L^G} \subseteq (\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$ generating the set $(WC)_L^G$ in $\mathcal{B} \otimes \mathcal{A}$ such that, for all $\mu \in \mathcal{U}_{\otimes}^{\mathcal{A}}$, $(WC)_L \subseteq (WC)_L^G$ where $(WC)_L$ is the largest winning component of $(\mathcal{B} \otimes \mathcal{A})[\mu]$, and, for all $\mu \in \mathcal{U}_{(WC)_L^G}$, the largest winning component of $(\mathcal{B} \otimes \mathcal{A})[\mu]$ is $(WC)_L^G$.*

Proof. Lemma 2 follows from a similar constructive argument as the one in the proof of Lemma 1 where the lower bound probability operators are replaced with upper bound probability operators and vice versa. $\square$

The set $(WC)_L^G$ is called the *Greatest Winning Component* of the product BMDP $\mathcal{B} \otimes \mathcal{A}$.

Theorem 2. *Let $\mathcal{B}$ be a BMDP, Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$ and $\overline{\Psi}$ be the complement of Ψ with corresponding DRA $\overline{\mathcal{A}}$. Let $(WC)_L^G$ and $(\overline{WC})_L^G$ be the greatest winning component, respectively, of the product BMDP $\mathcal{B} \otimes \mathcal{A}$ and $\mathcal{B} \otimes \overline{\mathcal{A}}$, and $\mathcal{U}_{(WC)_L^G}$ and $\mathcal{U}_{(\overline{WC})_L^G}$ be the memoryless* *policies generating these sets in the corresponding BMDP as defined in Lemma 2. An upper bound maximizing and lower bound minimizing switching policy* $\widehat{\mu}_{\Psi}^{up}$ *and* $\widecheck{\mu}_{\Psi}^{low}$ *in* $\mathcal{B}$ *with respect to* Ψ *are respectively induced by switching policies* $(\widehat{\mu}_{\Psi}^{up})_{\otimes}$ *in* $\mathcal{B}\otimes\mathcal{A}$ *and* $(\widecheck{\mu}_{\Psi}^{low})_{\otimes}$ *in* $\mathcal{B}\otimes\overline{\mathcal{A}}$ *such that*

$$(\widehat{\mu}_{\Psi}^{up})_{\otimes} = \underset{\mu\in\mathcal{U}_{(WC)_L^G}}{\arg\max}\ \widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu]}\big(\langle Q_j, s_0\rangle \models \Diamond(WC)_L^G\big) \tag{6}$$

$$(\widecheck{\mu}_{\Psi}^{low})_{\otimes} = \underset{\mu\in\mathcal{U}_{(\overline{WC})_L^G}}{\arg\max}\ \widehat{\mathcal{P}}_{(\mathcal{B}\otimes\overline{\mathcal{A}})[\mu]}\big(\langle Q_j, s_0\rangle \models \Diamond(\overline{WC})_L^G\big) \tag{7}$$

for all initial states Q_j *of* $\mathcal{B}$*.*

Proof. As shown in [24, Theorem 1], an upper bound on the probability of reaching an accepting BSCC in a product IMC $\mathcal{I}\otimes\mathcal{A}$ is achieved in an induced product MC ($\mathcal{M}_{\otimes}^{\mathcal{A}}$) with the largest possible set of winning components allowed by $\mathcal{I}\otimes\mathcal{A}$, which is the largest winning component $(WC)_L$ of $\mathcal{I}\otimes\mathcal{A}$, for all initial states of $\mathcal{I}\otimes\mathcal{A}$. Hence, the same arguments as in the proof of Theorem 1 proves (6). Symmetric arguments combined with Fact 1 prove (7). $\square$

We remark that replacing $(WC)_L^G$ in (6) by *the greatest accepting BSCC* $(U)_L^G \subseteq (WC)_L^G$ of $\mathcal{B}\otimes\mathcal{A}$ does not change the validity of (6). The set $(U)_L^G$ contains all states which belong to an accepting BSCC for at least one induced product MC under at least one policy in $\mathcal{B}\otimes\mathcal{A}$. The proof of the existence of a set of control policies generating this set is similar to the first part of the proof of Lemma 1. This substitution can be done because, by definition, $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[(\widehat{\mu}_{\Psi}^{up})_{\otimes}]}\Big((WC)_L^G \models \Diamond(U)_L^G\Big) = 1$, and leads to algorithmic simplifications as the full set $(WC)_L^G$ may not need to be computed explicitly. A similar reasoning holds by replacing $(\overline{WC})_L^G$ with the greatest accepting BSCC of $\mathcal{B}\otimes\overline{\mathcal{A}}$ in (7). The components $(WC)_L^G$ and $(\overline{WC})_L^G$ as well as the control actions generating these components are found via a graph search, as detailed in the next subsections.

4.2. WINNING COMPONENTS SEARCH ALGORITHMS

Now, we present graph-based algorithms for finding the greatest permanent winning component $(WC)_P^G$ of a product BMDP $\mathcal{B}\otimes\mathcal{A}$ defined in Lemma 1.

Furthermore, we show how to design a switching policy that effectively generates this greatest permanent component.

The search is decomposed in two parts: first, we determine a superset of the *greatest permanent accepting BSCC*, denoted by $(U)^G_P$, of $\mathcal{B} \otimes \mathcal{A}$ following Algorithm 1. The set $(U)^G_P$ contains all states which belong to a permanent accepting BSCC for some control policy in $\mathcal{B}\otimes\mathcal{A}$, and all such states are a part of $(WC)^G_P$ as seen in the proof of Lemma 1. We call the superset of $(U)^G_P$ returned by this algorithm an extended greatest permanent accepting BSCC, denoted by $(U_+)^G_P$. This set additionally satisfies $(U)^G_P \subseteq (U_+)^G_P \subseteq (WC)^G_P$. Although Algorithm 1 is driven by a search of the sets $(U)^G_P$, our implementation allows us to find additional members of $(WC)^G_P$ in some instances.

Then, by using an iterative technique which alternates between a graph search and a reachability maximization step in Algorithm 2, one can find the set of states which are not members of $(U_+)^G_P$ but for which the lower bound probability of reaching an accepting BSCC is equal to 1 nonetheless for some control policy, and effectively create $(WC)^G_P$.

4.2.1. GREATEST PERMANENT BSCC SEARCH ALGORITHMS

We now detail an algorithm for finding an extended greatest permanent accepting BSCC $(U_+)^G_P$ of a product BMDP $\mathcal{B} \otimes \mathcal{A}$.

We introduce the following notations and terminology: a set of states in a product $\mathcal{B} \otimes \mathcal{A}$ is said to be accepting if it satisfies the acceptance condition in Definition 14 and is said to be non-accepting otherwise. A state $\langle Q_\ell, s_j \rangle$ of $\mathcal{B}\otimes\mathcal{A}$ with labeling function L' is said to be *Rabin accepting with respect to the i^{th} Rabin pair* of $\mathcal{A}$ if $F_i \in L'(\langle Q_\ell, s_j \rangle)$; $\langle Q_\ell, s_j \rangle$ is said to be *Rabin non-accepting with respect to the i^{th} Rabin pair* of $\mathcal{A}$ if $E_i \in L'(\langle Q_\ell, s_j \rangle)$. A Rabin accepting state with respect to the i^{th} pair is said to be *unmatched* in a set of states C if, for all $\langle Q_\ell, s_j \rangle \in C$, $E_i \notin L'(\langle Q_\ell, s_j \rangle)$, and it is said to be *matched* otherwise. $Act(C)$ is a set containing all sets of actions allowed for each state in a set C, that is, if $C = \{q_0, q_1, \ldots, q_k\}$, $q_i \in Q \times S$, then, $Act(C) = \{A(q_0), A(q_1), \ldots, A(q_k)\}$. $At_P(B, C, Act(C))$ is a function which outputs the set of states in C which have a non-zero probability of transition to B for at least one adversary under all actions in $Act(C)$. In addition, this function removes all actions from the sets in $Act(C)$ for which a transition to B is possible under at least one adversary and returns the updated set of allowed actions for each state of C.

We provide a short description of the algorithm: Algorithm 1 first finds the largest possible set of Strongly Connected Components (SCC), denoted by S, that can be constructed in the product BMDP in line 4 and 5 assuming all actions are available, as the greatest permanent BSCCs are a subset of these by Definition 13. Set S is determined by applying a standard SCC search techniques on the graph G defined in line 4.

Then, the algorithms iteratively remove the actions and states which prevent these SCCs from being a permanent BSCC, that is, actions and states which allow for a transition outside of the SCCs, as captured by line 9. Note that a state is discarded in set C_i once its action set is empty. Then, new SCCs are computed with the remaining states and actions in line 12. If the algorithm finds an SCC S_k which does not allow any transition outside of S_k for any state and action available, then it is potentially a member of $(U_+)^G_P$ (line 13).

Next, the acceptance status of SCC S_k is checked at line 14. This is done by inspecting the states belonging to the SCC and comparing them with Definition 14. If S_k is not accepting, states which can revert the acceptance status of S_k are removed and new SCCs are computed with the remaining states in line 23. Otherwise, the algorithm enters the if-statement in line 14 for a further analysis of S_k.

An additional condition for S_k to be a part of $(U_+)^G_P$ is that no subset of states of S_k can form a non-accepting BSCC under any scenario allowed by the transition intervals of the product BSCC. Too make sure that no subset of S_k can form a non-accepting BSCC, we choose control actions for the states in S_k that maximize the lower bound probability of reaching the unmatched Rabin accepting states contained in S_k in line 14 to 17. If this lower bound is zero for some subset of S_k, then these states could potentially form a non-accepting BSCC inside S_k for some assignment of the probabilities under all available actions. The set of all such states is denoted by A_{bad}. If A_{bad} is empty, the algorithm found a control policy that guarantees S_k to be accepting for all possible adversaries of the induced product IMC, since no state of S_k can form a BSCC which doesn't contain at least one of the unmatched accepting states, and S_k is added to $(U_+)_P^G$ in line 18. Otherwise, the SCCs which can be formed by the states in A_{bad} and by the states in $S_k \setminus A_{bad}$ with the remaining actions are computed and added to S in line 20.

Algorithm 1 Find Extended Greatest Permanent Accepting BSCC

Input: Product BMDP $\mathcal{B} \otimes \mathcal{A}$

Output: Extended greatest permanent accepting BSCCs $(U_+)^G_P$ with corresponding policy $(\widehat{\mu}^{low}_\Psi)_\otimes$ for the states in this set

Initialize: $(U_+)^G_P := \emptyset$

Initially allow all actions for all states. Construct $G := (V, E)$ with a vertex for each state in $\mathcal{B} \otimes \mathcal{A}$ ($V = Q \times S$) and an edge between states $\langle Q_i, s_j \rangle$ and $\langle Q_{i'}, s_{j'} \rangle$ if $\widehat{T}(\langle Q_i, s_j \rangle, a, \langle Q_{i'}, s_{j'} \rangle) > 0$ for some $a \in A(\langle Q_i, s_j \rangle)$

Find all SCCs of G and list them in S

for $S_k \in S$ **do**

  $C_0 := \emptyset$, $i := 0$

  repeat

    $R_i := S_k \setminus \cup_{\ell=0}^{i} C_\ell$; $Tr_i := V \setminus R_i$; $(C_{i+1}, Act(R_i)) = At_P(Tr_i, R_i, Act(R_i))$; $i = i+1$

  until $C_i = \emptyset$ and no action is removed from $Act(R_i)$

  if $i \neq 1$ **then**

    Find all SCCs of R_i (with the remaining actions) and add them to S

  else

    if S_k is accepting **then**

      Find the set A of all unmatched Rabin accepting states of S_k

      For all states in S_k, maximize the lower bound probability of $\Diamond A$. Find the set of states A_{bad} whose lower bound probability of reaching A is zero after the maximization step

      if $A_{bad} = \emptyset$ **then**

        $(U_+)^G_P := (U_+)^G_P \cup S_k$ and save the actions computed in the maximization of $\Diamond A$ to $(\widehat{\mu}^{low}_\Psi)_\otimes$ for all states of S_k

      else

        Compute the SCCs formed by the states in A_{bad} and the states in $S_k \setminus A_{bad}$ with the remaining actions and add them to S

      end if

    else

      If S_k does not contain any Rabin accepting state, continue. Otherwise, for all Rabin accepting set of states A_i with respect to pair i in S_k, find the set A_i^{non} of all states in S_k which are non-accepting with respect to the same pair as A_i. Compute the SCCs formed by the states in $S_k \setminus A_i^{non}$ with the remaining actions and add them to S

    end if

  end if

end for

return $(U_+)^G_P$, $(\widehat{\mu}^{low}_\Psi)_\otimes$ for states in $(U_+)^G_P$

We offer the following reasoning as a proof sketch for the correctness of the algorithm, i.e, to show that the output $(U_+)^G_P$ of Algorithm 1 satisfies the chain of inequalities $(U)^G_P \subseteq (U_+)^G_P \subseteq (WC)^G_P$: for a set of states S_k to belong to a permanent BSCC of a given kind in a product IMC, the following conditions must hold: 1) its constituents are not allowed to transition outside of S_k under any adversary, 2) its constituents have to be reachable from one another under all adversaries, 3) its constituents have to fulfill the requirements for accepting and non-accepting BSCCs defined in Definition 14, 4) no subset of S_k is allowed to form a BSCC of the opposite acceptance status under any adversary. Condition 1) is guaranteed by lines 7 to 10; Condition 2) is not enforced and is the reason for outputting a superset of $(U)_P^G$. This is because, as long as the other 3 conditions are fulfilled, the states in the set S_k will still be permanently winning, although the transition bounds within S_k might allow these sets to be winning via different scenarios that are not only a BSCC formed by all the states of S_k (e.g. a subset of S_k always transitioning to another subset of S_k forming a BSCC); Condition 3) is enforced by the if-statement in line 14 and the corresponding else-statements of lines 22 to 24; Condition 4) is imposed by the remainder of the main for-loop. Lastly, the algorithm iteratively removes the minimum number of actions and states causing a set S_k to violate one of these conditions and analyze all of the remaining states, ensuring that the procedure does not skip any permanent component. Note that none of the removed states could form a permanent BSCC between each other under any policy. Indeed, if these states did not belong to a common SCC in S, this would be a contradiction. Therefore, by virtue of this fact, Algorithm 1 does not "miss" any permanent BSCCs and it must hold that $(U)_P^G \subseteq (U_+)_P^G$. Moreover, the previous discussion regarding Condition 2 ensures that all states in $(U_+)_P^G \backslash (U)_P^G$ are still permanently winning, guaranteeing that $(U_+)_P^G \subseteq (WC)_P^G$ and concluding the proof sketch.

This algorithm can be adapted to determine an extended greatest accepting $(U_+)_L^G$ by replacing all instances of the function $At_P(B, C, Act(C))$ with the function $At_{pot}(B, C, Act(C))$, where $At_{pot}(B, C, Act(C))$ returns the set of states of C which have a non-zero probability of transition to B for all adversaries under all allowed actions. This function also removes all actions from $Act(C)$ for which a non-zero probability of transition to B exists under all adversaries of the induced IMC and returns the updated set of allowed actions. In addition, all mentions of the term "lower bound" have to be replaced with "upper bound". The extended set is such that $(U)_L^G \subseteq (U_+)_L^G \subseteq (WC)_L^G$.

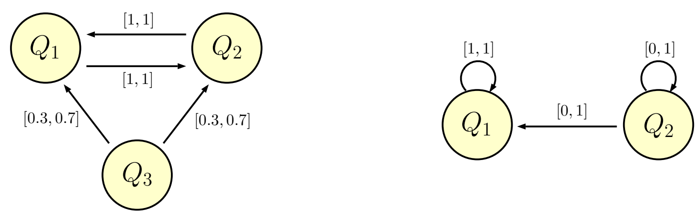


Figure 2: Depiction of the product IMCs in Example 1. On the left, state Q_3 transitions to the BSCC formed by Q_1 and Q_2 under all possible adversaries and is therefore a permanent sink state. On the right, state Q_2 is either a sink state with respect to state Q_1 or a BSCC itself for all realizations of the probability intervals.

4.2.2. GREATEST PERMANENT COMPONENTS SEARCH ALGORITHMS

Next, we present an algorithm which constructs the greatest permanent winning components $(WC)_P^G$ in a product BMDP $\mathcal{B} \otimes \mathcal{A}$ once an extended greatest permanent BSCC $(U_+)_P^G$ has been found.

In a product IMC $\mathcal{I} \otimes \mathcal{A}$, some states which are not in a permanent BSCC can still be a part of the permanent winning component of $\mathcal{I} \otimes \mathcal{A}$, as discussed in the second part of the proof of Lemma 1. These states are those which belong to a set of states C such that no transition outside the union of C and the permanent BSCCs of $\mathcal{I} \otimes \mathcal{A}$ is possible for any adversary, and such that no subset of C can form a non-accepting BSCC status under any adversary. We can further classify these states into *permanent sink* states, which cannot be a part of a BSCC under any scenario but transition to another winning set of state with lower bound probability 1, and states which allow non-deterministic scenarios where the state is sometimes a sink state with respect to another permanent winning set of states and sometimes a part of a winning component that reaches a non permanent accepting BSCC with probability one. The examples below, illustrated in Figure 2, present situations where these scenarios can occur.

Example 1. *Consider three states Q_1, Q_2 and Q_3 of a product IMC such that Q_1 and Q_2 form a permanent BSCC, with $\check{T}(Q_1, Q_2) = \check{T}(Q_2, Q_1) = 1$. Furthermore, $\check{T}(Q_3, Q_1) = \check{T}(Q_3, Q_2) = 0.3$ and $\widehat{T}(Q_3, Q_1) = \widehat{T}(Q_3, Q_2) = 0.7$. Clearly, Q_3 is not a member of the BSCC encompassing Q_1 and Q_2; yet,*

Q_3 always transitions to either Q_1 or Q_2 with probability probability 1 and is therefore a permanent sink state.

Now, consider two states Q_1 and Q_2 such that $\check{T}(Q_1, Q_1) = 1$, $\check{T}(Q_2, Q_1) = \check{T}(Q_2, Q_2) = 0$ and $\widehat{T}(Q_2, Q_1) = \widehat{T}(Q_2, Q_2) = 1$. While Q_1 is a permanent BSCC, Q_2 is neither a permanent sink state nor a permanent BSCC. However, all adversaries of the product IMC make Q_2 either a sink state with respect to Q_1 or a BSCC with itself.

Consequently, we describe a procedure in Algorithm 2 that finds all states in a product $\mathcal{B} \otimes \mathcal{A}$ for which a control policy induces one of the aforementioned scenarios given extended greatest permanent BSCCs $(U_+)^G_P$.

We explain the main features of this algorithm: first, the greatest permanent winning component $(WC)^G_P$ is initialized to the extended greatest permanent accepting BSCCs in line 3. Then, in line 5, the lower bound probability of reaching this component is maximized in the product BMDP to reveal the states which can be rendered permanent sinks with respect to $(WC)^G_P$, as these states yield a lower bound of 1 of reaching the component. The sink states are added to $(WC)^G_P$ in line 8.

Next, we define the greatest potential accepting BSCC $(U)^G_{pot}$ of a product BMDP, which are computed by taking the set difference between the greatest winning BSCC and the greatest permanent winning BSCC. States in $(U)^G_{pot}$ are those which could engender the second type of permanent components previously discussed. If $(U)^G_{pot}$ happened to contain a permanent sink state found in line 8, we compute the greatest accepting and non-accepting BSCC as well as their associated allowed actions with the remaining states in line 10 to update $(U)^G_{pot}$.

Then, in lines 12 to 17, for all BSCCs S which can be created in $(U)^G_{pot}$, we check whether there exists a policy such that no state of S can transition outside of the union of S and the current version of the greatest permanent winning component for any instantiation of the resulting transition intervals. If such a policy does not exist, states and actions for which a transition outside of the aforementioned set is possible are removed from S and the BSCCs which can be created inside the greatest BSCC of the remaining states are added to the list N of BSCCs to inspect in line 19. On the other hand, if S only contains valid states and corresponding actions, the algorithm enters the else-statement in line 20, where we need to choose a policy for the states in S which additionally does not allow the existence of a non-accepting BSCC within S under any adversary.

This step is done similarly as in Algorithm 1 by maximizing the lower bound probability of reaching the unmatched Rabin accepting states in S and removing the states yielding a lower bound probability of 0. If no such state is found, then we designed a policy that effectively makes S either a set of sink states or an accepting BSCC for all adversaries, and the states of S are added to the greatest permanent winning component $(WC)_P^G$. This process is described in line 21 to 28.

In the case that new states were added to $(WC)_P^G$ upon execution of the reachability maximization step and the graph search, which is checked in line 31 to 33, we return to the beginning of the while-loop and repeat this process with the augmented version of the greatest permanent winning component, as it could now allow previously discarded states to become permanently winning. Otherwise, the loop is exited and the algorithms return the true set $(WC)_P^G$ with its associated control actions.

A slight modification of Algorithm 2 can be employed to compute the greatest set $(WC)_L^G$ defined in Lemma 2. However, in this paper, we solely use the greatest accepting BSCC $(U_+)_L^G$ as our target set for computing the upper bound maximizing and lower bound minimizing policies $(\widehat{\mu}_\Psi^{up})_\otimes$ and $(\widetilde{\mu}_\Psi^{low})_\otimes$, as explained in Subsection 4.1.

Algorithm 2 Find Greatest Permanent Winning Components

Input: Product BMDP $\mathcal{B} \otimes \mathcal{A}$, extended greatest permanent accepting BSCC $(U_+)^G_P$, extended greatest accepting BSCCs $(U_+)^G_L$

Output: Greatest permanent winning component $(WC)^G_P$ with corresponding policy $(\widehat{\mu}^{low}_{\Psi})_{\otimes}$ for the states in this set

Initialize: $(WC)^G_P := (U_+)^G_P$, $(U)^G_{pot} := (U_+)^G_L \setminus (U_+)^G_P$, $(WC)^G_{P,prev} := (WC)^G_P$

repeat
    Maximize the lower bound probability of $\Diamond (WC)^G_P$ for all states $\langle Q_i, s_j \rangle$ in $\mathcal{B} \otimes \mathcal{A}$
    Construct the set L of all states with a lower bound equal to 1 that are not in $(WC)^G_P$
    for $Q \in L$ **do**
        $(WC)^G_P := (WC)^G_P \cup Q$, save the action $(\widehat{\mu}^{low}_{\Psi})_{\otimes}(Q)$ computed during maximization step
    end for
    Find the greatest accepting BSCC of $(U)^G_{pot} \setminus L$ using Algorithm 1 and set $(U)^G_{pot}$ to this new set of states
    Construct the set N of all accepting BSCCs constructed in $(U)^G_{pot}$ under some policy
    for $S_k \in N$ **do**
        Construct $G := (V, E)$ with a vertex for each state in $\mathcal{B} \otimes \mathcal{A}$ ($V = Q \times S$) and an edge between states $\langle Q_i, s_j \rangle$ and $\langle Q_{i'}, s_{j'} \rangle$ if $\widehat{T}(\langle Q_i, s_j \rangle, a, \langle Q_{i'}, s_{j'} \rangle) > 0$ for some $a \in A(\langle Q_i, s_j \rangle)$
        $C_0 := \emptyset$, $i := 0$
        repeat
            $R_i := S_k \setminus \cup_{\ell=0}^{i} C_\ell$; $Tr_i := V \setminus (R_i \cup (WC)^G_P)$; $(C_{i+1}, Act(R_i)) := At_P(Tr_i, R_i, Act(R_i))$; $i := i + 1$
        until $C_i = \emptyset$ and no action is removed from $Act(R_i)$
        if $i \neq 1$ **then**
            Find the greatest accepting BSCC of R_i (with remaining actions) using Algorithm 1, enumerate all accepting BSCCs constructed in this set under some policy, and add them to N
        else
            Find the set A of all unmatched Rabin accepting states of S_k
            For all states in S_k, maximize the lower bound probability of $\Diamond A$. Find the set of states A_{bad} whose lower bound probability of reaching A is zero after the maximization step
            if $A_{bad} = \emptyset$ **then**
                $(WC)^G_P := (WC)^G_P \cup S_k$, save corresponding actions in $(\widehat{\mu}^{low}_{\Psi})_{\otimes}$ for the states in S_k
                $(U)^G_{pot} := (U)^G_{pot} \setminus S_k$
            else
                Compute the greatest accepting BSCC of A_{bad} and $S_k \setminus A_{bad}$ using Algorithm 1, enumerate all accepting BSCCs constructed in this set under some policy, and add them to N
            end if
        end if
    end for
    $Y := (WC)^G_P \setminus (WC)^G_{P,prev}$
    $(WC)^G_{P,prev} := (WC)^G_P$
until $Y = \emptyset$
return $(WC)^G_P$, $(\widehat{\mu}^{low}_{\Psi})_{\otimes}$ for states in $(WC)^G_P$

In summary, we develop a procedure for computing policies that either maximize the lower bound probability or minimize the upper bound probability of satisfying an arbitrary ω-regular property in a BMDP. To this end, we show that these policies are induced by policies in the product between the BMDP and a DRA encoding the specification of interest for the maximization objective, or a DRA encoding the complement of the specification for the minimization objective. In Lemma 1, we remarked that a product BMDP always possesses a greatest permanent winning component. In Algorithms 1 and 2, we devise graph-based techniques for determining this component as well as the corresponding control actions for the states composing them. Finally, we show in Theorem 1 that, for the remaining states in the product BMDPs, the optimal policies are found by carrying out a lower bound reachability maximization computation on the greatest permanent winning component.

4.3. STATE SPACE REFINEMENT

4.3.1. QUALITY OF COMPUTED POLICY

In the previous subsections, we implemented a technique for computing an optimal switching policy in a BMDP subject to an ω-regular specification. However, recall that, in the problem at hand, BMDPs are used as abstractions of the underlying system (1) with respect to a partition of the system's continuous domain.

Here, we provide a measure of the suboptimality of the control strategy computed in a BMDP abstraction with respect to the abstracted system. While the discussion in this section focuses on optimality for the probability maximization problem with respect to specification Ψ, the same facts can straightforwardly be applied to the dual minimization problem by replacing the instances of $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$, $(\widehat{\mu}_{\Psi}^{up})_{\otimes}$ and $\mathcal{B} \otimes \mathcal{A}$ with $(\widecheck{\mu}_{\Psi}^{up})_{\otimes}$, $(\widecheck{\mu}_{\Psi}^{low})_{\otimes}$ and $\mathcal{B} \otimes \overline{\mathcal{A}}$ respectively, where $\overline{\mathcal{A}}$ is a DRA representing the complement specification $\overline{\Psi}$.

The value iteration algorithm used to design the policies $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ and $(\widehat{\mu}_{\Psi}^{up})_{\otimes}$ discussed in Theorem 1 and Theorem 2 provides useful information amenable to a quantitative measure of the quality of the lower bound maximizing policy

$(\widehat{\mu}_{\Psi}^{low})_{\otimes}$. In particular, for all states $\langle Q_j, s_i \rangle$, the algorithm determines a lower bound on the maximum lower bound probability of reaching an accepting BSCC achievable from $\langle Q_j, s_i \rangle$ over all memoryless policies of $\mathcal{B} \otimes \mathcal{A}$ choosing the lower bound maximizing action $a_{\ell,max} = (\widehat{\mu}_{\Psi}^{low})_{\otimes}(\langle Q_j, s_i \rangle)$ at state $\langle Q_j, s_i \rangle$, and an upper bound on the maximum upper bound probability of reaching an accepting BSCC achievable from $\langle Q_j, s_i \rangle$ over all memoryless policies of $\mathcal{B} \otimes \mathcal{A}$ choosing action a_ℓ at state $\langle Q_j, s_i \rangle$ for all actions $a_\ell \in A(\langle Q_j, s_i \rangle)$. Denoting these lower and upper bounds by $\check{p}_\ell$ and $\widehat{p}_\ell$ respectively for action a_ℓ, and the set of memoryless policies of $\mathcal{B} \otimes \mathcal{A}$ by $(\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$, this is formally stated as

$$\check{p}_{\ell,max} \leq \max_{\substack{\mu \in (\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem} \\ s.t. \\ \mu(\langle Q_j, s_i \rangle) = a_{\ell,max}}} \check{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[\mu]}(\langle Q_j, s_i \rangle \models \Diamond R) \ ,$$

where the subscript ℓ, max refers to the lower bound maximizing action and, for all actions $a_\ell \in A(\langle Q_j, s_i \rangle)$,

$$\widehat{p}_\ell \geq \max_{\substack{\mu \in (\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem} \\ s.t. \\ \mu(\langle Q_j, s_i \rangle) = a_\ell}} \widehat{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[\mu]}(\langle Q_j, s_i \rangle \models \Diamond R) \ ,$$

where $\Diamond R$ is a slight abuse of notation denoting the objective of reaching an accepting BSCC — which is generally not a fixed set of states as discussed in previous sections — in the product IMC $(\mathcal{B} \otimes \mathcal{A})[\mu]$.

Therefore, we introduce the *suboptimality factor* $\epsilon_{\langle Q_j, s_i \rangle}$ of state $\langle Q_j, s_i \rangle$ with respect to the lower bound maximizing policy $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ in the product BMDP $\mathcal{B} \otimes \mathcal{A}$ which is defined as

$$\epsilon_{\langle Q_j, s_i \rangle} = \max_{\ell \neq \ell, max} \widehat{p}_\ell - \check{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[(\widehat{\mu}_{\Psi}^{low})_{\otimes}]} \left(\langle Q_j, s_i \rangle \models \Diamond (WC)_P^G \right) \ . \qquad (8)$$

The quantity $\epsilon_{\langle Q_j, s_i \rangle}$ represents an upper bound on the maximum improvement in the probability of satisfying Ψ, using a memoryless policy with respect to the DRA states, any continuous state in Q_j could achieve by choosing another fixed action from the one prescribed by $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ when the product state is $\langle Q_j, s_i \rangle$, as the maximum satisfaction probability attainable when applying a different action is upper bounded by $\max_{\ell \neq \ell, max} \widehat{p}_\ell$. Therefore, the smaller $\epsilon_{\langle Q_j, s_i \rangle}$ is, the more certain we are that $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ is close to the best memoryless (in the product) policy for all states in Q_j when the automaton state is s_i.

Furthermore, the bounds computed by the value iteration algorithm can additionally be used to show that certain actions are *suboptimal* or *optimal* at a given state of a product BMDP $\mathcal{B} \otimes \mathcal{A}$ and, by extension, that the modes represented by these actions are suboptimal or optimal for some continuous states of the abstracted system for policies that are memoryless in the product. By comparing these bounds for all actions in an action space of a given state of the product BMDP $\mathcal{B} \otimes \mathcal{A}$, some of these actions may appear to surely perform worse or better than others at that particular state, as illustrated in the example below.

Example 2. *Consider a state $\langle Q_j, s_i \rangle$ of the product BMDP $\mathcal{B} \otimes \mathcal{A}$ with a set of actions $A(\langle Q_j, s_i \rangle) = \{a_1, a_2, a_3\}$, and $(\widehat{\mu}_{\Psi}^{low})_{\otimes}(\langle Q_j, s_i \rangle) = a_1$. Suppose the probabilities of reaching an accepting BSCC from $\langle Q_j, s_i \rangle$ under all 3 actions are described by the following intervals:*

- $(I_{\langle Q_j, s_i \rangle})_{a_1} = [0.5, 0.8]$,
- $(I_{\langle Q_j, s_i \rangle})_{a_2} = [0.0, 0.7]$,
- $(I_{\langle Q_j, s_i \rangle})_{a_3} = [0.0, 0.45]$,

where the lower bounds correspond to a lower bound on the maximum lower bound probability of reaching an accepting BSCC from state $\langle Q_j, s_i \rangle$ achievable over all memoryless policies of $\mathcal{B} \otimes \mathcal{A}$ choosing the corresponding action at state $\langle Q_j, s_i \rangle$, and the upper bounds correspond to an upper bound on the maximum upper bound probability of reaching an accepting BSCC from state $\langle Q_j, s_i \rangle$ achievable over all memoryless policies of $\mathcal{B} \otimes \mathcal{A}$ choosing the corresponding action at state $\langle Q_j, s_i \rangle$.

Although action a_1 maximizes the lower bound probability of reaching an accepting BSCC at 0.5*, it appears that some continuous states of Q_j could potentially produce a higher probability — up to* 0.7 *— of reaching an accepting BSCC under action a_2, since a non-deterministic scenario of the product BMDP allows* *for this probability to occur under some policy choosing a_2. However, under no memoryless policy and adversary can action a_3 generate a higher probability of reaching an accepting BSCC than action a_1, since $0.45 < 0.5$, and can therefore be discarded.*

In spite of action a_3 being removed, the suboptimality factor of $\langle Q_j, s_i\rangle$ with respect to $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ in this case is $\epsilon_{\langle Q_j, s_i\rangle} = 0.7 - 0.5 = 0.2$, as there still exists an action achieving a higher upper bound probability of reaching an accepting BSCC, namely a_2 with 0.7, than the lower bound probability of reaching an accepting BSCC under the lower bound maximizing action, namely a_1 with 0.5.

Definition 22 (Optimal/Suboptimal Action)**.** *Consider a state $\langle Q_j, s_i\rangle$ of a product BMDP $\mathcal{B} \otimes \mathcal{A}$ with a set of actions $A(\langle Q_j, s_i\rangle)$. Let us denote by $\check{p}_\ell$ a lower bound on the maximum lower bound probability of reaching an accepting BSCC from $\langle Q_j, s_i\rangle$ achievable over all memoryless policies of $\mathcal{B} \otimes \mathcal{A}$ choosing action $a_\ell \in A(\langle Q_j, s_i\rangle)$ at state $\langle Q_j, s_i\rangle$, and by $\widehat{p}_\ell$ an upper bound on the maximum upper bound probability of reaching an accepting BSCC from $\langle Q_j, s_i\rangle$ achievable over all memoryless policies of $\mathcal{B} \otimes \mathcal{A}$ choosing action a_ℓ at state $\langle Q_j, s_i\rangle$. An action a_ℓ is said to be* suboptimal *for state $\langle Q_j, s_i\rangle$ with respect to $A(\langle Q_j, s_i\rangle)$ if there exists an action $a_k \in A(\langle Q_j, s_i\rangle)$, $k \neq \ell$, such that $\widehat{p}_\ell < \check{p}_k$. An action a_ℓ is said to be* optimal *for state $\langle Q_j, s_i\rangle$ with respect to $A(\langle Q_j, s_i\rangle)$ if, for all $a_k \in A(\langle Q_j, s_i\rangle)$, $k \neq \ell$, $\check{p}_\ell \geq \widehat{p}_k$.*

Definition 23 (Optimal/Suboptimal Mode)**.** *Let $\pi = x[0]x[1]x[2] \ldots x[k]$ be any finite path of (1) such that the word $L(x[0])L(x[1])L(x[2]) \ldots L(x[k])$ produces a run $s[0]s[1]s[2] \ldots s[k]$ in automaton $\mathcal{A}$ corresponding to property Ψ, where $x[k] =: x \in D$ and $s[k] = s_i \in S$. Let us denote by $\check{p}_\ell$ a lower bound on the maximum (respectively, minimum) probability of an infinite path with prefix π to satisfy Ψ in (1) over all policies of (1) choosing mode $a_\ell \in A$ for path π, and by $\widehat{p}_\ell$ an upper bound on the maximum (respectively, minimum) probability of an* *infinite path with prefix* π *to satisfy* Ψ *in* (1) *over all policies of* (1) *choosing mode* $a_\ell \in A$ *for path* π*. When the objective is to maximize (respectively, minimize) the probability of satisfying* Ψ*, a mode* a_ℓ *is said to be* suboptimal *for state* x *with respect to automaton state* s_i *and the set of modes* A *if there exists a mode* $a_k \in A$*,* $k \neq \ell$*, such that* $\widehat{p}_\ell < \check{p}_k$ *(respectively,* $\check{p}_\ell > \widehat{p}_k$*). A mode* a_ℓ *is said to be* optimal *for state* x *with respect to automaton state* s_i *and the set of modes* A *if, for all* $a_k \in A$*,* $k \neq \ell$*,* $\widehat{p}_k \leq \check{p}_\ell$ *(respectively,* $\widehat{p}_\ell \leq \check{p}_k$*).*

If the set of actions $A(\langle Q_j, s_i \rangle)$ of state $\langle Q_j, s_i \rangle$ contains an optimal action, then the suboptimality factor $\epsilon_{\langle Q_j, s_i \rangle}$ is set to 0.

4.3.2. REFINEMENT PROCEDURE

Now that a quantitative measure for the quality of the computed switching policy has been introduced, our next objective is to design a domain partition refinement scheme to address Subproblem 1.2 and achieve a user-defined level of optimality. In order to mitigate the state-space explosion phenomenon, the refinement algorithm should specifically target the states causing the most uncertainty in the domain partition.

We define the *greatest suboptimality factor* ϵ_{max} as

$$\epsilon_{max} = \max_{\langle Q_j, s_i \rangle \in (Q \times S)} \epsilon_{\langle Q_j, s_i \rangle} \tag{9}$$

which can be used as a natural precision criterion for a given domain partition P. A low factor ϵ_{max} ensures that no state in the original system is poorly controlled under the switching policy computed in the BMDP abstraction arising from P. Looser notions of optimality, such as *the average suboptimality factor* or *the fraction of states* below a fixed optimality threshold, are less sensitive to outliers and can alternatively be considered. We denote the desired suboptimality target by ϵ_{thr}. Note that a target ϵ_{thr} equal to 0 requires to find an optimal action for all states in $\mathcal{B} \otimes \mathcal{A}$ in the case of maximization or in $\mathcal{B} \otimes \overline{\mathcal{A}}$ for the case of minimization.

Formally, a partition P' is a refinement of a coarser partition P if all states in P is equal to the union of a set of states in P'. In the general case, abstractions constructed from a refinement P' of P will exhibit a lesser degree of non-determinism than abstractions constructed from P, allowing for the computation of higher-quality controllers with respect to the abstracted system.

Definition 24 (Partition Refinement)**.** *A partition P' is a refinement of a partition P if, for all states $Q_j \in P$, there exists a set of states $\{Q^k_{j'}\}^{m_j}_{k=0}$ in P' such that $Q_j = \cup^{m_j}_{k=0} Q^k_{j'}$.*

The proposed refinement procedure to achieve a target precision ϵ_{thr} is inspired by our technique in [24] where refinement was conducted for the purpose of verification in an IMC and whose main features are extended to the synthesis problem at hand. This new procedure is based on a heuristical scoring of the states in a partition P which highlights the regions of the state-space causing the most uncertainty with respect to the specification of interest and the set of actions at hand. Specifically, this score aims to capture how differently a partition state behaves between the extreme cases induced by the two maximizing (or minimizing) policies previously discussed, as well as how much this state influences other states which are known to be suboptimaly controlled.

Our scoring algorithm is presented in Algorithm 3 and is summarized as follows: first, we take as input a "best-case" product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_u$ and a "worst-case" product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_l$. For the case of maximization, the worst-case product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_l$ is a worst-case product MC induced by the IMC $(\mathcal{B} \otimes \mathcal{A})[(\widehat{\mu}^{low}_{\Psi})_{\otimes}]$ with respect to the objective of reaching an accepting BSCC, while the best-case product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_u$ is a best-case product MC induced by the IMC $(\mathcal{B} \otimes \mathcal{A})[(\widehat{\mu}^{up}_{\Psi})_{\otimes}]$. Similarly, for the case of minimization, the worst-case product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_l$ is a worst-case product MC induced by the IMC $(\mathcal{B} \otimes \overline{\mathcal{A}})[(\widecheck{\mu}^{up}_{\Psi})_{\otimes}]$ with respect to the objective of reaching an accepting BSCC, while the best-case product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_u$ is a best-case product MC induced by the IMC $(\mathcal{B} \otimes$

$\overline{\mathcal{A}})[\widetilde{\mu}_{\Psi}^{low}]$. Again, the aforementioned MCs are automatically constructed when applying the reachability value iteration algorithm used in Algorithms 1 and 2 and for designing the two maximizing (or minimizing) policies.

Next, for all state $\langle Q_j, s_i \rangle$ of the product BMDP $\mathcal{B} \otimes \mathcal{A}$ (or $\mathcal{B} \otimes \overline{\mathcal{A}}$) whose suboptimality factor is greater than the target ϵ_{thr}, we compute the probability $p_{\langle j,i \rangle \to \langle j',i' \rangle}$ of reaching any state $\langle Q_{j'}, s_{i'} \rangle$ from $\langle Q_j, s_i \rangle$ in the MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ on line 7 using the results in [30]. Then, for all states $\langle Q_{j'}, s_{i'} \rangle$ of the product BMDP that do not belong to a permanent component (as these do not require refinement), the quantity $p_{\langle j,i \rangle \to \langle j',i' \rangle} \cdot ||T^u_{\langle j',i' \rangle} - T^{\ell}_{\langle j',i' \rangle}||_2$ is added to the score $\sigma_{j'}$ of the partition state $Q_{j'}$ on line 10, where $T^u_{\langle j',i' \rangle}$ and $T^{\ell}_{\langle j',i' \rangle}$ are the rows corresponding to state $\langle Q_{j'}, s_{i'} \rangle$ in the transition matrices of $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ respectively. The term $||T^u_{\langle j',i' \rangle} - T^{\ell}_{\langle j',i' \rangle}||_2$ aims to capture how differently state $\langle Q_{j'}, s_{i'} \rangle$ behaves in the two extreme MCs, while $p_{\langle j,i \rangle \to \langle j',i' \rangle}$ is a term associated with how much state $\langle Q_{j'}, s_{i'} \rangle$ affects state $\langle Q_j, s_i \rangle$. Finally, from line 10 to 13, we additionally increment the score of states which have the potential of changing the qualitative connectivity structure of the "best" and "worst" case scenarios. These states are those which belong to a BSCC that is present in one of the scenarios and not in the other and have the potential of confirming or invalidating the existence of these BSCCs, that is, states which have an outgoing transition with a zero lower bound and a non-zero upper bound for at least one available (non-suboptimal) control action.

Once a score is attributed to each state of P via Algorithm 3, states with a score above a user-defined threshold are refined to generate a finer partition P'. A new switching policy is computed in a BMDP abstraction constructed from P', and more refinement steps are subsequently applied if necessary. The procedure terminates once the optimality factor ϵ_{max} becomes less than the target ϵ_{thr}.

It should be noted that a product IMC generally does not induce a unique worst-case and best-case MC, but rather induces sets of possible worst-case and best-case MCs yielding the same probabilities of reaching an accepting BSCC from all states [24]. Therefore, the choice of inputs for Algorithm 3 may not be unique. As previously discussed, we choose to input the MCs computed in the process of designing the control policies for the BMDP. Although selecting other MCs is possible, we claim that the design of Algorithm 3 renders the effect of choosing other input MCs negligible in the long-term behavior of the synthesis algorithm in all but pathological cases. The reasoning behind this claim is that a lot of discrepancies between different worst-case (or best-case) MCs occur in the transitions within permanent winning or losing components which belong to the set G defined in Line 5 of Algorithm 3, as the existence of these components depend on the qualitative structure of the IMC and not on the exact transition values, and have no influence on the computation of the refinement scores. Other large discrepancies between different such MCs may be found in the potential BSCCs stored in set R in Line 4. However, the relative difference captured by the term $||T^u_{\langle j',i' \rangle} - T^\ell_{\langle j',i' \rangle}||_2$ at such states is likely to be similar regardless, causing only a minor variation in refinement scores for two different input MCs, and the set R may quickly become empty after a few iterations of the refinement algorithm as the states causing this set to exist, namely states without zero lower bound and non-zero upper bound, are targeted in Line 12 to 14. Finally, different transitions between two worst-case (or best-case) MCs can be found outside of the aforementioned sets, but this scenario is improbable for abstractions computed from dynamical systems with continuous state-spaces. Indeed, this would require for at least two states outside these sets to have the exact same probability of reaching an accepting or a non-accepting BSCC in the extremal assignments of the transition probabilities, which is unlikely when using transition bounds derived from integrals over continuous sets. Such a scenario may be encountered on coarse abstractions with very few states that are all bound to be refined no matter which best or worst-case MC is chosen, and finding such states with equal reachability probabilities in high-dimensional MCs would be an isolated event with little impact on the refinement algorithm.

Algorithm 3 Refinement Scoring Algorithm

Input: Product BMDP $\mathcal{B} \otimes \mathcal{A}$, best-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$, worst-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$, threshold suboptimality factor ϵ_{thr}, suboptimality factors $\epsilon_{\langle Q_j, s_i \rangle}$ for all states $\langle Q_j, s_i \rangle$ of $\mathcal{B} \otimes \mathcal{A}$

Output: Refinement scores $\sigma = [\sigma_0, \sigma_1, \ldots, \sigma_{|Q|-1}]$ for all states of partition P

Initialize: $\sigma = [\sigma_0, \sigma_1, \ldots, \sigma_{|Q|-1}]$ where $\sigma_i = 0$

In R, list all states of $\mathcal{B} \otimes \mathcal{A}$ belonging to a BSCC that exists in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and not in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$, or vice-versa

In G, list all states of $\mathcal{B} \otimes \mathcal{A}$ with a probability of reaching an accepting BSCC of 0 in both $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ or of 1 in both $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$

for $\langle Q_j, s_i \rangle \in \mathcal{B} \otimes \mathcal{A}$ **do**

  if $\epsilon_{\langle Q_j, s_i \rangle} \geq \epsilon_{thr}$ **then**

    Compute the probability $p_{\langle j,i \rangle \to \langle j',i' \rangle}$ of reaching $\langle Q_{j'}, s_{i'} \rangle$ from $\langle Q_j, s_i \rangle$ in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$, for all $\langle Q_{j'}, s_{i'} \rangle \in \mathcal{B} \otimes \mathcal{A}$, using the technique in [30]

    for $\langle Q_{j'}, s_{i'} \rangle \in \mathcal{B} \otimes \mathcal{A}$ such that $\langle Q_{j'}, s_{i'} \rangle \notin G$ **do**

      $\sigma_{j'} = \sigma_{j'} + p_{\langle j,i \rangle \to \langle j',i' \rangle} \cdot ||T^u_{\langle j',i' \rangle} - T^\ell_{\langle j',i' \rangle}||_2$, where $T^u_{\langle j',i' \rangle}$ and $T^\ell_{\langle j',i' \rangle}$ are the rows corresponding to state $\langle Q_{j'}, s_{i'} \rangle$ in the transition matrices of $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ respectively

      if $\langle Q_{j'}, s_{i'} \rangle \in R$ **then**

        for $\langle Q_{j''}, s_{i''} \rangle \in \mathcal{B} \otimes \mathcal{A}$ such that $\langle Q_{j'}, s_{i'} \rangle$ and $\langle Q_{j''}, s_{i''} \rangle$ belong to a common BSCC in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ or $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ **do**

          if $\langle Q_{j''}, s_{i''} \rangle$ has an outgoing transition with a zero lower bound and a non-zero upper bound for at least one available (non-suboptimal) control action **then**

            $\sigma_{j''} = \sigma_{j''} + p_{\langle j,i \rangle \to \langle j',i' \rangle} \cdot ||T^u_{\langle j',i' \rangle} - T^\ell_{\langle j',i' \rangle}||_2$

          end if

        end for

      end if

    end for

  end if

end for

The fact that a partition P' is a refinement of a partition P allows us to make inferences about the properties of the states in P' from the synthesis computations performed on the states in P. First, as discussed previously, not all actions allowed in P may need to be considered in the refined partition P' when computing a new switching policy. Indeed, given a partition $Q_j = \cup_{k=0}^{m_j} Q_{j'}^k$ of a state $Q_j \in P$, it follows that a certainly suboptimal action with respect to the action set of a product state $\langle Q_j, s_i \rangle$ will also be suboptimal with respect to all $\left\langle Q_{j'}^k, s_i \right\rangle$ and can be eliminated in the synthesis procedure applied to P'.

Proposition 1. *Let $\mathcal{B}$ be a BMDP abstraction constructed from a partition P of the domain D of* (1)*, $\mathcal{A}$ be a DRA corresponding to specification Ψ, and P' be a refinement of P. Let $\{Q_{j'}^k\}_{k=0}^{m_j} \subseteq P'$, be a partition of state $Q_j \in P$. If action $a \in A(\langle Q_j, s_i \rangle)$ is suboptimal for state $\langle Q_j, s_i \rangle$ with respect to $A(\langle Q_j, s_i \rangle)$ in the product BMDP $\mathcal{B} \otimes \mathcal{A}$, then the mode of* (1) *represented by action a is suboptimal for all $x \in Q_j$ with respect to the automaton state s_i and the set of available modes, and, in particular, for all $x \in Q_{j'}^k$, $k = 0, 1 \ldots, m_j$.*

Proof. The proof assumes the objective of synthesis to be the maximization of the probability of satisfying Ψ. We denote by $\widehat{p}$ an upper bound on the maximum upper bound probability of reaching an accepting BSCC in $\mathcal{B} \otimes \mathcal{A}$ from $\langle Q_j, s_i \rangle$ achievable over all memoryless policies choosing action $a \in A(\langle Q_j, s_i \rangle)$ at state $\langle Q_j, s_i \rangle$. The assumption that a is suboptimal with respect to $A(\langle Q_j, s_i \rangle)$ in $\mathcal{B} \otimes \mathcal{A}$ implies that there exists an action $a' \in A(\langle Q_j, s_i \rangle)$ with a known a lower bound $\widecheck{p}'$ on the maximum lower bound probability of reaching an accepting BSCC in $\mathcal{B} \otimes \mathcal{A}$ from $\langle Q_j, s_i \rangle$ achievable over all memoryless policies choosing action $a' \in A(\langle Q_j, s_i \rangle)$ and such that $\widehat{p} < \widecheck{p}'$. Therefore, by virtue of $\mathcal{B}$ being an abstraction of (1), $\forall x \in Q_j$, it follows that $\widehat{p}_{mode} < \widecheck{p}'_{mode}$, where $\widehat{p}_{mode}$ and $\widecheck{p}'_{mode}$ are a lower bound and an upper bound on the maximum probability that an infinite path of (1) with prefix $\pi = x[0]x[1]x[2]$ ... $x[k]$, $x[k] =: x$, such that the word $L(x[0])L(x[1])L(x[2])$ ... $L(x[k])$ produces a run $s[0]s[1]s[2] \ldots s[k]$, with $s[k] = s_i$, satisfies Ψ over all the (memoryless in the product) policies of

(1) choosing the modes represented by actions a and a' respectively at path π. It follows that the mode represented by action a is suboptimal for all $x \in Q_j$ with respect to automaton state s_i and the set of available modes. In particular, this statement is true for all $x \in Q^k_{j'}$, $k = 0, 1 \ldots, m_j$, since $Q^k_{j'} \subseteq Q_j$, proving the proposition. Symmetric arguments prove this proposition in the case of minimization. □

Furthermore, out of the remaining actions, only a subset of them may be retained for the qualitative problems of constructing the largest and permanent components in P' using Algorithms 1 and 2. Indeed, all actions in $A(\langle Q_j, s_i \rangle)$ which were discarded during the graph search for $(WC)^G_L$ could not, under any policy and adversary, generate a winning component in $\mathcal{B} \otimes \mathcal{A}$. Therefore, we can define the set of actions $A_{qual}(\langle Q^k_{j'}, s_i \rangle) \subseteq A(\langle Q^k_{j'}, s_i \rangle)$ used specifically for the component graph search and containing all actions which, at state $\langle Q_j, s_i \rangle$, allowed for the existence of $(WC)^G_L$ with respect to the partition P.

Proposition 2. *Let $\mathcal{B}$ be a BMDP abstraction constructed from a partition P of the domain D of* (1)*, $\mathcal{A}$ be a DRA corresponding to specification Ψ, $\overline{\mathcal{A}}$ be a DRA corresponding to complement specification $\overline{\Psi}$, and P' be refinement of a partition P. If state $\langle Q_j, s_i \rangle$ is not a member of $(WC)^G_L$ in the product BMDP $\mathcal{B} \otimes \mathcal{A}$ (respectively, $\mathcal{B} \otimes \overline{\mathcal{A}}$) under any memoryless policy μ of $\mathcal{B} \otimes \mathcal{A}$ (respectively, $\mathcal{B} \otimes \overline{\mathcal{A}}$) such that $\mu(\langle Q_j, s_i \rangle) = a \in A(\langle Q_j, s_i \rangle)$, then, for all $x \in Q_j$, the probability that an infinite path with prefix $\pi = x[0]x[1]x[2]\ ...\ x[k]$, $x[k] =: x$, such that the word $L(x[0])L(x[1])L(x[2])\ ...\ L(x[k])$ produces a run $s[0]s[1]s[2] \ldots s[k]$, with $s[k] = s_i$ in automaton $\mathcal{A}$, satisfies Ψ is strictly less than 1 (respectively, strictly greater than 0) for all policies of* (1) *choosing the mode represented by action a at state x. In particular, this statement is true for all $x \in Q^k_{j'}$, $k = 0, 1 \ldots, m_j$, where $\{Q^k_{j'}\}^{m_j}_{k=0}$, $Q^k_{j'} \in P'$, is a partition of state $Q_j \in P$.*

Proof. The proof assumes the objective of synthesis to be the maximization of the probability of Ψ. If state $\langle Q_j, s_i\rangle$ is not a member of $(WC)^G_L$ under any memoryless policy μ such that $\mu(\langle Q_j, s_i\rangle) = a$, then it must be true that $\widehat{p} < 1$, where $\widehat{p}$ is an upper bound on the probability of $\langle Q_j, s_i\rangle$ to reach an accepting BSCC in $\mathcal{B}\otimes\mathcal{A}$ under all memoryless policies μ such that $\mu(\langle Q_j, s_i\rangle) = a$. Therefore, by virtue of $\mathcal{B}$ being an abstraction of (1), it follows that the probability of an infinite path with prefix $\pi = x[0]x[1]x[2] \ldots x[k]$, $x[k] =: x$, such that the word $L(x[0])L(x[1])L(x[2]) \ldots L(x[k])$ produces a run $s[0]s[1]s[2]\ldots s[k]$, with $s[k] = s_i$ in automaton $\mathcal{A}$ to satisfy Ψ is upper bounded by $\widehat{p}$ for all policies of (1) choosing the mode represented by action a for the path π and is thus strictly less than 1. In particular, this statement is true for all $x \in Q^k_{j'}$, $k = 0, 1 \ldots, m_j$, since $Q^k_{j'} \subseteq Q_j$, proving the proposition. Symmetric arguments prove the proposition with respect to the minimization objective. $\square$

An analogous proposition can be established with respect to the greatest BSCCs $(U)^G_L$ for Algorithm 1.

We remark that any state $\langle Q_j, s_i\rangle$ belonging to the greatest permanent winning components $(WC)^G_P$ of a BMDP abstraction $\mathcal{B}\otimes\mathcal{A}$ constructed from a partition P has to belong to the greatest permanent components with respect to a refined partition P' if the same control action applied to all $\langle Q_j, s_i\rangle \in (WC)^G_P$ in the abstraction resulting from P is applied to all their refinement states $\left\langle Q^k_{j'}, s_i\right\rangle$.

Proposition 3. *Let $\mathcal{B}$ be a BMDP abstraction constructed from a partition P of the domain D of* (1)*, $\mathcal{A}$ be a DRA corresponding to specification Ψ, $\overline{\mathcal{A}}$ be a DRA corresponding to complement specification $\overline{\Psi}$, and P' be refinement of a partition P. A policy μ of $\mathcal{B}$ induced by a policy in $\mathcal{B}\otimes\mathcal{A}$ (respectively, $\mathcal{B}\otimes\overline{\mathcal{A}}$ in the case of minimization) generating the greatest permanent winning component $(WC)^G_P$ of $\mathcal{B}\otimes\mathcal{A}$ (respectively, of $\mathcal{B}\otimes\overline{\mathcal{A}}$) selects an optimal mode (with the appropriate mode/action correspondence) for all $x \in Q_j$ such that $\langle Q_j, s_i\rangle \in (WC)^G_P$ with respect to the automaton state s_i and the set of available modes, and, in particular, for all $x \in Q^k_{j'}$, $k = 0, 1 \ldots, m_j$, where $\{Q^k_{j'}\}^{m_j}_{k=0}$,*

$Q^k_{j'} \in P'$*, is a partition of state* $Q_j \in P$*.*

Proof. The proof assumes the objective of synthesis to be the maximization of the probability of Ψ. A policy $(\mu)_\otimes$ generating $(WC)^G_P$ in $\mathcal{B} \otimes \mathcal{A}$ ensures that $\breve{\mathcal{P}}(\langle Q_j, s_i \rangle \models \Diamond (WC)^G_P) = 1$ for all $\langle Q_j, s_i \rangle \in (WC)^G_P$. The policy μ in $\mathcal{B}$ induced by $(\mu)_\otimes$ applied to all $x \in Q_j$ such that $\langle Q_j, s_i \rangle \in (WC)^G_P$ when the automaton state is s_i with the appropriate mode/action correspondence guarantees that, for all such x, the probability of an infinite path with prefix $\pi = x[0]x[1]x[2] \ldots x[k]$, $x[k] =: x$, such that the word $L(x[0])L(x[1])L(x[2]) \ldots L(x[k])$ produces a run $s[0]s[1]s[2] \ldots s[k]$, with $s[k] = s_i$ in automaton $\mathcal{A}$ to satisfy Ψ is equal to 1, by virtue of $\mathcal{B}$ being an abstraction of (1). Therefore, μ selects an optimal mode for all such x. In particular, this statement is true for all $x \in Q^k_{j'}$, $k = 0, 1 \ldots, m_j$, since $Q^k_{j'} \subseteq Q_j$, proving the proposition. Symmetric arguments prove the proposition with respect to the minimization case. $\square$

Therefore, by pruning all states which were a member of $(WC)^G_P$ in an abstraction constructed P, since an action engendering a fixed probability of reaching an accepting BSCC equal to 1 is known for such states, we can reduce the effective set of states for which a controller has to be synthesized in the abstraction arising from a refined partition P' after each refinement step.

Finally, additional crucial information can be exploited to tremendously reduce the number of operations performed in a refined partition. For example, in the numerical examples presented further, all states which were shown to be reachable from a given state Q_j under some action in partition P are stored in memory, and only these states or their subsets are inspected for computing the transitions from Q_j in the abstraction arising from a refined partition P'. This is justified by the fact that, if $\widehat{T}(Q_1, Q_2) = 0$ for any Q_1 and Q_2 in partition P, then it follows that $\widehat{T}(Q^k_1, Q^k_2) = 0$ for any $Q^k_1 \subseteq Q_1$ and $Q^k_2 \subseteq Q_2$.

This novel iterative approach that removes suboptimal actions at each refinement step is promising in terms of scalability compared to existing methods. For instance, prominant tools such as StocHy [19] and FAUST[2] [31] employ a single gridding approach where a unique (often conservative) partition of the domain guaranteeing a target abstraction error is created and used for computing a switching policy; in this case, all possible actions allowed by the original abstracted system have to be considered on possibly very fine partition grids, causing intractability issues when the action space is large. Here, the action space to be analyzed for the refined states reduces in size as the partition is progressively rendered finer. Therefore, the number of computations performed to synthesize a switching policy for an equivalent level of abstraction fineness is reduced compared to the aforementioned tools. Furthermore, the continuous domain grid in StocHy and FAUST^2 depends primarily on the properties of the abstracted system whereas our refinement is specification-guided, i.e, tailored to the specification under consideration, diminishing the generation of unnecessary discrete states. The iterative refinement proposed in [16] does not implement an action removal scheme and suffers from the same tractability issues discussed above. In addition, structural properties inherited from coarser partitions are not discussed and leveraged to lessen the computational burden of synthesis. Also, the termination criterion of the algorithm in [16] is a low abstraction error under the lower bound maximizing (or upper bound minimizing) policy which, unlike the suboptimality factor introduced in this work, does not directly capture the possible improvement one could achieve by choosing a different policy (which is memoryless in the product construction) on a refined abstraction. Lastly, the selection of states to be refined in [16] focuses on one-step transition errors and does not involve the inspection of the overall structure of the abstraction between the two extreme scenarios of the BMDP as in Algorithm 3.

Our specification-guided, refinement-based synthesis procedure for finite-mode systems is summarized in Algorithm 4. We assume that states selected by the scoring scheme are split in half along their greatest dimension. In this case, the worst-case growth of the BMDP abstraction throughout the procedure is $\mathcal{O}(|S| \cdot |Act| \cdot 2^{|Q|})$ when every state in the partition is refined. However, the iterative removal of considered actions, coupled with the scoring algorithm targeting only specific regions of the domain, mitigates this exponential growth in practice. The run-time complexity of the sub-components of Algorithm 4 is as follows: Algorithm 1 is exponential in $|S| \cdot |Q|$ as the number of SCCs to analyze may grow exponentially in the worst-case; consequently, Algorithm 2, which calls Algorithm 1, displays the same run-time complexity; the iterative reachability maximization algorithm on the winning components is polynomial in $|Act| \cdot |S| \cdot |Q|$ [16] and Algorithm 3, whose limiting factor is the computation reachability probabilities in MCs, is therefore polynomial in $|S| \cdot |Q|$.

Algorithm 4 Controller Synthesis for Finite-mode Systems

Input: Partition P_0 of domain D of (1), ω-regular property Ψ (complement property $\overline{\Psi}$) and corresponding DRA $\mathcal{A}$ ($\overline{\mathcal{A}}$), target controller precision ϵ_{thr}
Output: Maximizing (minimizing) switching policy $\widehat{\mu}_{\Psi}^{low}$ ($\widecheck{\mu}_{\Psi}^{up}$), final partition P_{fin}
Initialize: $\epsilon_{max} := 1$, $i := 0$
while $\epsilon_{max} > \epsilon_{thr}$ **do**
    Compute the sets $(WC)_P^G$ and $(WC)_L^G$ of the product BMDP $\mathcal{B} \otimes \mathcal{A}$ ($\mathcal{B} \otimes \overline{\mathcal{A}}$) constructed from P_i using Algorithms 1 and 2
    Compute the policies $\widehat{\mu}_{\Psi}^{low}$ and $\widehat{\mu}_{\Psi}^{up}$ ($\widecheck{\mu}_{\Psi}^{up}$ and $\widecheck{\mu}_{\Psi}^{low}$) of the BMDP $\mathcal{B}$ according to Subsections 4.1
    Compute ϵ_{max} using (9)
    if $\epsilon_{max} > \epsilon_{thr}$ **then**
        Compute a best-case and worst-case product MC $(\mathcal{M}_u)_{\otimes}^{\mathcal{A}}$ and $(\mathcal{M}_l)_{\otimes}^{\mathcal{A}}$ as discussed in Subsection 4.3.2
        Apply the scoring procedure in Algorithm 3 and refine all states above a user-defined threshold score to produce P_{i+1}
        Update the set of actions of all states in P_{i+1} for the component search and reachability problem as discussed in Subsection 4.3.2
        $i := i + 1$
    end if
end while
return $\widehat{\mu}_{\Psi}^{low}$ ($\widecheck{\mu}_{\Psi}^{up}$), $P_{fin} := P_i$

4.3.3. MONOTONICITY AND CONVERGENCE OF SYNTHESIS PROCEDURE

As pointed out in [24], it is possible to construct scenarios where, for two states Q_i and Q_j in a given partition, and two states Q_j' and Q_j'' generated from a refinement of Q_j, that is, $Q_j = Q_j' \cup Q_j''$, the inequality $\widehat{T}_{ex}(Q_i, a, Q_j) < \widehat{T}_{ex}(Q_i, a, Q_j') + \widehat{T}_{ex}(Q_i, a, Q_j'')$ holds for some mode a of system (1), where $\widehat{T}_{ex}(Q_i, a, Q_j)$ returns the least upper bound on the probability for any continuous state $x \in Q_i$ to transition to a state in Q_j under mode a. As a consequence, because the current implementations of the graph search and reachability maximization algorithms view the abstractions created from a partition and its refinements as being independent from one another, our synthesis algorithm may assign a larger amount of probability to the transition from state Q_i to the total refined states constituting Q_j in the refined abstractions than was allowed in the coarser ones. This phenomenon may cause:

- The set $(WC)_L^G$ to increase and the set $(WC)_P^G$ to decrease upon refinement. Specifically, given a state $\langle Q_j, s_i \rangle$ of a product BMDP $\mathcal{B} \otimes \mathcal{A}$ constructed from a partition P, and a state $\langle Q_j', s_i \rangle$ of a product BMDP $\mathcal{B}' \otimes \mathcal{A}$ constructed from a refinement P' of P, where $Q_j' \subset Q_j$, it is possible for $\langle Q_j', s_i \rangle$ to belong to $(WC)_L^G$ in $\mathcal{B}' \otimes \mathcal{A}$ while $\langle Q_j, s_i \rangle$ does not belong to this set in $\mathcal{B} \otimes \mathcal{A}$, and it is possible for $\langle Q_j, s_i \rangle$ to belong to $(WC)_P^G$ in $\mathcal{B} \otimes \mathcal{A}$ while $\langle Q_j', s_i \rangle$ does not belong to this set in $\mathcal{B}' \otimes \mathcal{A}$,

- The lower bound probabilities of reaching $(WC)_P^G$ to decrease from some states of the product BMDP for a fixed policy, and the upper bound probability of reaching $(WC)_L^G$ to increase from some states of the product BMDP for a fixed policy.

Therefore, a finer partition could provide "less certainty" and result in the synthesis of a switching policy yielding a smaller satisfaction lower bound (or greater upper bound in the case of minimization) for some states of the refined BMDP abstraction. This means that a monotone decrease of the greatest suboptimality factor ϵ_{max} is not guaranteed under the proposed iterative refinement method.

We address the first bullet point by saving the states that belong to the aforementioned components in the coarser abstraction before each refinement step and using the facts enunciated in Propositions 2 and 3; however, the second bullet point affects the monotonicity of the value iteration algorithm of [16] in its current state.

Nonetheless, under a continuity assumption on the dynamics and using adequate BMDP abstraction techniques, it seems that having the size of all discrete states which are not in a permanent component approach zero in the limit is sufficient for guaranteeing convergence of Algorithm 4, as seen in related case studies using iterative refinement [16], [24] and the case study presented further. We conjecture that the scoring and refinement procedure applied in Algorithm 4 satisfies this condition and therefore ensures convergence; however, we leave a thorough investigation and potential formal proof of these facts for future work.

In brief, we introduce a quantitative measure of the suboptimality of the devised switching policy in a BMDP abstraction with respect to the original continuous abstracted states. This suboptimality factor defined through (8) and (9) corresponds to an upper bound on the potential improvement any continuous state of the system could experience in the probability of satisfying the specification using memoryless (in the product) policies by choosing a different control action from the one prescribed by the computed policy. This factor is established in the BMDP abstraction through a comparison between the worst-case assignment of the probability intervals under the computed policy and the best-case assignment of these probabilities under a policy assuming the most optimistic outcome of the transition intervals. Furthermore, these worst-case and best-case scenarios are used to identify control actions that are certainly suboptimal for a given state as formalized in Proposition 1. Lastly, in Algorithm 4, we presented an iterative partition refinement heuristic which selectively targets certain regions of the state-space by comparing these two extreme scenarios with the objective of achieving a user-defined precision threshold. Some structural properties transmitted from coarser abstractions to refined ones are identified in Proposition 2 and 3, allowing to reduce the number of required computations after each refinement step.

While the techniques derived in this section are applicable to finite mode stochastic systems, they do not straightforwardly extend to the synthesis of control policies for stochastic systems with a continuous set of available inputs as stated in Problem 2, which is the focus of the next section.

5. CONTROLLER SYNTHESIS FOR CONTINUOUS INPUT SYSTEMS

In this section, we discuss synthesis for stochastic systems with a continuous set of inputs as defined in Problem 2. Recall that we focus on systems of the form (3) with state update equation $x[k+1] = \mathcal{F}(x[k]) + u[k] + w[k]$.

To synthesize controllers for such systems, we again construct a finite partition P of the continuous domain D of (3) to generate a CIMC abstraction $\mathcal{C}$ of the system. Note that the results presented in the lemmas and theorems of Section 4 for BMPDs are not altered if the set of available actions is infinite and consequently apply identically to CIMCs. Therefore, our approach is similar to the synthesis method for BMDPs, that is, a DRA representation $\mathcal{A}$ of the specification of interest Ψ is computed, and the problem is converted to a component search and a reachability maximization step in the product CIMC $\mathcal{C} \otimes \mathcal{A}$.

Definition 25 (Product Controlled Interval-valued Markov Chain)**.** *Let* $\mathcal{C} = (Q, U, \breve{T}, \widehat{T}, q_0, \Sigma, L)$ *be a CIMC and* $\mathcal{A} = (S, 2^{\Sigma}, \delta, s_0, Acc)$ *be a DRA. The* product $\mathcal{C} \otimes \mathcal{A} = (Q \times S, U, \breve{T'}, \widehat{T'}, q_0^{\otimes}, Acc', L')$ *is a CIMC defined similarly to product BMDP with the difference that a continuous set of inputs* $U \subset \mathbb{R}^m$ *replaces the finite set of actions* Act.

However, because the number of "modes" of (3) corresponding to different choices of input u can be viewed as being uncountably infinite, the techniques established in Section 4, which rely on exhaustive searches over all possible actions at all states of the abstraction, cannot be applied directly in this context. Instead, we need to consider the underlying continuous dynamics of the abstracted system and exploit their relationship with the bounds of the CIMC abstraction $\mathcal{C}$.

To propose a solution to this problem, we make the following assumptions on (3) which allow to derive closed-form expressions for the lower and upper bound transition maps $\check{T}$ and $\widehat{T}$ as a function of the input parameter u.

Assumption 1. *The partition P of the domain D of system* (3) *conforms to the labeling function of* (3) *and is rectangular, that is,* $\forall Q_j \in P$, $Q_j = [a_1^j, b_1^j] \times [a_2^j, b_2^j] \times \ldots \times [a_n^j, b_n^j]$.

Assumption 2. *For every discrete state Q_j in the partition P of D, a rectangular over-approximation of the one-step reachable set from Q_j under $\mathcal{F}$, denoted by $R_{Q_j} = [\check{r}_1^j, \widehat{r}_1^j] \times [\check{r}_2^j, \widehat{r}_2^j] \times \ldots \times [\check{r}_n^j, \widehat{r}_n^j]$, is available.*

Assumption 3. *The random disturbance $w[k]$ in* (3) *is of the form $w[k] = \begin{bmatrix} w_1[k] & w_2[k] & \ldots & w_n[k] \end{bmatrix}^T$, where each $w_i \in W_i \subset \mathbb{R}$ has probability density function $f_{w_i}(x_i)$, W_i is an interval, and the collection $\{w_i\}_{i=1}^n$ is mutually independent. We denote by $F_{w_i}(x) = \int_{-\infty}^{x} f_{w_i}(\sigma) d\sigma$ the cumulative distribution function for w_i. Moreover, the probability density function f_{w_i} for each random variable w_i is symmetric and unimodal with mode c_i.*

For systems which cannot satisfy Assumption 1, derivations of probability bounds using over and under-approximations of labeled regions are found in [18] and can be extended to our synthesis framework to allow for a rectangular partition. Assumption 2 is relevant for wide classes of systems. For example, it was shown that a rectangular over-approximation of the reachable set from any box state could be efficiently computed under mixed-monotone dynamics, which include the well-known class of monotone systems [32] [33]. Note that, under this assumption, an over-approximation of the reachable set of state Q_j under $\mathcal{F}$ with an additive input $u \in U$ is a shifted version of the rectangular set R_{Q_j}, denoted by $R^u_{Q_j}$.

Remark 1. *Let $R_{Q_j} = [\widecheck{r}^j_1, \widehat{r}^j_1] \times [\widecheck{r}^j_2, \widehat{r}^j_2] \times \ldots \times [\widecheck{r}^j_n, \widehat{r}^j_n] \supseteq \{\mathcal{F}(x) : x \in Q_j\}$ be an over-approximation of the one-step reachable set from discrete state $Q_j \in P$ under the state update map $\mathcal{F}(x)$. Then, $R^u_{Q_j} = [\widecheck{r}^j_1 + u_1, \widehat{r}^j_1 + u_1] \times [\widecheck{r}^j_2 + u_2, \widehat{r}^j_2 + u_2] \times \ldots \times [\widecheck{r}^j_n + u_n, \widehat{r}^j_n + u_n] \supseteq \{F(x) + u : x \in Q_j\}$ is an over-approximation of the one-step reachable set from Q_j under the state update map $\mathcal{F}(x) + u$.*

In [27], we showed that under Assumptions 1 to 3 and for a fixed u, an upper bound on the probability of transition from state Q_j to state Q_ℓ is computed by placing the mode c of disturbance w, restricted to the reachable set $R^u_{Q_j}$, as close as possible to the center of Q_ℓ. A lower bound on this probability is computed by placing the mode of w as far as possible from the center of Q_ℓ.

Fact 3 ([27])**.** *For system* (3) *under Assumptions 1 to 3, an upper and lower bound on the probability of transition from state Q_j to state Q_ℓ, $Q_j, Q_\ell \in P$, under input $u = [u_1, u_2, \ldots, u_n] \in U$, are given by*

$$\begin{aligned}
\widehat{T}_{Q_j \xrightarrow{u} Q_\ell} &= \prod_{i=1}^{n} \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x_i - s^{j\to\ell}_{i,max})\, dx_i, \\
&= \prod_{i=1}^{n} \Big(F_{w_i}(b_i^\ell - s^{j\to\ell}_{i,max}) - F_{w_i}(a_i^\ell - s^{j\to\ell}_{i,max}) \Big),
\end{aligned}$$

$$\begin{aligned}
\widecheck{T}_{Q_j \xrightarrow{u} Q_\ell} &= \prod_{i=1}^{n} \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x_i - s^{j\to\ell}_{i,min})\, dx_i \\
&= \prod_{i=1}^{n} \Big(F_{w_i}(b_i^\ell - s^{j\to\ell}_{i,min}) - F_{w_i}(a_i^\ell - s^{j\to\ell}_{i,min}) \Big)
\end{aligned}$$

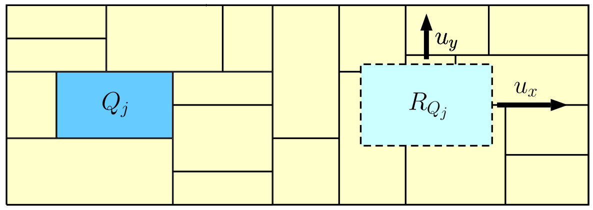


Figure 3: 2D depiction of the synthesis problem for system (3). Every state Q_j has a reachable set R_{Q_j} under $\mathcal{F}$ which is shifted when input u is applied. The permanent component construction problem requires positioning R_{Q_j} such that all instances of noise inside R_{Q_j} ensures the satisfiability of the specification. If no input can achieve this, the lower bound reachability maximization problem requires positioning R_{Q_j} such that the probability of reaching a permanent component is maximized in the worst instance of noise inside R_{Q_j}.

where F_{w_i} is the cumulative distribution function for w_i and

$$s^{j\to\ell}_{i,max} = \begin{cases} s^{\ell}_{i,max}, & \text{if } s^{\ell}_{i,max} \in [\check{r}^{j}_{i} + u_i, \widehat{r}^{j}_{i} + u_i] \\ \widehat{r}^{j}_{i} + u_i, & \text{if } s^{\ell}_{i,max} > \widehat{r}^{j}_{i} + u_i \\ \check{r}^{j}_{i} + u_i, & \text{if } s^{\ell}_{i,max} < \check{r}^{j}_{i} + u_i, \end{cases} \tag{10}$$

$$s^{j\to\ell}_{i,min} = \begin{cases} \check{r}^{j}_{i} + u_i, & \text{if } s^{j\to\ell}_{i,max} > \frac{\check{r}^{j}_{i} + \widehat{r}^{j}_{i}}{2} + u_i \\ \widehat{r}^{j}_{i} + u_i, & \text{otherwise} \end{cases}, \tag{11}$$

with $s^{\ell}_{i,max} = \frac{a^{\ell}_{i} + b^{\ell}_{i}}{2} - c_i$.

According to Remark 1, given a CIMC abstraction $\mathcal{C}$ of (3), for every state $\langle Q_j, s_i \rangle$ of the product CIMC $\mathcal{C} \otimes \mathcal{A}$ (or $\mathcal{C} \otimes \overline{\mathcal{A}}$ when the objective is to minimize the probability of satisfying Ψ), the goal is to shift the reachable set R_{Q_j} of Q_j via the application of an input u so as to maximize the lower bound probability of reaching a permanent winning component from $\langle Q_j, s_i \rangle$, as illustrated in Figure 3. As in the finite-mode case, this is achieved by first solving a qualitative problem, which we call *component construction problem*, where the greatest permanent winning component of $\mathcal{C} \otimes \mathcal{A}$ ($\mathcal{C} \otimes \overline{\mathcal{A}}$ for minimization) is created; then, a quantitative problem is solved where an input maximizing the lower bound probability of reaching these components is computed for all states of $\mathcal{C} \otimes \mathcal{A}$.

In the following sections, we first provide a solution to Subproblem 2.1 and show that, although the input space U of a CIMC $\mathcal{C}$ is uncountably infinite, the qualitative problem can be converted to a finite-mode component search by carefully selecting a finite number of inputs of U, which are identified geometrically under the stated assumptions. Subsequently, we derive an optimization problem for solving the quantitative problem and obtain the desired policies for the CIMC abstraction $\mathcal{C}$ of the system. Finally, the refinement of the partition P, from which the CIMC abstraction $\mathcal{C}$ arises, is addressed so as to reach a set level of optimality for the control policies with respect to the abstracted system.

5.1. COMPONENTS CONSTRUCTION

In this subsection, we discuss the problem of generating the greatest permanent component $(WC)_P^G$ in a product CIMC $\mathcal{C} \otimes \mathcal{A}$ when $\mathcal{C}$ abstracts (3) under Assumptions 1 to 3, that is, the transition bounds between the states of $\mathcal{C}$ are given as in Fact 3.

First, we remark that if all density functions f_{w_i} of the disturbance vector $w[k]$ have infinite support, the probability of making a transition between any two states of $\mathcal{C}$ has a non-zero lower bound for all choices of input. In this case, the IMC abstraction induced by some policy of $\mathcal{C}$ always induces MCs where all possible transitions have a non-zero probability, greatly simplifying the component construction problem. Here, we remove this restriction and alternatively assume that each w_i has a probability density function living on a bounded interval support.

Assumption 4. *All probability density functions f_{w_i} of the disturbance vector $w[k] = \begin{bmatrix} w_1[k] & w_2[k] & \dots & w_n[k] \end{bmatrix}^T$ of system (3) have a bounded interval support, that is $W_i = [\breve{w}_i, \widehat{w}_i] \subset R$ and $f_{w_i}(x_i) = 0 \;\; \forall x_i \notin W_i$.*

Recall that, in an IMC, a transition between two states Q_j and Q_i can be classified into three different categories:

- An "off" transition if $\widehat{T}(Q_j, Q_i) = 0$,
- An "on" transition if $\breve{T}(Q_j, Q_i) > 0$,
- A transition which could be either "on" or "off" depending on the assumed transition values if $\breve{T}(Q_j, Q_i) = 0$ and $\widehat{T}(Q_j, Q_i) > 0$.

The connectivity properties of an IMC $\mathcal{I}$ dictate which states belong to a permanent winning component or a largest winning component in the product between $\mathcal{I}$ and an automaton $\mathcal{A}$. Provided that the partition P of the system's domain is finite, the number of possible connectivity structures of an IMC abstraction arising from this partition is finite as well. Therefore, in the case of a CIMC abstraction, the objective is to find all connectivity structures which are achievable with the set of inputs U, choose an input $u \in U$ for all such structures and for all states Q_j of $\mathcal{C}$, and feed the resulting finite-input BMDP $\mathcal{B}$ into the component search algorithms introduced in Section 4 in order to compute the permanent winning component of the product CIMC $\mathcal{C} \otimes \mathcal{A}$, where $\mathcal{C}$ is the CIMC abstraction of (3) with domain partition P. The same procedure can be applied to find the greatest winning $(WC)_L^G$ of $\mathcal{C} \otimes \mathcal{A}$.

Fact 4. *The problem of computing the greatest permanent winning component $(WC)_P^G$ as well as the greatest winning component $(WC)_L^G$ of a product CIMC $\mathcal{C} \otimes \mathcal{A}$ can be converted to a component search in a product BMDP.*

Finding the appropriate actions for state Q_j is done by partitioning the input space U into regions such that the resulting IMCs upon application of an input in different regions are qualitatively different, as illustrated in Figure 4. We achieve this by first finding the subsets of U where, for each state Q_i reachable by Q_j under some input, the transition from Q_j to Q_i behaves differently

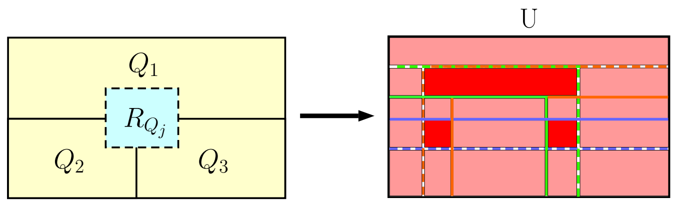


Figure 4: Sketch example of the component construction problem. The reachable set R_{Q_j} of state Q_j induces a partition of the input space U where each region produces a qualitatively different set of transitions. Dashed lines separate regions of U where the transition to some state is turned "on" or "off", solid lines separate regions where the lower bound probability of transition to some state is zero and non-zero. Blue lines correspond to state Q_1, green to Q_2 and orange to Q_3. Dark red regions highlight inputs causing several transitions to have a zero lower bound and a non-zero upper bound; such regions may need to be further partitioned.

("on", "off" or either), formalized below as *trigger regions*.

Definition 26 (Trigger Region)**.** *For any states Q_j and Q_i of P, the trigger regions of Q_j with respect to Q_i are subsets of the input space U defined as follows:*

- *The "off" trigger region $U^f_{Q_j}(Q_i) \subseteq U$ is the set of inputs such that $\widehat{T}(Q_j, u, Q_i) = 0$, $\forall u \in U^f_{Q_j}(Q_i)$,*
- *The "on" trigger region $U^o_{Q_j}(Q_i) \subseteq U$ is the set of inputs such that $\breve{T}(Q_j, u, Q_i) > 0$, $\forall u \in U^o_{Q_j}(Q_i)$,*
- *The "undecided" trigger region $U^n_{Q_j}(Q_i) \subseteq U$ is the set of inputs such that $\breve{T}(Q_j, u, Q_i) = 0$ and $\widehat{T}(Q_j, u, Q_i) > 0$, $\forall u \in U^n_{Q_j}(Q_i)$.*

Note that some of these triggers regions may evaluate to the empty set for some choices of partition P. In addition, the union of all trigger regions of state Q_j with respect to state Q_i is equal to the input space U. For system (3) with Assumptions 1 to 4, these trigger regions for state Q_j are geometrically identifiable due to the structure of both the disturbance and the over-approximation of the one-step reachable state of Q_j highlighted in Remark 1. The "off" trigger region corresponds to shifted reachable sets of Q_j where disturbance w cannot reach Q_i, the "on" trigger region corresponds to shifted reachable sets where any position of the disturbance results in an overlap with Q_i, and the "undecided" trigger region corresponds to shifted reachable sets where some positions of the disturbance cause an overlap with Q_i and some do not.

Proposition 4. *The trigger regions of state $Q_j \in P$ with respect to state $Q_i \in P$ and input space U under dynamics* (3) *with partition P and satisfying Assumptions 1 to 4 are given by*

$$
\begin{aligned}
U^f_{Q_j}(Q_i) &= \{u \in \mathbb{R}^n : \exists k \;\; \widehat{r}^j_k + u_k + \widehat{w}_k \le a^i_k \\
&\qquad \textit{or}\; \widecheck{r}^j_k + u_k + \widecheck{w}_k \ge b^i_k\} \cap\, U, \\
U^o_{Q_j}(Q_i) &= \Big\{u \in \mathbb{R}^n : \forall k \;\; \Big(\frac{\widehat{r}^j_k + \widecheck{r}^j_k}{2} + u_k \ge \frac{a^i_k + b^i_k}{2} - c_i \\
&\qquad \textit{and}\; \widehat{r}^j_k + u_k + \widecheck{w}_k \le b^i_k\Big) \;\textit{or}\; \Big(\frac{\widehat{r}^j_k + \widecheck{r}^j_k}{2} + u_k \le \frac{a^i_k + b^i_k}{2} - c_i \\
&\qquad \textit{and}\; \widecheck{r}^j_k + u_k + \widehat{w}_k \ge a^i_k\Big)\Big\} \cap\, U\;, \\
U^n_{Q_j}(Q_i) &= \Big(\mathbb{R}^n \setminus (\, U^o_{Q_j}(Q_i) \cup U^f_{Q_j}(Q_i)\,)\Big) \cap\, U\;.
\end{aligned}
$$

It follows that different overlaps of the trigger regions of state Q_j induce qualitatively different profiles for the outgoing transitions of Q_j.

Definition 27 (Trigger Regions Overlap)**.** *A Trigger Regions Overlap $\mathcal{H}_{Q_j} \subseteq U$ of state $Q_j \in P$ is a subset of the input space U such that*

$$
\mathcal{H}_{Q_j}(t_1, t_2, \ldots, t_{|P|}) = \bigcap_{i \in \{1,2,\ldots,|P|\}} U^{t_i}_{Q_j}(Q_i) \;\;,
$$

where $t_i \in \{f, o, n\}$, $\forall i$.

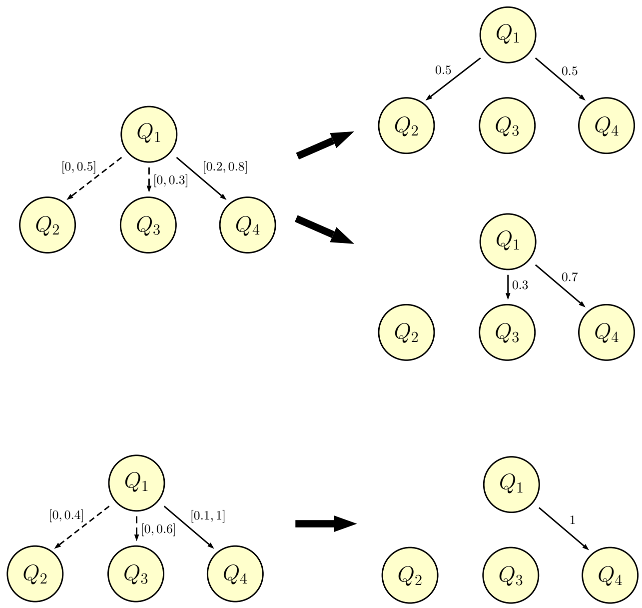


Figure 5: Two IMC transition profiles with similar transition types but different qualitative structures as discussed in Example 3. The transitions from Q_1 to the three other states are of the same type in both cases; however, while the transitions from Q_1 to Q_2 and Q_3 cannot be set to zero simultaneously for any adversary in the top example, this can be achieved in the bottom example.

Note that an overlap of two or more undecided trigger regions could produce qualitatively different transitions for several subset of its inputs and have to be further examined, as illustrated in the following example depicted in Figure 5.

Example 3. *Consider the following two transition profiles from state Q_1 to three states Q_2, Q_3 and Q_4:*

- $T(Q_1, Q_2) = [0, 0.5]$, $T(Q_1, Q_3) = [0, 0.3]$ *and* $T(Q_1, Q_4) = [0.2, 0.8]$,
- $T(Q_1, Q_2) = [0, 0.4]$, $T(Q_1, Q_3) = [0, 0.6]$ *and* $T(Q_1, Q_4) = [0.1, 1]$.

Although $T(Q_1, Q_2)$ and $T(Q_1, Q_3)$ are undecided in both cases and $T(Q_1, Q_4)$ is "on" in both cases, the two profiles are qualitatively different. In the first case, no probability assignment can simultaneously turn off the transitions from

Q_1 to Q_2 and from Q_1 to Q_3; however, in the second case, it is possible to turn off these two transitions at the same time by assigning a probability of 1 to the transition from Q_1 to Q_4.

For all states $Q_j \in P$, we denote the set of overlaps with 2 or more undecided trigger regions by $\mathcal{H}^n_{Q_j}$, and all other overlaps by $\mathcal{H}^S_{Q_j}$.

In summary, we remark that the components construction problem in a product CIMC $\mathcal{C} \otimes \mathcal{A}$ is solved by converting it to a component search in a finite-action product BMDP $\mathcal{B} \otimes \mathcal{A}$. The construction of $\mathcal{B}$ is achieved by partitioning the input space of all states Q_j of $\mathcal{C}$ into trigger region overlaps yielding qualitatively different transition profiles, and by choosing one control action per overlap in $\mathcal{H}^S_{Q_j}$, and possibly more than one control actions per overlap in $\mathcal{H}^n_{Q_j}$. Indeed, we observed in Example 3 that, for every overlap in the set $\mathcal{H}^n_{Q_j}$ of a state Q_j, we have to distinguish the sets of inputs allowing for different combinations of inactive uncertain transitions. We show that the overlaps are geometrically identified for system (3) under Assumption 1 to 4.

The input selection procedure is detailed in Algorithm 5. This algorithm chooses the minimum energy input in all overlaps in $\mathcal{H}^S_{Q_j}$ and performs a search over from the overlaps in $\mathcal{H}^n_{Q_j}$ in order to find control inputs allowing for different combinations of inactive uncertain transitions. We emphasize that the optimization problem on line 20 is non-convex under our system assumptions and is in general hard to solve. Note that Algorithm 5 in its current state may select more actions than needed from the overlaps in $\mathcal{H}^n_{Q_j}$. Indeed, our procedure is likely to choose different actions for two distinct combinations of achievable "off" uncertain transitions S and S', where none of these combinations is a strict subset of the other, while a single action may be able to accommodate these two combinations at once. Consequently, the resulting BMDP $\mathcal{B}$ may have a larger action space than necessary. This could be addressed by considering multiple such combinations at once in the constraints on line 20, at the cost of having to potentially solve a greater number of optimization problems.

Algorithm 5 Input Selection for State Q_j

1: **Input**: Sets of overlaps $\mathcal{H}^S_{Q_j}$ and $\mathcal{H}^n_{Q_j}$ of state Q_j
2: **Output**: Finite set of actions $A(Q_j)$
3: **Initialize**: $A(Q_j) := \emptyset$
4: **for** $\mathcal{H}_i \in \mathcal{H}^S_{Q_j}$ **do**
5:     $u^* := \arg\min_{u \in \mathcal{H}_i} ||u||_2^2$, $A(Q_j) \leftarrow u^*$
6: **end for**
7: **for** $\mathcal{H}_i \in \mathcal{H}^n_{Q_j}$ **do**
8:     $L := \emptyset$, $O := \emptyset$, $Y := \emptyset$
9:     For all states Q_k such that $U^o_{Q_k} \cap \mathcal{H}_i \neq \emptyset$, $O \leftarrow Q_k$
10:     For all states Q_k such that $U^n_{Q_k} \cap \mathcal{H}_i \neq \emptyset$, $Y \leftarrow Q_k$
11:     $L \leftarrow Y$
12:     **for** $S \in L$ **do**
13:         **for** $u \in A(Q_j)$ **do**
14:             Check if $\sum_{q \in O} \widehat{T}(Q_j, u, q) + \sum_{q \in Y \setminus S} \widehat{T}(Q_j, u, q) \geq 1$
15:         **end for**
16:         **if** Feasible for some $u \in A(Q_j)$ **then**
17:             Continue for-loop (Line 13)
18:         **end if**
19:         Solve $u^* = \arg\min_{u \in \mathcal{H}_i} ||u||_2^2$ such that $\sum_{q \in O} \widehat{T}(Q_j, u, q) + \sum_{q \in Y \setminus S} \widehat{T}(Q_j, u, q) \geq 1$
20:         **if** Feasible **then**
21:             $A(Q_j) \leftarrow u^*$
22:         **else**
23:             Add the $\binom{|S|}{|S|-1}$ combinations of $|S| - 1$ states of S (which are not already in L and for which no superset of states previously returned a feasible solution) to L
24:         **end if**
25:     **end for**
26: **end for**
27: **return** $A(Q_j)$

Algorithm 6 Component Construction Method for (3)

Input: Domain Partition P, input Space U, DRA $\mathcal{A}$ of specification Ψ

Output: Winning components $(WC)_P^G$ and $(WC)_L^G$ of product CIMC $\mathcal{C} \otimes \mathcal{A}$ constructed from P

Create a BMDP $\mathcal{B}$ with the same states as P and with each action set $A(Q_j)$ initialized to the empty set

Compute the overlap sets for all $Q_j \in P$ using Proposition 4 and according to Definition 27

for $Q_j \in P$ **do**

  Compute the set of actions $A(Q_j)$ using Algorithm 5 as well as their corresponding transition profiles

end for

return $(WC)_P^G$ and $(WC)_L^G$ and their corresponding control actions by applying the component search in Algorithm 1 and 2 to $\mathcal{B} \otimes \mathcal{A}$

Algorithm 6 summarizes the component construction procedure and outputs the greatest permanent winning component $(WC)_P^G$ of a product CIMC $\mathcal{C} \otimes \mathcal{A}$, as well as its greatest winning component $(WC)_L^G$, where $\mathcal{C}$ serves as a CIMC abstraction of system (3).

5.2. REACHABILITY MAXIMIZATION

To devise an optimal control policy for system (3) abstracted by a CIMC $\mathcal{C}$, we now have to find the control inputs in the continuous set U maximizing the lower bound probability of reaching $(WC)_P^G$ in a product CIMC according to Theorem 1.

Our approach is inspired from the lower bound reachability maximization algorithm for BMDPs in [16]. In this algorithm, the procedure for computing a control policy maximizing the lower bound probability of reaching a target set of states G in a finite-action BMDP is based on value iteration and is as follows:

1. Initialize a probability vector $W^0 = [p_1^0, p_2^0, \ldots, p_m^0]$ where $p_i^0 = 1$ if $p_i \in G$ and 0 otherwise.

2. At each time step k, construct an ascending ordering $\mathcal{O}_k = q_1 q_2 \ldots q_m$, $q_i \in Q$, of the states such that $p_1^k \leq p_2^k \leq \ldots \leq p_m^k$.

3. For each state Q_j and for each action in $A(Q_j)$, allocate as much probability mass z_1^j as possible to state q_1, then allocate as much probability mass z_2^j as possible to state q_2 with the amount of probability left, etc., in order to construct the worst possible assignment of the probabilities allowed by the IMC under each action with respect to the objective of reaching G.

4. For each state, pick the action from $A(Q_j)$ that yields the highest worst-case probability $p_i^{k+1} = \sum_{j=1}^{m} p_j^k z_j^i$ of reaching G.

5. Update the probability vector W^{k+1} such that $p_i^{k+1} = \sum_{j=1}^{m} p_j^k z_j^i$, with p_i^{k+1} being the computed probability under the chosen action at state Q_i, and construct a new ordering $\mathcal{O}^{k+1}$. Repeat this process until vector W converges [34] and the last selected actions are the lower bound reachability maximizing actions for all states.

We propose to follow the same procedure for computing lower bound maximizing policies in the product CIMC $\mathcal{C} \otimes \mathcal{A}$. However, while finite-mode systems rely on exhaustive search over every possible action to choose the most optimal one at each step k of the above algorithm, systems with a continuous set of inputs U require solving an optimization problem at Step 3 of the above algorithm to find the reachability maximizing input u for all states $\langle Q_j, s_i \rangle$ of the product CIMC $\mathcal{C} \otimes \mathcal{A}$.

We first note that the transition bound functions in $\mathcal{C} \otimes \mathcal{A}$ are determined by the transition bound functions in $\mathcal{C}$, as seen in the definition of a product CIMC. We formulate an optimization problem that outputs the best action $u \in U$ for state $\langle Q_j, s_i \rangle$ at some time step k of the aforementioned algorithm. Consider the set of states $\{q_\ell\}_{\ell=1}^{m}$ which are reachable by $\langle Q_j, s_i \rangle$ under some input, that is $\exists u \in U$ such that $\widehat{T}(\langle Q_j, s_i \rangle, u, q_\ell) > 0,\ i = 1, 2, \ldots, m$. We denote the probability of reaching the desired component from state q_ℓ at the current time step of the algorithm by p_ℓ. Consider an ascending ordering $\mathcal{O} = q_1 q_2 q_3 \ldots q_m$

of the states reachable by $\langle Q_j, s_i \rangle$ such that $p_1 \leq p_2 \leq \ldots \leq p_m$. Step 3 and 4 of the reachability maximization algorithm for the continuous input case are formulated as the optimization program

$$\begin{aligned} &\max_{u \in U} \ \sum_{\ell=1}^{m} p_\ell z_\ell && (12) \\ &s.t. \quad z_\ell = \min\left\{ \widehat{T}\big(\left\langle Q_j, s_i \right\rangle, u, q_\ell\big), \ \ 1 - \sum_{k=1}^{\ell-1} z_k - \sum_{k=\ell+1}^{m} \widecheck{T}\big(\left\langle Q_j, s_i \right\rangle, u, q_k\big) \right\}, \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \ell = 1, 2, 3, \ldots, m \ , \end{aligned}$$

where the lower and upper bound terms are given by (10) and (11) for the specific case of system (3) under Assumption 1 to 3, rendering this problem non-convex. The constraints ensure that, for a given input u, each state in $\mathcal{O}$ is allocated either its upper bound probability of transition or the maximum probability mass allowed by the lower bound transition probability of the following states in $\mathcal{O}$ and the probability mass distributed to the preceding states in $\mathcal{O}$. In the case study section of this paper, we tackle optimization problem (12) using numerical heuristics.

Unlike in the finite-mode case, this value iteration procedure for continuous input sets is not guaranteed to converge in a finite number of steps. Therefore, we suggest computing the maximum change in the reachability probability among all states of $\mathcal{C} \otimes \mathcal{A}$ at each step of the algorithm, and terminating the procedure once this change reaches a user-defined convergence threshold.

5.3. STATE SPACE REFINEMENT

Finally, we discuss partition refinement for system (3) to address Subproblem 2.2.

The quality of the controller designed in the CIMC abstraction $\mathcal{C}$ with respect to continuous states of (3) can be assessed as in Section 4 for the finite-mode system case. In light of Subsection 4.3, we need to construct a best-case and a worst-case product MC induced by the product CIMC $\mathcal{C} \otimes \mathcal{A}$ to determine the suboptimality factor of each state of $\mathcal{C} \otimes \mathcal{A}$. In particular, when devising a maximizing control policy, a best-case MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ is constructed by solving an upper bound reachability maximization problem on the greatest winning component $(WC)_L^G$ of the product CIMC $\mathcal{C} \otimes \mathcal{A}$, where $\mathcal{C}$ is the CIMC abstraction of (3) under the current partition P. When devising a minimizing control policy, a best-case MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ is constructed by solving an upper bound reachability maximization problem on the greatest winning component of the product CIMC $\mathcal{C} \otimes \overline{\mathcal{A}}$, where $\mathcal{C}$ is the CIMC abstraction of (3). These upper bound reachability maximization problems are addressed using a similar procedure as in Subsection 5.2, with the difference that the ordering $\mathcal{O} = q_1 q_2 q_3 \ldots q_m$ in the optimization program (12) is now descending with respect to the probability of reaching the target set G, that is $p_1 \geq p_2 \geq \ldots \geq p_m$.

Propositions 1 to 3, which discuss some properties that are passed from a partition to its refinements for the finite-mode case, are also valid in this continuous input framework. In particular, as in the finite-mode case, subsets of the input space U which can be shown to be certainly suboptimal may be removed. To find such subsets, we suggest building a partition $U(\langle Q_j, s_i \rangle) = \{U_n(\langle Q_j, s_i \rangle)\}_{n=1}^{k}$ of the input space for all states $\langle Q_j, s_i \rangle$ of $\mathcal{C} \otimes \mathcal{A}$. Then, for all subsets U_n, an upper bound maximization step on $(WC)_L^G$ is conducted; subsets yielding an upper bound on the maximum upper bound probability of reaching an accepting BSCC from $\langle Q_j, s_i \rangle$ which is lower than the lower bound produced by $(\widehat{\mu}_{\Psi}^{low})_{\otimes}(\langle Q_j, s_i \rangle)$ (respectively, by $(\widetilde{\mu}_{\Psi}^{up})_{\otimes}(\langle Q_j, s_i \rangle)$ for the case of minimization) are suboptimal with respect to the entire input set of $\langle Q_j, s_i \rangle$ and are removed from $U(\langle Q_j, s_i \rangle)$, as depicted in Figure 6. Note that a finer discretization of the input space $U(\langle Q_j, s_i \rangle)$ for the update step may result in the removal of a greater volume of suboptimal inputs from $U(\langle Q_j, s_i \rangle)$ at each iteration of the synthesis procedure, allowing to "zoom in" on better inputs for state $\langle Q_j, s_i \rangle$ in fewer iterations at the expense of having to solve a larger number of optimization problems per iteration.

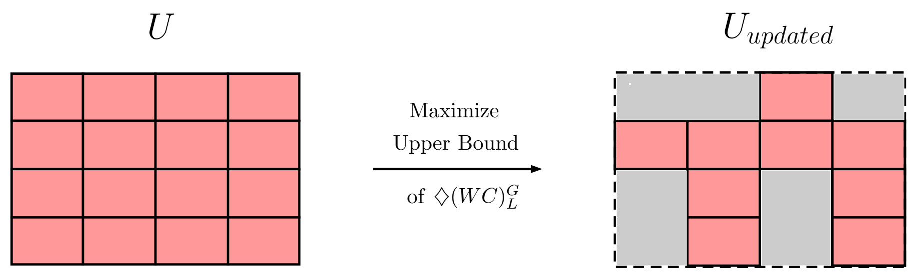


Figure 6: Sketch of an input space update before refinement of the domain partition. The original input space U of the considered state is gridded and the upper bound probability of reaching $(WC)^G_L$ is maximized for all subsets of the grid. The subsets producing suboptimal bounds are shown in gray and are discarded.

Finally, once $(\mathcal{M}^{\mathcal{A}}_{\otimes})_u$ and $(\mathcal{M}^{\mathcal{A}}_{\otimes})_l$ are generated and all input sets are updated, the scoring and refinement procedure are performed identical to the finite-mode case. After refinement, the trigger regions and overlaps of a state Q_j calculated in Algorithm 6 have to be re-computed only if there exists a state $\langle Q_j, s_i \rangle$ for some i which belonged to the difference between the greatest winning component and the greatest permanent winning component in the previous abstraction, as only such a state could potentially be a member of a new permanent winning set of states in the refined abstraction as a consequence of Proposition 2, and if either Q_j has been refined into children states in which case the trigger regions of the children states have to be determined, or a state that was reachable from Q_j under some action in the input space $U(Q_j)$ has been refined into children states with respect to which the trigger regions have to be evaluated.

The controller synthesis algorithm for continuous input systems is summarized in Algorithm 7. The run-time complexity of most sub-algorithms of Algorithm 7 has already been presented in Section 4. Additionally, as previously discussed, the input selection in Algorithm 5 grows combinatorially in $|Q|$ and the computation of overlaps in Algorithm 6 runs exponentially in $|Q|$. Lastly, the computational complexity of the algorithm strongly depends on the optimization method used for solving the reachability maximization step.

Algorithm 7 Controller Synthesis for Continuous Input Systems

Input: Partition P_0 of domain D of (1), ω-regular property Ψ (complement property $\overline{\Psi}$) and corresponding DRA $\mathcal{A}$ ($\overline{\mathcal{A}}$), target controller precision ϵ_{thr}

Output: Maximizing (minimizing) switching policy $\widehat{\mu}_{\Psi}^{low}$ ($\widecheck{\mu}_{\Psi}^{up}$), final partition P_{fin}

Initialize: $\epsilon_{max} := 1$, $i := 0$

while $\epsilon_{max} > \epsilon_{thr}$ **do**

  Compute the sets $(WC)_P^G$ and $(WC)_L^G$ of the product CIMC $\mathcal{C} \otimes \mathcal{A}$ ($\mathcal{C} \otimes \overline{\mathcal{A}}$) constructed from P_i using Algorithm 6

  Compute the policies $\widehat{\mu}_{\Psi}^{low}$ and $\widehat{\mu}_{\Psi}^{up}$ ($\widecheck{\mu}_{\Psi}^{up}$ and $\widecheck{\mu}_{\Psi}^{low}$) of the CIMC $\mathcal{C}$ according to Subsection 5.2

  Compute ϵ_{max} using (9)

  if $\epsilon_{max} > \epsilon_{thr}$ **then**

    Compute the best-case and worst-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ as discussed in Subsection 5.3.

    Construct a partition $\{U_n(\langle Q_j, s_m\rangle)\}_{n=1}^{k}$ of the input space $U(\langle Q_j, s_m\rangle)$ of all states $\langle Q_j, s_m\rangle$ of the product CIMC $\mathcal{C} \otimes \mathcal{A}$ ($\mathcal{C} \otimes \overline{\mathcal{A}}$)

    for $U_n(\langle Q_j, s_m\rangle) \in U(\langle Q_j, s_m\rangle)$ **do**

      Maximize the upper bound probability of $\Diamond (WC)_L^G$ from $\langle Q_j, s_m\rangle$ with the set of inputs $U_n(\langle Q_j, s_m\rangle)$

    end for

    Apply the scoring procedure in Algorithm 3 and refine all states in P_i with a score above a user-defined threshold to produce P_{i+1}

    Update the set of inputs of all states in the product CIMC $\mathcal{C} \otimes \mathcal{A}$ ($\mathcal{C} \otimes \overline{\mathcal{A}}$) constructed from P_{i+1} as discussed in Subsection 5.3.

    $i := i + 1$

  end if

end while

return $\widehat{\mu}_{\Psi}^{low}$ ($\widecheck{\mu}_{\Psi}^{up}$), $P_{fin} := P_i$

6. CASE STUDY

We now present a numerical example to demonstrate the synthesis procedures derived in previous sections. The code used to generate this example was written in Python 2.7 and is available at `https://github.com/gtfactslab/StochasticSynthesis`. All computations were conducted on the Partnership for an Advanced Computing Environment (PACE) Georgia Tech cluster [35] which offered 120GB of memory. The examples in Section 6.1 were performed on a single core, while those in Section 6.2 were distributed over 4 cores.

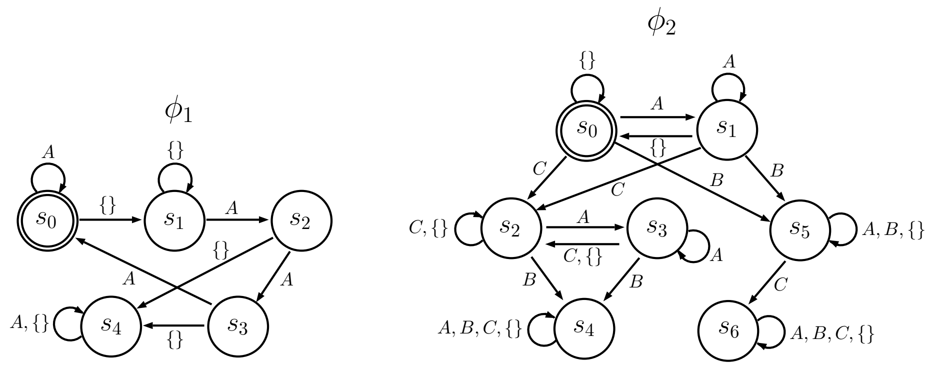
We consider a stochastic model of a bistable switch with dynamics

$$\begin{aligned} x_1[k+1] &= x_1[k] + (\ -ax_1[k] + x_2[k]\ ) \cdot \Delta T + u_1 + w_1 \\ x_2[k+1] &= x_2[k] + \left(\frac{(x_1[k])^2}{(x_1[k])^2+1} - bx_2[k] \right) \cdot \Delta T + u_2 + w_2 \ , \end{aligned} \tag{13}$$

where w_1 and w_2 are independent truncated Gaussian random variables sampled at each time step. $w_1 \sim \mathcal{N}(\mu = -0.3; \sigma^2 = 0.1)$ and is truncated on $[-0.4, -0.2]$; w_2 is similarly defined. We will consider two sets of inputs in this case study: the continuous set $U = [-0.05, 0.05] \times [-0.05, 0.05]$ and the finite set $U_{fin} = \{[0,0]^T, [0.05,0]^T, [-0.05,0]^T, [0,0.05]^T, [0,-0.05]^T\}$ which is a subset of U. The domain D of (13) is $[0.0, 4.0] \times [0.0, 4.0]$. To keep the system self-contained in D, we assume that any time the disturbance would push the trajectory outside of D, it is actually maintained on the boundary of D. We choose the parameters $a = 1.3$, $b = 0.25$ and $\Delta T = 0.05$. Our goal is to synthesize a controller for (13) that maximizes the probability of satisfying the LTL specifications

$$\begin{aligned} \phi_1 &= \Box((\neg A \wedge \bigcirc A) \rightarrow (\bigcirc \bigcirc A \wedge \bigcirc \bigcirc \bigcirc A)) \ , \\ \phi_2 &= (\Box\Diamond A \rightarrow \Diamond B) \wedge (\Diamond C \rightarrow \Box \neg B) \ , \end{aligned}$$

where ϕ_1 translates to " always remain in an A state for at least 2 more time steps when entering an A state" and ϕ_2 translates to "reach a B state if the trajectory always eventually returns to an A state, and never reach a B state if the trajectory reaches a C state" in natural language. The DRA corresponding to specification ϕ_1 contains 5 states and has 1 Rabin pair, while the DRA representing ϕ_2 contains 7 states and has 3 Rabin pairs. Schematic representations



$Acc = \{(\emptyset,\ \{s_0, s_1\})\}$ $\quad$ $Acc = \{(\{s_1\},\ \{s_0\}), (\{s_3\}, \{s_2\}), (\emptyset,\ \{s_5\})\}$

Figure 7: Possible DRAs for specification ϕ_1 (Left) and specification ϕ_2 (Right), where double-edged circles represent the initial states of the DRAs. Note that these DRAs assume the convention that the state initialization "counts as a transition", i.e., when a state of the BMDP Q_j is chosen as initial state, the product BMDP transitions from $\langle Q_j, s_0\rangle$ to $\langle Q_j, \delta(s_0, L(Q_j))\rangle$.

of these DRAs are found in Figure 7. Initial partitions of the domain D along with the labeling of the states are presented in the next subsections. First, we synthesize controllers using the finite set of inputs U_{fin}. Second, we devise control policies from the continuous set of inputs U. Finally, we compile some observations and concluding remarks in a discussion subsection.

6.1. FINITE-MODE SYNTHESIS

First, we synthesize a switching policy for maximizing the probability of satisfying ϕ_1 and ϕ_2 in (13) using the finite set U_{fin}, where each input corresponds to one mode, and applying the synthesis Algorithm 4 for finite-mode systems with a target precision $\epsilon_{thr} = 0.30$. At each refinement step, states of the current partition with a refinement score that is greater than 5% of the maximum score are chosen to be refined and split in half along their greatest dimension. The deterministic portion of the dynamics of system (13) are known to be monotone. Therefore, BMDP abstractions of (13) for rectangular partitions of D are efficiently computed using the technique in [27] for each mode.

The initial partition of the domain D for specification ϕ_1 is given in Figure 8 (Left), and the initial partition for specification ϕ_2 is in Figure 9 (Left). At each refinement step, the states selected for refinement are split in half along their greatest dimension.

The component search algorithm is conducted at each iteration of the while loop of Algorithm 4 until the set of potential accepting BSCCs $(U)^G_{pot}$ becomes empty, in which case the component construction procedure is skipped and the lower bound maximization problem in Line 6 is performed on the latest known version of the greatest permanent winning component $(WC)^G_P$. As no new permanent accepting BSCCs can be constructed anywhere else in the state space in this scenario, an under-approximation of $(WC)^G_P$ containing all possible permanent BSCCs without all permanent sink states is sufficient for the reachability problem. Note that $(WC)^G_P$ can be updated if permanent sink states with a lower bound of 1 are constructed during the lower bound maximization step.

The controller synthesis procedure for specification ϕ_1 terminated in 13 hours and 27 minutes with a greatest suboptimality factor $\epsilon_{max} = 0.2999$, and created 18418 states in 18 refinement steps, corresponding to 92090 states in the product BMDP constructed from the final partition. The final refined partition is shown in Figure (8) (Right). For specification ϕ_2, the procedure terminated in 38 minutes with a greatest suboptimality factor $\epsilon_{max} = 0.2998$ and created 7711 states in 15 refinement steps, corresponding to 53977 states in the product BMDP constructed from the final partition. The final refined partition is shown in Figure (9) (Right).

The cumulative execution time against the number of refinement steps is plotted in Figure 10 for specification ϕ_1 (Left) and specification ϕ_2 (Right). The average number of actions left at each state of the product BMDP $\mathcal{B} \otimes \mathcal{A}$ after each refinement step is displayed in Figure 11 for specification ϕ_1 (Left) and specification ϕ_2 (Right). Lastly, three possible metrics of precision for the computed controller — namely, the greatest suboptimality factor, average suboptimality factor of the product BMDP and fractions of states above the target precision ϵ_{thr} — as a function of the number of refinement steps are shown in Figure (12) for specification ϕ_1 (Left) and specification ϕ_2 (Right).

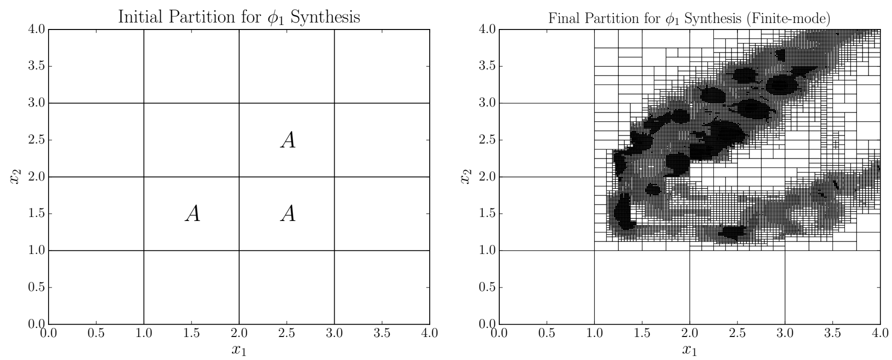


Figure 8: Initial domain partition with state labeling (Left) and final domain partition upon synthesis of a controller for maximizing the probability of satisfying ϕ_1 in (13) using the finite set of inputs U_{fin} after 18 refinement steps (Right). The final partition contains 18418 states, corresponding to 92090 states in the resulting product BMDP abstraction.

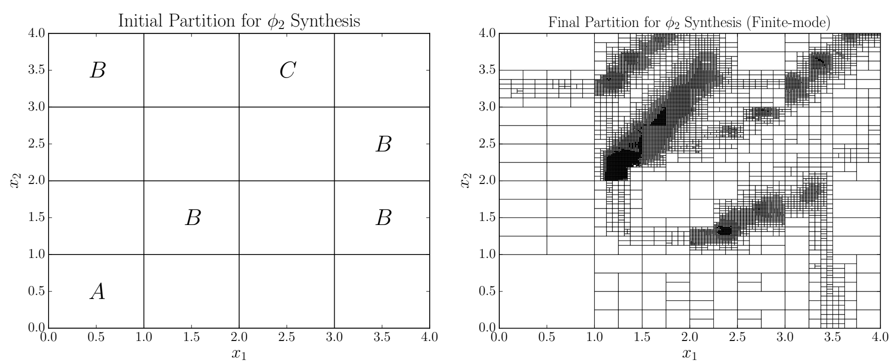


Figure 9: Initial domain partition with state labeling (Left) and final domain partition upon synthesis of a controller for maximizing the probability of satisfying ϕ_2 in (13) using the finite set of inputs U_{fin} after 15 refinement steps (Right). The final partition contains 7711 states, corresponding to 53977 states in the resulting product BMDP abstraction.

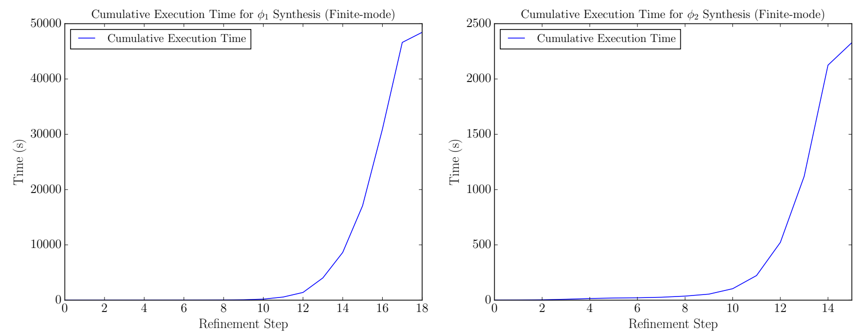


Figure 10: Cumulative execution time of the synthesis procedure with the finite input set U_{fin} as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right). The synthesis procedure for ϕ_1 terminated in 13 hours and 27 minutes; the synthesis procedure for ϕ_2 terminated in 38 minutes

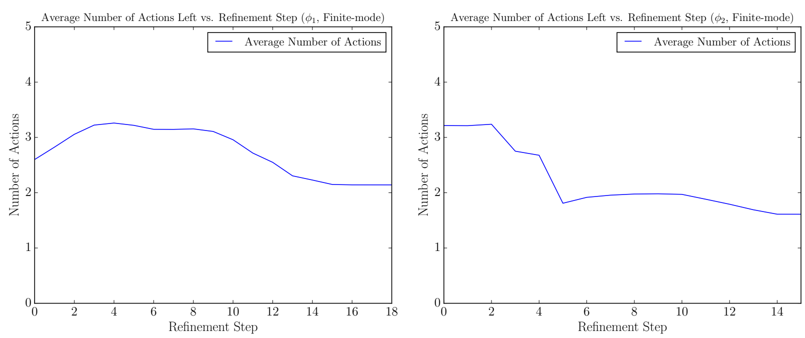


Figure 11: Average number of actions left at each state of the product BMDP as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right).

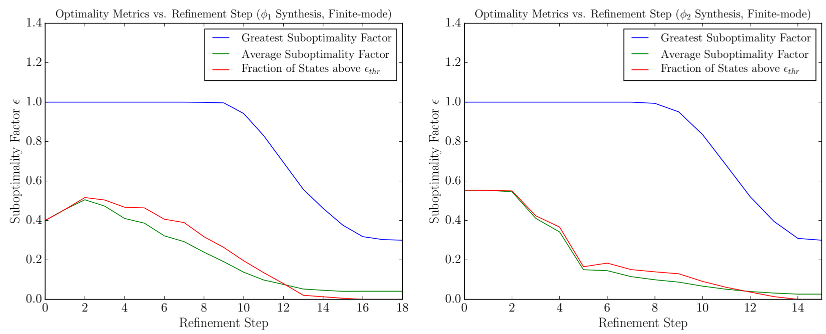


Figure 12: Different metrics of precision for the controller computed from the finite input set U_{fin} as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right). The synthesis algorithm reaches the target $\epsilon_{thr} = 0.30$ for both specifications. This means that the probability of satisfying the specifications can only increase by a maximum of 0.30 from all possible states of the abstracted system by choosing another switching policy.

6.2. CONTINUOUS INPUT SET SYNTHESIS

Next, we generate a control policy from the set of continuous inputs U by applying Algorithm 7. The desired threshold precision is chosen to be $\epsilon_{thr} = 0.30$. At each refinement step, states of the current partition with a refinement score that is greater than 1% of the maximum score are chosen to be refined and split in half along their greatest dimension. Tight rectangular over-approximation of the deterministic reachable set of (13) are obtained efficiently from the results in [32] thanks to the monotone property of the state update map. The input space of all states in the product CIMC is stored as a union of rectangles. When evaluating the optimality of the synthesized controller before every refinement step, we partition each rectangle of the input space of all states into 4 rectangles of equal area. This allows the input spaces to always remain a union of rectangles in case some sub-regions of the input space were removed, as in Figure 6, which facilitates the computation of the overlaps in Algorithm 6.

The non-convex optimization problem in Algorithm 5, line 14, and the non-convex optimization problem (12) are solved by gridding each rectangle U_i of the input space of interest with an N-by-N meshgrid, where $N = \max\{N_{min},$

$\lceil N_{init} \cdot \frac{Area(U_i)}{Area(U)} \rceil\}$ with $N_{min} = 3$ and $N_{init} = 12$, and using a convex solver from all points of the grid. The component construction algorithm is conducted at each iteration of the while loop of Algorithm 7 until the set of potential accepting BSCCs $(U)^G_{pot}$ becomes empty, as in the finite-mode examples. The threshold of convergence for the reachability value iteration scheme is set to 0.01.

The controller synthesis procedure for specification ϕ_1 was manually terminated after 12 refinement steps which lasted 22 hours and 32 minutes with a greatest suboptimality factor $\epsilon_{max} = 0.8705$, and created 16079 states, corresponding to 80395 states in the product BMDP constructed from the final partition. The final refined partition is displayed in Figure 13 (Right). The procedure for specification ϕ_2 was manually terminated after 14 refinement steps which lasted 73 hours with a greatest suboptimality factor $\epsilon_{max} = 0.7754$, and created 24607 states in 14 refinement steps, corresponding to 172249 states in the product BMDP constructed from the final partition. The final refined partition is displayed in Figure 14 (Right).

The cumulative execution time against the number of refinement steps is plotted in Figure 16 for specification ϕ_1 (Left) and specification ϕ_2 (Right). The original input space for all states of the system is shown in Figure 15, along with the reduced input space with respect to specification ϕ_1 and ϕ_2 upon refinement for 2 states of the system. Finally, the greatest suboptimality factor, average suboptimality factor of the product CIMC and fractions of states above the target precision ϵ_{thr} as a function of the number of refinement steps are shown in Figure (17) for specification ϕ_1 (Left) and specification ϕ_2 (Right).

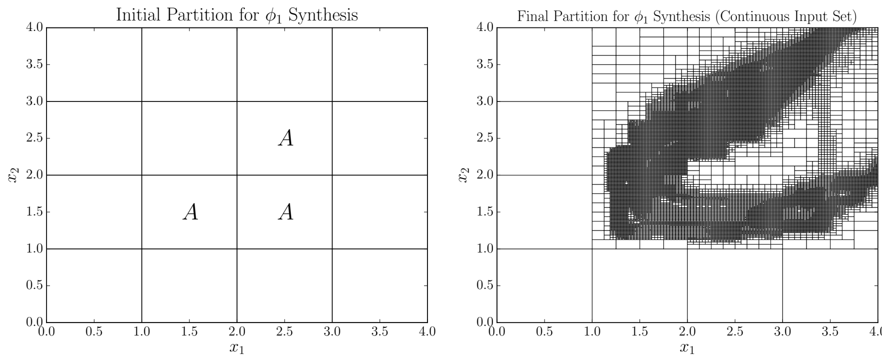


Figure 13: Initial domain partition with state labeling (Left) and final domain partition upon synthesis of a controller for maximizing the probability of satisfying ϕ_1 using the continuous set of inputs U after 12 refinement steps (Right). The final partition contains 16079 states, corresponding to 80395 states in the resulting product CIMC abstraction.

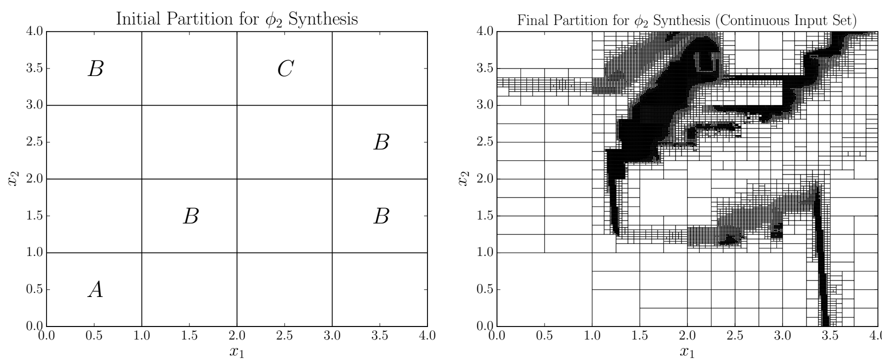


Figure 14: Initial domain partition with state labeling (Left) and final domain partition upon synthesis of a controller for maximizing the probability of satisfying ϕ_2 using the continuous set of inputs U after 14 refinement steps (Right). The final partition contains 24607 states, corresponding to 172249 states in the resulting product CIMC abstraction.

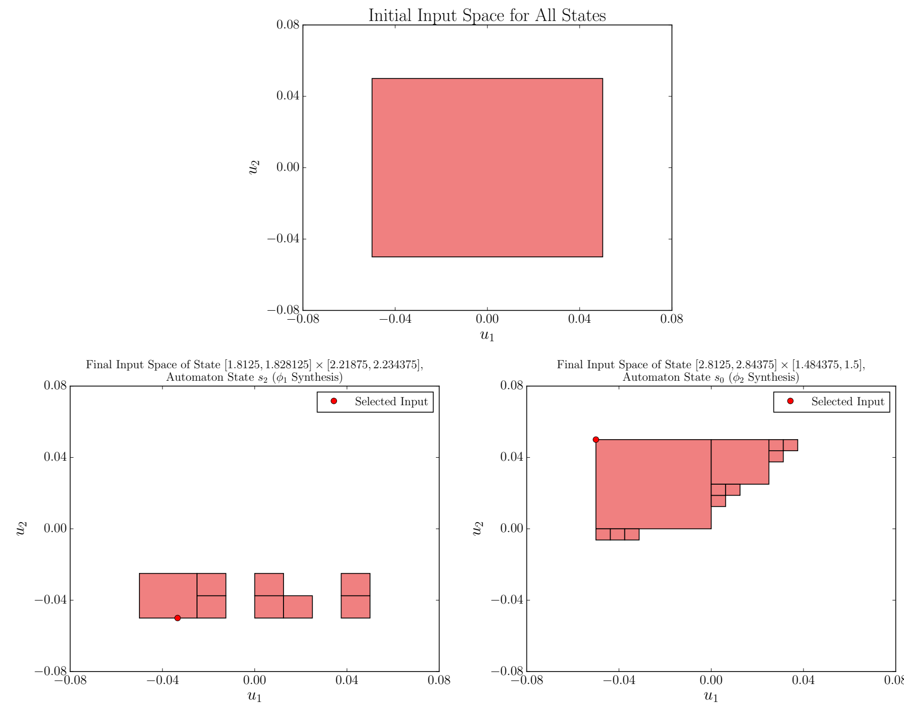


Figure 15: Plot of the initial input space U (Top) for all states of the state space. The reduced input space of state $[1.8125, 1.828125] \times [2.21875, 2.234375]$ with automaton state s_2 with respect to specification ϕ_1 upon refinement is shown in the bottom left plot. The reduced input space of state $[2.8125, 2.84375] \times [1.484375, 1.5]$ with automaton state s_0 with respect to specification ϕ_2 upon refinement is shown in the bottom right plot.

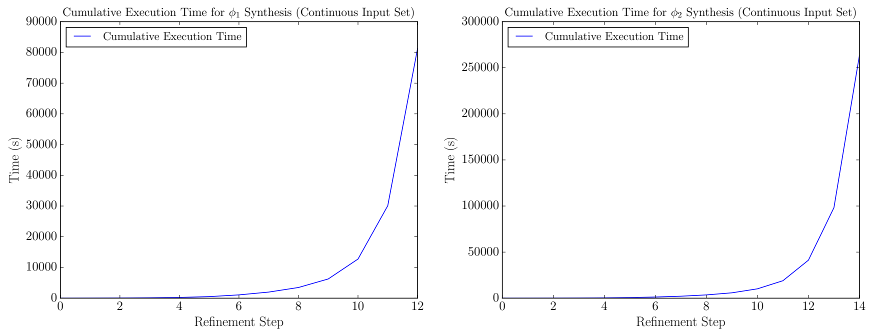


Figure 16: Cumulative execution time of the synthesis procedure with the continuous input set U as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right).

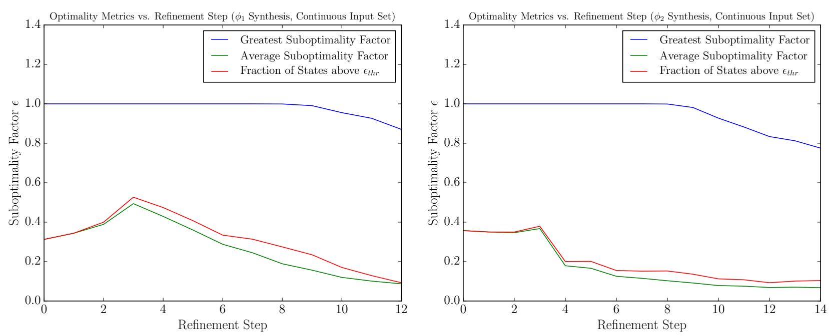


Figure 17: Different metrics of precision for the computed controller with the continuous input set as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right). The synthesis algorithm is manually terminated before reaching the target $\epsilon_{thr} = 0.30$ for both specifications.

6.3. DISCUSSION

The synthesis algorithms presented in the previous sections successfully designed controllers from both the finite set of inputs U_{fin} and the continuous set of inputs U. Moreover, the algorithms conducted synthesis for two different complex specifications that existing tools could not accommodate, and automatically produced a targeted domain refinement for the two cases so as to achieve a higher level of optimality for the computed controllers. We also consider our approach to be an improvement over related synthesis works in terms of scalability; for instance, our finite-mode algorithm is orders of magnitude faster than the technique used for the synthesis case study in [16], which designed a switching policy for a 3-mode 2D linear system with a simple reachability specification over the course of several days.

To further demonstrate the synthesis procedure, in Figure 18 (Top), we display the verification of system (13) against ϕ_1 without any available input with respect to a satisfaction threshold of 0.8 from the work in [24], where the initial states in green have a probability of satisfying the specification which is greater than 0.8, the states in red have a probability which is below 0.8, and the states in yellow are undecided at the level of precision of the available partition. In the bottom left, we display the verification of system (13) under the computed switching policy in the finite-mode section, and in the bottom right, we show the verification of system (13) under the computed control policy from the continuous set of inputs. As expected, moving counter-clockwise through the plots, we observe that some red regions of the state-space are converted to green regions.

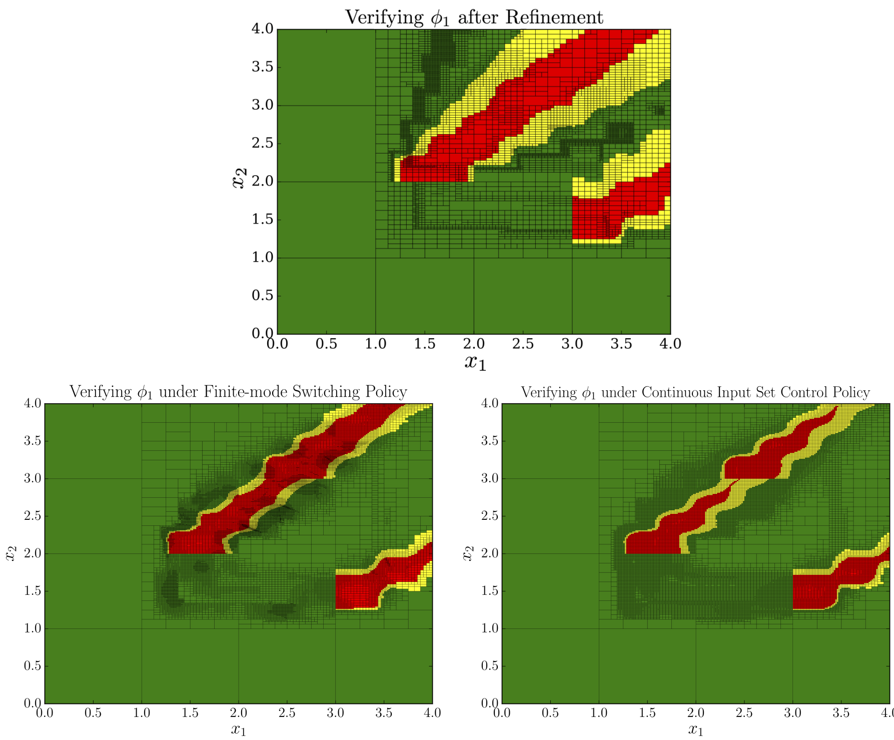


Figure 18: Verification of system (13) against ϕ_1 with respect to a satisfaction threshold of 0.8 without any input (Top), and under both the switching policy computed from the finite input set U_{fin} (Bottom Left) and the control policy computed from the continuous input space U (Bottom Right). The initial states in green have a probability of satisfying the specification which is greater than 0.8, the states in red have a probability which is below 0.8, and the states in yellow are undecided. The controlled versions of (13) convert some red regions of the state-space in the uncontrolled case to green regions.

It is evident that computing controllers from a continuous set of inputs requires a more significant amount of computational effort compared to the finite input case. The largest portion of the continuous-input synthesis algorithm is expended solving the optimization problems for the value iteration step of the procedure, which is the clear scalability bottleneck of our current implementation. Moreover, we notice that the greatest suboptimality factor decreases at a slower rate as a function of refinement steps in the continuous input case than in the finite mode case, which causes a much finer partition of the domain and is the reason for the manual termination in the former example. We explain this phenomenon by observing that the suboptimality factor is more dependent on the abstraction error when using the continuous set of inputs. To see this, consider an optimal input u^* computed for a state of the product CIMC $\mathcal{C} \otimes \mathcal{A}$, yielding an interval of satisfaction $[a, b]$ for this state. Now, consider another input $u^* + \epsilon$ for a small disturbance ϵ. Assuming the dynamics of interest are continuous, it follows that the interval of satisfaction under the disturbed input is $[a + \epsilon_a, b + \epsilon_b]$. Therefore, the suboptimality factor for this state will be at least $b + \epsilon_b - a \approx b - a$, which is the size of the satisfaction interval of the considered state under the computed optimal input. Nonetheless, the algorithm still results in overall progress towards the goal optimality across all metrics as it performs more refinement steps.

7. CONCLUSION

In this paper, we developed abstraction-based controller synthesis techniques for stochastic systems with ω-regular objectives. First, we showed a method to compute switching policies in stochastic systems with a finite number of modes by performing a permanent component search and a reachability maximization task in an abstraction of the dynamics. We proposed a specification-guided domain partition refinement scheme which targets states causing the most uncertainty in the abstraction and discards the system modes that are guaranteed to be suboptimal. We extended these results to stochastic systems with a continuous set of inputs and designed a synthesis method for the specific class of affine-in-input and affine-in-disturbance systems. Finally, we presented a numerical example where controller synthesis is conducted for both finite and continuous input sets on a nonlinear system with complex temporal logic tasks.

Future works will further explore the relationship between original partitions and their refined versions to reduce the number of operations performed in the components search and reachability algorithms after each refinement step and consequently improve scalability of our technique. An adaptation of these algorithms to guarantee a monotone decrease of the suboptimality factor throughout the synthesis procedure will also be investigated. Obtaining formal convergence guarantees of the refinement heuristic is another important issue.

References

[1] A. Pnueli, The temporal logic of programs, in: 18th Annual Symposium on Foundations of Computer Science (sfcs 1977), IEEE, pp. 46–57.

[2] W. Thomas, Automata on infinite objects, in: Formal models and semantics, Elsevier, 1990, pp. 133–191.

[3] P. Tabuada, G. J. Pappas, Linear time logic control of discrete-time linear systems, IEEE Transactions on Automatic Control 51 (12) (2006) 1862–1877.

[4] M. Kloetzer, C. Belta, A fully automated framework for control of linear systems from temporal logic specifications, IEEE Transactions on Automatic Control 53 (1) (2008) 287–297.

[5] B. Yordanov, J. Tumova, I. Cerna, J. Barnat, C. Belta, Temporal logic control of discrete-time piecewise affine systems, IEEE Transactions on Automatic Control 57 (6) (2011) 1491–1504.

[6] M. Rungger, M. Mazo Jr, P. Tabuada, Specification-guided controller synthesis for linear systems and safe linear-time temporal logic, in: Proceedings of the 16th international conference on Hybrid systems: computation and control, ACM, 2013, pp. 333–342.

[7] M. Rungger, M. Zamani, Scots: A tool for the synthesis of symbolic controllers, in: Proceedings of the 19th international conference on hybrid systems: Computation and control, ACM, 2016, pp. 99–104.

[8] J. Liu, N. Ozay, Finite abstractions with robustness margins for temporal logic-based control synthesis, Nonlinear Analysis: Hybrid Systems 22 (2016) 1–15.

[9] S. Sadraddini, C. Belta, Formal synthesis of control strategies for positive monotone systems, IEEE Transactions on Automatic Control 64 (2) (2018) 480–495.

[10] C. Belta, B. Yordanov, E. A. Gol, Formal methods for discrete-time dynamical systems, Vol. 89, Springer, 2017.

[11] C. Baier, J.-P. Katoen, K. G. Larsen, Principles of model checking, MIT press, 2008.

[12] M. L. Bujorianu, J. Lygeros, M. C. Bujorianu, Bisimulation for general stochastic hybrid systems, in: International Workshop on Hybrid Systems: Computation and Control, Springer, 2005, pp. 198–214.

[13] A. Abate, A. D'Innocenzo, M. D. Di Benedetto, S. S. Sastry, Markov set-chains as abstractions of stochastic hybrid systems, in: International Workshop on Hybrid Systems: Computation and Control, Springer, 2008, pp. 1–15.

[14] A. Abate, A. D'Innocenzo, M. D. Di Benedetto, Approximate abstractions of stochastic hybrid systems, IEEE Transactions on Automatic Control 56 (11) (2011) 2688–2694.

[15] M. Kwiatkowska, G. Norman, D. Parker, Prism 4.0: Verification of probabilistic real-time systems, in: International conference on computer aided verification, Springer, 2011, pp. 585–591.

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

[17] R. Givan, S. Leach, T. Dean, Bounded-parameter Markov decision processes, Artificial Intelligence 122 (1-2) (2000) 71–109.

[18] N. Cauchi, L. Laurenti, M. Lahijanian, A. Abate, M. Kwiatkowska, L. Cardelli, Efficiency through Uncertainty: Scalable Formal Synthesis for

Stochastic Hybrid Systems, arXiv e-prints (2019) arXiv:1901.01576`arXiv:1901.01576`.

[19] N. Cauchi, K. Degiorgio, A. Abate, StocHy: automated verification and synthesis of stochastic processes, arXiv preprint arXiv:1901.10287 (2019).

[20] K. Y. Rozier, Linear temporal logic symbolic model checking, Computer Science Review 5 (2) (2011) 163–203.

[21] E. M. Wolff, U. Topcu, R. M. Murray, Robust control of uncertain Markov decision processes with temporal logic specifications, in: Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, IEEE, 2012, pp. 3372–3379.

[22] E. M. Hahn, V. Hashemi, H. Hermanns, M. Lahijanian, A. Turrini, Interval markov decision processes with multiple objectives: From robust strategies to pareto curves, ACM Transactions on Modeling and Computer Simulation (TOMACS) 29 (4) (2019) 1–31.

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

[24] M. Dutreix, S. Coogan, Specification-guided verification and abstraction refinement of mixed monotone stochastic systems, IEEE Transactions on Automatic Control (2020).

[25] R. Majumdar, K. Mallik, S. Soudjani, Symbolic Controller Synthesis for Büchi Specifications on Stochastic Systems, arXiv e-prints (2019) arXiv:1910.12137`arXiv:1910.12137`.

[26] O. Hernández-Lerma, J. B. Lasserre, Discrete-time markov control processes, volume 30 of applications of mathematics, Springer 10 (1996) 978–1.

[27] M. Dutreix, S. Coogan, Efficient Verification for Stochastic Mixed Monotone Systems, in: International Conference on Cyber-Physical Systems, 2018.

[28] K. Sen, M. Viswanathan, G. Agha, Model-checking Markov chains in the presence of uncertainties, in: International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2006, pp. 394–410.

[29] S. Haddad, B. Monmege, Interval iteration algorithm for mdps and imdps, Theoretical Computer Science 735 (2018) 111–131.

[30] J.-P. Katoen, Model checking meets probability: A gentle introduction.

[31] S. Soudjani, C. Gevaerts, A. Abate, Faust 2: Formal abstractions of uncountable-state stochastic processes, arXiv preprint arXiv:1403.3286 (2014).

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

[33] M. Hirsch, H. Smith, Monotone dynamical systems, Handbook of differential equations: Ordinary differential equations 2 (2005) 239–357.

[34] D. Wu, X. Koutsoukos, Reachability analysis of uncertain systems using bounded-parameter Markov decision processes, Artificial Intelligence 172 (8-9) (2008) 945–954.

[35] PACE, Partnership for an Advanced Computing Environment (PACE) (2017).
URL `http://www.pace.gatech.edu`

[36] K. Chatterjee, L. De Alfaro, T. A. Henzinger, Trading memory for randomness, in: First International Conference on the Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings., IEEE, 2004, pp. 206–217.

Proof of Lemma 1

We provide a constructive proof for this lemma. Consider a product BMDP $\mathcal{B} \otimes \mathcal{A}$ with set of states $Q \times S$, set of policies $\mathcal{U}_{\otimes}^{\mathcal{A}}$ and set of memoryless policies $(\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$. We define the greatest permanent accepting BSCC $(U)_P^G \subseteq Q \times S$ as the set of all states of $\mathcal{B} \otimes \mathcal{A}$ such that, if $q \in (U)_P^G$, then there exists a policy in $\mathcal{U}_{\otimes}^{\mathcal{A}}$ such that q belongs to a permanent accepting BSCC in $\mathcal{B} \otimes \mathcal{A}$.

The first part of the proof consists in showing that there exists a set of memoryless policies $\mathcal{U}_{(U)_P^G} \subseteq (\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$ such that, under all product IMCs induced by a policy in $\mathcal{U}_{(U)_P^G}$, all states in $(U)_P^G$ belong to a permanent winning component simultaneously and, therefore, $(U)_P^G \subseteq (WC)_P^G$.

The second part of the proof shows that, for any other states of $\mathcal{B} \otimes \mathcal{A}$ which can be made a permanent winning component under some policy in $\mathcal{U}_{\otimes}^{\mathcal{A}}$, there exists a set of memoryless policies in $(\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$ (which is a subset of $\mathcal{U}_{(U)_P^G}$), such that all these states are a permanent winning component simultaneously.

I] Proof of existence of memoryless policies generating the greatest permanent accepting BSCC as a permanent winning component

First, we constructively show that, if there exists a policy $\mu_1 \in \mathcal{U}_{\otimes}^{\mathcal{A}}$ generating a permanent accepting BSCC $B_1 \subseteq Q \times S$ in $(\mathcal{B} \otimes \mathcal{A})[\mu_1]$, and if there exists another policy $\mu_2 \in \mathcal{U}_{\otimes}^{\mathcal{A}}$ generating a permanent accepting BSCC $B_2 \subseteq Q \times S$ in $(\mathcal{B} \otimes \mathcal{A})[\mu_2]$, then there has to exist a set of memoryless policies in $(\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$ causing the set $B_1 \cup B_2$ to be a permanent winning component in $\mathcal{B} \otimes \mathcal{A}$. Consider a policy $\mu_3 \in (\mathcal{U}_{\otimes}^{\mathcal{A}})_{mem}$ constructed as follows:

1) For the states in B_1, consider the following reasoning: by virtue of B_1 being a permanent accepting BSCC for some policy, it has to hold that, for some state $q_{acc} \in B_1$, $F_i \in L'(q_{acc})$ and $E_i \notin L'(q)$ $\forall q \in B_1$, for some i. Moreover, as B_1 is a permanent BSCC under μ_1, for any state $q \in B_1$, there exists a sequence of inputs chosen by μ_1 such that the lower bound probability of reaching q_{acc}

is 1, that is, $\breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_1]}(q \models \Diamond q_{acc}) = 1$. Since reachability problems in BMDPs have memoryless optimal policies [29], it must be true that a memoryless policy μ_1^{mem} choosing no other actions than the ones prescribed by μ_1 at all states $q \in B_1$ and guaranteeing $\breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_1^{mem}]}(q \models \Diamond q_{acc}) = 1$ for all $q \in B_1$ exists as well. For all $q \in B_1$, set $\mu_3(q) = \mu_1^{mem}(q)$.

2) For all states $q \in B_2 \setminus (B_1 \cap B_2)$, apply the same reasoning with respect to the problem of reaching $(B_1 \cap B_2)$ instead of q_{acc}, that is, there exists a memoryless policy μ_2^{mem} choosing no other actions than the ones prescribed by μ_2 such that $\breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_2^{mem}]}(q \models \Diamond(B_1 \cap B_2)) = 1$ for all $q \in B_2 \setminus (B_1 \cap B_2)$. For all $q \in B_2 \setminus (B_1 \cap B_2)$, set $\mu_3(q) = \mu_2^{mem}(q)$.

3) For all states $q \in (Q \times S) \setminus (B_1 \cup B_2)$, choose any action in $Act(q)$ as $\mu_3(q)$.

As B_1 is a permanent BSCC under μ_1, no state of B_1 can transition outside of B_1 under μ_3, that is, it holds that $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Diamond(Q \times S) \setminus B_1) = 0$. Moreover, since $\breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Diamond q_{acc}) = 1$ for all $q \in B_1$, it follows that any trajectory starting in B_1 will always return to q_{acc}, that is, $\breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Box\Diamond q_{acc}) = 1$ for all $q \in B_1$, and will additionally never reach a state $q_{n-acc} \in Q \times S$ satisfying $E_i \in L'(q_{n-acc})$. Therefore, any trajectory starting in B_1 satisfies the Rabin acceptance condition with lower bound probability 1, and B_1 is a member of the permanent winning component of $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$. Furthermore, for all $q \in B_2 \setminus (B_1 \cap B_2)$, we have $\breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Diamond B_1) = \breve{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Diamond(B_1 \cap B_2)) = 1$ and thus, $B_2 \setminus (B_1 \cap B_2)$ is a member of the permanent winning component of $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$. Therefore, $B_1 \cup B_2$ is a member of the permanent winning component of $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$.

Iteratively applying this logic with $B_1 \cup B_2$ and any other member of $(U)_P^G$ shows that there exists a set of policies in $\mathcal{U}_{(U)_P^G} \subseteq (\mathcal{U}_\otimes^\mathcal{A})_{mem}$ such that all states in $(U)_P^G$ belong to a permanent winning component simultaneously.

II] Proof of existence of greatest permanent winning component and of memoryless policies generating this component

Now, we consider the set $R = (Q \times S) \setminus (U)_P^G$ of all states of $\mathcal{B} \otimes \mathcal{A}$ which do not belong to $(U)_P^G$.

We define the set $\mathcal{U}^{out}_{(U)_P^G}$ of all policies which are history-dependent outside of $(U)_P^G$ and generate $(U)_P^G$ with an (arbitrary) memoryless policy on the states in $(U)_P^G$.

For a policy $\mu \in \mathcal{U}^{out}_{(U)_P^G}$, the set of all states $C \subseteq R$ that belong to the permanent winning component $(WC)_P$ of $(\mathcal{B} \otimes \mathcal{A})[\mu]$ without being a member of $(U)_P^G$ — that is, $C \cup (U)_P^G = (WC)_P$ and $C \cap (U)_P^G = \emptyset$ — has to satisfy two conditions:

a) C does not allow a transition outside of $C \cup (U)_P^G$ under any adversary of $(\mathcal{B} \otimes \mathcal{A})[\mu]$, that is, $\widehat{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[\mu]}\Big(q \models \Diamond\Big((Q \times S) \setminus \big(C \cup (U)_P^G\big)\Big)\Big) = 0$ for all $q \in C$,

b) No subset of C can form a losing component under any adversary of $(\mathcal{B} \otimes \mathcal{A})[\mu]$, that is, no state in C is a member of the largest losing component $(LC)_L$ of the product IMC $(\mathcal{B} \otimes \mathcal{A})[\mu]$, or $C \cap (LC)_L = \emptyset$.

With these two conditions fulfilled, all states in C either transition to $(U)_P^G$ or reach an accepting BSCC formed within C under all adversaries of $(\mathcal{B} \otimes \mathcal{A})[\mu]$, and therefore reach an accepting BSCC with lower bound probability 1.

Now, we constructively show that, if there exists a policy $\mu_1 \in \mathcal{U}^{out}_{(U)_P^G}$ inducing a product IMC $(\mathcal{B} \otimes \mathcal{A})[\mu_1]$ with permanent winning component $(WC^1)_P$ and with a set of states $C_1 \in R$ satisfying conditions a) and b) such that $C_1 \cup (U)_P^G = (WC^1)_P$ and $C_1 \cap (U)_P^G = \emptyset$, and if there exists a policy $\mu_2 \in \mathcal{U}^{out}_{(U)_P^G}$ inducing a product IMC $(\mathcal{B} \otimes \mathcal{A})[\mu_2]$ with permanent winning component $(WC^2)_P$ and with a set of states $C_2 \in R$ satisfying conditions a) and b) such that $C_2 \cup (U)_P^G = (WC^2)_P$ and $C_2 \cap (U)_P^G = \emptyset$, then there has to exist a memoryless policy $\mu_3 \in \mathcal{U}_{(U)_P^G}$ inducing a product IMC $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$ with permanent winning component $(WC^3)_P$ and with the set of states $(C_1 \cup C_2) \in R$ satisfying conditions a) and b) such that $(C_1 \cup C_2) \cap (U)_P^G = \emptyset$. Consider a policy $\mu_3 \in \mathcal{U}_{(U)_P^G}$ constructed as follows:

1) For all state $q \in C_1$, consider the following reasoning inspired by the arguments in the proof of [36, Theorem 8] on the optimality of memoryless policies in MDPs for Rabin objectives: for any state $q \in C_1$, it must be true that any trajectory initiated at q under policy μ_1 reaches with lower bound probability 1 a set $K \subseteq C_1$ such that the continuation of any trajectory that reaches K is confined to $K \cup (U)_P^G$ and either reaches $(U)_P^G$ or visits an unmatched accepting Rabin state in K infinitely often. Consider the arbitrarily ordered set $(K_1, K_2, \ldots, K_m)$ of all such sets which can by reached by some initial state $q \in C_1$ under μ_1. Due to the optimality of memoryless policies for reachability problems in BMDPs and the properties of the K sets, there must exist a memoryless policy $\mu_1^{K_1}$ such that, for all $q \in K_1$, $\check{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_1^{K_1}]}\Big(q \models \Diamond((U)_P^G) \cup A\Big) = 1$, where A is the set of all unmatched Rabin accepting states in K_1, and $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_1^{K_1}]}\Big(q \models \Diamond(Q \times S) \setminus ((U)_P^G \cup K_1)\Big) = 0$. Set $\mu_3(q) = \mu_1^{K_1}(q)$ for all $q \in K_1$. Apply the same procedure recursively to $K_2 \setminus K_1$ and replacing $(U)_P^G$ with $(U)_P^G \cup K_1$, then to $K_3 \setminus (K_2 \cup K_1)$ etc. For the states $q \in C_1$ outside the K sets, design μ_3 such that $\check{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}\Big(q \models \Diamond(U)_P^G \cup K_1 \cup \ldots \cup K_m\Big) = 1$, which again can be achieved with a memoryless choice of actions due to the optimality of memoryless policies for reachability and the fact that μ_1 satisfies this condition.

2) For all state $q \in C_2 \setminus (C_1 \cap C_2)$, choose the actions in μ_3 by following the same reasoning as in 1) after replacing C_1 with $q \in C_2 \setminus (C_1 \cap C_2)$ and $(U)_P^G$ with $(U)_P^G \cup C_1$.

3) For all state $q \in (Q \times S) \setminus (C_1 \cup C_2)$ (not in $(U)_P^G$, since actions are already fixed in this set), choose any action in $Act(q)$ as $\mu_3(q)$.

By construction, the set $C_1 \cup C_2$ satisfy condition a) and b), as no subset of $C_1 \cup C_2$ can form a losing component under the actions prescribed by μ_3 and no trajectory can leave $C_1 \cup C_2 \cup (U)^G_P$. Therefore, $C_1 \cup C_2$ is a subset of the permanent winning component $(WC^3)_P$ of $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$.

Replacing the set $(U)^G_P$ from the beginning of section II] with $C_1 \cup C_2 \cup (U)^G_P$ and applying the same process iteratively proves the existence of a set $(WC)^G_P$ satisfying the properties enunciated in the lemma and of a set of memoryless policies $\mathcal{U}_{(WC)^G_P}$ generating $(WC)^G_P$.