VERIFICATION AND SYNTHESIS FOR STOCHASTIC SYSTEMS WITH TEMPORAL LOGIC SPECIFICATIONS

A Dissertation
Presented to
The Academic Faculty

By

Maxence Dominique Henri Dutreix

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2020

Copyright © Maxence Dominique Henri Dutreix 2020

VERIFICATION AND SYNTHESIS FOR STOCHASTIC SYSTEMS WITH TEMPORAL LOGIC SPECIFICATIONS

Approved by:

Dr. Samuel Coogan, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Fumin Zhang
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Ye Zhao
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Yorai Wardi
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Kyriakos Vamvoudakis
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: February 25, 2020

À Frederick.

ACKNOWLEDGEMENTS

First, I would like to sincerely thank my research advisor Samuel Coogan for providing me with the opportunity to pursue this PhD program. Not only am I grateful for his invaluable guidance throughout my studies, but I foremost want to express my deepest appreciation to him for being an understanding and considerate person to work with. I feel honored to be his first PhD student to graduate and wish him the best for his future research and academic career.

I would also like to recognize the support that I received from my colleagues in the FACTS lab for the entire duration of my PhD. The time we spent together as a lab and as friends outside the lab helped me overcome the difficult phases of my research. Being a part of a multicultural group has broadened my perspective on different cultures and their personal stories have taught me and inspired me tremendously.

Finally, I want to acknowledge all teachers, school officials and classmates from L'École élémentaire du Conseil des XV in Strasbourg to Georgia Tech in Atlanta who, each in their own way, contributed to my journey as a student and, by extension, to the elaboration of this document.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

The objective of this thesis is to first provide a formal framework for the verification of discrete-time, continuous-space stochastic systems with complex temporal specifications. Secondly, the approach developed for verification is extended to the synthesis of controllers that aim to maximize or minimize the probability of occurrence of temporal behaviors in stochastic systems. As these problems are generally undecidable or intractable to solve, approximation methods are employed in the form of finite-state abstractions arising from a partition of the original system's domain for which analysis is greatly simplified. The abstractions of choice in this work are Interval-valued Markov Chains (IMC) which, unlike conventional discrete-time Markov Chains, allow for a non-deterministic range of probabilities of transition between states instead of a fixed probability.

Techniques for constructing IMC abstractions for two classes of systems are presented. Due to their inherent structure that facilitates estimations of reachable sets, mixed monotone systems with additive disturbances are shown to be efficiently amenable to IMC abstractions. Then, an abstraction procedure for polynomial systems that uses stochastic barrier functions computed via Sum-of-Squares programming is derived.

Next, an algorithm for computing satisfaction bounds in IMCs with respect to so-called ω-regular properties is detailed. As probabilistic specifications require finding the set of initial states whose probability of fulfilling some behavior is below or above a certain threshold, this method may yield a set of states whose satisfaction status is undecided. An iterative specification-guided partition refinement method is proposed to reduce conservatism in the abstraction until a precision threshold is met.

Finally, similar interval-based finite abstractions are utilized to synthesize control policies for ω-regular objectives in systems with both a finite number of modes and a continuous set of available inputs. A notion of optimality for these policies is introduced and a partition refinement scheme is presented to reach a desired level of optimality.

CHAPTER 1
INTRODUCTION

Reliance on complex systems, incorporating a myriad of interacting components and subject to growingly demanding tasks, continues to increase exponentially. Their applications to safety-critical environments have driven a great deal of interest in the implementation of dependable verification and control apparatus. Indeed, failure of automated equipment may engender catastrophic financial, human and ecological consequences [1]. Avoiding such scenarios is evidently conditional on the fidelity of the available models with respect to the actual behavior of the physical systems of interest.

While control theory has established a rich, mathematically grounded foundation for the study of purely deterministic dynamics, many real-life processes exhibit random behaviors which can drastically interfere with intended operations if not properly reckoned with. Stochasticity traditionally manifests itself in systems models in the form of quantifiable disturbances. As the evolution of these systems throughout time cannot be predicted exactly, their analysis requires a distinct framework from their deterministic counterparts and raises an abundance of open questions.

To address the latter, this thesis provides theoretical contributions to the formal verification and synthesis of stochastic dynamical systems. We focus on large classes of stochastic system models with specific, physically-motivated structures amenable to efficient formal verification and control, and contrive algorithms that are applicable to a wide range of pertinent system properties.

1.1 Verification and Control of Stochastic Systems

Stochastic systems theory can be regarded as a fairly young field of study. Whereas control theory emerged towards the middle of the 19th century, a modern and rigorous formulation of probability theory materialized only in the second quarter of the 20th century, enabled by the axiom system of Kolmogorov. This coherent mathematical framework became the long-awaited standard for reasoning about real-world uncertainties, and opened the door to a plethora of new opportunities for theoretical scientists and engineers alike.

Despite its somewhat recent apparition, the field of stochastic system analysis and control flourished for the past five decades, unsurprisingly driven by its numerous and lucrative applications to finance. Often lagging their deterministic analog, specialized areas such as stochastic optimal control, stochastic network control and stochastic adaptive control naturally came into existence as the necessity to account for the intrinsic randomness in natural phenomena became apparent for many technological purposes. For examples, weather events, message losses in communication channels, machine sensors and social structures like traffic networks, all exhibit some degree of measurable stochasticity. The tremendous increase in computers' capabilities has helped in the physical manifestation of the theoretical developments of the field, culminating with the implementation of now ubiquitous practical instruments such as Kalman filters. Yet, many crucial challenges still remain unresolved for state-of-the-art tools. In particular, the very nature of stochastic problems, which imposes the consideration of multiple possible future scenarios, renders the study of high-dimensional random systems particularly prone to computational complications like the curse of dimensionality.

The recent advent of systems comprising a mixture of continuous and discrete dynamics, often referred to under the broad umbrella term of *cyber-physical systems*, has further complexified the accommodation of uncertain stochastic events. Traditional control techniques have encountered limited success with both the prediction and command of cyber-physical behaviors characterized by an intertwining of continuous quantities and discrete inputs, outputs and mode switches. Paradigm examples of such systems include automobiles, aeroplanes, medical monitors, fleets of robots and smart electricity grids. The unavoidable presence of internal and external random disturbances clearly poses additional obstacles to the complete automation of cyber-physical systems.

Not only are systems becoming more complex, but the tasks they are expected to perform are growing more exigent as well. Fundamental system properties, such as safety and stability, often need to be enforced alongside additional crucial performance objectives. For instance, an intelligent agent may be required to navigate a rough environment with several obstacles in order to sequentially and repeatedly transport an object to multiple target regions. Formal symbolic languages known as temporal logics are particularly well-suited for reasoning about time and expressing complex behaviors in an unambivalent manner.

Classical control theory fell short of delivering the adequate machinery for handling convoluted temporal specifications. Engineers eventually turned to advanced techniques relying on finite-state representations of continuous dynamics to tackle the verification of systems against such properties. These finite-state models, or *abstractions*, aim to capture all possible behaviors of a system using a finite set of transitions, often at the cost of introducing conservatism and nondeterminism. In addition, both deterministic and stochastic dynamics can be abstracted using finite transition systems, making this approach very versatile. The finite number of configurations of these abstractions enables the utilization of brute-force verification methods commonly referred as *model checking*, characterized by their high computational effort but also their formidable reliability.

1.2 Model Checking and its Applications to Stochastic Systems Analysis

The concept of model checking emerged as the need for highly reliable hardware and software became increasingly pressing due to the tremendous costs of potential technological failure. Model checking is an automated process for verifying whether a system possesses a desired property. Given an unambiguous mathematical description of the system, a model checker provides a trustworthy positive or negative answer upon an exhaustive examination of all possible states and behaviors of the model. Intensive research in this area in the past three decades led to the development of dependable verification tools, and to the deployment of these tools in the design loop of numerous technologies, such as computer hardware [2] [3], communication protocols [4], airplanes [5] and spacecrafts [6], with unequivocal success.

The applicability range of model checking has recently been extended to the study of probabilistic transition systems, which led to the development of extremely dependable tools for the verification of Markov Chains with complex temporal objectives [7]. Substantial work has been conducted in parallel on the efficient construction of Markov Chain abstractions for stochastic dynamical systems to leverage the power of model checking towards the verification of such systems [8] [9]. Unfortunately, these abstraction methods suffer from a restrictive conservatism due to the fact that continuous-state stochastic systems cannot, in general, be abstracted exactly by a Markov Chain. To this day, the verification of such systems is therefore limited to low-dimensional models for which the required computational effort is not prohibitive. Moreover, a unified approach for all "interesting" system properties has yet to be erected, as specifications coming from different temporal logics sometimes demand distinct verification frameworks. All these shortcomings equally hindered the implementation of robust and scalable synthesis procedures for controlled stochastic dynamical systems.

Building on the most recent results drawn from the model checking literature, this dissertation focuses on enhancing the scalibility and versatiliy of existing techniques to enable a computationally efficient verification and synthesis for stochastic dynamical systems with complex temporal objectives. To this end, we investigate a largely unexplored type of stochastic finite-state abstractions known as *Interval-valued Markov Chains* (IMC). We conduct a thorough analysis of efficient abstraction techniques for wide classes of systems, and detail the appropriate theory for performing verification against the highly-expressive ω-regular temporal properties in IMCs. Automated procedures for reducing the conservatism of IMC abstractions with the least possible computational impact are set forth. We introduce equivalent interval-valued transition systems to serve as abstractions of controlled stochastic systems, and derive novel synthesis algorithms for computing optimal controllers in these abstractions for ω-regular specifications. Finally, we contrive an automated refinement process to decrease conservatism in these abstractions and achieve a desired level of controller optimality.

1.3 Preview of Thesis

The objective of this thesis is the implementation of efficient verification and synthesis techniques for discrete-time, continuous-state stochastic systems with complex temporal logic objectives. In Chapter 2, we review the main literature on the topic of verification and synthesis for stochastic systems. In Chapter 3, we introduce the preliminaries of our approach which relies on finite-state stochastic abstractions of the continuous dynamics in the form of interval-valued Markov chains, bounded-parameter Markov decision processes and controlled interval-valued Markov chains arising from a partition of the system's domain. In Chapter 4, we present a methodology for constructing interval-valued Markov chain abstractions and bounded-parameter Markov decision process abstractions for two classes of systems, namely affine-in-disturbance mixed monotone systems and polynomial systems. In Chapter 5, we detail an algorithm used to conduct verification of IMCs against ω-regular specifications. Applying this algorithm on IMC abstractions allows to obtain probabilistic guarantees with respect to the abstracted continuous states. We study a so-called specification-guided refinement method of the domain partition may the conservatism of the abstraction need to be reduced. In Chapter 6, we extend the theory developed in Chapter 5 to the synthesis of controllers for stochastic systems with both a finite number of modes and a continuous set of available inputs using bounded-parameter Markov decision process abstractions and controlled interval-valued Markov chain abstractions. We introduce a metric for quantifying the optimality of the designed controllers with respect to the abstracted continuous states and present an algorithm for enhancing the controller optimality through a specification-guided refinement of the domain partition. In Chapter 7, we demonstrate the results obtained in the previous chapters in several case studies.

IMC and BMDP Abstraction of Stochastic Mixed Monotone Systems and Polynomial Systems

In Chapter 4, we treat the problem of abstracting discrete-time, continuous-state stochastic systems with finite-state models. The construction of such abstractions is traditionally achieved by partitioning the domain of the system and determining the possible transitions between the resulting discrete regions using reachable set computations. However, in the case of stochastic systems, a quantitative component—namely, a probability—characterizes each transition, which adds an extra layer of complexity to this problem. Furthermore, standard probabilistic transition systems such as Markov chains cannot, in general, exactly abstract the behavior of continuous-state stochastic dynamical systems. Indeed, two distinct continuous states from a given discrete partition state may yield different probabilities of transition to other regions of the state-space. In order to encapsulate all feasible behaviors of the system in a finite number of states and transitions, we consider augmented Markov chains, known as Interval-valued Markov Chains (IMC), where the transition probabilities are constrained to a nondeterministic range of values, as depicted in Figure 1.1. Given an arbitrary stochastic system, finding non-trivial transition probability intervals between all discrete states is a complicated task. Hence, we focus on two specific classes of systems that are efficiently amenable to IMC abstractions.

In Section 4.1, we propose an IMC abstraction technique for discrete-time *affine-in-disturbance mixed monotone systems*. A system with governing equation

$$x[k+1] = \mathcal{F}(x[k]) + w[k] \tag{1.1}$$

belongs to the aforementioned class of systems if $\mathcal{F}$ is mixed monotone and $w[k]$ is a ran-

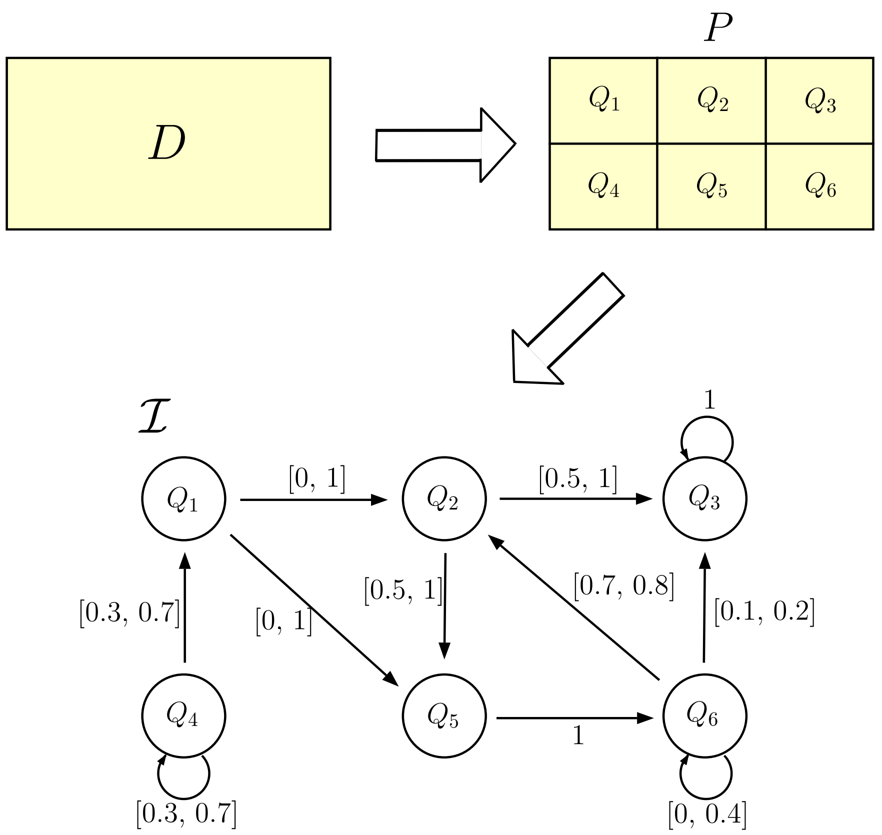


Figure 1.1: A finite-state IMC abstraction $\mathcal{I}$ of a stochastic system over a continuous domain D. A partition P of D is generated and bounds on the transition probabilities between states are estimated.

dom variable. Generalizing the well-known concept of monotonicity, a function $\mathcal{F}$ is mixed monotone if there exists a decomposition function $g(x, y)$ which is increasing in its first argument, decreasing in its second argument, and if the original dynamics are recovered by evaluating g with the same value in both arguments, that is, $\mathcal{F}(x) = g(x, x)$. Many relevant systems were shown to possess mixed monotone properties, such as transportation networks [10], [11] and biological processes [12].

It was shown in previous works that an over-approximation of the one-step reachable set from any hyperrectangular region under a mixed-monotone map $\mathcal{F}$ can be efficiently computed by evaluating a corresponding decomposition function g at the least and greatest point of the hyperrectangle [13]. Moreover, these approximations were proved to be tight in some instances. Supposing the domain of (1.1) admits a rectangular partition, an over-approximation of the deterministic reachable set from any discrete state is therefore easily calculated with only two function evaluations, regardless of the dimension of the system.

Due to the additive nature of the disturbance, an upper and lower bound on the probability of transition from one state in the partition to another are determined by finding the positions of the probability density function of the disturbance inside the deterministic reachable set of the origin state that respectively maximize and minimize the probability overlap with the destination state. We derive closed-form solutions for these constrained minimizing and maximizing disturbance shifts under additional symmetry and unimodality assumptions on the disturbance term, which allow to compute the desired bounds upon evaluation of two integrals. Therefore, our abstraction procedure grows linearly in the number of dimensions of the system. Furthermore, we suggest an alternate formulation for the bounds when the symmetry assumption on the disturbance is relaxed. This formulation nonetheless assumes that the disturbance can be reasonably approximated by another disturbance which is symmetric.

The results of this section appear in [14].

In Section 4.2, we present an IMC abstraction method for polynomial systems of the form

$$x[k+1] = \mathcal{F}(x[k], w[k]) \ , \tag{1.2}$$

where $\mathcal{F}$ is a polynomial function in both $x[k]$ and $w[k]$. Stochastic barrier functions emerged as promising tools for providing probabilistic guarantees amenable to IMC abstractions for polynomial dynamical systems. Stochastic barrier functions are used as a probabilistic certificate of set invariance for stochastic dynamical systems. Specifically, one can derive an upper bound on the probability that a system will reach some region of the domain if one can show the existence of a barrier function whose expectation against the system dynamics behaves in a certain way over the domain.

First, we present the main properties of stochastic barrier functions for discrete-time systems which allow to compute guarantees of set invariance over a finite-time horizon. Then, we introduce a formulation of the stochastic barrier function discrete-time framework over a single transition where two barrier functions are computed. By means of this formulation, we show that an IMC abstraction of stochastic polynomial systems is created from a finite partition of its domain by finding two stochastic barrier functions per transition.

Enabled by (1.2) being a polynomial system, the search of stochastic barrier functions is cast as optimization problems in the form of *Sum-of-Squares Programs* (SOSP). Existing tools can convert SOSPs to semidefinite programs, which are conveniently convex and therefore efficient to solve. Thus, a lower bound and and an upper bound on the probability of transition between any two states in the domain partition are determined by solving two SOSPs. Each SOSP involves the computation of two barrier functions providing one-step probabilistic guarantees of reachability. Applying this procedure to all pairs of states in the partition generates an IMC abstraction of (1.2).

The results of this section appear in [15] and [16].

The theory developed in this chapter is straightfowardly extended to the abstraction of switched stochastic systems of the form

$$x[k+1] = \mathcal{F}_a(x[k], w_a[k]) \; , \tag{1.3}$$

where a belongs to a finite set of distinct modes. Such systems are abstracted by *Bounded-parameter Markov Decision Processes* (BMDP) where a finite number of actions representing the modes of the original system is available at each state, with each action inducing a different transition profile characterized by intervals of transition probabilities. Assuming each mode of the system results in a state update map $\mathcal{F}_a$ belonging to one of the two aforementioned classes, we can construct an IMC abstraction for each mode of the system which is equivalent to a BMDP abstraction of the overall system.

IMC-based Verification of Stochastic Systems with ω-regular Objectives

In Chapter 5, we present an IMC abstraction-based verification procedure for stochastic systems subject to probabilistic ω-regular specifications. Formally, considering a stochastic system with dynamics

$$x[k+1] = \mathcal{F}(x[k], w[k]) \ , \tag{1.4}$$

our objective is to find the set of initial states of (1.4) satisfying a probabilistic specification of the form

$$\phi = \mathcal{P}_{\bowtie p_{sat}}[\Psi] \ , \tag{1.5}$$

where $\bowtie \in \{\leq, <, \geq, >\}$, $p_{sat} \in [0, 1]$, and Ψ is an ω-regular property. An initial state x of (1.4) satisfies (1.5) if a trajectory generated by (1.4) from x has a probability of satisfying Ψ which is greater or less than p_{sat}, depending on the choice for $\bowtie$.

To address this problem, we assume that an IMC abstraction of (1.4) is constructed from a finite partition of its domain. Any probabilistic guarantees computed for the discrete IMC states with respect to Ψ can be mapped to the continuous abstracted states of (1.4). Because the transition probabilities in IMCs are constrained to intervals, the probability of satisfying a property Ψ has to be specified as an interval as well from all discrete states of the abstraction.

First, in Section 5.1, we derive an algorithm for computing the tightest interval on the probability of satisfying an ω-regular property Ψ for all initial states of an IMC. The proposed solution extends the theory of verification of standard discrete-time Markov chains against ω-regular properties. In the latter case, the Cartesian product between the Markov chain and *a Deterministic Rabin Automaton* (DRA) encoding the property of interest is constructed. Then, the probability of reaching special sets of states— namely, the accepting *Bottom Strongly Connected Components* (BSCC) of the product Markov chain which are a function the acceptance conditions of the DRA—is computed from the initial states of the product. These reachability probabilities correspond to the probabilities of satisfaction for the Markov chain states.

In a similar way, we define the Cartesian product between an IMC and a DRA. However, we show that, in general, the set of accepting BSCCs in a product IMC is not fixed and depends on the assumed transition values for each interval. Instead, we demonstrate that any product IMC induces a *largest losing component* and a *largest winning component*. We establish that an interval on the probability of satisfying the desired property is found by solving a reachability problem on these components. We devise graph-based algorithms for finding the largest components of a product IMC.

Applying this verification procedure on an IMC abstraction of (1.4) yields an interval on the probability of satisfying Ψ for all continuous states of (1.4). Therefore, some of these states may be undecided with respect to (1.5) if the threshold p_{sat} belongs to their satisfaction interval. In order to reduce the conservatism of the IMC abstraction of (1.4) and achieve a lower volume of undecided states, the standard approach consists in refining the domain partition of (1.4) and construct a new, less conservative IMC abstraction from the refined partition.

To mitigate the state-space explosion phenomenon caused by excessive refinement, we suggest a heuristical scoring procedure to target the states in the partition which are most likely to reduce conservatism under refinement. The procedure relies on a quantitative and qualitative comparison of the paths reaching an accepting BSCC in the product IMC between the worst-case and best-case assignment of the transition probabilities computed at the time of verification.

The results of this chapter appear in [17].

Synthesis of Controllers for Stochastic Systems using Interval-Valued Probabilistic Abstractions

In Chapter 6, we present an interval-valued abstraction-based approach to the synthesis of control policies for stochastic systems with ω-regular objectives. Given the controlled stochastic system

$$x[k+1] = \mathcal{F}(x[k], u[k], w[k]) \; , \tag{1.6}$$

where $u[k]$ denotes a control input, our goal is to devise a control strategy that either maximizes or minimizes the probability of satisfying an ω-regular property Ψ for any initial state of the system.

In Section 6.1, we study the simplified case where only a finite number of inputs, or *modes*, are available, that is,

$$x[k+1] = \mathcal{F}_a(x[k], w_a[k]) \; , \tag{1.7}$$

where $a \in \{1, 2, \ldots, m\}$ denotes the discrete modes of the system, and the noise term w_a is allowed to be mode-dependent. Here, we aim to find a switching policy that, at each time step, selects the best mode a so as to maximize or minimize the probability of satisfying Ψ for the subsequent execution of the system.

We undertake this problem via approximation methods, and assume that a BMDP abstraction $\mathcal{B}$ of (1.7) constructed from a finite partition of the system's domain is available. In other words, an IMC abstraction of (1.7) is created for every mode a of (1.7). As BMDPs are interval-valued transition systems, minimizing the probability of satisfying a property

Ψ in $\mathcal{B}$ is equivalent to minimizing the upper bound probability of satisfying Ψ from all initial states of $\mathcal{B}$. Likewise, maximizing the probability of satisfying Ψ corresponds to maximizing the lower bound probability of satisfying Ψ from all initial states.

To devise optimal switching policies in a BMDP $\mathcal{B}$, we use an automaton-based approach. We define the Cartesian product between a BMDP $\mathcal{B}$ and a DRA representing the specification Ψ. Next, we show that computing optimal switching policies in a BMDP with respect to Ψ amounts to solving both a qualitative problem and a quantitative problem in the product BMDP: first, we construct the so-called *greatest permanent winning component* of the product BMDP for maximization, or the *greatest permanent losing component* of the product BMDP for minimization. Then, for all states outside of these components, the optimal policy is computed by maximizing the lower bound probability of reaching these components. We detail graph-based algorithms for finding the latter components and determine the corresponding control actions for generating them.

The computed switching policy in the BMDP abstraction $\mathcal{B}$ is likely to be suboptimal when mapped onto the continuous abstracted states of (1.7). We propose a methodology for quantitatively assessing the optimality of the designed policy for every discrete state of the product BMDP, and for identifying the modes which are certaintly optimal or not optimal at each state. To reduce the suboptimality of the switching policy down to a user-defined threshold, we present a partition refinement technique inspired from the refinement algorithm for verification in Chapter 5.

In Section 6.2, we treat the case where the input $u[k]$ takes values in a continuous set and where system (1.6) is affine in disturbance and input, that is,

$$x[k+1] = \mathcal{F}(x[k]) + u[k] + w[k] \; . \tag{1.8}$$

The problem of computing an optimal control policy for such systems is addressed in a similar fashion as in the finite-mode case. First, we construct a finite-state abstraction of (1.8) from a partition of its domain in the form of a *Controlled Interval-valued Markov Chain* (CIMC) $\mathcal{C}$, where the probabilities of transition between all states are also given as an interval which is dependent on an input drawn from a continuous set.

Then, we synthesize an optimal controller in the CIMC abstraction $\mathcal{C}$ in two steps: first, we show that a greatest permanent winning component and a greatest permanent losing component can be constructed in the Cartesian product between $\mathcal{C}$ and the DRA representing the specification of interest Ψ as in a BMDP. We demonstrate that this qualitative step is achieved by converting the CIMC $\mathcal{C}$ into a BMDP $\mathcal{B}$ through the selection of a finite number of actions from the continuous set of available inputs. We discuss how to find the required finite set of actions under certain assumptions on the noise term and the geometry of the domain partition. Next, for the states which do not belong to the greatest permanent components, an optimal input is computed by maximizing the lower bound probability of reaching these components through the iterative resolution of optimization problems.

Finally, a similar domain partition refinement scheme as in the finite-mode problem is proposed so as to attain a user-defined level of optimality for the devised control policy.

The results of this chapter appear in [18].

CHAPTER 2
LITERATURE REVIEW

Numerous techniques for the verification of continuous-state stochastic systems have been put forth in the literature. These can be classified into two fundamentally different approaches commonly referred to as *abstraction-free* and *abstraction-based*, which are respectively reviewed in Section 2.1 and Section 2.2. Related literature on the synthesis of controllers for stochastic systems with temporal logic objectives, which is the focus of Section 2.3, almost exclusively employ abstraction-based approaches.

2.1 Abstraction-Free Verification for Stochastic Systems

Abstraction-free techniques use a thorough analysis of a system's vector field in order to derive probabilistic properties without having to directly generate any trajectory. The work in [19] introduces *stochastic Lyapunov functions* which, in the same spirit as in the deterministic case, serve as a certificate of stability for the equilibrum points of stochastic differential equations. In particular, if there exists a stochastic Lyapunov function whose value decreases in expectation in some neighborhood of an equilibrium point, then the latter is guaranteed to be stable in probability.

The notion of *stochastic barrier certificate* is presented in [20] and provides a Lyapunov-like framework to compute an upper bound on the probability of some continuous-time stochastic process to exit a safe region of the state-space over an infinite-time horizon. This upper bound is found by showing the existence of a function whose infinitesimal generator with respect to the stochastic process is strictly negative over the safe region, i.e. this barrier function has to be a supermartingale and decreases everywhere in expectation. The search of such a function is carried out via a *Sum-of-Squares* (SOS) program and its maximum value over the set of initial conditions determines the probability bound. Unfortunately, the supermartingale requirement is often overly restrictive for a large number of systems, making it impossible to find a barrier function fulfilling this criterion.

As an extension to this work, [21] introduces the concept of *c-martingale* stochastic barrier functions, whose infinitesimal generator is now restricted to be less than a positive constant over the safe region of interested. An SOS problem is again solved in order to find an appropriate function which is used as a certificate for bounding the probability of the system being unsafe on a finite-time horizon. However, this bound is likely to be conservative as it assumes a “worst-case scenario” for the generator value over the whole domain and thus fails to fully capture the system’s dynamics. C-martingales are further utilized by [22] in the context of discrete-time stochastic barrier functions. In this work, the authors manage to decompose any finite-time safety *Linear Temporal Logic* (LTL) specification into a sequence of reachability objectives in the automaton corresponding to the complement property. An upper bound probability on these reachability objectives is computed by solving an SOS program and translates into a lower bound probability of satisfying the original LTL specification.

A common observation from all the mentioned papers is that the computed bounds appear to be quite conservative when validated against Monte Carlo simulations, especially for large noise values. Calculating tighter bounds usually requires solving a higher order SOS problem, which can dramatically impact the computational complexity of the discussed procedures in a negative way and even more so in high-dimensional systems. Additionally, it is still unclear how these abstraction-free techniques could be applied to more involved temporal logic specifications beyond safety and simple reachability.

2.2 Abstraction-Based Verification for Stochastic Systems

Abstraction-based verification methods rely on explicit computations of the system’s trajectories over a finite partition of the domain of interest. In a discrete-time setting, these trajectories are typically estimated via under- or over-approximation of one-step reachable sets. A state-of-the-art tool for the verification of stochastic systems against complex specifications is the FAUST^2 model checker [8]. In FAUST^2, the system's continuous domain is partitioned into a finite number of discrete states, each of them being reduced to a single representative point. Propagating the system's dynamics from these points generates an approximate Markov chain representation of the original continuous-domain system [23], which can then be easily verified for a large number of specifications with off-the-shelf softwares such as PRISM [7]. Probabilistic guarantees obtained on the approximate MC can in turn be mapped back to the original abstracted states in the form of a probability interval when some characteristics of the underlying dynamics, such as global or local Lipschitz constants, are known.

The work in [24] also suggests a methodology to compute a gridding parameter for the domain partition that ensures an upper bound on the size of the interval of satisfaction for all discrete initial states and for all specifications from the class of *Probabilistic Computation Tree Logic* (PCTL) properties. One advantage of this approach is that the approximate Markov chain abstraction does not need to be recomputed if the specification of interest is changed. However, these techniques generally rely on a conservatively fine gridding of the continuous state-space making the verification process computationally intractable. Additionally, this approach overlooks qualitative aspects that are important to certain specifications, e.g. the creation of absorbing states as pointed out in [25], making these *dynamics-guided* partitioning techniques inadequate for high-dimensional systems.

In [26], the authors address the problem of verifying discrete-time stochastic systems against PCTL specifications using *Interval-valued Markov Chains* (IMC) abstractions. IMCs are Markovian transition systems wherein the probability of transition between states is given to lie within an interval rather than being a single, well-defined number. The complexity of verifying such models is naturally increased compared to standard MCs model-checking algorithms. First, the authors create a polytopic partition of the system's domain, which induces a finite-state abstraction of the system in the form of an IMC. Then, any

PCTL specification can be converted into a reachability problem in the IMC abstraction and a polynomial-time algorithm from computing the lower and upper bound probabilities of reaching the accepting states from any initial state is presented. This results in a lower and an upper bound probability of satisfying the specification for any continuous state abstracted by the IMC. If this interval of satisfaction is too large for a significant volume of the state-space, the polytopic partition is refined using one-step pre and post operations on the most uncertain states of the partition. Subsequent partitions may be refined as well until some precision threshold is met. This *specification-guided* approach to state-space partition and refinement differs from the one previously discussed in [24]. At each refinement step, the most uncertain states with respect to the specification at hand are targeted, avoiding some unnecessary refinement arising in the dynamics-guided approach. However, the main drawback is that verification has to be carried out every time a finer partition is created, as opposed to being performed only once in the dynamics-guided case. Although this paper introduces critical verification tools, it suffers from a number of shortcomings: first, the IMC abstractions are built using sampling methods which negatively impact the robustness of the verification procedure; then, the proposed method is only suited for specifications in the logic PCTL, which cannot express important behaviors such as *liveness* properties, *persistence* properties or *implications* [27]; finally, the refinement technique only considers one-step transitions and fails to truly account for the overall structure of the system paired with the specification at hand.

The verification of IMC abstractions for more expressive specifications from the class of *ω-regular properties* is discussed in [28], where the authors convert an IMC into a Markov Decision Process (MDP) which only accounts for the extreme values of the transition probability intervals; however, the suggested method has a computational complexity which grows exponentially in the size of the abstraction, making it unsuitable for the verification of complex systems.

Other forms of abstractions for discrete-time stochastic systems include *Markov Set-*

Chains [29] [24]. Markov Set-chains non-deterministically select a transition matrix from a set defined by two bounding transition matrices at each time step and evolve accordingly. Although Markov Set-chains abstractions provide useful information such as the asymptotic behavior of stochastic systems, it is unclear from these publications how they could be used for the verification of highly complex behaviors.

2.3 Controller Synthesis for Stochastic Systems

The design of controllers for stochastic systems also comes with its unique challenges. Indeed, due to the intrinsic non-determinism of random dynamics, synthesizing optimal control laws for stochastic systems amounts to maximizing or minimizing the probability of occurrence of some behavior. The literature on stochastic controller synthesis for temporal logic specifications is sparse and, in general, restricts its scope to a finite number of available inputs. Related papers almost exclusively employ finite-state abstractions of the original dynamics.

One exception to this is the work in [16], which uses stochastic control barrier functions to achieve a user-defined probability of system safety in both a continuous-time and discrete-time framework in an abstraction-free manner. The system dynamics and controller expressions are both assumed to be polynomials in order to convert the synthesis procedure to an SOS optimization program whose objective is to ensure a given upper bound on the probability of the system reaching an unsafe region with a low control effort.

The work in [26] addresses the problem of finding an optimal switching policy in continuous-state, discrete-time switched stochastic systems with a finite number of modes and with respect to PCTL specifications. The proposed approach consists in partitioning the continuous domain and compute an IMC abstraction for each possible mode, generating a *Bounded-Parameter Markov Decision Process* (BMDP) abstraction of the switched system. Then, a probability maximizing switching policy is found by maximizing the lower bound probability of reaching an accepting state in the BMDP from the desired initial state; likewise, a minimizing policy can be established by minimizing the upper bound probability of reaching an accepting state. However, as explained previously, PCTL lacks in expressiveness compared to other logic systems.

The problem of verifying BMDPs against co-safe LTL specifications is discussed in [30] where synthesis is reduced to a reachability maximization task in the product of the BMDP with an automaton representation of the property. Again, restricting to the co-safe LTL class significantly reduces the scope of this technique in terms of expressiveness.

The more recent verification and synthesis tool StocHy [9], which exploits the techniques presented in [31], very efficiently performs synthesis for stochastic switched linear systems with additive Gaussian noise, but is limited to co-safe LTL as well.

In [32], a synthesis algorithm is presented for uncertain *Markov Decision Processes* (MDP) subject to LTL specifications. Uncertain MDPs are similar to BMDPs with the difference that the non-deterministic transition probabilities are drawn from an arbitrary set of valid transition matrices for each mode and not necessarily from a Cartesian product of probability intervals. Such stochastic models are of interest to us as they can serve as systems abstractions. Unfortunately, [32] makes strong simplifying and unrealistic assumptions on the connectivity structure of the product between the uncertain MDP and the *Deterministic Rabin Automaton* (DRA) corresponding to the specification. The synthesis of control strategies for interval Markov decision processes with multi-objectives that include ω-regular properties was discussed in [33]; however, the qualitative structure of the transition system is again assumed to be invariant, which alleviates key difficulties associated with the problem.

Other publications such as [34] rely on standard MDP abstractions of stochastic systems in order to carry out synthesis for more advanced LTL specifications. However, MDP abstractions from a domain partition can only be approximate, which causes issues with respect to the robustness of the designed controllers.

The problem of synthezing controllers was extended to continuous-input stochastic systems subject to subsets of ω-regular properties in few related works. For instance, the theory developed in [35] uses abstraction-based methods for approximating the maximal winning region of discrete-time continuous-input systems with Büchi objectives. Approximate abstractions are employed in [36] for synthesizing controllers from a continuous set of input maximizing the probability of satisfying syntactically co-safe LTL specifications. A thorough investigation of the reachability problem for similar systems was conducted in [37].

In summary, this literature survey provides evidence that current controller synthesis techniques for stochastic systems against complex specifications are still in a nascent stage of development. Existing tools lack in scalability — it takes several days for [26] to design a controller for 2D linear dynamics with simple reachability objectives — and are restricted to very specific classes of objectives. In addition, synthesis problems in the discussed papers mostly consider systems with a finite number of modes and, to the best of our knowledge, similar problems allowing for continuous sets of inputs have been only scarcely treated. These observations motivate the work presented in the following chapters.

CHAPTER 3
VERIFICATION AND SYNTHESIS FOR STOCHASTIC SYSTEMS: A FINITE-STATE ABSTRACTION APPROACH

The study of stochastic systems involves a technical machinery that is distinct from that commonly employed in the context of deterministic systems. Consequently, we use this chapter to establish some necessary preliminaries before expounding our contributions in the remainder of the dissertation.

The first objective of this chapter is to introduce the mathematical framework used to describe stochastic systems throughout this work in Section 3.1. Then, in Section 3.2, the wide class of ω-regular system specifications, which are the focus of this document, is presented. In Section 3.3, the verification and controller synthesis problems against such specifications are subsequently defined in the context of stochastic systems. Lastly, in Section 3.4, we review three types of stochastic transition systems, namely *Interval-valued Markov Chains* (IMC), *Bounded-Parameter Markov Decision Processes* (BMDP) and *Controlled Interval-valued Markov Chains* (CIMC), which can serve as finite-state abstractions of stochastic systems and are the main verification and synthesis tools discussed in the next chapters.

3.1 Stochastic Systems Models

Systems are typically represented as a set of quantities, also known as *a state*, evolving in time from an initial condition. Throughout this dissertation, we make the assumption that systems evolve in discrete-time, that is, the state of the system has well-defined values only at discrete instances of time.

Given the state of an uncontrolled stochastic system at a time k, its state at the next time step $k + 1$ is determined by both its current state and a stochastic realization of some random quantity. Mathematically, such a system takes the form of a *Stochastic Difference Equation* (SDE)

$$x[k+1] = \mathcal{F}(x[k], w[k]) \;, \tag{3.1}$$

where $x[k] \in D \subset \mathbb{R}^n$ is the state of the system on a domain D at time $k \in \mathbb{N}$, $w[k] \in W \subset \mathbb{R}^m$ is a random variable living on a support W and sampled at each time step k, and $\mathcal{F} : D \times W \to D$ is a function. At time $k = 0$, the system is set to an *initial state* $x[0]$ and thereafter evolves according to (3.1). An infinite sequence of states $\pi = x[0]x[1]x[2]\ldots$ generated by (3.1), where $x[1] = \mathcal{F}(x[0], w[0])$, $x[2] = \mathcal{F}(x[1], w[1])\ldots$, is called *an infinite path*. As the sequence of stochastic realizations $w[0]w[1]\ldots$ may be different for each execution of the system, (3.1) may produce different paths from the same initial condition $x[0]$. We assume throughout this work that the set of all possible initial conditions is the entire domain D.

For complex systems, it is generally not possible to find a closed-form solution of (3.1) where $x[k]$ is written explicitly in terms of k, $w[k]$ and $x[0]$, and one must rely on the analysis of $\mathcal{F}$ in the recursive system equation in order to make inferences on the behavior of (3.1).

When an external agent interacts with the stochastic system under consideration, affecting the state transitions throughout time, the model additionally incorporates a control parameter and the resulting SDE becomes

$$x[k+1] = \mathcal{F}(x[k], u[k], w[k]) \;, \tag{3.2}$$

where $w[k]$ is a random variable, $u[k] \in U \subseteq \mathbb{R}^\ell$ is a time-dependent control input, and everything else is defined as in (3.1). The set of all finite paths of (3.2) is denoted by $Paths_{fin}$. A function $\mu : Paths_{fin} \to U$ assigning an input to each finite path in (3.2) is called a *control policy* and the set of all control policies of (3.2) is denoted by $\mathcal{U} = \{\mu \mid$

$\mu : Paths_{fin} \rightarrow U\}$. Although the paths generated by (3.2) under a sequence of inputs $u[0]u[1]u[2]...$ are still stochastic in general, a well-designed control policy can influence the random evolution of the system and in turn maximize the probability of occurrence of some desired performance objective.

In this work, we distinguish the case where the set of available inputs U is uncountably-infinite from the case where U is a countably finite set of possible actions. In the latter case, we can view (3.2) as a finite-mode switched stochastic system with equation

$$x[k+1] = \mathcal{F}_a(x[k], w_a[k]) \; , \tag{3.3}$$

where $a \in A := \{0, 1, \ldots, N\}$ is a finite set of modes. At each time step, the external agent chooses a mode a and the system performs a transition according to the dynamics defined by $\mathcal{F}_a$ and a random realization of w_a. Note that, in this framework, we allow the noise term to be a function of the selected mode a.

3.2 ω-regular Specifications

We are interested in studying the behavior of stochastic systems which is fully characterized by the sequences of states produced according to the SDE models. Important behaviors are, for example, *safety* specifications, for which a path generated by a system must remain indefinitely inside a "safe" subset of the considered domain, or *reachability* specifications, where the path must reach a desired subset of goal states.

Expressing high-level system objectives is accomplished through the use of various symbolic *temporal logic* systems which allow to reason about the occurrence of events throughout time. Each logic structure possesses its own set of operators and syntactic rules for constructing *temporal properties* over a set of possible events. Different temporal logics usually display distinct levels of expressiveness, where certain properties can be enunciated in one logic and not in the other, or vice versa. Frequently utilized temporal logics include

Linear Temporal Logic (LTL) [38] and *Computation Tree Logic* (CTL) [39], whose most common temporal operators are

- The 'Always' operator $\Box$, which enforces an event to hold for all time (e.g. $\Box a$ means that event a is always true),
- The 'Eventually' operator $\Diamond$, which enforces an event to hold at some point in time (e.g. $\Diamond a$ means that event a has to occur at least once in the future),
- The 'Next' operator $\bigcirc$, which asks for a specific event to occur next,
- The 'Until' operator U, which requires some event to be true until another event occurs (e.g. aUb means that event a has to hold true as long as b hasn't occurred),
- The 'Implication' operator $\rightarrow$, which indicates that the occurrence of some event implies the occurrence of another event (e.g. $a \rightarrow b$ means that event b has to occur if event a is triggered).

These operators can be combined with each other to express even richer properties, such as *persistence* specifications — e.g. $\Diamond\Box a$, for which event a eventually holds true forever — or *liveness* specifications — e.g. $\Box\Diamond a$, for which event a has to occur infinitely often.

To employ these symbolic systems in the context of system analysis, a label is assigned to all states of the system domain of interest. These labels associate each transition performed by the system to an event. Hence, we define a *labeling function* $L : D \rightarrow \Pi$ mapping every state in the domain D to an element of Π, where Π is a *finite alphabet* of *atomic propositions*. Any infinite path $\pi = x[0]x[1]x[2]\ldots$ generated by a system induces an *infinite word* called a *trace* $\mathcal{L}(\pi) = L(x[0])L(x[1])L(x[2])\ldots$ from which the satisfaction of a specification can be assessed.

A temporal property ϕ defined over a finite alphabet Π can be viewed as a subset of $(\Pi)^\omega$, the set of all infinite words constructed from a concatenation of the elements in Π [40, Chapter 3]. Thus, ϕ is a set of infinite words representing admissible behaviors with respect to a system of interest. We introduce the *satisfaction relation* $\models$ between an infinite path π and a property $\phi \subseteq (\Pi)^\omega$ as

$$\pi \models \phi \Leftrightarrow \mathcal{L}(\pi) \in \phi \,. \tag{3.4}$$

If the trace of a path π does not belong to ϕ, then π violates ϕ, denoted by $\pi \not\models \phi$.

Example 1. *Consider a labeled system with alphabet $\Pi = \{a, b\}$ and the LTL specification $\phi = \Diamond a$. A path inducing a trace $\mathcal{L}_1 = bbbba\ldots$ satisfies ϕ, while a path inducing a trace $\mathcal{L}_2 = bbbbb\ldots$ without ever reaching an "a" state does not satisfy ϕ.*

Verification and synthesis tools for stochastic systems found in the recent literature often restrict their scope to specifications belonging only to LTL or only to CLT, or to a subset of these two logics. Here, we aim to develop a set of techniques that are directly applicable to all "interesting" system specifications.

The class of *ω-regular properties*, which is a strict superset of both LTL and CTL in expressiveness [27], stands out as a good candidate for fulfilling this criterion. Informally speaking, this class encompasses all infinite-time temporal properties whose satisfaction can be assessed using a finite amount of memory. Any ω-regular properties Ψ over alphabet Π has an *ω-regular expression* form G_Ψ

$$G_\Psi = E_1.F_1^\omega + \ldots + E_n.F_n^\omega \,, \tag{3.5}$$

where $n \geq 1$ and $E_1, \ldots, E_n, F_1, \ldots F_n$ are *regular expressions* over Π such that $\epsilon \notin \mathcal{L}(F_i)$, for all $1 \leq i \leq n$, with ω being the infinite repetition operator, ϵ being the empty word and . being the concatenation operator [40, Section 4.3.1]. A regular expression is semantically interpreted as a set of finite words exhibiting some pattern described using a finite number of symbols. For in-depth definitions of a regular expression and the aforementioned operators, we refer the interested reader to [40, Section A.2]. The ω-regular property Ψ is the set of all words $\mathcal{L}(G_\Psi)$ induced by the ω-regular expression G_Ψ, that is,

$$\Psi = \mathcal{L}(G_\Psi) = \mathcal{L}(E_1).\mathcal{L}(F_1^\omega) \cup \ldots \cup \mathcal{L}(E_n).\mathcal{L}(F_n^\omega) \,, \tag{3.6}$$

where $\mathcal{L}(E_i)$ and $\mathcal{L}(F_i^\omega)$ are the sets of all words induced by E_i and F_i^ω respectively, for all $1 \leq i \leq n$.

Example 2. *The LTL specification $\Diamond a$ over alphabet Π has an ω-regular expression $(\Pi)^*.a.(\Pi)^{\omega_r}$, where $*$ denotes the finite repetition operator and ω_r denotes the infinite repetition operator. The LTL specification $\Box\Diamond a$ has an ω-regular expression $((\Pi)^*.a)^{\omega_r}$.*

3.3 The Probabilistic Verification Problem and the Probabilistic Synthesis Problem

As discussed in Section 3.1, stochastic systems can produce different paths from the same initial state due to their random nature, as opposed to deterministic systems, for which a path generated from a given initial condition will always be the same. This fact calls for a distinct framework for assessing the fulfillment of a specification in a stochastic sense.

In this context, we are most interested in determining whether *the probability* that a stochastic system satisfies a specification from some initial condition is above or below a fixed threshold. This probability quantifies the frequency of "success" of the system when executed several times with the same initial state. Furthermore, if the system admits a control input, a natural objective consists in devising a control policy that increases the probability of fulfilling a goal specification, or reduces the risk of an unwanted behavior.

It is therefore necessary to carefully define a probability measure over the paths of (3.1). We denote by $\mathcal{B}(D)$ the σ-algebra generated by the domain $D \subset \mathbb{R}^n$ of (3.1), and introduce the Borel-measurable stochastic kernel $T : D \times \mathcal{B}(D) \to [0,1]$, where $A \in \mathcal{B}(D)$ is a Borel set of D, induced by the stochastic dynamics, that is, $T(x, A) = Pr(x' = \mathcal{F}(x, w) \in A \mid x)$. The function T quantifies the probability of making a transition from any state x to any (Borel) subset of D. Additionally, we denote the set of all paths of

(3.1) by $Paths = Paths_{fin} \cup Paths_{\omega}$, where $Paths_{fin}$ and $Paths_{\omega}$ refer to the sets of all finite and infinite paths of (3.1) respectively. The *cylinder set* $Cyl(x_0 A_1 A_2 \ldots A_k)$, with $A_i \in \mathcal{B}(D) \quad \forall i \leq k$, is the set of all paths $\pi \in Paths$ such that $x[0] = x_0 \in D$ and $x[i] \in A_i,\ \ i = 1, 2 \ldots k$, where $x[i]$ is the state in the i-th position of π [25]. Cylinder sets can be viewed as the set of all (finite and infinite) paths starting at x_0 and making a transition to the set A_i at time step i, $\forall i \leq k$. For a chosen initial state x_0, a probability measure on the σ-algebra $\mathcal{B}(Paths)$ is derived from probabilities on cylinder sets given by

$$Pr(Cyl(x_0 A_1 A_2 \ldots A_k)) = \int_{A_1} \int_{A_2} \cdots \int_{A_k} T(x_{k-1}, dx_k) \ldots T(x_1, dx_2) T(x_0, dx_1) \ . \tag{3.7}$$

Now, consider an ω-regular property Ψ. Let $\Phi_{\Psi}^{x_0}$ denote the set of all satisfying paths with respect to Ψ starting from initial state x_0, that is $\Phi_{\Psi}^{x_0} = \{\pi \in Paths \mid \mathcal{L}(\pi) \in \Psi, x[0] = x_0\}$. It can be shown that $\Phi_{\Psi}^{x_0}$ is a measurable set and that the measure on $\Phi_{\Psi}^{x_0}$ corresponds to the probability for the system to satisfy Ψ from x_0, denoted by $p_{\Psi}^{x_0}$. The labeling function L of (3.1) induces a partition of D such that, for all elements $L_i \in \Pi$, $A_{L_i} = \{x \in D \mid L(x) = L_i\}$ and $D = \bigcup_{i=1}^{|\Pi|} A_{L_i}$. We make the further assumption that $A_{L_i} \in \mathcal{B}(D), \quad \forall i$. As discussed in [40, Remark 10.57], the set $\Phi_{\Psi}^{x_0}$ arises as a countable union and intersection of cylinder sets $Cyl(x_0 A_{L_1} \ldots A_{L_k})$ starting in x_0 and ranging over acceptable sequences of Borel sets $A_{L_1} \ldots A_{L_k}$ with respect to the property Ψ, where each A_{L_i} is a set with a fixed label from the finite partition induced by L. As per (3.7), $\Phi_{\Psi}^{x_0}$ is therefore measurable and $p_{\Psi}^{x_0}$ is consequently well-defined.

Identical probability measures on paths can be established for controlled stochastic systems of the form (3.2) under a fixed control policy $\mu \in \mathcal{U}$. We denote the transition kernel of such systems by $T(x, u, A)$, with $x \in D$, $u \in U$ and $A \in \mathcal{B}(D)$. Additionally, we denote by $(p_{\Psi}^{x_0})_{\mu}$ the probability of satisfying property Ψ from initial state x_0 under policy μ.

To formally inquire about the probabilistic behavior of (3.1), we introduce a *probability* *operator* $\mathcal{P}_{\bowtie p_{sat}}$ over ω-regular properties, with $\bowtie \in \{\leq, <, \geq, >\}$, $p_{sat} \in [0, 1]$. The latter allows us to construct probabilistic ω-regular specifications of the form

$$\phi = \mathcal{P}_{\bowtie p_{sat}}[\Psi] \, , \tag{3.8}$$

where Ψ is an ω-regular specification. For any initial state $x_0 \in D$, we define the satisfaction relation $\models$ over such specifications, where

$$x_0 \models \mathcal{P}_{\bowtie p_{sat}}[\Psi] \Leftrightarrow p_{\Psi}^{x_0} \bowtie p_{sat} \, , \tag{3.9}$$

with $p_{\Psi}^{x_0}$ being the probability that a random path starting in x_0 satisfies property Ψ. In brief, x_0 satisfies ϕ if the probability of satisfying Ψ from x_0 is above or below a threshold p_{sat}.

We now have all the tools to state the first main problem treated in this dissertation, which is the *Probabilistic Verification Problem.*

<u>Probabilistic Verification Problem</u>

"Given a stochastic system (3.1) *and a probabilistic ω-regular formula of the form* (3.8)*, find the set of initial states of* (3.1) *satisfying* (3.8)*."*

For system (3.2) allowing a control action at each time step, our objective is to design probability maximizing or minimizing control policies $\widehat{\mu}_{\Psi}$ and $\breve{\mu}_{\Psi}$ with respect to some ω-regular specification Ψ, that is,

$$\breve{\mu}_{\Psi} = \arg\min_{\mu \in \mathcal{U}} (p_{\Psi}^{x_0})_{\mu} \tag{3.10}$$

$$\widehat{\mu}_{\Psi} = \arg\max_{\mu \in \mathcal{U}} (p_{\Psi}^{x_0})_{\mu} \tag{3.11}$$

for any initial state $x_0 \in D$, where $(p_{\Psi}^{x_0})_{\mu}$ is the probability that a random path starting in

x_0 satisfies property Ψ under policy μ. This second main problem explored in this work is referred to as the *Probabilistic Synthesis Problem*.

Probabilistic Synthesis Problem

"Given a stochastic system (3.2) *and an ω-regular specification Ψ, find a control policy $\widehat{\mu}_\Psi$ ($\widecheck{\mu}_\Psi$) that maximizes (minimizes) the probability of satisfying Ψ for any initial state x_0."*

In the next section, we discuss our suggested approach for addressing these two problems.

3.4 Finite-State Abstractions: Interval-valued Markov Chains, Bounded-Parameter Markov Decision Processes and Controlled Interval-valued Markov Chains

The probabilistic verification and synthesis problems were shown to be, in general, infeasible to solve exactly [23, 41]. This means that, for most systems and specifications of interest, it is not possible to find a closed-form classifier that separates initial states satisfying (3.8) from those that do not. Likewise, it is usually equally impossible to compute closed-form functions corresponding to the optimal control policies (3.10) and (3.11). Nevertheless, it is common to resort to *approximation* methods for tackling these problems.

A prevalent technique consists in constructing a *finite-state abstraction* of the stochastic system at hand in the form of a *probabilistic transition system.* Performing verification and controller synthesis on such abstractions yields bounded-error probabilistic guarantees which are mapped onto the original continuous system states. These finite-state abstractions typically arise from a *finite partition P* of the continuous domain D of the system.

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

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

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

3.4.1 Solving the Verification Problem using Interval-valued Markov Chain Abstractions

Let us first consider the uncontrolled system (3.1). For such a partition P of the domain D of (3.1), 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, that is, $T(x, Q_\ell)$ and $T(x', Q_\ell)$ may be different for two continuous states x and x' in Q_j. Therefore, we cannot use partition P to exactly abstract the system into a standard finite Markov Chain. Instead, we aim to produce an Interval-valued Markov Chain (IMC) abstraction of the system where the transition probabilities between states are constrained within some bounds. Such an abstraction is depicted in Figure 1.1.

Definition 2 (Interval-Valued Markov Chain)**.** *An* Interval-Valued Markov Chain (IMC) *[14] is a 6-tuple $\mathcal{I} = (Q, \check{T}, \widehat{T}, q_0, \Pi, L)$ where:*

- *Q is a finite set of states,*
- *$\check{T} : Q \times Q \to [0,1]$ maps pairs of states to a lower transition bound so that $\check{T}_{Q_j \to Q_\ell} := \check{T}(Q_j, Q_\ell)$ denotes the lower bound of the transition probability from state Q_j to state Q_ℓ,*
- *$\widehat{T} : Q \times Q \to [0,1]$ maps pairs of states to an upper transition bound so that $\widehat{T}_{Q_j \to Q_\ell} := \widehat{T}(Q_j, Q_\ell)$ denotes the upper bound of the transition probability from state Q_j to state Q_ℓ,*
- *$q_0 \subseteq Q$ is a set of initial states,*
- *Π is a finite set of atomic propositions,*
- *$L : Q \to \Pi$ is a labeling function from states to Π,*

and $\check{T}$ *and* $\widehat{T}$ *satisfy* $\check{T}(Q_j, Q_\ell) \leq \widehat{T}(Q_j, Q_\ell)$ *for all* $Q_j, Q_\ell \in Q$ *and*

$$\sum_{Q_\ell \in Q} \check{T}(Q_j, Q_\ell) \leq 1 \leq \sum_{Q_\ell \in Q} \widehat{T}(Q_j, Q_\ell) \tag{3.12}$$

for all $Q_j \in Q$.

In the interest of clarity, we assume in this work that any state of an IMC $\mathcal{I}$ can serve as an initial state.

An IMC $\mathcal{I}$ is interpreted as an *Interval Markov Decision Process* (IMDP) [42] if, at each time step k, the environment non-deterministically chooses a transition matrix T_k where each entry satisfies the bounds defined by the transition bound functions of $\mathcal{I}$ and the next transition occurs according to T_k. A mapping ν from a finite path $\pi = q_0 q_1 \ldots q_k$ in $\mathcal{I}$ to a transition matrix T_k is called an *adversary*. The behavior of $\mathcal{I}$ under some adversary ν reduces to that of a Markov chain denoted by $\mathcal{I}[\nu]$. The set of all adversaries of $\mathcal{I}$ is denoted by $\nu_{\mathcal{I}}$. The IMDP interpretation of IMCs is assumed throughout this work. For more details on possible semantic interpretations of an IMC, see [42].

Definition 3 (IMC Abstraction)**.** *Given the system* (3.1) *evolving on a domain* $D \subset \mathbb{R}^n$ *and a partition* $P = \{Q_j\}_{j=1}^m$ *of* D*, an IMC* $\mathcal{I} = (Q, \check{T}, \widehat{T}, q_0, \Pi, L)$ *is an* abstraction *of* (3.1) *if:*

- $P = Q$*, that is, the set of states of the IMC is the partition* P*,*
- *For all* $Q_j, Q_\ell \in P$*,*

$$\check{T}_{Q_j \to Q_\ell} \leq \inf_{x \in Q_j} T(x, Q_\ell), \text{ and} \tag{3.13}$$

$$\widehat{T}_{Q_j \to Q_\ell} \geq \sup_{x \in Q_j} T(x, Q_\ell), \tag{3.14}$$

- $P = q_0$*, i.e., the set of initial states of the IMC 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,*

- *$\mathcal{I}$ is interpreted as an IMDP.*

The fact that two continuous states within the same discrete state of the abstraction may engender different transition probabilities is captured by interpreting the IMC as an IMDP.

Performing verification on an IMC abstraction, which is an over-approximation of all possible continuous-state behaviors of (3.1), provides probabilistic guarantees with respect to the original system's states. A consequence of model checking an IMC $\mathcal{I}$, whose transitions are characterized by transition probability intervals, is that the probability of satisfying property Ψ from any of its initial states Q_j must be specified as an interval $I_j = [p^j_{min}, p^j_{max}]$ as well, where $\mathcal{P}_{\mathcal{I}[\nu]}(Q_j \models \Psi) \in I_j,\ \forall \nu \in \nu_{\mathcal{I}}$, that is, p^j_{min} is a lower bound on the probability of satisfying Ψ from state Q_j over all possible adversaries of $\mathcal{I}$ and p^j_{max} is an upper bound on the probability of satisfying Ψ from state Q_j over all possible adversaries of $\mathcal{I}$. For any initial state Q_j in an IMC $\mathcal{I}$, we define the satisfaction relation $\models$ for formulas of the type (3.8) where

$$Q_j \models \mathcal{P}_{\bowtie p_{sat}}[\Psi] \Leftrightarrow (p_\Psi^{Q_j})_\nu \bowtie p_{sat} \ \ \forall \nu \in \nu_{\mathcal{I}} \ , \tag{3.15}$$

with $(p_\Psi^{Q_j})_\nu$ being the probability that the word generated by a random path starting in Q_j satisfies property Ψ under adversary ν. We denote the set of initial states satisfying ϕ in $\mathcal{I}$ by $(Q_\phi^{yes})_{\mathcal{I}}$, while states that do not satisfy ϕ are in $(Q_\phi^{no})_{\mathcal{I}}$. Note that any Q_j such that $p_{sat} \in]p^j_{min}, p^j_{max}[$ if $\bowtie \in \{\leq, \geq\}$, or $p_{sat} \in [p^j_{min}, p^j_{max}]$ if $\bowtie \in \{<, >\}$, is undecided with respect to ϕ in (3.8) and we write $Q_j \in (Q_\phi^{?})_{\mathcal{I}}$. The remaining states either satisfy ϕ or do not satisfy ϕ.

Fact 1. *Let $\mathcal{I}$ be an IMC abstraction of* (3.1) *induced by a partition $P = \{Q_j\}_{j=1}^m$ of D. For any formula of the form* (3.8)*, it holds that:*

- $Q_j \in (Q_\phi^{yes})_{\mathcal{I}} \Rightarrow x \in Q_\phi^{yes} \quad \forall x \in Q_j$
- $Q_j \in (Q_\phi^{no})_{\mathcal{I}} \Rightarrow x \in Q_\phi^{no} \quad \forall x \in Q_j$.

Given an IMC abstraction $\mathcal{I}$ of (3.1) generated from a partition P of D, our approach for addressing the stochastic verification problem is thus to implement a technique for determining non-trivial values of p_{min}^j and p_{max}^j and sort all states of P into the sets $(Q_\phi^{yes})_{\mathcal{I}}$, $(Q_\phi^{no})_{\mathcal{I}}$ and $(Q_\phi^{?})_{\mathcal{I}}$.

3.4.2 Solving the Synthesis Problem using Bounded-Parameter Markov Decision Processes Abstractions and Controlled Interval-valued Markov Chain Abstractions

Next, we consider systems of the form (3.3) with a finite number of modes. Recall that, at each time step, an external agent chooses a mode and the next stochastic transition of (3.3) takes place according to the dynamics defined by the selected mode. As in the uncontrolled case, for a fixed mode a of (3.3), the probability of making a transition from a discrete state Q_j of a given partition P to another state Q_ℓ is not uniquely defined as $T(x, a, Q_\ell)$ and $T(x', a, Q_\ell)$ may not be the same for two continuous states $x, x' \in Q_\ell$. Hence, systems of the form (3.3) are abstracted by Bounded-parameter Markov Decision Processes (BMDP) from a partition P of the domain D, where the probability of transition between any two states is given as an interval of possible probabilities for any action of the BMDP representing a mode of the original system, as illustrated in Figure 3.1.

Definition 4 (Bounded-parameter Markov Decision Process)**.** *A* Bounded-parameter Markov Decision Process (BMDP) *[43] is a 7-tuple* $\mathcal{B} = (Q, Act, \check{T}, \widehat{T}, q_0, \Pi, L)$ *where:*

- Q *is a finite set of states,*
- Act *is a finite set of actions,*
- $\check{T} : Q \times Act \times Q \to [0, 1]$ *maps pairs of states and an action to a lower transition*

bound so that $\widecheck{T}_{Q_j \xrightarrow{a} Q_\ell} := \widecheck{T}(Q_j, a, Q_\ell)$ *denotes the lower bound of the transition probability from state* Q_j *to state* Q_ℓ *under action* $a \in A(Q_j)$*,*

- $\widehat{T} : Q \times Act \times Q \to [0,1]$ *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,*
- Π *is a finite set of atomic propositions,*
- $L : Q \to \Pi$ *is a labeling function from states to* Π*,*

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

$$\sum_{Q_\ell \in Q} \widecheck{T}(Q_j, a, Q_\ell) \leq 1 \leq \sum_{Q_\ell \in Q} \widehat{T}(Q_j, a, Q_\ell) \tag{3.16}$$

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

Denoting the set of all finite paths of a BMDP $\mathcal{B}$ by $(Paths_{fin})_{\mathcal{B}}$, a switching policy $\mu : (Paths_{fin})_{\mathcal{B}} \to Act$ 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 $\mathcal{U}_{\mathcal{B}} = \{\mu \mid \mu : (Paths_{fin})_{\mathcal{B}} \to Act\}$. 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 an (possibly countably infinite-state) IMC. As will be discussed further, only finite-memory policies need to be considered in this work, which induce finite-state IMCs. The IMC *induced* by policy μ in BMDP $\mathcal{B}$ is denoted by $\mathcal{B}[\mu]$.

Definition 5 (BMDP Abstraction)**.** *Given the system* (3.3) *evolving on a domain* $D \subset \mathbb{R}^n$ *and a partition* $P = \{Q_j\}_{j=1}^m$ *of* D*, a BMDP* $\mathcal{B} = (Q, Act, \widecheck{T}, \widehat{T}, q_0, \Pi, L)$ *is an* abstraction *of* (3.3) *if:*

- $P := Q$*, 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* (3.3)*,*
- *For all $Q_j, Q_\ell \in P$ and action $a \in Act$,*

$$\widecheck{T}_{Q_j \xrightarrow{a} Q_\ell} \leq \inf_{x \in Q_j} T(x, a, Q_\ell), \text{ and} \tag{3.17}$$

$$\widehat{T}_{Q_j \xrightarrow{a} Q_\ell} \geq \sup_{x \in Q_j} T(x, a, Q_\ell), \tag{3.18}$$

- *$P = q_0$, 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,*
- *Any IMC $\mathcal{I}_a$ induced by an action $a \in Act$ is interpreted as an IMDP.*

Model checking a BMDP $\mathcal{B}$ under switching policy μ against specification Ψ is equivalent to verifying an IMC. Because the probability of satisfying a specification Ψ in an IMC is not uniquely defined and depends on the instanciation of a non-deterministic adversary, the verification of an IMC induced by a policy μ in a BMDP does not result in a fixed probability but in an interval of satisfaction $(I_j)_\mu = [(p^j_{min})_\mu, (p^j_{max})_\mu]$ for all initial states Q_j, where $\mathcal{P}_{\mathcal{B}[\mu][\nu]}(Q_j \models \Psi) \in (I_j)_\mu,\ \forall \nu \in \nu_{\mathcal{B}[\mu]}$. A policy μ for a BMDP abstraction of (3.3) maps to a policy for (3.3) in the natural way, *i.e.*, at state $x \in Q_j$, the control action prescribed by μ at discrete state Q_i is applied to (3.3). It then holds that the exact probability of satisfying Ψ from any initial state $x \in Q_j$ for (3.3) is contained within the bounds $(I_j)_\mu$ [26]. Therefore, given a BMDP abstraction $\mathcal{B}$ of (3.3) generated from a partition P of the domain D, our approach to the stochastic synthesis problem for finite-mode systems is to find policies $\widehat{\mu}^{low}_\Psi$ and $\widecheck{\mu}^{up}_\Psi$ that respectively maximize the lower bound probability and minimize the upper bound probability of satisfying Ψ for all initial states Q_j of $\mathcal{B}$. Specifically, given a system of the form (3.3), a partition P of its domain D, a BMDP abstraction $\mathcal{B}$ of (3.3) arising from P, any initial state $Q_j \in Q$ of $\mathcal{B}$ and an ω-regular property Ψ, we

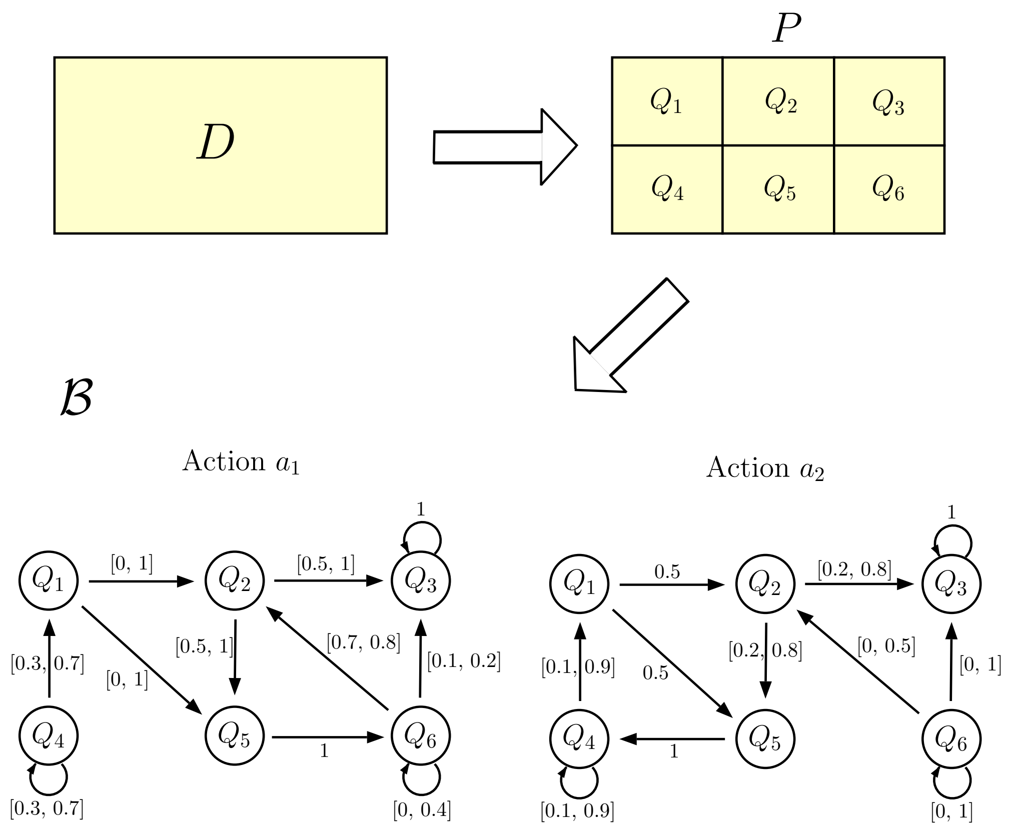


Figure 3.1: A finite-state BMDP abstraction $\mathcal{B}$ over a continuous domain D. A partition P of D is generated and bounds on the transition probabilities between states are estimated for two actions a_1 and a_2 of $\mathcal{B}$.

want to compute switching policies $\check{\mu}_{\Psi}^{up} \in \mathcal{U}_{\mathcal{B}}$ 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.*,

$$\check{\mu}_{\Psi}^{up} = \arg\min_{\mu \in \mathcal{U}_{\mathcal{B}}} \widehat{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) \tag{3.19}$$

$$\widehat{\mu}_{\Psi}^{low} = \arg\max_{\mu \in \mathcal{U}_{\mathcal{B}}} \check{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi)\ . \tag{3.20}$$

Then, we focus our attention on controlled stochastic systems of the general form (3.2) where an input is drawn from a continuous set at every time step. Solving the stochastic synthesis 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. 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 (3.2).

Definition 6 (Controlled Interval-valued Markov Chain)**.** *A* Controlled Interval-valued Markov Chain (CIMC) *is a 7-tuple $\mathcal{C} = (Q, U, \check{T}, \widehat{T}, q_0, \Pi, L)$ 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 7 (Controlled Interval-valued Markov Chain Abstraction)**.** *Given the system* (3.2) *evolving on a domain $D \subset \mathbb{R}^n$ and a partition $P = \{Q_j\}_{j=1}^m$ of D, a CIMC $\mathcal{C} = (Q, U, \check{T}, \widehat{T}, q_0, \Pi, L)$ is an* abstraction *of* (3.2) *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 $\mu : (Paths_{fin})_{\mathcal{C}} \to U$ 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 $\mathcal{U}_{\mathcal{C}} = \{\mu \mid \mu : (Paths_{fin})_{\mathcal{C}} \to U\}$. A policy μ applied to a CIMC $\mathcal{C}$ induces an IMC denoted by $\mathcal{C}[\mu]$. As in the finite-mode case, 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, ruling out the possibility of using an exhaustive search. Formally, given a system of the form (3.2), a partition P of its domain D, a CIMC abstraction $\mathcal{C}$ of (3.2) arising from P, any initial state $Q_j \in Q$ of $\mathcal{C}$ and an ω-regular property Ψ, we want to compute control policies $\check{\mu}_{\Psi}^{up} \in \mathcal{U}_{\mathcal{C}}$ 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.*,

$$\breve{\mu}_{\Psi}^{up} = \arg\min_{\mu \in \mathcal{U}_{\mathcal{C}}} \widehat{\mathcal{P}}_{\mathcal{C}[\mu]}(Q_j \models \Psi) \tag{3.21}$$

$$\widehat{\mu}_{\Psi}^{low} = \arg\max_{\mu \in \mathcal{U}_{\mathcal{C}}} \breve{\mathcal{P}}_{\mathcal{C}[\mu]}(Q_j \models \Psi) \ . \tag{3.22}$$

In light of the proposed approaches, the remainder of this dissertation will first focus on the development of efficient finite-state abstraction techniques for specific classes of discrete-time stochastic dynamical systems. Then, verification and synthesis algorithms for ω-regular specifications applicable to these abstractions are developed in subsequent chapters and used to provide solutions to the stochastic verification problem and the stochastic synthesis problem.

CHAPTER 4
IMC AND BMDP ABSTRACTION TECHNIQUES

In this chapter, we discuss several techniques for constructing IMC and BMDP abstractions applicable to specific classes of stochastic dynamical systems. These finite abstractions are amenable to formal verification and synthesis, and therefore serve as the main instruments to address the stochastic verification problem and the stochastic synthesis problem enunciated in Chapter 3. Unlike sampling techniques employed in related works, our objective is to compute transition probability intervals that are guaranteed to be correct while being non-trivial — that is, ranging from 0 to 1 for all transitions.

First, in Section 4.1, we present an efficient IMC abstraction procedure for the wide class of *mixed monotone systems* with additive disturbance. Mixed monotonicity generalizes the property of monotonicity for dynamical systems for which trajectories maintain a partial ordering on states [44], [45], [46]. Many physical systems have been shown to be monotone or mixed monotone such as biological systems [12] and transportation networks [10], [47], [11].

Next, in Section 4.2, we detail an IMC abstraction method for *polynomial* stochastic systems which is based on so-called *stochastic barrier functions*. Stochastic barrier functions are used to provide probabilistic guarantees of set invariance and reachability without requiring explicit computations of reachable sets. In particular, we show that an IMC abstraction of polynomial systems can be constructed from a discrete partition of the continuous domain by solving two Sum-of-Squares (SOS) optimization programs per transition.

Note that any finite-mode system whose individual modes belong to one of the above classes of systems can be abstracted by a BMDP using the techniques presented in this chapter.

4.1 Abstraction of Mixed Monotone Systems with Additive Disturbance

In this section, we suggest an efficient and scalable IMC abstraction technique for affine-in-disturbance stochastic mixed monotone systems. Systems with mixed monotone state update maps exhibit considerable structure useful for analysis and control. As formally defined further, mixed monotone functions are *order-preserving* with respect to a so-called *decomposition function*. This property allows us to compute tight over-approximation of reachable sets from rectangular sets by evaluating the decomposition function at only two points, regardless of the dimension of the domain. Therefore, given a rectangular partition of the domain of a mixed monotone system, we can efficiently over-approximate the reachable set of any state in the partition. By restricting the additive disturbance to such reachable sets, lower and upper bounds on the probability of transition between any two state are efficiently determined under certain assumptions on the structure of the noise and the partition.

Consider the affine-in-disturbance stochastic system

$$x[k+1] = \mathcal{F}(x[k]) + w[k] \ , \tag{4.1}$$

where $x[k] \in D \subset \mathbb{R}^n$ is the state of the system on a domain D at time k, $\mathcal{F} : D \to D$ is a mixed monotone function and $w[k] \in W \subseteq \mathbb{R}^m$ is a *unimodal*, *symmetric* random disturbance with a diagonal covariance matrix as defined below. In the following definitions and throughout this section, all inequalities between vectors are interpreted element-wise.

Definition 8 (Mixed monotone function)**.** *[14] A function $\mathcal{F} : D \to D$ is* mixed monotone *if there exists a* decomposition function $g : D \times D \to D$ *satisfying [48, 13]:*

- $\forall x \in D : \mathcal{F}(x) = g(x, x)$
- $\forall x^1, x^2, y \in D : x^1 \leq x^2$ *implies* $g(x^1, y) \leq g(x^2, y)$
- $\forall x, y^1, y^2 \in D : y^1 \leq y^2$ *implies* $g(x, y^2) \leq g(x, y^1)$.

Mixed monotonicity generalizes the notion of monotonicity in dynamical systems, which is recovered when $g(x, y) = F(x)$ for all x, y.

Assumption 1. *$\mathcal{F}$ in* (4.1) *is mixed monotone with decomposition function $g(x, y)$.*

Definition 9 (Unimodal distribution)**.** *[14] For a random disturbance $\omega \in \Omega \subset \mathbb{R}$ with Ω an interval, its probability density function $f_\omega : \mathbb{R} \to \mathbb{R}$ is* unimodal *if f_ω is differentiable on Ω and there exists a unique number $c \in \mathbb{R}$, referred as the mode of the distribution, such that, for $x \in \Omega$:*

- $x < c \Rightarrow f'_\omega(x) \geq 0$,
- $x = c \Rightarrow f'_\omega(x) = 0$, *and*
- $x > c \Rightarrow f'_\omega(x) \leq 0$.

Definition 10 (Symmetric distribution)**.** *[14] For a random disturbance $\omega \in \Omega \subset \mathbb{R}$ with Ω an interval, its probability density function $f_\omega : \mathbb{R} \to \mathbb{R}$ is* symmetric *if there exists a number $d \in \mathbb{R}$ such that $f_\omega(d - x) = f_\omega(d + x)$ for all x.*

Assumption 2. *The random disturbance $w[k]$ in* (4.1) *is of the form $w[k] = [\ w_1[k], w_2[k], \ldots, w_n[k]\ ]^T$, 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. Furthermore, the probability density function f_{w_i} for each random variable w_i is symmetric and unimodal with mode c_i.*

For mixed monotone $\mathcal{F}$ with decomposition function g, for $x, y, z \in D$ satisfying $x \leq z \leq y$, we have $g(x, y) \leq \mathcal{F}(z) \leq g(y, x)$. This leads to the following proposition.

Proposition 1 ([13, Theorem 1])**.** *Let $\mathcal{F} : D \to D$ be mixed monotone with decomposition function $g : D \times D \to D$, and let $a, b \in D$ satisfy $a \leq b$. Then*

$$\{\mathcal{F}(x) : a \leq x \leq b\} \subseteq \{z : g(a, b) \leq z \leq g(b, a)\}\ . \tag{4.2}$$

Proposition 1 implies that the one-step reachable set from the rectangular region bounded between a and b is over-approximated by the rectangular region bounded by the two points $g(a, b)$ and $g(b, a)$. This property will prove key for efficient computation of IMC abstractions.

To exploit the mixed monotonicity property of $\mathcal{F}$, we make the additional assumption that the domain D of (4.1) admits a *rectangular* partition P.

Definition 11 (Rectangular Partition)**.** *A* rectangular partition P *of the domain* $D \subset \mathbb{R}^n$ *is a collection of discrete states* $P = \{Q_j\}_{j=1}^m,\ Q_j \subset D$*, satisfying*

- $Q_j = \{x : a^j \leq x \leq b^j\}$ *for some* $a^j, b^j \in \mathbb{R}^n$ *such that* $a^j \leq b^j,\ \forall j = 1, \ldots, m$*,*
- $\bigcup_{j=1}^m Q_j = D$*,*
- $\mathbf{int}(Q_j) \cap \mathbf{int}(Q_\ell) = \emptyset\ \ \forall j, \ell,\ j \neq \ell$,

where **int** *denotes interior.*

Assumption 3. *The partition* P *of the domain* D *of* (4.1) *is rectangular.*

We decompose our procedure for bounding the transition probability from a state $Q_1 \in P$ to a state $Q_2 \in P$ in two steps: first, we compute the rectangular over-approximation of the $\mathcal{F}$-reachable set from state Q_1 by exploiting the mixed monotonicity property of $\mathcal{F}$. Next, we determine the positions of f_w within this rectangular over-approximation that respectively minimize and maximize its overlap with state Q_2. In the next section, we exploit the characteristics of w previously evoked to streamline the calculation of these extremum points.

Proposition 2. *Consider system* (4.1) *under Assumptions 1–2. Let* $Q_1 = \{x : a^1 \leq x \leq b^1\}$ *and* $Q_2 = \{x : a^2 \leq x \leq b^2\}$ *be two nonempty rectangular sets with least point* a^j *and greatest point* b^j *for* $j = 1, 2$*. Then*

$$\min_{x \in Q_1} Pr(\mathcal{F}(x) + w \in Q_2) \;\geq\; \prod_{i=1}^{n} \min_{z_i \in [g_i(a^1,b^1), g_i(b^1,a^1)]} \int_{a_i^2}^{b_i^2} f_{w_i}(x - z_i)dx \tag{4.3}$$

and

$$\max_{x \in Q_1} Pr(\mathcal{F}(x) + w \in Q_2) \;\leq\; \prod_{i=1}^{n} \max_{z_i \in [g_i(a^1,b^1), g_i(b^1,a^1)]} \int_{a_i^2}^{b_i^2} f_{w_i}(x - z_i)dx \tag{4.4}$$

where g_i denotes the i-th element of $g(x,y)$, the decomposition function of $\mathcal{F}$.

Proof. By Proposition 1, we observe

$$\{\mathcal{F}(x) : x \in Q_1\} \subseteq \{z : g(a^1,b^1) \leq z \leq g(b^1,a^1)\} \, . \tag{4.5}$$

To prove (4.3), we have

$$\begin{aligned}
&\min_{x \in Q_1} Pr(\mathcal{F}(x) + w \in Q_2) \\
&\geq \min_{z: g(a^1,b^1) \leq z \leq g(b^1,a^1)} Pr(z + w \in Q_2) && (4.6)\\
&= \min_{z: g(a^1,b^1) \leq z \leq g(b^1,a^1)} \prod_{i=1}^{n} Pr(z_i + w_i \in [a_i^2, b_i^2]) && (4.7)\\
&= \prod_{i=1}^{n} \min_{z_i: g_i(a^1,b^1) \leq z_i \leq g_i(b^1,a^1)} Pr(z_i + w_i \in [a_i^2, b_i^2]) && (4.8)
\end{aligned}$$

where (4.6) follows from (4.5), (4.7) follows from the mutual independence of all components of w in Assumption 1, and (4.8) holds because $g(a^1,b^1) \leq z \leq g(b^1,a^1)$ if and only if $g_i(a^1,b^1) \leq z_i \leq g_i(b^1,a^1)$ for all $i = 1, \ldots, n$. Then (4.3) holds because $Pr(z_i + w_i \in [a_i^2, b_i^2]) = \int_{a_i^2}^{b_i^2} f_{w_i}(x - z_i)dx$. Finally, (4.4) holds by a symmetric argument as above, replacing $\min$ with $\max$. $\square$

Before generalizing to higher dimensions, we treat a 1-dimensional version of our original problem. In Lemma 1, we prove that for a fixed interval $[a,b] \subset \mathbb{R}$, there exists a

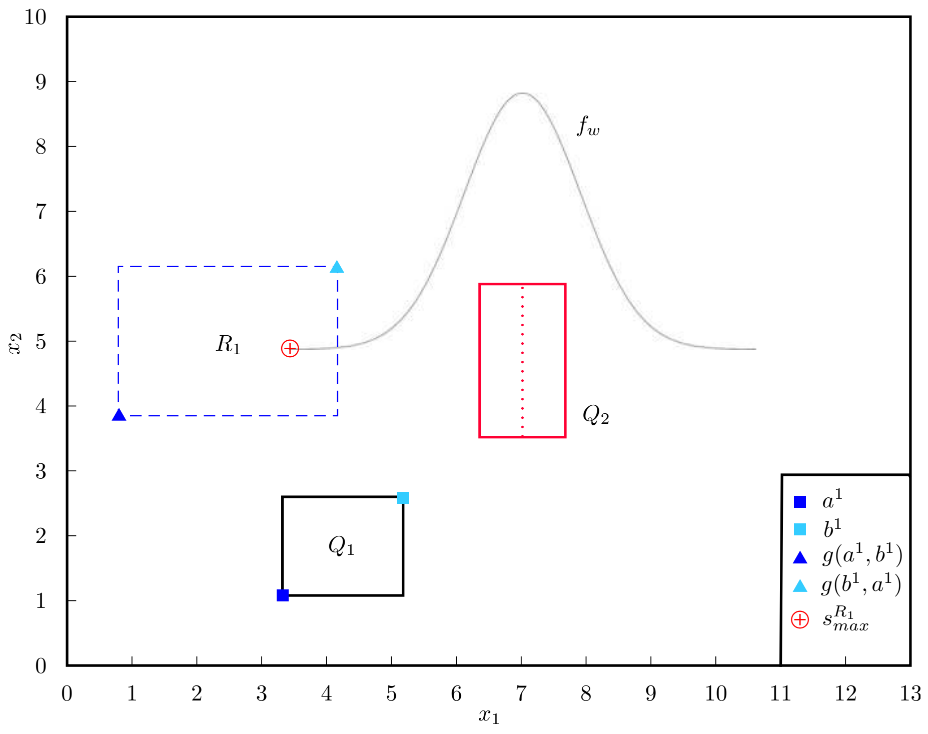


Figure 4.1: Schematic depiction of the procedure for computing an upper bound on the probability of transition from Q_1 to Q_2. First, the one-step reachable set R_1 from Q_1 is over-approximated by evaluating the decomposition function at only two extremal points, regardless of the state-space dimension. Then, the distribution of $z + w$ is positioned as close to the center of Q_2 as possible under the restriction that $z \in R_1$. A lower bound on the transition probability is achieved by positioning the distribution as far from the center of Q_2 as possible.

unique position for a unimodal and symmetric distribution which maximizes its integral over $[a, b]$.

Lemma 1. *Let $\omega \in \Omega \subset \mathbb{R}$ with Ω an interval be a random variable with symmetric and unimodal probability density function $f_\omega : \mathbb{R} \to \mathbb{R}$ and mode $c \in \mathbb{R}$. For any $a, b \in \mathbb{R}$ satisfying $a \leq b$ and any $r_1, r_2 \in \mathbb{R}$ satisfying $r_1 \leq r_2$, let*

$$s_{max} = \frac{a+b}{2} - c \tag{4.9}$$

and define

$$s^r_{max} = \arg\min_{s\in[r_1,r_2]} |s_{max} - s| = \begin{cases} s_{max}, & \textit{if } \ s_{max} \in [r_1, r_2] \\ r_2, & \textit{if } \ s_{max} > r_2 \\ r_1, & \textit{if } \ s_{max} < r_1 \ , \end{cases} \tag{4.10}$$

$$s^r_{min} = \arg\max_{s\in[r_1,r_2]} |s_{max} - s| = \begin{cases} r_1, & \textit{if } \ s_{max} > \frac{r_1+r_2}{2} \\ r_2, & \textit{otherwise} \ . \end{cases} \tag{4.11}$$

Then

$$\max_{s\in[r_1,r_2]} \int_a^b f_\omega(x-s)\ dx = \int_a^b f_\omega(x - s^r_{max})\ dx \tag{4.12}$$

$$\min_{s\in[r_1,r_2]} \int_a^b f_w(x-s)\ dx = \int_a^b f_\omega(x - s^r_{min})\ dx \ . \tag{4.13}$$

Proof. For $s \in \mathbb{R}$, define $H(s) = \int_a^b f_\omega(x-s)\ dx$. We claim

$$H(s_{max}) = \max_{s\in\mathbb{R}} H(s) \ , \tag{4.14}$$

and, moreover, for all $s_1, s_2 \in \mathbb{R}$ such that $|s_{max} - s_1| \geq |s_{max} - s_2|$, it holds that $H(s_1) \leq H(s_2)$, that is, $H(s)$ monotonically decreases as $|s_{max} - s|$ increases. Assuming the claim to be true, it follows that $\max_{s\in[r_1,r_2]} H(s) = H(s^r_{max})$ and $\min_{s\in[r_1,r_2]} H(s) = H(s^r_{min})$, *i.e.*, (4.12) and (4.13), completing the proof.

To prove the claim, we have, for all $s \in \mathbb{R}$,

$$\begin{aligned}
&H(s_{max}) - H(s) \\
&= \int_a^b f_\omega(x - s_{max})\, dx - \int_a^b f_\omega(x - s)\, dx && (4.15) \\
&= \int_{a-s_{max}}^{b-s_{max}} f_\omega(x)\, dx - \int_{a-s}^{b-s} f_\omega(x)\, dx && (4.16) \\
&= \int_{a-s_{max}}^{a-s} f_\omega(x)\, dx - \int_{b-s_{max}}^{b-s} f_\omega(x)\, dx && (4.17) \\
&= \int_0^{s_{max}-s} \left[f_\omega(x + \tfrac{a-b}{2} - c) - f_\omega(x + \tfrac{b-a}{2} - c) \right] dx\ . && (4.18)
\end{aligned}$$

Moreover, because f_ω is symmetric and unimodal with mode c, $f_\omega(x + \frac{a-b}{2} - c) - f_\omega(x + \frac{b-a}{2} - c)$ is an odd function of x and is negative for $x > 0$ and positive for $x < 0$. Therefore, the integral in (4.18) is nonnegative and monotonically decreases as $|s_{max} - s|$ increases, thus proving the claim. $\square$

When $s_{max} \in [r_1, r_2]$ in Lemma 1, the lemma confirms the intuitive idea that the integral of a unimodal, symmetric distribution over some interval $I = [a, b]$ is maximized when the peak of its probability distribution lies at the center of I. However, for the type of systems considered in this work, the shift of such distributions will always be restricted to take values within a given rectangular set $[r_1, r_2]$ so that, when $s_{max} \notin [r_1, r_2]$, the shift $s \in [r_1, r_2]$ maximizing the overlap of the density function over I is the one closest to the global maximizing shift s_{max}. Conversely, a shift $s \in [r_1, r_2]$ minimizing this overlap is the one furthest from s_{max}.

Theorem 1 combines Lemma 1 and Proposition 2 in order to provide a procedure for efficiently constructing an IMC abstraction for (4.1) given a rectangular partition of its domain D.

Theorem 1. *Consider system* (4.1) *under Assumptions 1–2 and let $P = \{Q_j\}_{j=1}^m$ be a rectangular partition of D with each $Q_j = \{x : a^j \leq x \leq b^j\}$ for some $a^j, b^j \in \mathbb{R}^n$*

satisfying $a^j \leq b^j$. *For all* $Q_j, Q_\ell \in P$, *let*

$$s^\ell_{i,max} = \frac{a^\ell_i + b^\ell_i}{2} - c_i \text{ for } i = 1, \ldots, n, \tag{4.19}$$

$$\check{r}^j = g(a^j, b^j)\ , \tag{4.20}$$

$$\hat{r}^j = g(b^j, a^j)\ , \tag{4.21}$$

and define

$$\begin{aligned}\widehat{T}_{Q_j \to Q_\ell} &= \prod_{i=1}^{n} \int_{a^\ell_i}^{b^\ell_i} f_{w_i}(x_i - s^{j\to\ell}_{i,max})\, dx_i\ , && (4.22)\\ &= \prod_{i=1}^{n} \left(F_{w_i}(b^\ell_i - s^{j\to\ell}_{i,max}) - F_{w_i}(a^\ell_i - s^{j\to\ell}_{i,max}) \right), && (4.23)\end{aligned}$$

$$\begin{aligned}\widecheck{T}_{Q_j \to Q_\ell} &= \prod_{i=1}^{n} \int_{a^\ell_i}^{b^\ell_i} f_{w_i}(x_i - s^{j\to\ell}_{i,min})\, dx_i && (4.24)\\ &= \prod_{i=1}^{n} \left(F_{w_i}(b^\ell_i - s^{j\to\ell}_{i,min}) - F_{w_i}(a^\ell_i - s^{j\to\ell}_{i,min}) \right) && (4.25)\end{aligned}$$

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

$$s^{j\to\ell}_{i,max} = \begin{cases} s^\ell_{i,max}, & \textit{if } s^\ell_{i,max} \in [\check{r}^j_i, \hat{r}^j_i] \\ \hat{r}^j_i, & \textit{if } s^\ell_{i,max} > \hat{r}^j_i \\ \check{r}^j_i, & \textit{if } s^\ell_{i,max} < \check{r}^j_i\ , \end{cases} \tag{4.26}$$

$$s^{j\to\ell}_{i,min} = \begin{cases} \check{r}^j_i, & \textit{if } s^\ell_{i,max} > \frac{\check{r}^j_i + \hat{r}^j_i}{2} \\ \hat{r}^j_i\ , & \textit{otherwise}\ . \end{cases} \tag{4.27}$$

Then $\mathcal{I} = (P, \widecheck{T}, \widehat{T})$ *is an IMC abstraction of* (4.1).

Proof. For all $Q_j, Q_\ell \in P$ and $i = 1, \ldots, n$, by Lemma 1,

$$\begin{aligned}\min_{z_i \in [g_i(a^j,b^j), g_i(b^j,a^j)]} & \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x - z_i)dx \\ &= \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x_i - s_{i,min}^{j\to\ell})\, dx_i\ , \qquad (4.28)\\ \max_{z_i \in [g_i(a^j,b^j), g_i(b^j,a^j)]} & \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x - z_i)dx \\ &= \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x_i - s_{i,max}^{j\to\ell})\, dx_i\ , \qquad (4.29)\end{aligned}$$

Then, by Proposition 2,

$$\min_{x\in Q_j} Pr(\mathcal{F}(x) + w \in Q_\ell) \geq \prod_{i=1}^{n} \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x_i - s_{i,min}^{j\to\ell})\, dx_i \qquad (4.30)$$

$$\max_{x\in Q_j} Pr(\mathcal{F}(x) + w \in Q_\ell) \leq \prod_{i=1}^{n} \int_{a_i^\ell}^{b_i^\ell} f_{w_i}(x_i - s_{i,max}^{j\to\ell})\, dx_i\ , \qquad (4.31)$$

so that (4.22)–(4.24) implies (3.13)–(3.14). Furthermore, (4.30)–(4.31) implies $\check{T}(Q_j, Q_\ell) \leq \widehat{T}(Q_j, Q_\ell)$ and (3.13)–(3.14) implies (3.12) so that $\mathcal{I} = (P, \check{T}, \widehat{T})$ is a valid IMC, concluding the proof. $\square$

Given a system of the form (4.1) satisfying Assumptions 1 to 2, and a rectangular partition P of its domain D, Theorem 1 shows that an IMC abstraction of (4.1) can be computed efficiently. Specifically, for any state in P, we establish an over-approximation of its one-step reachable set by evaluating the system's decomposition function at only two points. Likewise, finding the maximizing and minimizing shifts inside the reachable sets decouples along each coordinate and involves a number of operations and conditional statements that is linear in the dimension n of the state-space, according to (4.26) and (4.27). Finally, we see in (4.22) and (4.24) that n integral evaluations are needed per transition bound. Presuming the cumulative distribution function F_{w_i} for each w_i is available to us, this last step amounts to $2n$ function evaluations per bound. The practical implications of Theorem 1 are implemented in Algorithm 1.

Algorithm 1 Computation of an IMC abstraction for a rectangular partition P

```
 1: Input: Partition $P = \{Q_j\}_{j=1}^m$, probability density functions $f_{w_i}$ and modes $c_i \in \mathbb{R}$ for
    each component of disturbance, cumulative distribution functions $F_{w_i}$ of $f_{w_i}$, system
    decomposition function $g$
 2: Output: IMC abstraction $\mathcal{I}$ of (4.1)
 3:
 4: for $j = 1,\ 2, \dots,\ n$ do
 5:     Set $\widehat{r}^j = g(b^j, a^j)$ and $\widecheck{r}^j = g(a^j, b^j)$
 6:     for $\ell = 1,\ 2, \dots,\ n$ do
 7:         for $i = 1,\ 2, \dots,\ n$ do
 8:             Compute $s_{i,max}^{\ell}$ according to (4.19)
 9:             Compute $s_{i,max}^{j\to\ell}$ and $s_{i,min}^{j\to\ell}$ according to (4.26) and (4.27)
10:         end for
11:         Compute $\widehat{T}_{Q_j \to Q_\ell}$ and $\widecheck{T}_{Q_j \to Q_\ell}$ according to (4.22) and (4.24)
12:     end for
13: end for
14:
15: return $\mathcal{I} = (P, \widecheck{T}, \widehat{T})$
```

In Theorem 1, we exploited the crucial facts that each component of the random disturbance w of system (4.1) was unimodal and symmetric in order to efficiently construct an IMC approximation. Unfortunately, real-world systems rarely encounter disturbances displaying these two properties. In such instances, one could resort to purely numerical techniques to generate an IMC. We instead develop an alternate solution by approximating the original distribution with another one which is unimodal and symmetric. Then, the tools previously derived can be utilized on the approximation distribution. Now, we introduce a method for generating an IMC abstraction of the original system: we first compute an IMC using the approximation distribution, and then adjust its transition bounds appropriately.

To that end, consider random disturbance $w \in \mathbb{R}^n$ of (4.1) and suppose the collection $\{w_i\}_{i=1}^n$ remains mutually independent, but we no longer assume that each w_i is unimodal and symmetric, *i.e.*, the second part of Assumption 2 no longer holds. However, we assume that each w_i is reasonably approximated by a unimodal and symmetric distribution. To this end, many metrics exist to quantify the similarity between two probability distributions. Here, the maximum absolute difference between the original distribution w and its approximation v is our metric of choice, and we replace Assumption 2 with the following assumption.

Assumption 4. *There exists a mutually independent collection of random variables $\{v_i\}_{i=1}^n$ and constants $\{\delta_i\}_{i=1}^n$ such that $v_i \in V_i$, the probability density function f_{v_i} for each v_i is unimodal and symmetric with mode $\tilde{c}_i$, and*

$$\delta_i \geq \max_{x_i \in \mathbb{R}} \left| f_{v_i}(x_i) - f_{w_i}(x_i) \right| . \tag{4.32}$$

In Assumption 4, recall that f_{w_i} is the probability density function of $w_i \in W_i$, the i-th component of the random disturbance w.

The main result of this section, Theorem 2 below, states that we are able to determine an upper and a lower bound on the probabilities of transition between any two states in system (4.1) subject to disturbance w through an efficient computation of the bounds assuming instead that the system is subject to the random disturbance v.

Theorem 2. *Consider system* (4.1) *under Assumptions 1 and 4, and let $P = \{Q_j\}_{j=1}^m$ be a rectangular partition of D with each $Q_j = \{x : a^j \leq x \leq b^j\}$ for some $a^j, b^j \in \mathbb{R}^n$ satisfying $a^j \leq b^j$. For all $Q_j, Q_\ell \in P$, let*

$$\tilde{s}^{\ell}_{i,max} = \frac{a_i^\ell + b_i^\ell}{2} - \tilde{c}_i \tag{4.33}$$

and let $\widecheck{r}^j = g(a^j, b^j)$ and $\widehat{r}^j = g(b^j, a^j)$. Define

$$\widehat{T}^*_{Q_j \to Q_\ell} = \prod_{i=1}^n \left(\int_{a_i^\ell}^{b_i^\ell} f_{v_i}(x_i - \tilde{s}^{j \to \ell}_{i,max})\, dx_i + \delta_i (b_i^\ell - a_i^\ell) \right) , \tag{4.34}$$

$$\widecheck{T}^*_{Q_j \to Q_\ell} = \prod_{i=1}^n \left(\int_{a_i^\ell}^{b_i^\ell} f_{v_i}(x_i - \tilde{s}^{j \to \ell}_{i,min})\, dx_i - \delta_i (b_i^\ell - a_i^\ell) \right) \tag{4.35}$$

where

$$\tilde{s}^{j\to\ell}_{i,max} = \begin{cases} \tilde{s}^{\ell}_{i,max}, & \textit{if } \tilde{s}^{\ell}_{i,max} \in [\check{r}^j_i, \hat{r}^j_i] \\ \hat{r}^j_i, & \textit{if } \tilde{s}^{\ell}_{i,max} > \hat{r}^j_i \\ \check{r}^j_i, & \textit{if } \tilde{s}^{\ell}_{i,max} < \check{r}^j_i\ , \end{cases} \tag{4.36}$$

$$\tilde{s}^{j\to\ell}_{i,min} = \begin{cases} \check{r}^j_i, & \textit{if } \tilde{s}^{\ell}_{i,max} > \frac{\check{r}^j_i + \hat{r}^j_i}{2} \\ \hat{r}^j_i, & \textit{otherwise}\ . \end{cases} \tag{4.37}$$

Then $\mathcal{I} = (P, \check{T}^*, \hat{T}^*)$ *is an IMC abstraction of* (4.1).

Although similar to Theorem 1, Theorem 2 relaxes Assumption 2 and considers an arbitrary disturbance w to system (4.1). It assumes the existence of a random disturbance v that is unimodal, symmetric and characterized by its maximum absolute difference with w as stated in Assumption 4. This allows us to efficiently compute an IMC abstraction for (4.1) by applying the equations in Theorem 1 to disturbance v with the addition of an error term in the bounds (4.34) and (4.35). The error terms solely involve the multiplication of two known quantities and do not significantly affect the complexity of computing the IMC as compared to Theorem 1. However, it should be noted that the conservatism of an IMC generated from Theorem 2 strongly depends on the δ_i parameters. The latter are scaled by the size of the destination states and added to the IMC bounds, meaning that a large maximum absolute distance between w and v can only be compensated by a reduction of the states' size and an increase in the number of states in the partition. Therefore, excessively large δ_i are susceptible to cause a curse of dimensionality. As such, even though this method can be applied to arbitrary disturbances, it is most practical if w originally displays a probability density profile that is almost symmetric and unimodal.

The proof of Theorem 2 requires the following lemma in which we first restrict ourselves to a one-dimensional framework.

Lemma 2. *Let $\omega \in \Omega \subset \mathbb{R}$ with Ω an interval be a random variable with probability density function $f_\omega : \mathbb{R} \to \mathbb{R}$. Let $\nu \in \Omega \subset \mathbb{R}$ be another random variable with symmetric and unimodal probability density function $f_\nu : \mathbb{R} \to \mathbb{R}$ with mode $\tilde{c}$, and let δ satisfy $\delta \geq \max_{x\in\mathbb{R}} |f_\omega(x) - f_\nu(x)|$. For any $a, b \in \mathbb{R}$ satisfying $a \leq b$ and any $r_1, r_2 \in \mathbb{R}$ satisfying $r_1 \leq r_2$,*

$$\begin{aligned} &\max_{s\in[r_1,r_2]} \int_a^b f_\omega(x-s)\,dx \\ &\qquad \leq \max_{s\in[r_1,r_2]} \int_a^b f_\nu(x-s)\,dx + \delta(b-a) \qquad (4.38) \\ &\min_{s\in[r_1,r_2]} \int_a^b f_\omega(x-s)\,dx \\ &\qquad \geq \min_{s\in[r_1,r_2]} \int_a^b f_\nu(x-s)\,dx - \delta(b-a)\;. \qquad (4.39) \end{aligned}$$

Proof. Let s^ω_{max} and s^ω_{min} satisfy

$$\int_a^b f_\omega(x - s^\omega_{max})\,dx = \max_{s\in[r_1,r_2]} \int_a^b f_\omega(x-s)\,dx\;, \qquad (4.40)$$

$$\int_a^b f_\omega(x - s^\omega_{min})\,dx = \min_{s\in[r_1,r_2]} \int_a^b f_\omega(x-s)\,dx\;. \qquad (4.41)$$

$$(4.42)$$

Since $|f_\omega(x) - f_\nu(x)| \leq \delta\ \forall x$, we have

$$f_\omega(x - s^\omega_{max}) \leq f_\nu(x - s^\omega_{max}) + \delta\ \ \forall x$$

and it follows that

$$\begin{aligned}
&\int_a^b f_\omega(x - s^\omega_{max})\,dx \\
&\qquad \le \int_a^b f_\nu(x - s^\omega_{max})\,dx + \delta(b-a) \\
&\qquad \le \max_{s\in[r_1,r_2]} \int_a^b f_\nu(x-s)\,dx + \delta(b-a)\ .
\end{aligned}$$

Similarly,

$$f_\omega(x - s^\omega_{min}) \ge f_\nu(x - s^\omega_{min}) - \delta \ \ \forall x$$

so that

$$\int_a^b f_\omega(x - s^\omega_{min})\,dx \tag{4.43}$$

$$\ge \int_a^b f_\nu(x - s^\omega_{min})\,dx - \delta(b-a) \tag{4.44}$$

$$\ge \min_{s\in[r_1,r_2]} \int_a^b f_\nu(x-s)\,dx - \delta(b-a)\ . \tag{4.45}$$

$\square$

Lemma 2 is the enabling step in the proof Theorem 2.

Proof. For all $Q_j, Q_\ell \in P$ and $i = 1, \ldots, n$, by Lemma 1 and Lemma 2,

$$\min_{z_i \in [g_i(a^1,b^1), g_i(b^1,a^1)]} \int_{a_i^2}^{b_i^2} f_{w_i}(x - z_i)dx \geq \int_{a_i^\ell}^{b_i^\ell} f_{v_i}(x_i - \tilde{s}_{i,min}^{j \to \ell})\, dx_i - \delta_i(b_i^\ell - a_i^\ell)\ , \tag{4.46}$$

$$\min_{z_i \in [g_i(a^1,b^1), g_i(b^1,a^1)]} \int_{a_i^2}^{b_i^2} f_{w_i}(x - z_i)dx \leq \int_{a_i^\ell}^{b_i^\ell} f_{v_i}(x_i - \tilde{s}_{i,max}^{j \to \ell})\, dx_i + \delta_i(b_i^\ell - a_i^\ell)\ . \tag{4.47}$$

The theorem then follows from Proposition 2 by following the same argument as in the proof of Theorem 1 □

4.2 Abstraction of Polynomial Systems

We now present an IMC abstraction method for polynomial stochastic systems. These systems are relevant to numerous applications in a wide array of fields such as biology [49], physics [50], kinematics [51] and geology [52]. Furthermore, many nonlinear systems with intricate dynamics can be approximated by polynomial state updates [53] [54], which renders the study of polynomial systems especially attractive. Our proposed solution utilizes so-called stochastic barrier functions to compute bounds on the probability that a polynomial system transitions from one region of the state-space to another in one time step. Stochastic barrier functions are versatile tools that do not require explicit computations of reachable sets and serve as Lyapunov-like probabilistic certificates of forward set invariance. We show further that polynomial dynamics enable the search for stochastic barrier functions to be converted to Sum-of-Squares (SOS) optimization programs, which are known to be convex. Finding stochastic barrier functions for every possible transition between the discrete states of a domain partition effectively constructs a non-trivial IMC abstraction of a stochastic polynomial system.

Consider the stochastic system with dynamics

$$x[k+1] = \mathcal{F}(x[k], w[k]) \; , \tag{4.48}$$

where $x[k] \in D \subset \mathbb{R}^n$ is the state of the system on a domain D at time k, $\mathcal{F} : D \to D$ is a function and $w[k] \in W \subseteq \mathbb{R}^m$ is a random disturbance. In this section, we focus on systems with polynomial dynamics in x and w.

Assumption 5. *The function $\mathcal{F}$ in* (4.48) *is a polynomial in x and w.*

Hence, the main problem of this section consists in devising an IMC abstraction procedure for (4.48) under Assumption 5.

4.2.1 Stochastic Barrier Functions

In order to propose a solution to this problem, we first introduce the concept of stochastic barrier function.

Stochastic barrier functions are utilized as a probabilistic certificate of set invariance for stochastic systems. Specifically, by showing the existence of a non-negative function satisfying a particular set of constraints over the domain of the system, one can ensure that the probability of reaching a given set of states from a set of initial conditions is no greater than some bound. The following general theorem is a corollary of [55, Chapter 3, Theorem 3] and quantitatively demonstrates how stochastic barrier functions provide a bound on the probability of set invariance for discrete-time processes over a finite-time horizon.

Theorem 3. *Given the stochastic difference equation* (4.48) *and the sets $\mathcal{X} \subset \mathbb{R}^n$, $\mathcal{X}_u \subseteq \mathcal{X}, \mathcal{X}_0 \subseteq \mathcal{X} \setminus \mathcal{X}_u$. Consider the process $x[k]$ evolving according to* (4.48)*. Suppose there* *exists a function B such that*

$$B(x) \leq \gamma \; \forall x \in \mathcal{X}_0 \tag{4.49}$$

$$B(x) \geq 1 \; \forall x \in \mathcal{X}_u \tag{4.50}$$

$$B(x) \geq 0 \; \forall x \in \mathcal{X} \tag{4.51}$$

$$\mathbb{E}_w[B(F(x,w)) \mid x] \leq \frac{B(x)}{\alpha} + \beta \quad \forall x \in \mathcal{X} \setminus \mathcal{X}_u \tag{4.52}$$

for some $\alpha \geq 1$, $0 \leq \beta < 1$ and $\gamma \in [0,1)$, where $\mathbb{E}_w$ denotes the expectation with respect to random variable w. Define

$$\rho_u := Pr\{x[k] \in \mathcal{X}_u \text{ for } 0 \leq k \leq N \mid x[0] \in \mathcal{X}_0\} \;, \tag{4.53}$$

$$\rho_B := Pr\left\{ \sup_{0 \leq k \leq N} B(x) \geq 1 \mid x[0] \in \mathcal{X}_0 \right\} \;. \tag{4.54}$$

Then

- *If $\alpha > 1$ and $\frac{\beta\alpha}{\alpha-1} \leq 1$,*

$$\rho_u \leq \rho_B \leq 1 - \left(1 - \gamma\right) \prod_{0}^{N-1} \left(1 - \beta\right) \;. \tag{4.55}$$

- *If $\alpha > 1$ and $\frac{\beta\alpha}{\alpha-1} > 1$,*

$$\rho_u \leq \rho_B \leq \gamma\alpha^{-N} + \frac{(1-\alpha^{-N})\alpha\beta}{(\alpha - 1)} \;. \tag{4.56}$$

- *If $\alpha = 1$,*

$$\rho_u \leq \rho_B \leq \gamma + \beta N \;. \tag{4.57}$$

If a function $B(x)$ satisfies the conditions of Theorem 3, then $B(x)$ is called a stochastic barrier function.

While this theorem applies to stochastic systems over an arbitrary finite time-horizon, one-step transition guarantees are sufficient for creating an IMC abstraction of (4.48). In (4.52), the condition that the expectation of B evaluated against the stochastic process (4.48) is less than a function of B itself captures the fact that different regions of the safe set $\mathcal{X} \setminus \mathcal{X}_u$ have different probabilities of reaching the unsafe set $\mathcal{X}_u$, which is factored in the bounds on the probability of reaching $\mathcal{X}_u$ over multiple transitions from some initial condition. When focusing on a single transition, such considerations are irrelevant as the transition probabilities for the whole initial set $\mathcal{X}_0$ does not influence the probability of attaining the unsafe set in one transition from some particular x_0.

In light of these facts, we can make some simplifications on the conditions and bounds of Theorem 3 which are amenable to more efficient implementations in the following subsection. We now study stochastic barrier functions over a finite-time horizon of two time steps, i.e., the current time and the next time. Let $\mathcal{X}_0 \subseteq \mathcal{X}$ be a set of possible initial conditions, and $\mathcal{X}_1 \subseteq \mathcal{X}$ be a compact set of the domain. The following theorem establishes that the probability of reaching set $\mathcal{X}_1$ from any initial state $x_0 \in \mathcal{X}_0$ in one time step can be upper-bounded by finding a function $B(x)$ whose expectation against the stochastic dynamics is upper-bounded by a constant over the set $\mathcal{X}_0$.

Theorem 4. *Given the stochastic differential equation in* (4.48) *and the sets $\mathcal{X} \subset \mathbb{R}^n$, $\mathcal{X}_0 \subseteq \mathcal{X}, \mathcal{X}_1 \subseteq \mathcal{X}$. Consider the process $x[k]$ evolving according to* (4.48)*. Suppose that there exists a function $B : \mathcal{X} \to \mathbb{R}$, such that*

$$B(x) \geq 1 \; \forall x \in \mathcal{X}_1 \; , \tag{4.58}$$

$$B(x) \geq 0 \; \forall x \in \mathcal{X} \; , \tag{4.59}$$

$$\mathbb{E}_w \Big[B(\mathcal{F}(x, w)) \mid x \Big] \leq \alpha \quad \forall x \in \mathcal{X}_0 \tag{4.60}$$

for some $\alpha \geq 0$, where $\mathbb{E}_w$ denotes the expectation with respect to w. Given $x_0 := x[0]$, *define* $\rho_u(x_0) := Pr\{x[1] \in \mathcal{X}_1 \mid x_0\}$*. Then, for any initial state* $x_0 \in \mathcal{X}_0$*,*

$$\rho_u(x_0) \leq Pr\left\{B(x[1]) \geq 1 \mid x_0\right\} \leq \alpha\ . \tag{4.61}$$

Proof. This theorem follows from Markov's inequality:

$$\rho_u(x_0) \leq Pr\left\{B(x[1]) \geq 1 \mid x_0\right\} \leq \frac{\mathbb{E}_w\Big[B(\mathcal{F}(x_0, w)) \mid x_0\Big]}{1} \leq \alpha\ .$$

□

A function B satisfying the conditions of Theorem 4 is a special-case of stochastic barrier function over the set of initial conditions $\mathcal{X}_0$ for a single transition. Numerical procedures for finding such functions for particular polynomial systems under Assumption 5 are developed in the next subsections.

4.2.2 Barrier Function-based IMC Abstraction

Now, we present a stochastic barrier function-based approach to the IMC abstraction problem for general systems of the form (4.48).

We first define the exact transition bounds on the probability of transition between any two states in a domain partition P.

Definition 12 (Exact Transition Bounds)**.** *Let P be a partition of the domain D of* (4.48)*. For all $Q_i, Q_j \in P$, the* exact *transition lower bound $\check{T}_{ex}(Q_i, Q_j)$ and upper bound $\widehat{T}_{ex}(Q_i, Q_j)$ on the transition from Q_i to Q_j are given by*

$$\check{T}_{ex}(Q_i, Q_j) = \inf_{x \in Q_i} T(x, Q_j)\ ,$$

$$\widehat{T}_{ex}(Q_i, Q_j) = \sup_{x \in Q_i} T(x, Q_j)\ ,$$

where T denotes the stochastic transition kernel of system (4.48).

Remark that, in an IMC abstraction of (4.48), it must hold that, for all $Q_i, Q_j \in P$,

$$\check{T}_{Q_i \to Q_j} \leq \check{T}_{ex}(Q_i, Q_j) \leq \widehat{T}_{ex}(Q_i, Q_j) \leq \widehat{T}_{Q_i \to Q_j} \; . \tag{4.62}$$

Consider two states Q_i and Q_j from a partition P of the domain D of (4.48). Our goal is to determine a lower bound $\check{T}(Q_i, Q_j)$ and an upper bound $\widehat{T}(Q_i, Q_j)$ on the probability of making a transition from any continuous state in Q_i to a state in Q_j. Then, an IMC abstraction of (4.48) can be constructed by applying this methodology to all possible pairs of states in P.

We assume henceforth that an over-approximation and an under-approximation of any discrete state in P can be represented as the zero-superlevel set of some polynomial function, as well as the domain D itself. This assumption is reasonable in many examples found in the literature where the discrete states arise from polytopic or rectangular partitions of the domain. For instance, a technique for computing such approximations with arbitrary precision for hyperrectangular states is presented in [15].

Assumption 6. *For any state Q_i in a partition P of domain D, there exists an over-approximation $\widehat{\mathcal{X}}_{Q_i} \supset Q_i$ and an under-approximation $\check{\mathcal{X}}_{Q_i} \subset Q_i$ such that $\widehat{\mathcal{X}}_{Q_i} = \{x \in \mathbb{R}^n \mid s_{\widehat{\mathcal{X}}_{Q_i}}(x) \geq 0\}$ and $\check{\mathcal{X}}_{Q_i} = \{x \in \mathbb{R}^n \mid s_{\check{\mathcal{X}}_{Q_i}}(x) \geq 0\}$, where $s_{\widehat{\mathcal{X}}_{Q_i}}$ and $s_{\check{\mathcal{X}}_{Q_i}}$ are polynomials. Also, there exists an over-approximation $\widehat{\mathcal{X}} \supset D$ of D such that $\widehat{\mathcal{X}} = \{x \in \mathbb{R}^n \mid s_{\widehat{\mathcal{X}}}(x) \geq 0\}$, where $s_{\widehat{\mathcal{X}}}$ is a polynomial.*

Finding bounds on the probability of making a transition from Q_i to Q_j in one time step can be converted to two reachability problems over a one time step time horizon. Indeed, by viewing Q_i as the set $\mathcal{X}_0$ in Theorem 4, determining upper bounds on the probability of reaching $\mathcal{X}_1 = Q_j$ and $\mathcal{X}_1 = D \setminus Q_j$ induces an interval on the probability of making a transition from Q_i to Q_j. We formalize this in terms of the over and under-approximation representations of these states in the following lemma.

Lemma 3. *Let $\mathcal{X}_0$ and $\mathcal{X}_1$ be the sets as defined in Theorem 4. Recall the exact bounds on the probability of transition from Q_i to Q_j are $\check{T}_{ex}(Q_i, Q_j)$ and $\widehat{T}_{ex}(Q_i, Q_j)$, where Q_i and Q_j are states in the partition P. Let $\widehat{\rho}_u$ be an upper bound on the probability for system* (4.48) *to reach $\mathcal{X}_1$ when $\mathcal{X}_0 = \widehat{\mathcal{X}}_{Q_i}$ and $\mathcal{X}_1 = \widehat{\mathcal{X}}_{Q_j}$, and let $\check{\rho}_u$ be a similarly defined upper bound when $\mathcal{X}_0 = \widehat{\mathcal{X}}_{Q_i}$ and $\mathcal{X}_1 = D \setminus \check{\mathcal{X}}_{Q_j}$. Then,*

$$\widehat{\rho}_u \geq \widehat{T}_{ex}(Q_i, Q_j) \ , \tag{4.63}$$

$$1 - \check{\rho}_u \leq \check{T}_{ex}(Q_i, Q_j) \ . \tag{4.64}$$

Proof. By assumption, the probability of making a transition to $\widehat{\mathcal{X}}_{Q_j}$ from any state $x \in \widehat{\mathcal{X}}_{Q_i}$ in one time step is upper bounded by $\widehat{\rho}_u$. Since $Q_j \subset \widehat{\mathcal{X}}_{Q_j}$, the probability of making a transition from $\widehat{\mathcal{X}}_{Q_i}$ to Q_j cannot be greater than $\widehat{\rho}_u$ as well. As $Q_i \subset \widehat{\mathcal{X}}_{Q_i}$, the latter also holds true for all $x \in Q_i$, proving (4.63).
Furthermore, the probability of making a transition to $D \setminus \check{\mathcal{X}}_{Q_j}$ from any state $x \in \widehat{\mathcal{X}}_{Q_i}$ in one time step is upper bounded by $\check{\rho}_u$. Therefore, the probability of making a transition to $\check{\mathcal{X}}_{Q_j}$ is at least $1 - \check{\rho}_u$. Since $\check{\mathcal{X}}_{Q_j} \subset Q_j$, the probability of making a transition to Q_j from $\widehat{\mathcal{X}}_{Q_i}$ cannot be less than $1 - \check{\rho}_u$ as well. As $Q_i \subset \widehat{\mathcal{X}}_{Q_i}$, the latter also holds true for all $x \in Q_i$, proving (4.64). □

In the next subsection, we describe a numerical procedure for computing polynomial barrier functions fulfilling the requirements of Theorem 4 for systems satisfying Assumption 5.

4.2.3 Numerical Procedure for Barrier Function Computation

This section proposes a numerical algorithm based on the equations in Theorem 4 for computing the bounds discussed in Lemma 3. As an example, in this subsection, we assume the dynamics of the system under consideration to satisfy Assumption 5. As we wish to find transition bounds that are as tight as possible, we seek to formulate an optimization problem that minimizes the computed upper bound probability of system (4.48) making a transition to a set $\mathcal{X}_1$ in one time step as established in Theorem 4. Specifically, for a given initial set $\mathcal{X}_0$ and a set $\mathcal{X}_1$, we are interested in finding the minimum upper bound α on ρ_u such that a barrier function B satisfying conditions (4.58) – (4.60) exists.

Imposing the restriction that B is a polynomial function, this problem is converted to a *Sum-of-Squares Program* (SOSP) as defined below.

Definition 13 (Sum-of-Squares Polynomial)**.** *Define* $\mathbb{R}[x]$ *as the set of all polynomials in* $x \in \mathbb{R}^n$*. Then*

$$\Sigma[x] := \left\{ s(x) \in \mathbb{R}[x] : s(x) = \sum_{i=1}^{m} g_i(x)^2, g_i(x) \in \mathbb{R}[x] \right\}$$

is the set of Sum-of-Squares polynomials*. It is noted that if* $s(x) \in \Sigma[x]$*, then* $s(x) \geq 0 \; \forall \; x$.

Definition 14 (Sum-of-Squares Program)**.** *Given* $p_i(x) \in \mathbb{R}[x]$ *for* $i = 0, \ldots, m$*, the problem of finding* $q_i(x) \in \Sigma[x]$ *for* $i = 1, \ldots, \hat{m}$ *and* $q_i(x) \in \mathbb{R}[x]$ *for* $i = \hat{m}+1, \ldots, m$ *such that*

$$p_0(x) + \sum_{i=1}^{m} p_i(x) q_i(x) \in \Sigma[x]$$

is a Sum-of-Squares Program *(SOSP). SOSPs can be efficiently converted to semidefinite programs, which are convex, using tools such as SOSTOOLS [56].*

Finding an SOS polynomial barrier functions B fulfilling constraints (4.58) – (4.60) over the sets $\mathcal{X}_0$ and $\mathcal{X}_1$ can be encoded as an SOSP. Assume $\mathcal{X}_0 = \{x \in \mathbb{R}^n \mid s_{\mathcal{X}_0}(x) \geq 0\}$, $\mathcal{X}_1 = \{x \in \mathbb{R}^n \mid s_{\mathcal{X}_1}(x) \geq 0\}$ and $\mathcal{X} = \{x \in \mathbb{R}^n \mid s_{\mathcal{X}}(x) \geq 0\}$, where $s_{\mathcal{X}_0}$, $s_{\mathcal{X}_1}$ and $s_{\mathcal{X}}$ are polynomials. The SOSP $S(s_{\mathcal{X}_0}, s_{\mathcal{X}_1}, s_{\mathcal{X}})$ in Algorithm 2 finds an upper bound on the probability of making a transition from $\mathcal{X}_0$ to $\mathcal{X}_1$ in one time step by setting α to be the objective function to minimize. Note that the optimization program resulting from this barrier function formulation is convex and therefore amenable to efficient resolution.

Algorithm 2 Upper bounding SOSP $\mathcal{S}(s_{\mathcal{X}_0}, s_{\mathcal{X}_1}, s_{\mathcal{X}})$

```
1: Input: Polynomial representations $s_{\mathcal{X}_0}, s_{\mathcal{X}_1}, s_{\mathcal{X}}$ of regions $\mathcal{X}_0$, $\mathcal{X}_1$ and domain $D$
2: Output: Upper bound $\alpha^*$ on the probability of making a transition from $\mathcal{X}_0$ to $\mathcal{X}_1$ in
   one time step
3: Solve
```

$$\alpha^* = \min_{\substack{\alpha,\, B(x), \\ \lambda_{\mathcal{X}}(x),\, \lambda_{\mathcal{X}_0}(x),\, \lambda_{\mathcal{X}_1}(x)}} \alpha$$

subject to

$$\begin{aligned} B(x) - \lambda_{\mathcal{X}}(x) s_{\mathcal{X}}(x) &\in \Sigma[x] \\ B(x) - \lambda_{\mathcal{X}_1}(x) s_{\mathcal{X}_1}(x) - 1 &\in \Sigma[x] \\ -\mathbb{E}_w[B(\mathcal{F}(x,w)) \mid x] + \alpha - \lambda_{\mathcal{X}_0}(x) s_{\mathcal{X}_0}(x) &\in \Sigma[x] \\ \lambda_{\mathcal{X}}(x),\ \lambda_{\mathcal{X}_0}(x),\ \lambda_{\mathcal{X}_1}(x) &\in \Sigma[x] \end{aligned}$$

```
4: return $\alpha^*$
```

The constraints of the SOSPs are derived from the *Positivstellensatz condition* for converting constraints on sets to SOSPs as detailed in [56]. The expectation term in the SOSP is computed by expanding $B(\mathcal{F}(x, w))$ and determining the moments of the noise terms, which can be accomplished efficiently for certain classes of disturbance, such as Gaussian or Poisson random variables, and results in a polynomial in x when $\mathcal{F}$ is a polynomial. An important hyperparameter of this algorithm is the degree of the barrier and λ polynomials. Searching for high degree polynomials allows to find tighter bounds, at a cost of increased computational complexity.

According to Lemma 3, an upper and lower bound on the probability of transition between any two states in a partition P of the domain D can be found using function $\mathcal{S}$. Algorithm 3 summarizes the IMC abstraction procedure for system (4.48) with a given domain partition P.

Theorem 5. *Given a system of the form* (4.48) *and partition P of its domain D, an IMC abstraction of* (4.48) *is computed via Algorithm 3.*

Proof. For any states Q_i and Q_j of a partition P of the domain D of (4.48), Algorithm 3 computes an upper bound and a lower bound on the probability of making a transition from any continuous state in Q_i to Q_j in line 6 to 10, from Lemma 3. Moreover, Algorithm 3

Algorithm 3 Barrier function-based IMC Abstraction

```
1:  Input: Domain $D$, Domain partition $P$
2:  Output: IMC Abstraction $\mathcal{I}$
3:  Compute an over-approximation representation $s_{\widehat{\mathcal{X}}}$ of the domain $D$
4:  for $Q_i \in P$ do
5:      for $Q_j \in P$ do
6:          Compute the over and under-approximation representations $s_{\widehat{\mathcal{X}}_{Q_i}}$, $s_{\widehat{\mathcal{X}}_{Q_j}}$ and $s_{D\backslash\widecheck{\mathcal{X}}_{Q_j}}$
7:          $\widehat{\rho} := \mathcal{S}(s_{\widehat{\mathcal{X}}_{Q_i}}, s_{\widehat{\mathcal{X}}_{Q_j}}, s_{\widehat{\mathcal{X}}})$
8:          $\widecheck{\rho} := \mathcal{S}(s_{\widehat{\mathcal{X}}_{Q_i}}, s_{D\backslash\widecheck{\mathcal{X}}_{Q_j}}, s_{\widehat{\mathcal{X}}})$
9:          $\widehat{T}_{Q_i \to Q_j} := \widehat{\rho}$
10:         $\widecheck{T}_{Q_i \to Q_j} := 1 - \widecheck{\rho}$
11:     end for
12: end for
13: return $\mathcal{I} = (Q, \widecheck{T}, \widehat{T})$
```

applies this bounding procedure to every pair of states in P, proving the theorem. $\square$

In brief, we implement IMC abstraction techniques that ensure the correctness of the computed transition bounds for two classes of stochastic systems. For affine-in-disturbance mixed monotone systems, we show that these bounds are found by evaluating a decomposition function at only two points of the origin state and performing a number of integrations which grows linearly with the system dimensions for each transition. For polynomial systems, we cast the computation of each bound to an SOSP which is converted to a convex optimization problem as a semidefinite program.

The techniques developed in this chapter are naturally extended to switched stochastic systems with state update equation $x[k+1] = \mathcal{F}_a(x[k], w_a[k])$ with $a \in A$, where A is a finite set of modes. Indeed, assuming that this system is of the form (4.1) or (4.48) for all modes a, one can construct an IMC abstraction for each individual mode from the same partition P of the domain using the algorithms devised in Section 4.1 and Section 4.2. As per Definition 4, concatenating the resulting outgoing transition probability intervals for each state of the partition under all possible modes produces a BMDP abstraction of the switched stochastic system, where each action of the BMDP corresponds to a mode of the abstracted system.

CHAPTER 5
SPECIFICATION-GUIDED VERIFICATION AND ABSTRACTION REFINEMENT FOR STOCHASTIC SYSTEMS WITH OMEGA-REGULAR OBJECTIVES

The verification of stochastic systems against complex temporal tasks is generally addressed by constructing finite-state abstractions of the dynamics for which probabilistic guarantees are derived using formal analysis. In Chapter 4, we elaborated procedures for abstracting affine-in disturbance mixed monotone systems and polynomial systems as IMCs from a partition of their continuous domain. In this chapter, we leverage IMC abstractions to provide a solution to the stochastic verification problem.

In Section 5.1, we develop a technique for computing an interval on the probability of satisfying an arbitrary ω-regular property for any initial state of an IMC. We employ an automaton-based approach and build a Deterministic Rabin Automaton (DRA) representing the property of interest. A transition system satisfies an ω-regular property if it generates a trajectory which induces an accepting run in the corresponding DRA. For standard Markov Chains (MC), the probability of generating an accepting run in a DRA is computed by constructing the Cartesian product between the MC and the DRA, finding a special set of states known as accepting *Bottom Strongly Connected Components* (BSCC), and determining the probability of reaching an accepting BSCC from the initial states of the product construction. However, we show that the set of accepting BSCCs in the product between an IMC and a DRA is not fixed and depends on the assumed transition values for the intervals of the IMC, preventing us from applying the same machinery as for standard MCs. Instead, we prove that a product IMC generates a *largest winning component* and a *largest losing component*. A winning and a losing component in a MC are sets of states which respectively reach an accepting and a non-accepting BSCC with probability

1. We show that an interval on the probability of satisfying the desired property from any initial state of the IMC is found by computing the upper bound probability of reaching these largest components from the corresponding initial state of the product IMC. Solving these reachability problems effectively produces the worst and best-case instantiations of the probability intervals of the IMC with respect to the property at hand.

The intervals of satisfactions computed in an IMC abstraction directly apply to the continuous abstracted states. As explained in Chapter 3, because these guarantees are specified as intervals, it may not be possible to determine whether a probabilistic ω-regular specifications is satisfied or violated for some subsets of states. In Section 5.2, in order to decrease the total volume of these undecided states, we present a partition refinement algorithm whose goal is to reduce conservatism in the resulting abstraction. Because partition refinement can substantially enlarge the number of states in the IMC abstraction, we have to carefully select the states which are most likely to decrease the overall uncertainty under refinement so as to lessen the state-space explosion effect. To select the states to be refined, we adopt a *specification-guided* approach, that is, the regions selected for refinement are the ones causing the most uncertainty with respect to the particular specification at hand. Specifically, we implement a heuristical scoring procedure that quantitatively captures how much uncertainty is generated by each state in the IMC with respect to the uncertain states. This score is assigned by exploring and comparing the paths generated by the best and worst-case adversary of the IMC abstraction from the uncertain states, and the states with the greatest scores are chosen for refinement. The subsequent finer partition may be iteratively refined as well until the fractional volume of uncertain states reaches a user-defined threshold of precision.

Our refinement-based verification procedure is illustrated in Figure 5.1.

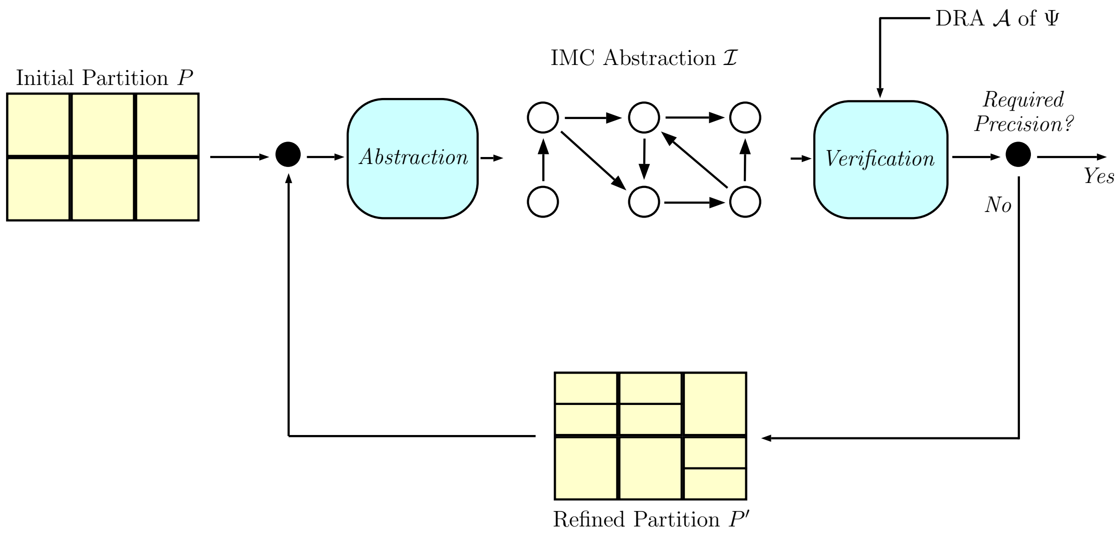


Figure 5.1: Depiction of our refinement-based verification procedure. First, an IMC abstraction $\mathcal{I}$ of the system is constructed from an initial partition P of the domain. Then, verification is performed on $\mathcal{I}$. If the computed probabilistic guarantees satisfy a user-defined precision criterion, the procedure terminates. Otherwise, the partition undergoes a refinement step where specific regions of the state-space are targeted, and a finer IMC abstraction is constructed and analyzed.

5.1 Verification of IMCs against ω-regular Specifications

Consider the stochastic system

$$x[k+1] = \mathcal{F}(x[k], w[k]) \tag{5.1}$$

evolving on a domain D and subject to a probabilistic ω-regular specification $\phi = \mathcal{P}_{\bowtie p_{sat}}[\Psi]$ defined in (3.8) in Section 3.3. Recall the objective of the probabilistic verification problem which is to find the set of initial states of (5.1) satisfying ϕ. To this end, we assume that an IMC abstraction $\mathcal{I}$ of (5.1) constructed from a partition P of D is available to us.

Now, for all initial states Q_j of $\mathcal{I}$, we require a lower bound and an upper bound on the probability of satisfying the ω-regular property Ψ to assess their satisfaction with respect to ϕ, and by extension the satisfaction of the continuous abstracted states of (5.1). We thus seek to compute the greatest lower bound $\widecheck{\mathcal{P}}_{\mathcal{I}}(Q_j \models \Psi)$ and least upper bound $\widehat{\mathcal{P}}_{\mathcal{I}}(Q_j \models \Psi)$

such that, for any adversary $\nu \in \nu_{\mathcal{I}}$,

$$\breve{\mathcal{P}}_{\mathcal{I}}(Q_j \models \Psi) \leq \mathcal{P}_{\mathcal{I}[\nu]}(Q_j \models \Psi) \leq \widehat{\mathcal{P}}_{\mathcal{I}}(Q_j \models \Psi) \; . \tag{5.2}$$

Our approach draws from the verification of regular Markov Chains (MC) against ω-regular properties using automata-based methods [40, Section 10.3].

Definition 15 (Markov Chain)**.** *A* Markov Chain *(MC)* $\mathcal{M} = (Q, T, q_0, \Pi, L)$ *is defined similarly to an IMC with the difference that the transition probability function or transition matrix* $T : Q \times Q \to [0, 1]$ *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 a temporal specification Ψ from initial state Q_j in a Markov Chain $\mathcal{M}$ is denoted by $\mathcal{P}_{\mathcal{M}}(Q_j \models \Psi)$.

First, we generate a *Deterministic Rabin Automaton* (DRA) $\mathcal{A}$ that recognizes the language induced by property Ψ.

Definition 16 (Deterministic Rabin Automaton)**.** *A* Deterministic Rabin Automaton (DRA) *[40] 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* $E_i, F_i \subset S$*, 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 it contains an infinite number of states belonging to F_i and a finite number of states in E_i 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 $\mathcal{A}$.

Note that a property is ω-regular if and only if it is accepted by a DRA. Therefore, a DRA representation always exists for any ω-regular property. Several algorithms can efficiently generate a DRA for a large subset of ω-regular expressions [57] [58]. Then, we construct the Cartesian product between the IMC abstraction $\mathcal{I}$ and the DRA $\mathcal{A}$ encoding the specification of interest Ψ. This product IMC is denoted by $\mathcal{I} \otimes \mathcal{A}$.

Definition 17 (Product Interval-valued Markov Chain)**.** *Let* $\mathcal{I} = (Q, \check{T}, \widehat{T}, q_0, \Pi, L)$ *be an Interval-valued Markov Chain and* $\mathcal{A} = (S, \Pi, \delta, s_0, Acc)$ *be a Deterministic Rabin Automaton. The* product $\mathcal{I} \otimes \mathcal{A} = (Q \times S, \check{T}', \widehat{T}', q_0^{\otimes}, Acc', L')$ *is an Interval-valued Markov Chain where:*

- $Q \times S$ *is a set of states,*
- $$\check{T}'_{\langle Q_j, s\rangle \rightarrow \langle Q_\ell, s'\rangle} = \begin{cases} \check{T}'_{Q_j \rightarrow Q_\ell}, & \textit{if } \ s' = \delta(s, L(Q_\ell)) \\ 0, & \textit{otherwise} \end{cases}$$
- $$\widehat{T}'_{\langle Q_j, s\rangle \rightarrow \langle Q_\ell, s'\rangle} = \begin{cases} \widehat{T}'_{Q_j \rightarrow Q_\ell}, & \textit{if } \ s' = \delta(s, L(Q_\ell)) \\ 0, & \textit{otherwise} \end{cases}$$
- $q_0^{\otimes} = \{(Q_j, s_0) : Q_j \in q_0\}$ *is a 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 $H \in L'(\langle Q_j, s\rangle)$ if and only if $s \in H$, for all $H \in Acc'$ and for all j.*

We further introduce the notion of *induced Markov Chain* which will prove key for our proposed solution.

Definition 18 (Induced Markov Chain)**.** *A Markov Chain $\mathcal{M} = (Q, T, q_0, \Pi, L)$ is said to be* induced *by IMC $\mathcal{I} = (Q, \check{T}, \widehat{T}, q_0, \Pi, L)$ if they share the same Q, q_0, Π and L, and for all $Q_j, Q_\ell \in Q$, $\check{T}(Q_j, Q_\ell) \leq T(Q_j, Q_\ell) \leq \widehat{T}(Q_j, Q_\ell)$. A transition matrix T satisfying this inequality is said to be induced by $\mathcal{I}$.*

A MC induced by $\mathcal{I} \otimes \mathcal{A}$ is called a *product Markov Chain*, and we use the notation $\mathcal{M}_{\otimes}^{\mathcal{A}}$ to denote such an induced MC. In general, we cannot view an induced MC as a product of a MC with $\mathcal{A}$, $\mathcal{M} \otimes \mathcal{A}$. Regular MCs are interpreted as memoryless adversaries of $\mathcal{I}$, while induced MCs represent a larger class of memory-dependent adversaries of $\mathcal{I}$.

It is known that the probability of satisfying Ψ from initial state Q_j in a MC equals that of reaching an accepting *Bottom Strongly Connected Component* (BSCC) from initial state $\langle Q_j, s_0\rangle$ in the product MC with $\mathcal{A}$ [40].

Definition 19 (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. Moreover, B is accepting when at least one of its states maps to the accepting set of a Rabin pair, while no state in B maps to the non-accepting set of that same pair as formalized next.

Definition 20 (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:*

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

Definition 21 (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* non-accepting *if it is not accepting.*

We denote by U^A and U^N the set of states that respectively belong to an accepting and a non-accepting BSCC in a product MC.

Note that each product MC $\mathcal{M}_{\otimes}^{\mathcal{A}}$ induced by $\mathcal{I} \otimes \mathcal{A}$ simulates the behavior of $\mathcal{I}$ under some adversary $\nu \in \nu_{\mathcal{I}}$. Indeed, for any two states Q_j and Q_ℓ in $\mathcal{I}$ and some states s, s', s'' and s''' in $\mathcal{A}$, we allow $T_{\langle Q_j, s \rangle \rightarrow \langle Q_\ell, s' \rangle}$ and $T_{\langle Q_j, s'' \rangle \rightarrow \langle Q_\ell, s''' \rangle}$ to assume different values in $\mathcal{M}_{\otimes}^{\mathcal{A}}$, which means that the transition probability between Q_j and Q_ℓ is permitted to change depending on the history of the path in $\mathcal{I}$ as encoded in the state of $\mathcal{A}$.

Also note that the adversary is history-independent or *memoryless* in the product automaton, that is, the adversary's chosen transition probability only depends on the current state of the IMC and the current state of the DRA $\mathcal{A}$. For reachability problems in IMCs, it was shown in [59] that memoryless adversaries yield the same bounds as the memory-dependent ones. The following facts establish that, therefore, such memoryless (in the product) adversaries are sufficient for IMC verification.

Fact 2. *[40, p. 792, Theorem 10.56] [59] We denote the set of adversaries of $\mathcal{I}$ that are* *induced by memoryless adversaries in the product IMC $\mathcal{I} \otimes \mathcal{A}$ by $(\nu_{\mathcal{I}})^{\mathcal{A}}_{\otimes} \subseteq \nu_{\mathcal{I}}$. It holds that*

$$\inf_{\nu \in \nu_{\mathcal{I}}} \mathcal{P}_{\mathcal{I}[\nu]}(Q_i \models \Psi) = \inf_{\nu \in (\nu_{\mathcal{I}})^{\mathcal{A}}_{\otimes}} \mathcal{P}_{\mathcal{I}[\nu]}(Q_i \models \Psi) \tag{5.3}$$

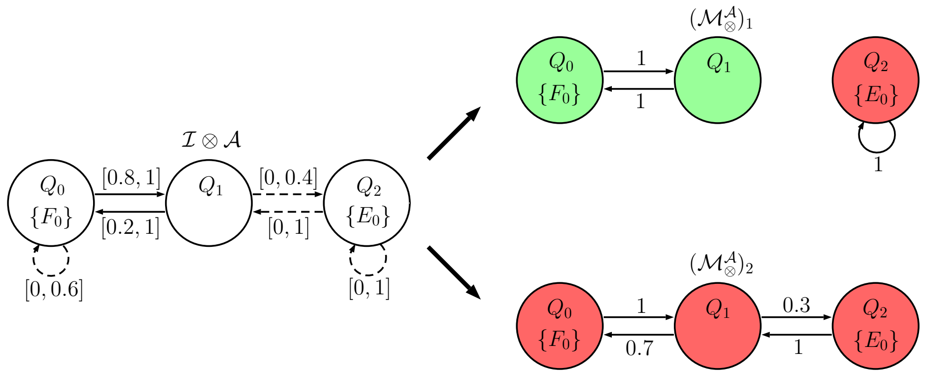
$$\sup_{\nu \in \nu_{\mathcal{I}}} \mathcal{P}_{\mathcal{I}[\nu]}(Q_i \models \Psi) = \sup_{\nu \in (\nu_{\mathcal{I}})^{\mathcal{A}}_{\otimes}} \mathcal{P}_{\mathcal{I}[\nu]}(Q_i \models \Psi) \, . \tag{5.4}$$

Fact 3. *For any adversary $\nu \in (\nu_{\mathcal{I}})^{\mathcal{A}}_{\otimes}$ in $\mathcal{I}$, it holds that $\mathcal{P}_{\mathcal{I}[\nu]}(Q_i \models \Psi)$ = $\mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_\nu}(\langle Q_i, s_0\rangle \models \Diamond U^A)$, where $(\mathcal{M}^{\mathcal{A}}_{\otimes})_\nu$ denotes the product MC induced by $\mathcal{I} \otimes \mathcal{A}$ corresponding to adversary ν.*

Consequently, computing $\check{\mathcal{P}}_{\mathcal{I}}(Q_i \models \Psi)$ and $\widehat{\mathcal{P}}_{\mathcal{I}}(Q_i \models \Psi)$ amounts to finding the product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ that respectively minimize and maximize the probability of reaching an accepting BSCC from $\langle Q_i, s_0\rangle$. Such reachability problems in IMCs were solved when the destination states are fixed for all induced MCs [26] [28].

However, in general, the sets U^A and U^N are not fixed in product IMCs and vary as a function of the assumed values for each transition. Specifically, U^A and U^N are determined by transitions that can be turned "on" or "off", i.e. those whose lower bound is zero and upper bound is non-zero, as seen in the example in Figure 5.2. In the product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_1$ induced by $\mathcal{I} \otimes \mathcal{A}$, the set U^A is $\{Q_0, Q_1\}$ while $\{Q_2\}$ is non-accepting. However, in $(\mathcal{M}^{\mathcal{A}}_{\otimes})_2$, another product MC induced by $\mathcal{I} \otimes \mathcal{A}$, all states are in U^N.

First, we show that a product IMC always induces a largest *Losing Component* and *Winning Component*. These components contain states that reach a BSCC with probability 1. Upper and lower bounds on Ψ are computed by solving a reachability problem for these sets. We further introduce the notion of *Permanent Losing Components* and *Permanent Winning Components*. These components are those which cannot be 'destroyed' for any product MC induced by a product IMC and play a crucial role in the refinement algorithm discussed in the next section. Second, we describe a graph search algorithm to find these components.



$\mathcal{I} \otimes \mathcal{A}$
Q_0
$\{F_0\}$
Q_1
Q_2
$\{E_0\}$
$[0.8, 1]$
$[0.2, 1]$
$[0, 0.4]$
$[0, 1]$
$[0, 0.6]$
$[0, 1]$
$(\mathcal{M}_{\otimes}^{\mathcal{A}})_1$
1
1
1
$(\mathcal{M}_{\otimes}^{\mathcal{A}})_2$
1
0.7
0.3
1

Figure 5.2: Two product MCs $(\mathcal{M}_{\otimes}^{\mathcal{A}})_1$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_2$ induced by a product IMC $\mathcal{I} \otimes \mathcal{A}$. Accepting BSCCs are shown in green; non-accepting BSCCs are red. In $(\mathcal{M}_{\otimes}^{\mathcal{A}})_1$, $U^A = \{Q_0, Q_1\}$ and $U^N = \{Q_2\}$; in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_2$, $U^A = \emptyset$ and $U^N = \{Q_0, Q_1, Q_2\}$.

5.1.1 Computation of Satisfiability Bounds in IMCs

Previous works highlighted the crucial role of BSCCs in product MCs [40, Theorem 10.56]. As the probability of reaching an accepting BSCC in a product MC determines the probability of satisfying some property in the original abstraction, we now further introduce the notions of winning and losing components. These components include states that may not belong to a BSCC but from which any path is bound to reach a BSCC.

Definition 22 (Winning/Losing Component)**.** *[60] A* winning (losing) component WC (LC) *of a product MC* $\mathcal{M}_{\otimes}^{\mathcal{A}}$ *is a set of states satisfying* $\mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(WC \models \Diamond U^A) = 1$ ($\mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(LC \models \Diamond U^N) = 1$)*, where* U^A *(*U^N*) is the set of states belonging to an accepting (non-accepting) BSCC in* $\mathcal{M}_{\otimes}^{\mathcal{A}}$*.*

It naturally follows that the probability of eventually reaching a BSCC from some initial state is equal to that of reaching a winning or losing component.

Corollary 1. *In any product MC* $\mathcal{M}_{\otimes}^{\mathcal{A}}$*,*

$$\mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond U^A) = \mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond WC) \tag{5.5}$$

$$\mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond U^N) = \mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond LC)\ . \tag{5.6}$$

For any initial state in a product IMC, our goal is thus to find induced product MCs that minimize and maximize the probability of reaching a winning component.

We refer to the technical appendix for all lemmas and proofs leading to the proposed solution. The key observation is that any product IMC induces a *largest winning component* and a *largest losing component*. The largest winning component is the set of states of the product IMC belonging to a winning component for at least one induced product MC, while the largest losing component is the analogous set for losing components. Definitions of *permanent* and *potential* components follow directly from that of largest components.

Definition 23 (Largest Winning/Losing Components)**.** *A state $\langle Q_i, s_j\rangle \in Q \times S$ of a product IMC $\mathcal{I} \otimes \mathcal{A}$ is a member of the* Largest Winning (Losing) Component $(WC)_L$ $\big((LC)_L \big)$ *if there exists a product MC induced by $\mathcal{I} \otimes \mathcal{A}$ such that $\langle Q_i, s_j\rangle$ belongs to a winning (losing) component.*

Definition 24 (Permanent Winning/Losing Components)**.** *A state $\langle Q_i, s_j\rangle \in Q \times S$ of a product IMC $\mathcal{I} \otimes \mathcal{A}$ is a member of the* Permanent Winning (Losing) Component $(WC)_P$ $\big((LC)_P \big)$ *of $\mathcal{I} \otimes \mathcal{A}$ if $\langle Q_i, s_j\rangle$ belongs to a winning (losing) component for all product MCs induced by $\mathcal{I} \otimes \mathcal{A}$.*

Definition 25 (Potential Winning/Losing Components)**.** *A state $\langle Q_i, s_j\rangle \in Q \times S$ of a product IMC $\mathcal{I} \otimes \mathcal{A}$ is a member of the* Potential Winning (Losing) Component $(WC)_?$ $\big((LC)_? \big)$ *of $\mathcal{I} \otimes \mathcal{A}$ if $\langle Q_i, s_j\rangle \in (WC)_L \setminus (WC)_P$ $\big(\langle Q_i, s_j\rangle \in (LC)_L \setminus (LC)_P \big)$.*

Note that the sets $(WC)_?$ and $(LC)_?$ may intersect, and by extension $(WC)_L$ and $(LC)_L$, while $(WC)_P$ and $(LC)_P$ are disjoint. An important result established in this paper is that any product IMC induces a set of product MCs where all members of the largest winning component belong to a winning component simultaneously. A product IMC induces an analogous set of product MCs for the largest losing component. We provide proofs in Lemmas 8 to 11 of the Appendix.

We now state the main result of this section, which establishes that bounds on the probability of satisfying an ω-regular property in an IMC can be computed by solving a reachability maximization problem on a fixed set of states in a product IMC. These sets are the largest components of the product IMC. Furthermore, solving these problems induce sets of best and worst-case product MCs where the probabilities of reaching a winning component are respectively maximized and minimized for all initial states of the product IMC. The lemmas used in the proof of this theorem are found in the technical Appendix.

Theorem 6. *Let $\mathcal{I}$ be an IMC and $\mathcal{A}$ be a DRA corresponding to ω-regular property Ψ. Let $(WC)_L$ and $(LC)_L$ be the largest winning and losing components and $(WC)_P$ and $(LC)_P$ be the permanent winning and losing components of the product IMC $\mathcal{I} \otimes \mathcal{A}$. Then for any initial state Q_i of $\mathcal{I}$,*

$$\widecheck{\mathcal{P}}_{\mathcal{I}}(Q_i \models \Psi) = \; 1 - \widehat{\mathcal{P}}_{\mathcal{I}\otimes\mathcal{A}}(\; \langle Q_i, s_0 \rangle \models \Diamond (LC)_L \;) \tag{5.7}$$

$$\widehat{\mathcal{P}}_{\mathcal{I}}(Q_i \models \Psi) = \; \widehat{\mathcal{P}}_{\mathcal{I}\otimes\mathcal{A}}(\; \langle Q_i, s_0 \rangle \models \Diamond (WC)_L \;) \; . \tag{5.8}$$

Moreover, there exists a set of induced product MCs $(\mathcal{M}_{\otimes}^{\mathcal{A}})_{worst}$, where, $\forall \mathcal{M}_i \in (\mathcal{M}_{\otimes}^{\mathcal{A}})_{worst}$, the sets of all losing and winning components of $\mathcal{M}_i$ are $(LC)_L$ and $(WC)_P$ respectively, and, $\forall \, \langle Q_i, s_0 \rangle \in (Q \times S)$, $\mathcal{P}_{\mathcal{M}_i}(\langle Q_i, s_0 \rangle \models \Diamond (LC)_L) = \widehat{\mathcal{P}}_{\mathcal{I}\otimes\mathcal{A}}(\langle Q_i, s_0 \rangle \models \Diamond (LC)_L)$. Likewise, there exists a set of induced product MCs $(\mathcal{M}_{\otimes}^{\mathcal{A}})_{best}$, where, $\forall \mathcal{M}_i \in (\mathcal{M}_{\otimes}^{\mathcal{A}})_{best}$, the sets of all losing and winning components of $\mathcal{M}_i$ are $(LC)_P$ and $(WC)_L$ respectively, and, $\forall \, \langle Q_i, s_0 \rangle \in (Q \times S)$, $\mathcal{P}_{\mathcal{M}_i}(\langle Q_i, s_0 \rangle \models \Diamond (WC)_L) = \widehat{\mathcal{P}}_{\mathcal{I}\otimes\mathcal{A}}(\langle Q_i, s_0 \rangle \models \Diamond (WC)_L)$.

Proof. $\widecheck{\mathcal{P}}_{\mathcal{I}}(Q_i \models \Psi)$ is equivalent to a lower bound on the probability of reaching an accepting BSCC from $\langle Q_i, s_0 \rangle$ in $\mathcal{I} \otimes \mathcal{A}$. Equation (5.7) follows from Lemma 7 and the following reasoning: assume (5.7) is not true. This implies that there exists an induced product MC where the probability of reaching an non-accepting BSCC from $\langle Q_i, s_0 \rangle$ is greater than the highest probability of reaching $(LC)_L$, which is a contradiction to Lemma

9 and 13. Next, we denote by $\mathcal{D}$ the set of induced product MCs with set of winning components $(WC)_P$ and set of losing components $(LC)_L$ constructed in Lemma 14. Lemma 12 and Lemma 13 guarantee that the probability of reaching an accepting BSCC from all $\langle Q_i, s_0 \rangle$ is minimized in induced product MCs with the smallest set of winning components and the largest set of losing components respectively. Therefore, $(\mathcal{M}_{\otimes}^{\mathcal{A}})_{\text{worst}} \subseteq \mathcal{D}$. Equation (5.8) and the existence of $(\mathcal{M}_{\otimes}^{\mathcal{A}})_{\text{best}}$ are proved identically. $\square$

The equalities highlighted by this theorem are central to the elaboration of our verification procedure. We first solve a qualitative problem, which is to find the largest components of the product IMC. This can be achieved via graph search and will be the focus of the next section. Then, we compute upper and lower bound probabilities of reaching these components from all states in the product IMC using existing algorithms found in the literature [26] [28]. By doing so, we construct a best-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u \in (\mathcal{M}_{\otimes}^{\mathcal{A}})_{\text{best}}$ and a worst-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l \in (\mathcal{M}_{\otimes}^{\mathcal{A}})_{\text{worst}}$ which respectively maximizes and minimizes the probability of reaching an accepting BSCC from all initial states. Note that the transition values between states inside the components do not affect the reachability probabilities and do not need to be considered.

5.1.2 Components Graph Search Algorithm

We present a graph-based algorithm for finding $(WC)_P$, $(WC)_L$, $(LC)_P$ and $(LC)_L$ in a product IMC, divided into Algorithm 4 and Algorithm 5. We define the sets of potential and permanent BSCCs $(U^A)_?$, $(U^A)_P$, $(U^N)_?$ and $(U^N)_P$ analogously to the potential and permanent winning and losing components. Algorithm 4 takes a product IMC as input and returns its potential and permanent BSCCs. Algorithm 5 takes as inputs a product IMC and its permanent and potential BSCCs and outputs $(WC)_P$, $(WC)_L$, $(LC)_P$ and $(LC)_L$. The potential components are the set difference between the largest and permanent components.

We employ the following notations: a digraph G is said to be generated by an in-

```
Algorithm 4 Find Potential and Permanent BSCCs
1:  Input: Product IMC $\mathcal{I} \otimes \mathcal{A}$
2:  Output: Potential and permanent BSCCs $(U^A)_?$, $(U^A)_P$, $(U^N)_?$, $(U^N)_P$
3:  Initialize: $(U^A)_?$, $(U^A)_P$, $(U^N)_?$, $(U^N)_P := \emptyset$
4:  Construct $G = (V, E)$ with a vertex for each state in $\mathcal{I} \otimes \mathcal{A}$ and an edge between states
    $Q_i$ and $Q_j$ if $\widehat{T}(Q_i, Q_j) > 0$
5:  Find all SCCs of $G$ and list them in $S$
6:  for $S_k \in S$ do
7:    $C_0 := \emptyset$, $i := 0$
8:    repeat
9:      $R_i := S_k \setminus \cup_{\ell=0}^{i} C_\ell$;  $Tr_i := V \setminus R_i$;  $C_{i+1} := At_?(Tr_i, R_i)$;  $i := i + 1$
10:   until $C_i = \emptyset$
11:   if $i \neq 1$ then
12:     Find all SCCs of $R_i$ and add them to $S$
13:   else
14:     if $S_k$ is accepting then
15:       In $C$, list all states in $S_k$ mapping to some accepting set $F_i$ if no other state in
          $S_k$ maps to $E_i$. Find all SCCs of $S_k \setminus At_?(C, S_k)$ and add them to $S$.
16:     else
17:       For all sets $F_i$ to which at least one state in $S_k$ is mapped, set $S'_k = S_k$, list all
          states mapping to $E_i$ in $C$, find all SCCs of $S'_k \setminus At_?(C, S'_k)$ and add them to $S$.
18:     end if
19:     if $At_P(V \setminus S_k, S_k) \neq \emptyset$ then
20:       $(U^A)_? := (U^A)_? \cup \{S_k\}$ or $(U^N)_? := (U^N)_? \cup \{S_k\}$ depending on the acceptance
          status of $S_k$.
21:     else
22:       $(U^A)_P := (U^A)_P \cup \{S_k\}$ or $(U^N)_P := (U^N)_P \cup \{S_k\}$ depending on the ac-
          ceptance status $S_k$ and if no other state in $S_k$ belongs to a potential BSCC of
          the opposite acceptance status. Else, $(U^A)_? := (U^A)_? \cup \{S_k\}$ or $(U^N)_? :=$
          $(U^N)_? \cup \{S_k\}$.
23:     end if
24:   end if
25: end for
26: return $(U^A)_?, (U^A)_P, (U^N)_?, (U^N)_P$
```

duced MC $\mathcal{M}_{\otimes}^{\mathcal{A}}$ with transition matrix T and states $Q \times S$ if G has a representative vertex for all states in $\mathcal{M}_{\otimes}^{\mathcal{A}}$, and an edge exists between two such vertices if $T(Q_i, Q_j) > 0$, $Q_i, Q_j \in Q \times S$. $Reach(S, G)$ denotes the set of vertices in graph G from which there exists a path to the set of vertices S; $At_?(S, G)$ denotes the set of vertices in G from which there exists a path to S for all graphs G' generated by an induced product MC of $\mathcal{I} \otimes \mathcal{A}$, where G and G' share the same set of vertices; and $At_P(S, G)$ denotes the set of vertices in G from which there exists a path to S for at least one graph G' generated by an induced product MC of $\mathcal{I} \otimes \mathcal{A}$. A detailed description of the algorithms is found below.

Algorithm 4:

Line 4: We first assume all transitions with a non-zero upper bound to be "on" and generate a graph $G = (V, E)$ with a vertex for all states and an edge for all transitions in $\mathcal{I} \otimes A$.

Line 5: Next, we find all strongly connected components (SCC) of G and list them in S.

Line 6 to 10: For all SCC $S_k \in S$, we want to determine if there exists an induced MC where S_k is a BSCC. To this end, for all the states S_k^j in S_k, we check whether all outgoing transitions to states not in S_k can be turned "off" for some induced product MC, that is if the transition lower bounds from S_k^j to states in $Tr_i = V \setminus R_i$ are 0 and the sum of the transition upper bounds from S_k^j to states in S_k is greater than 1, which is captured by the use of the function $At_?$. Otherwise, S_k^j is said to be *leaky* in all induced product MCs and S_k^j is added to the set C_i, which contains all leaky states of S_k found at iteration i. Note that $R_0 = S_k$ and that all leaky states previously found are removed from S_k at each iteration via variable R_i. The loop terminates when all states have been checked and no more leaky states are found, that is $C_i = \emptyset$.

Line 11 to 13: If S_k contained leaky states that were previously removed, we compute all SCCs formed by the remaining states in R_i and add them to the list of SCCs of G. If S_k did not contain any leaky state, it is a member of a largest set of BSCCs and the mapping of the states in S_k with respect to the Rabin Pairs decides whether S_k is accepting and $S_k \in (U^A)_L$ or non-accepting and $S_i \in (U^N)_L$.

Line 14 to 15: If $S_k \in (U^A)_L$, it could still contain potential non-accepting BSCCs, since $(U^A)_L$ and $(U^N)_L$ may comprise intersecting sets. Treat all states causing S_k to be accepting as leaky (states mapping to some F_i in the Rabin pairs when no states in S_k maps to E_i), remove from S_k all states that have a permanent path to the leaky states, compute all SCCs formed by the remaining states and add them to S.

Line 16 to 17: If $S_k \in (U^N)_L$, potential accepting BSCCs may lie inside S_k. For all sets F_i in the Rabin pairs to which at least one state in S_k is mapped, create a "copy" S'_k of S_k where all states causing S_k to be non-accepting are considered leaky (the states mapping to E_i), remove from S'_k all states that have a permanent path to the leaky states, compute all SCCs formed by the remaining states and add them to S.

Line 19 to 22: We check whether some state in B leaks outside of B for at least one induced MC. If so, the BSCC is not permanent. Otherwise, B is permanent if and only if no BSCC of the opposite acceptance status is found inside of B.

Algorithm 5:

Inspired by the Classical Algorithm for Buchi MDPs [61], we perform a graph search to find the permanent and largest winning and losing components of $\mathcal{I} \otimes \mathcal{A}$. The permanent components only arise from permanent BSCCs, while the largest components stem from both potential and permanent BSCCs.

Line 3: We generate a graph $G = (V, E)$ where transitions with a non-zero upper bound are assumed to be "on".

Line 4 to 11: To find the largest winning component of the product IMC $\mathcal{I} \otimes \mathcal{A}$, we find the set R_i of all states from which there is a path to $(U^A)_? \cup (U^A)_P$ in G. Other states in G are "trap states" denoted by Tr_i. Then, we iteratively remove the set of states C_i from R_i that "leak" to Tr_i for all induced MCs, and compute the new set R_{i+1} of states that have a path to $(U^A)_? \cup (U^A)_P$ once the leaky states are discarded. The iteration stops when no more leaky states are found, that is $C_i = \emptyset$. The remaining states belong to the largest winning component $(WC)_L$. The same procedure is applied with respect to $(U^N)_? \cup (U^N)_P$ to find the largest losing component $(LC)_L$.

Line 12 to 20: To find the permanent winning component of the product IMC $\mathcal{I} \otimes \mathcal{A}$,

Algorithm 5 Find Largest and Permanent Components

```
 1: Input: Product IMC $\mathcal{I} \otimes \mathcal{A}$ and its potential and permanent BSCCs
    $(U^A)_?, (U^A)_P, (U^N)_?, (U^N)_P$
 2: Output: Largest and permanent components $(WC)_L$, $(LC)_L$, $(WC)_P$, $(LC)_P$
 3: Construct $G = (V, E)$ with a vertex for each state in $\mathcal{I} \otimes \mathcal{A}$ and an edge between states
    $Q_i$ and $Q_j$ if $\widehat{T}(Q_i, Q_j) > 0$
 4: for $B \in \{(U^A)_? \cup (U^A)_P, (U^N)_? \cup (U^N)_P\}$ do
 5:     $C_0 := \emptyset, V_0 := V, i := 0$
 6:     repeat
 7:         $R_i := Reach(B \cap V_i, V_i); Tr_i := V_i \setminus R_i; C_{i+1} := At_?(Tr_i, V_i)\ ; V_{i+1} := V_i \setminus C_{i+1}$
            $i := i + 1$
 8:     until $C_i = \emptyset$
 9:     $W := V \setminus \cup_{k=1}^{i} C_k$
10:     $(WC)_L := W$ if $B = (U^A)_? \cup (U^A)_P$ or $(LC)_L := W$ if $B = (U^N)_? \cup (U^N)_P$
11: end for
12: for $B \in \{(U^A)_P, (U^N)_P\}$ do
13:     In $D$, list the states of $V$ belonging to the largest component of the opposite accep-
        tance status from $B$
14:     $C_0 := \emptyset, V_0 := V \setminus D, i := 0$
15:     repeat
16:         $R_i := Reach(B \cap V_i, V_i); Tr_i := V_i \setminus R_i; C_{i+1} := At_P(Tr_i, V_i); V_{i+1} := V_i \setminus C_{i+1}$
            $i := i + 1$
17:     until $C_i = \emptyset$
18:     $W := V_0 \setminus \cup_{k=1}^{i} C_k$
19:     $(WC)_P := W$ if $B = (U^A)_P$ or $(LC)_P := W$ if $B = (U^N)_P$
20: end for
21: return $(WC)_L$, $(LC)_L$, $(WC)_P$, $(LC)_P$
```

we first discard all states which belong to the largest losing component $(LC)_L$ from the set of edges V to be analyzed. Then, we repeat the same procedure as in Algorithm 4 with respect to $(U^A)_P$ until no more leaky states are found, that is $C_i = \emptyset$, except that leaky states are now those which have a path to the trap states Tr_i in at least one induced MC of the product IMC. The remaining states upon completion of this iterative removal belong to $(WC)_P$. The same procedure is applied with respect to $(U^N)_P$ to find the permanent losing component $(LC)_P$.

To summarize, it is known that verification against temporal logic specifications in discrete-time MCs can be accomplished by solving a reachability problem on a product MC constructed from a Rabin automaton corresponding to the specification to be verified. The heart of this approach relies on analyzing winning and losing components of the product MC. These ideas do not directly extend to IMCs because BSCCs are not uniquely determined in this case; this is because some transitions can have a lower transition bound equal to 0 but an upper transition bound that is non-zero. Instead, we introduced the concepts of largest winning and losing components. In Theorem 6, we show that upper and lower bounds on the probability of satisfaction are obtained from these components. Algorithms 4 and 5 provide means for computing these components. Note that the proposed algorithm allows to perform verification of IMCs without constructing an exponentially large Markov decision process, as done in [28].

5.2 Refinement of the Domain Partition

Given a partition P of the domain D and a specification $\phi = \mathcal{P}_{\bowtie p_{sat}}[\Psi]$, the verification procedure derived in Section 5.1 assigns each discrete state of P to one of the sets Q_{ϕ}^{yes}, Q_{ϕ}^{no} or $Q_{\phi}^{?}$ as seen in Subsection 3.4.1. One aims to find a partition P that yields a low volume of undecided states in $Q_{\phi}^{?}$. To this end, we suggest a specification-guided iterative partition refinement method. Specifically, we first generate a rough partition P of D and successively refine P into finer partitions by targeting the best candidate states for reducing the uncertainty in the abstraction with respect to the transition intervals and, consequently, to the probability of satisfying Ψ.

Definition 26 (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 $\{Q_{j'}^k\}_{k=0}^{m_j}$ in P' such that $Q_j = \cup_{k=0}^{m_j} Q_{j'}^k$.*

The states to refine are chosen after comparing the behavior of the system in the best and worst-case probability assignment scenarios computed during verification. The procedure stops when a user-defined criterion is reached. Here, we terminate when the fractional volume of uncertain states is less than a threshold $V_{stop} \in [0, 1]$.

We seek to analyze the behavior of accepting paths in the best= and worst-case product MCs $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ obtained at the time of verification and illustrated in Figure 5.3.

Algorithm 6 State-Space Refinement Scoring Procedure

```
1: Input: Worst-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ and best-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ induced by
   the product $\mathcal{I} \otimes \mathcal{A}$
2: Output: Scores $\sigma = [\sigma_0, \ldots, \sigma_N]$ for all states in $\mathcal{I}$
3: Initialize: $\sigma_i = 0$, with $\sigma_i$ the score of the $i$-th state of $\mathcal{I}$, $p_{stop} \in (0,1)$ user-defined
   probability threshold
4: for $Q_\ell \in Q_\phi^?$ do
5:     $\pi := q_0 := \langle Q_\ell, s_0 \rangle$ in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$
6:     repeat
7:         if $\mathcal{P}(\pi) < p_{stop}$ or $Exp(\pi) = R(\pi)$ then
8:             $\pi := \pi^-$
9:         else
10:            if $Exp(\pi) \neq \emptyset$ then
11:                $q_i \rightarrow Exp(\pi)$, where $q_i$ is any state in $R(\pi) \setminus Exp(\pi)$; $\pi := \pi^+(q_i)$
12:            else
13:                $Exp(\pi) := \cup_i \pi_i$
14:                if $Last(\pi) \in (WC)_? \cup (LC)_?$ then
15:                    $\sigma_j := \sigma_j + \mathcal{P}(\pi)(p_{max} - p_{min})$ for all states $\langle Q_j, s_i \rangle$ in the potential BSCC
                       of $Last(\pi)$ with an outgoing transition which can be either zero or non-
                       zero, $p_{max}$ and $p_{min}$ are the probabilities of reaching an accepting BSCC
                       from $Last(\pi)$ in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ respectively; $\pi := \pi^-$
16:                else if $Last(\pi) \in (WC)_P \cup (LC)_P$ then
17:                    $\pi := \pi^-$
18:                else
19:                    $\sigma_j := \sigma_j + \mathcal{P}(\pi)(p_{max} - p_{min})$, where $j$ corresponds to $\langle Q_j, s_i \rangle :=$
                       $Last(\pi)$, $p_{max}$ and $p_{min}$ are as in line 15;
20:                    $\pi := \pi^+(q_i)$ where $q_i$ is any state in $R(\pi)$
21:                end if
22:            end if
23:        end if
24:    until $\pi = \emptyset$
25: end for
26: return $\sigma$
```

In particular, for every undecided state Q_j in $Q_\phi^?$, we look at all paths starting from $\langle Q_j, s_0 \rangle$ in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and assign a score to the states encountered along them depending on how these states behave in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$. We inspect a path until it reaches a state that belongs to either $(WC)_L$ or $(LC)_L$, or when its probability of occurrence in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ falls below a threshold p_{stop}. States with high scores are targeted for refinement.

We introduce some notation: for a finite path $\pi = q_0 q_1 \ldots q_k$ in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$, $Last(\pi)$ denotes the last state q_k of π; π_i denotes the i-th state of π; $\mathcal{P}(\pi) = T(q_0, q_1) \cdot T(q_1, q_2) \cdot \ldots \cdot T(q_{k-1}, q_k)$, $\mathcal{P}(q_0) = 1$, is the probability of path π in $(\mathcal{M}^{\mathcal{A}}_{\otimes})_u$; $R(\pi)$ is the set of states that are one-step reachable from $Last(\pi)$ in $(\mathcal{M}^{\mathcal{A}}_{\otimes})_u$; $Exp(\pi)$ denotes all continuations of π from $R(\pi)$ that have been explored and is initialized to the empty set for all π; π^- is the path obtained by removing the last state of π and $\pi^+(q_i)$ is the path with q_i appended to π. $V_?$ is the fractional volume of uncertain states and is equal to the sum of the volume of all states in $Q^?_\phi$ divided by the volume of the domain D. Our procedure is as follows:

1. Compute a refinement score for all states in $\mathcal{I}$ according to Algorithm (6), which is described below:

   Line 6 to 24: This loop terminates when $\pi = \emptyset$, that is, when all paths starting from $\langle Q_j, s_0 \rangle$ have been explored.

   Line 7 to 8: If $\mathcal{P}(\pi) < p_{stop}$ or $Exp(\pi) = R(\pi)$, the probability of the path is below the pre-defined exploration threshold or all continuations of π have been explored. Thus, we return to the previous state in the path.

   Line 10 to 13: If $Exp(\pi) \neq \emptyset$, add q_i to $Exp(\pi)$, where q_i is some unexplored state in $R(\pi)$ and extend the path to q_i. Else, π is a path fragment which has not been explored yet. Add all states in π to $Exp(\pi)$ to avoid loops.

   Line 14 to 15: If $Last(\pi) \in (WC)_?$ or $Last(\pi) \in (LC)_?$, the path reached a state in a potential component. We want to target the states which can either confirm or refute that $Last(\pi)$ belongs to such a component. These states are the ones inside the potential BSCCs that $Last(\pi)$ belongs to (or makes a transition to with probability 1) that have outgoing transitions which can be either "on" or "off", as depicted in Figure 5.4. A potential "certainty gain" is added to the score of all such states and the path is returned to its previous state. If $Last(\pi)$ belongs to both $(WC)_?$ and $(LC)_?$, then the scoring scheme is applied to all intersecting potential BSCCs related to $Last(\pi)$. This heuristical gain quantifies a potential reduction in the width of the satisfaction interval of Q_ℓ in the scenario that the refinement of the considered states provides perfect information, i.e., the probability of reaching an accepting BSCC from $Last(\pi)$ becomes a fixed number.

Line 16 to 17: If $Last(\pi) \in (WC)_P$ or $Last(\pi) \in (LC)_P$, the path reached a region of the state-space that does not require refinement as it belongs to a permanent component. The path returns to its previous state.

Line 18 to 20: Else, $Last(\pi)$ does not belong to a winning or losing component for any refinement of the product IMC. The potential "certainty gain" one can hope for by refining $\langle Q_j, s_i \rangle = Last(\pi)$ is added to the score of Q_j. The path is continued to an unexplored state.

2. Refine the states in P_k with scores above a user-defined threshold to generate P_{k+1}.

3. Generate an IMC abstraction of the system with respect to P_{k+1}, perform model-checking and compute $V_?$.

4. If $V_? > V_{stop}$, return to step 1. Else, terminate.

It is not difficult to construct examples demonstrating that the volume of uncertain states $V_?$ need not decrease monotonically at each step of the refinement algorithm using our abstraction technique. This is because, when a parent state is refined to two children states, the sum of the upper transition bounds for the children states may be greater than the upper transition bound of the original parent state. Nevertheless, when $\mathcal{F}$ is continuous, the size of the reachable sets, and consequently the error in the transitions, approaches zero as the grid size decreases. Thus, in the limit, the volume of uncertain states $V_?$ decreases to zero.

A common refinement approach consists in splitting the chosen states in the partition in half along their greatest dimension. As the scoring procedure in Algorithm 6 may select the

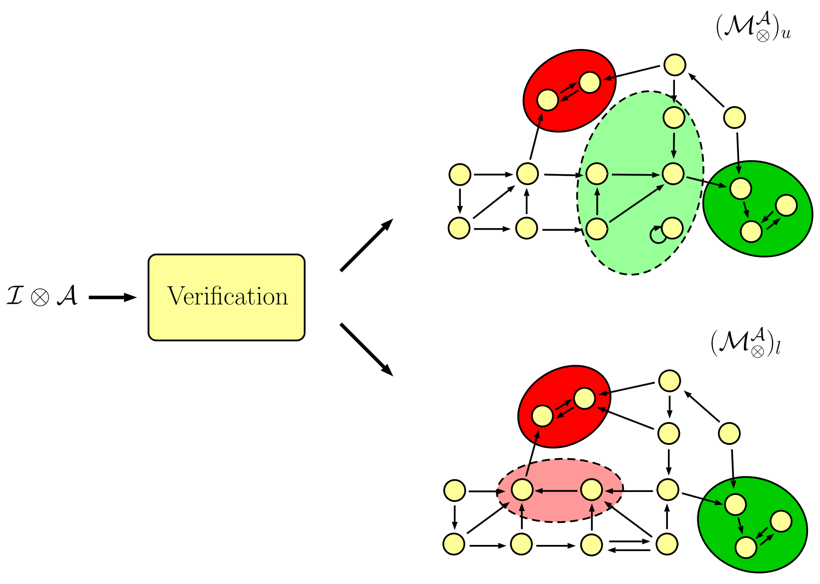

$\mathcal{I} \otimes \mathcal{A}$
Verification
$(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$
$(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$

Figure 5.3: Our IMC verification algorithm generates a best- and worst-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$. Winning and losing components are respectively in red and green; permanent and potential components are circled in bold and dotted lines respectively. Comparing the behavior of the paths in the two scenarios is the basis of our refinement algorithm.

entire state-space of IMC $\mathcal{I}$ for refinement, the worst-case growth of the size of the product IMC $\mathcal{I} \otimes \mathcal{A}$ is exponential and scales in $\mathcal{O}(|S| \cdot 2^{|Q|})$, where $|S|$ and $|Q|$ are the number of states of automaton $\mathcal{A}$ and IMC $\mathcal{I}$ respectively. However, because this path-based scoring procedure aims to target states which are most likely to reduce the volume of undecided states in the partition with respect to the specification under consideration, our refinement algorithm tends to focus on specific regions of the state-space with the effect of mitigating state-space explosion.

In summary, we develop an abstraction-based verification procedure for discrete-time stochastic systems with probabilistic ω-regular objectives. Given a partition of the system domain, an IMC abstraction of the dynamics is generated, and bounds on the probability of satisfying the property from all initial states of the IMC are computed by solving a graph-based qualitative problem and a value iteration quantitative problem in the product between the abstraction and an automaton. We implement a heuristical partition refinement tech-

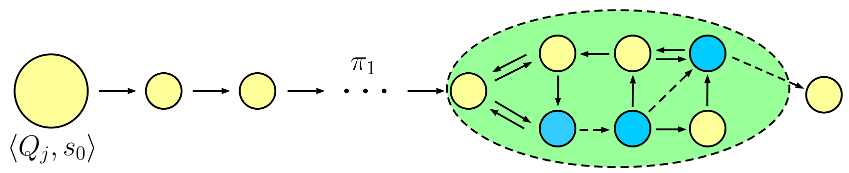


Figure 5.4: For all undecided states Q_j of the IMC abstraction $\mathcal{I}$, we inspect all paths starting from $\langle Q_j, s_0 \rangle$ in $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ to determine which states to refine. Above is an example of a path π_1. A score is assigned to all states along π_1 as detailed in Algorithm 6. In particular, if π_1 reaches a member of a potential BSCC, a score is assigned to the states which could possibly destroy the BSCC under refinement. These states are shown in blue and have outgoing transitions with lower bound 0.

nique that compares the two extreme non-deterministic scenarios induced by the product between the abstraction and the automaton and targets specific partition states accordingly in order to reduce the conservatism of the derived bounds if necessary.

CHAPTER 6
CONTROLLER SYNTHESIS FOR STOCHASTIC SYSTEMS WITH OMEGA-REGULAR OBJECTIVES

This chapter addresses the synthesis of controllers for stochastic dynamical systems with temporal logic objectives as defined by the probabilistic synthesis problem, building on the automaton-based verification methodology delineated in Chapter 5. We distinguish the instance where a finite number of inputs or *modes* is available with that offering a continuous set of possible inputs, as the two situations necessitate different solutions.

In Section 6.1, we investigate the synthesis of switching policies for finite-mode systems subject to ω-regular specifications. The proposed strategy employs interval-valued finite-state abstractions, known as *Bounded-Parameter Markov Decision Processes* (BMDP), constructed from a partition of the continuous system domain. A BMDP is a probabilistic transition system allowing a finite set of actions at each state and induces an IMC for a fixed switching policy. A BMDP abstraction can alternatively be viewed as a collection of IMC abstractions, each one of them corresponding to a mode of the underlying continuous-state system. Devising an optimal switching in the BMDP abstraction engenders a near-optimal policy with respect to the abstracted continuous states.

Therefore, we present a technique for computing optimal switching policies in BMDPs with ω-regular objectives. A maximizing policy aims at maximizing the lower bound probability of satisfying an objective for all initial states of the BMDP, whereas a minimizing policy minimizes the upper bound probability of satisfying the objective. We show that this task is decomposed into a qualitative problem and a quantitative problem. The qualitative problem requires building the greatest possible permanent winning or losing component in the product between the BMDP and a DRA encoding the ω-regular property. In the case of maximization, the greatest permanent winning component is constructed, while in the case of minimization, the greatest permanent losing component is constructed. We detail graph-based algorithms for finding these components and their associated control actions. For every state of the product BMDP that do not belong to these components, the quantitative problem demands to compute a policy that maximizes the lower bound probability of reaching the permanent components, which is accomplished through a value iteration scheme.

As the efficacy of the computed policy with respect to the abstracted system depends on the quality of the domain partition from which the BMDP abstraction originates, we suggest an approach for quantitatively assessing the optimality of the policy. First, we design another switching policy that maximizes the upper bound probability of reaching a winning or losing component for each state of the product BMDP, instead of maximizing the lower bound probability as previously done. As opposed to accommodating the worst-case scenario with respect to the objective (minimization or maximization), this policy creates the most favorable best-case scenario. Then, a suboptimality factor comparing the probabilities of satisfaction resulting from the two policies is assigned to each state of the product BMDP. This technique is also used to detect actions which are necessarily optimal or suboptimal at each state of the product BMDP. To proceed to a refinement of the domain partition, the worst-case and best-case product MCs induced by the two policies are provided as inputs to a refinement algorithm inspired by the one in Section 5.2 for the purpose of verification. The synthesis procedure terminates once the suboptimality factor of every state reaches a user-defined precision threshold.

In Section 6.2, we treat the problem of computing control policies for stochastic systems with a continuous set of available inputs when the specification is ω-regular. We focus on the class of systems which are affine in input and in disturbance. The type of finite-state abstractions used for such systems is the *Controlled Interval-valued Markov Chain* (CIMC), where an input drawn from a continuous set determines an interval on the probability of transition between any two states. As in the finite-mode case, computing an optimal control policy in a CIMC abstraction results in a near-optimal policy for the original abstracted system.

Finding an optimal control policy in a CIMC is similarly divided into a qualitative and quantitative problem. These problems are tackled in the Cartesian product between the CIMC and a DRA representing the ω-regular specification of interest. First, we show that the qualitative problem is addressed by constructing a BMDP through the selection of a finite number of actions from the continuous set of inputs. This is due to the fact that, because CIMCs are endowed with a finite number of states, the number of possible qualitative transition configurations between these states is also finite. Then, the greatest permanent winning and losing components of the product between the CIMC and the DRA are built by applying the finite-mode algorithm to the constructed BMDP. Correctly choosing the necessary finite set of actions from the continuous set of inputs depends on the dynamics at hand and the geometry of the partition states. We show how to select these actions under certain assumptions on the dynamics and on the structure of the noise. Then, for the states which are not a member of the greatest permanent components, we show that an optimal input is computed through an iterative resolution of optimization problems.

The optimality of the computed controller with respect to the original abstracted states is quantitatively measured for this case as well, and an iterative, targeted partition refinement algorithm inspired from the finite-mode case is proposed with the aim of reaching a user-defined level of optimality.

6.1 Synthesis for Finite-mode Switched Stochastic Systems

Recall the general SDE for discrete-time, continuous-state stochastic systems with a finite number of modes presented in Section 3.1 as

$$x[k+1] = \mathcal{F}_a(x[k], w_a[k]) \ , \tag{6.1}$$

where $a \in A$, with A being a finite set of modes.

As stated in the probabilistic synthesis problem, our objective is to determine switching policies $\widecheck{\mu}_{\Psi}$ and $\widehat{\mu}_{\Psi}$ that respectively minimize and maximize the probability of satisfying property Ψ for any path in system (6.1) and, by extension, for any initialization to x of (6.1).

Problem 1: *Given a system of the form* (6.1)*, any initial state $x \in D$ and an ω-regular property Ψ, find switching policies $\widecheck{\mu}_{\Psi} \in \mathcal{U}$ and $\widehat{\mu}_{\Psi} \in \mathcal{U}$ that respectively minimize and maximize the probability of satisfying Ψ from x,* i.e.,

$$\widecheck{\mu}_{\Psi} = \arg\min_{\mu \in \mathcal{U}} (p^{x}_{\Psi})_{\mu} \tag{6.2}$$

$$\widehat{\mu}_{\Psi} = \arg\max_{\mu \in \mathcal{U}} (p^{x}_{\Psi})_{\mu}\ . \tag{6.3}$$

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 the domain D of (6.1) into a finite collection of states P to construct a finite-state abstraction of the stochastic dynamics in the form of a BMDP abstraction, defined in Definitions 4 and 5 in Subsection 3.4.2. Techniques for constructing non-trivial BMDP abstractions for certain classes of stochastic systems are presented in Chapter 4.

Given a BMDP abstraction $\mathcal{B}$ of (6.1) generated from a partition P of the domain D, our approach to Problem 1 is to find policies $\widehat{\mu}^{low}_{\Psi}$ and $\widecheck{\mu}^{up}_{\Psi}$ that respectively maximize the lower bound probability and minimize the upper bound probability of satisfying Ψ for all initial states Q_j of $\mathcal{B}$.

Subproblem 1.1: *Given a system of the form* (6.1)*, a partition P of its domain D,* *a BMDP abstraction $\mathcal{B}$ of* (6.1) *arising from P, any initial state $Q_j \in Q$ of $\mathcal{B}$ and an ω-regular property Ψ, compute the 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.,

$$\widecheck{\mu}_{\Psi}^{up} = \arg\min_{\mu \in \mathcal{U}_{\mathcal{B}}} \widehat{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) \tag{6.4}$$

$$\widehat{\mu}_{\Psi}^{low} = \arg\max_{\mu \in \mathcal{U}_{\mathcal{B}}} \widecheck{\mathcal{P}}_{\mathcal{B}[\mu]}(Q_j \models \Psi) \ . \tag{6.5}$$

If $\mathcal{B}$ is a BMDP abstraction of (6.1), then a unique control action is assigned to all continuous states abstracted by some Q_i in $\mathcal{B}$. In this case, the optimality 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. 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* (6.1) *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 optimality with respect to the objective of minimizing or maximizing the probability of satisfying Ψ in* (6.1).

6.1.1 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 seen in Chapter 5, for every such property, there exists a corresponding DRA representation $\mathcal{A}$ which is used to determine whether a sequence of states visited by a transition system satisfies the encoded property. Similar to Chapter 5 where the Cartesian product of an IMC with a DRA is introduced, we define the Cartesian product $\mathcal{B} \otimes \mathcal{A}$ between a BMDP and a DRA.

Definition 27 (Product Bounded-Parameter Markov Decision Process)**.** *Let $\mathcal{B} = (Q, Act, \widecheck{T}, \widehat{T}, q_0, \Pi, L)$ be a BMDP and $\mathcal{A} = (S, \Pi, \delta, s_0, Acc)$ be a DRA. The* product $\mathcal{B} \otimes \mathcal{A} = (Q \times S, Act, \widecheck{T'}, \widehat{T'}, q_0^{\otimes}, Acc', L')$ *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 $s_i \in S$,*
- $$\widecheck{T'}_{\langle Q_j, s\rangle \xrightarrow{a} \langle Q_\ell, s'\rangle} = \begin{cases} \widecheck{T'}_{Q_j \xrightarrow{a} Q_\ell}, & \textit{if } s' = \delta(s, L(Q_\ell)) \\ 0, & \textit{otherwise} \end{cases}$$
- $$\widehat{T'}_{\langle Q_j, s\rangle \xrightarrow{a} \langle Q_\ell, s'\rangle} = \begin{cases} \widehat{T'}_{Q_j \xrightarrow{a} Q_\ell}, & \textit{if } s' = \delta(s, L(Q_\ell)) \\ 0, & \textit{otherwise} \end{cases}$$
- *$q_0^{\otimes} = \{(Q_j, s_0) : Q_j \in q_0\}$ 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 $H \in L'(\langle Q_j, s \rangle)$ if and only if $s \in H$, for all $H \in Acc'$ and for all j.*

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 27. 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 28 (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 29 (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 $\mathcal{B}$, we asserted in Chapter 5 that the probability of satisfying Ψ in the induced IMC $\mathcal{B}[\mu]$ from some initial states is equal to the probability of reaching an accepting BSCC from the corresponding initial states in the product IMC $\mathcal{B}[\mu] \otimes \mathcal{A}$. 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 $\mathcal{B}[\mu][\nu]_{\otimes}^{\mathcal{A}}$.

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 17 and 27 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.

Our objective consists in computing policies that maximize the lower bound probability and minimize the upper bound probability of reaching an accepting BSCC from all initial states of the resulting product IMC $\mathcal{B}[\mu] \otimes \mathcal{A}$. Furthermore, as discussed in [59] and Chapter 5, reachability probabilities in IMCs are minimized and maximized by *memoryless* adversaries which depend solely on the current state of the IMC. Therefore, only adversaries, and by extension switching policies, that are memoryless in the product $\mathcal{B} \otimes \mathcal{A}$ (thus finite-memory in $\mathcal{B}$) need to be considered for solving Subproblem 1.1.

Definition 30 (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 $\pi = q[0]q[1]\dots q[k]$ of $\mathcal{B}$, it holds that $\mu(\pi) = \mu(q[k])$.*

Definition 31 (Memoryless Adversary)**.** *An adversary $\nu \in \mathcal{I}_{\nu}$ of an IMC $\mathcal{I}$ is said to be* memoryless *if, for all finite paths $\pi = q[0]q[1]\dots q[k]$ of $\mathcal{I}$, it holds that $\nu(\pi) = \nu(q[k])$.*

Fact 4. *We denote the set of policies of a BMDP $\mathcal{B}$ which are induced by memoryless policies in the product $\mathcal{B} \otimes \mathcal{A}$ by $(\mathcal{U}_{ind})^{\mathcal{A}}_{\otimes} \subseteq \mathcal{U}_{\mathcal{B}}$. For any IMC $\mathcal{B}[\mu]$ induced by a policy $\mu \in (\mathcal{U}_{ind})^{\mathcal{A}}_{\otimes}$, we denote the set of adversaries which are induced by memoryless adversaries in the product IMC $\mathcal{B}[\mu] \otimes \mathcal{A}$ by $(\nu_{\mathcal{B}[\mu],Ind})^{\mathcal{A}}_{\otimes} \subseteq \nu_{\mathcal{B}[\mu]}$. For any initial state Q_j of $\mathcal{B}$, it holds that*

$$\sup_{\mu \in \mathcal{U}_{\mathcal{B}}} \inf_{\nu \in \nu_{\mathcal{B}[\mu]}} \mathcal{P}_{\mathcal{B}[\mu][\nu]}(Q_j \models \Psi) = \sup_{\mu \in (\mathcal{U}_{ind})^{\mathcal{A}}_{\otimes}} \inf_{\nu \in (\nu_{\mathcal{B}[\mu],Ind})^{\mathcal{A}}_{\otimes}} \mathcal{P}_{\mathcal{B}[\mu][\nu]}(Q_j \models \Psi) \, ,$$

$$\inf_{\mu \in \mathcal{U}_{\mathcal{B}}} \sup_{\nu \in \nu_{\mathcal{B}[\mu]}} \mathcal{P}_{\mathcal{B}[\mu][\nu]}(Q_j \models \Psi) = \inf_{\mu \in (\mathcal{U}_{ind})^{\mathcal{A}}_{\otimes}} \sup_{\nu \in (\nu_{\mathcal{B}[\mu],Ind})^{\mathcal{A}}_{\otimes}} \mathcal{P}_{\mathcal{B}[\mu][\nu]}(Q_j \models \Psi) \, .$$

Before presenting a solution to Subproblem 1.1, we first recall some basic results established in Chapter 5 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 values for the transitions with zero lower bound and non-zero upper bound which cause certain edges to be either "on" or "off". To resolve this, we showed 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.

Moreover, it was shown in Theorem 6 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 $\mathcal{B}$.

The intuitive interpretation of this theorem 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 bound and lower bound probabilities of satisfying Ψ from the initial states of $\mathcal{B}[\mu]$. In the induced product MC corresponding to the best-case scenario, the set of winning components is as large as it can possibly be while the set of losing components is reduced to the smallest possible set of permanent losing components; the opposite holds true in the induced product MC corresponding to the worst-case scenario.

Recall our objective which is to find switching policies $\widecheck{\mu}_{\Psi}^{up}$ 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 the worst possible best-case scenario with respect to the probability of reaching an accepting BSCC in the product $\mathcal{B} \otimes \mathcal{A}$ for the minimization case. To this end, we first state in the following lemma that there exist sets of switching policies of $\mathcal{B} \otimes \mathcal{A}$ resulting in the greatest possible set of permanent winning components and the greatest possible set of permanent losing components in the corresponding induced product IMCs.

Lemma 4. *Let $\mathcal{B}$ be a BMDP and Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$. The set of memoryless switching policies of the product $\mathcal{B} \otimes \mathcal{A}$ is denoted by $\mathcal{U}_{\otimes}^{\mathcal{A}}$. There exists a set of switching policies $\mathcal{U}_{(WC)_P^G} \subseteq \mathcal{U}_{\otimes}^{\mathcal{A}}$ 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 $(WC)_P^G$. Likewise, there exists a set of switching policies $\mathcal{U}_{(LC)_P^G} \subseteq \mathcal{U}_{\otimes}^{\mathcal{A}}$ generating the set $(LC)_P^G$ in $\mathcal{B} \otimes \mathcal{A}$ with the same properties with respect to losing components.*

A constructive proof of this lemma is provided in the Appendix. The sets $(WC)_P^G$ and $(LC)_P^G$ are respectively called the *Greatest Permanent Winning Component* and the *Greatest Permanent Losing Component* of the product BMDP $\mathcal{B} \otimes \mathcal{A}$.

From Lemma 4, we infer 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 $\widecheck{\mu}_{\Psi}^{up}$ in $\mathcal{B} \otimes \mathcal{A}$ that generates the set $(LC)_P^G$ and, for all states not in $(LC)_P^G$, maximizes the lower bound probability of reaching this set. Because optimal switching policies for reachability objectives are memoryless, it follows that the policy $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ maximizing the lower bound probability of reaching an accepting BSCC is the same for all initial states of $\mathcal{B} \otimes \mathcal{A}$. Likewise, the policy $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ minimizing the upper bound probability of reaching an accepting BSCC is the same for all initial states of $\mathcal{B} \otimes \mathcal{A}$.

Theorem 7. *Let $\mathcal{B}$ be a BMDP and Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$. Let $(WC)_P^G$ and $(LC)_P^G$ be the greatest permanent winning and losing component, respectively, of the product BMDP $\mathcal{B} \otimes \mathcal{A}$, and $\mathcal{U}_{(WC)_P^G}$ and $\mathcal{U}_{(LC)_P^G}$ be the policies generating these sets as defined in Lemma 4. A lower bound maximizing and upper bound minimizing switching policy $\widehat{\mu}_\Psi^{low}$ and $\widecheck{\mu}_\Psi^{up}$ in $\mathcal{B}$ with respect to Ψ are respectively induced by switching policies $(\widehat{\mu}_\Psi^{low})_\otimes$ and $(\widecheck{\mu}_\Psi^{up})_\otimes$ in $\mathcal{B} \otimes \mathcal{A}$ such that*

$$(\widehat{\mu}_\Psi^{low})_\otimes = \underset{\mu \in \mathcal{U}_{(WC)_P^G}}{\arg\max}\, \widecheck{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu]}\big(\langle Q_j, s_0 \rangle \models \Diamond (WC)_P^G \big) \tag{6.6}$$

$$(\widecheck{\mu}_\Psi^{up})_\otimes = \underset{\mu \in \mathcal{U}_{(LC)_P^G}}{\arg\max}\, \widecheck{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu]}\big(\langle Q_j, s_0 \rangle \models \Diamond (LC)_P^G \big) \tag{6.7}$$

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

Proof. We first prove equation (6.6). 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 Theorem 6, it is 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 is shown in Lemma 13 in the Appendix 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. 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, whose optimal policies are memoryless, there exists a policy in

$\mathcal{U}_{(WC)_P^G}$ maximizing the lower bound probability of reaching $(WC)_P^G$ simultaneously for all states 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 (6.6). A symmetric argument with respect to non-accepting BSCCs and losing components can be used to prove (6.7). $\square$

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 [26]. An algorithm for finding the sets $(WC)_P^G$ and $(LC)_P^G$ as well as their associated control actions are presented in the next subsection.

We additionally 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 optimality 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* $(WC)_L^G$ and *Greatest Losing Component* $(LC)_L^G$ in $\mathcal{B} \otimes \mathcal{A}$, whose existence is established in the lemma below.

Lemma 5. *Let $\mathcal{B}$ be a BMDP and Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$. The set of memoryless switching policies of the product $\mathcal{B} \otimes \mathcal{A}$ is denoted by $\mathcal{U}_\otimes^\mathcal{A}$. There exists a set of switching policies $\mathcal{U}_{(WC)_L^G} \subseteq \mathcal{U}_\otimes^\mathcal{A}$ 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 $(WC)_L^G$. Likewise, there exists a set of switching policies $\mathcal{U}_{(LC)_L^G} \subseteq \mathcal{U}_\otimes^\mathcal{A}$ generating the set $(LC)_L^G$ in $\mathcal{B} \otimes \mathcal{A}$ with the same properties with respect to losing components.*

Proof. Lemma 5 follows from a similar constructive argument as the one in the proof of Lemma 4. $\square$

The sets $(WC)_L^G$ and $(LC)_P^L$ are respectively called the *Greatest Winning Component* and the *Greatest Losing Component* of the product BMDP $\mathcal{B} \otimes \mathcal{A}$.

Theorem 8. *Let $\mathcal{B}$ be a BMDP and Ψ be an ω-regular property with corresponding DRA $\mathcal{A}$. Let $(WC)_L^G$ and $(LC)_L^G$ be the greatest winning and losing component, respectively, of the product BMDP $\mathcal{B} \otimes \mathcal{A}$, and $\mathcal{U}_{(WC)_L^G}$ and $\mathcal{U}_{(LC)_L^G}$ be the policies generating these sets as defined in Lemma 5. 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 $(\widehat{\mu}_\Psi^{up})_\otimes$ and $(\widecheck{\mu}_\Psi^{low})_\otimes$ in $\mathcal{B} \otimes \mathcal{A}$ such that*

$$(\widehat{\mu}_\Psi^{up})_\otimes = \underset{\mu \in \mathcal{U}_{(WC)_L^G}}{\arg\max}\, \widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu]}\big(\langle Q_j, s_0\rangle \models \Diamond (WC)_L^G\big) \tag{6.8}$$

$$(\widecheck{\mu}_\Psi^{low})_\otimes = \underset{\mu \in \mathcal{U}_{(LC)_L^G}}{\arg\max}\, \widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu]}\big(\langle Q_j, s_0\rangle \models \Diamond (LC)_L^G\big) \tag{6.9}$$

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

Proof. As shown in Theorem 6, 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 7 proves (6.8). A symmetric argument with respect to non-accepting BSCCs and losing components proves (6.9). $\square$

We remark that replacing $(WC)_L^G$ and $(LC)_L^G$ in (6.8) and (6.9) by *the greatest accepting and non-accepting BSCCs* $(U^A)_L^G \subseteq (WC)_L^G$ and $(U^N)_L^G \subseteq (LC)_L^G$ respectively does not change the validity of (6.8) and (6.9). The set $(U^A)_L^G$ (respectively, $(U^N)_L^G$) contains all states which belong to an accepting (respectively, non-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 these sets is similar to the first part of the proof of Lemma 4. This substitution can be done because, by definition, $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[(\widehat{\mu}_\Psi^{up})_\otimes]}\Big((WC)_L^G \models \Diamond (U^A)_L^G\Big) = \widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[(\widecheck{\mu}_\Psi^{low})_\otimes]}\Big((LC)_L^G \models \Diamond (U^N)_L^G\Big) = 1$, and leads to algorithmic simplifications as the full sets $(WC)_L^G$ and $(LC)_L^G$ may not need to be computed explicitly. The components $(WC)_L^G$ and $(LC)_L^G$ as well as the control actions generating these components are found via a graph search, as detailed in the next subsections.

6.1.2 Winning and Losing Components Search Algorithms

Now, we present graph-based algorithms for finding the greatest permanent winning component $(WC)_P^G$ and the greatest permanent losing component $(LC)_P^G$ of a product BMDP $\mathcal{B} \otimes \mathcal{A}$ defined in Lemma 4. Furthermore, we show how to design a switching policy that effectively generates these greatest permanent components.

The search is decomposed in two parts: first, we determine a superset of the *greatest permanent accepting BSCC*, denoted by $(U^A)_P^G$, and the *greatest permanent non-accepting BSCC*, denoted by $(U^N)_P^G$, of $\mathcal{B} \otimes \mathcal{A}$ following Algorithm 7 and Algorithm 8. The sets $(U^A)_P^G$ and $(U^N)_P^G$ contain all states which belong to a permanent accepting and non-accepting BSCC respectively for some control policy in $\mathcal{B} \otimes \mathcal{A}$, and all such states are a part of $(WC)_P^G$ and $(LC)_P^G$ as seen in the proof of Lemma 4. We call the supersets of $(U^A)_P^G$ and $(U^N)_P^G$ returned by these algorithms an extended greatest permanent accepting BSCC and an extended greatest permanent non-accepting BSCC, denoted by $(U_+^A)_P^G$ and $(U_+^N)_P^G$ respectively. These sets additionally satisfy $(U^A)_P^G \subseteq (U_+^A)_P^G \subseteq (WC)_P^G$ and $(U^N)_P^G \subseteq (U_+^N)_P^G \subseteq (LC)_P^G$. Although Algorithm 7 and Algorithm 8 are driven by a search of the sets $(U^A)_P^G$ and $(U^N)_P^G$, our implementation allows us to find additional members of $(WC)_P^G$ and $(LC)_P^G$ in some instances.

Then, by using an iterative technique which alternates between a graph search and a reachability maximization step in Algorithm 9 and Algorithm 10, one can find the set of states which are not members of $(U_+^A)_P^G$ or $(U_+^N)_P^G$ but for which the lower bound probability of reaching an accepting BSCC is equal to 1 nonetheless for some control policy, and effectively create $(WC)_P^G$ and $(LC)_P^G$.

GREATEST PERMANENT BSCC SEARCH ALGORITHMS

We now detail an algorithm for finding an extended greatest permanent accepting BSCC $(U_+^A)_P^G$, and an extended greatest permanent non-accepting BSCC $(U_+^N)_P^G$ 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 20 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 algorithms: Algorithm 7 and 8 first find 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 19. 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 $(U_+^A)_P^G$ or $(U_+^N)_P^G$ (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 20. If S_k does not have the desired acceptance status, states which can revert the acceptance status of S_k are removed and new SCCs are computed with the remaining states in line 23 of Algorithm 7 and line 27 of Algorithm 8. 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 $(U_+^A)_P^G$ (respectively, $(U_+^N)_P^G$) is that no subset of states of S_k can form a non-accepting BSCC (respectively, accepting BSCC) under any scenario allowed by the transition intervals of the product BSCC. To verifiy this, the approach is slightly different for both algorithms:

- In Algorithm 7, to 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 $(U_+^A)_P^G$ 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.

- In Algorithm 8, to make sure that no subset of S_k can form an accepting BSCC, we first check whether S_k contains Rabin accepting states. If it does not, then S_k is a member of $(U_+^N)_P^G$ as explicitly shown in line 16. If S_k contains Rabin accepting states, for all sets of Rabin accepting states with respect to pair i, we sequentially solve reachability problems and choose control actions for the states in S_k that maximize the lower bound probability of reaching the Rabin non-accepting states with respect to pair i contained in S_k in line 17 to 19. If this lower bound is zero for some actions at some state, we discard these actions for this state and start again from the first set of Rabin accepting states. The process continues until all the Rabin pairs have been considered without removing any action or until a state has an empty action set. A state with an empty action set could potentially form a BSCC which does not contain a Rabin non-accepting i for some matched Rabin accepting state in S_k, and therefore which could potentially be accepting. If no state has an empty action set, the algorithm found a control policy which guarantees that all states in S_k reach a Rabin non-accepting state with respect to pair i for all matched Rabin accepting states in S_k with respect to pair i with lower bound probability 1, therefore no state of S_k can form an accepting BSCC and S_k is added to $(U_+^N)_P^G$ in line 20 to 21. Otherwise, the SCCs which can be formed by the states in $S_k \setminus A_i$ for all sets of Rabin accepting pairs with respect to pair i are computed with the original set of actions of S_k and added to S in line 23.

We offer the following reasoning as a proof sketch for the correctness of the algorithms: for a set of states S_k to belong to a permanent BSCC of a given kind in a product IMC, its constituents are not allowed to transition outside of S_k under any adversary, its constituents have to fulfill the requirements for accepting and non-accepting BSCCs defined in Definition 20, and no subset of S_k is allowed to form a BSCC of the opposite acceptance status under any adversary. The first condition is guaranteed by lines 7 to 10 in both algorithms; the second condition is enforced by the if-statement in line 14 in both algorithms and the corresponding else-statements of lines 22 to 24 in Algorithm 7 and lines 26 to 28 in Algorithm 8; the third condition is imposed by the remainder of the main for-loop in both

Algorithm 7 Find Extended Greatest Permanent Accepting BSCC

```
1: Input: Product BMDP $\mathcal{B} \otimes \mathcal{A}$
2: Output: Extended greatest permanent accepting BSCCs $(U_+^A)_P^G$ with corresponding policy $(\widehat{\mu}_\Psi^{low})_\otimes$ for the states in this set
3: Initialize: $(U_+^A)_P^G := \emptyset$
4: 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 $a \in A(\langle Q_i, s_j \rangle)$
5: Find all SCCs of $G$ and list them in $S$
6: for $S_k \in S$ do
7:     $C_0 := \emptyset, i := 0$
8:     repeat
9:         $R_i := S_k \setminus \cup_{\ell=0}^{i} C_\ell$;  $Tr_i := V \setminus R_i$;  $(C_{i+1}, Act(R_i)) = At_P(Tr_i, R_i, Act(R_i))$;  $i = i + 1$
10:    until $C_i = \emptyset$ and no action is removed from $Act(R_i)$
11:    if $i \neq 1$ then
12:        Find all SCCs of $R_i$ (with the remaining actions) and add them to $S$
13:    else
14:        if $S_k$ is accepting then
15:            Find the set $A$ of all unmatched Rabin accepting states of $S_k$
16:            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
17:            if $A_{bad} = \emptyset$ then
18:                $(U_+^A)_P^G := (U_+^A)_P^G \cup S_k$ and save the actions computed in the maximization of $\Diamond A$ to $(\widehat{\mu}_\Psi^{low})_\otimes$ for all states of $S_k$
19:            else
20:                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 $S$
21:            end if
22:        else
23:            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 $S$
24:        end if
25:    end if
26: end for
27: return $(U_+^A)_P^G$ , $(\widehat{\mu}_\Psi^{low})_\otimes$ for states in $(U_+^A)_P^G$
```

algorithms. Lastly, the algorithms iteratively remove 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 procedures do 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.

These algorithms can be adapted to determine an extended greatest accepting $(U_+^A)_L^G$ and an extended greatest non-accepting BSCCs $(U_+^N)_L^G$ by replacing all instances of the function $At_P(B, C, Act(C))$ with the function $At_?(B, C, Act(C))$, where $At_?(B, C, Act(C))$ 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 sets are such

Algorithm 8 Find Extended Greatest Permanent Non-Accepting BSCC

```
1: **Input**: Product BMDP $\mathcal{B} \otimes \mathcal{A}$
2: **Output**: Extended greatest permanent non-accepting BSCC $(U_+^N)_P^G$ with corresponding policy $(\widetilde{\mu}_\Psi^{up})_\otimes$ for the states in this set
3: **Initialize**: $(U_+^N)_P^G := \emptyset$
4: 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 $a \in A(\langle Q_i, s_j \rangle)$
5: Find all SCCs of $G$ and list them in $S$
6: **for** $S_k \in S$ **do**
7:     $C_0 := \emptyset, i := 0$
8:     **repeat**
9:         $R_i := S_k \setminus \cup_{\ell=0}^{i} C_\ell;\ \ Tr_i := V \setminus R_i;\ \ (C_{i+1}, Act(R_i)) = At_P(Tr_i, R_i, Act(R_i));\ \ i = i + 1$
10:    **until** $C_i = \emptyset$ and no action is removed from $Act(R_i)$
11:    **if** $i \neq 1$ **then**
12:        Find all SCCs of $R_i$ (with the remaining actions) and add them to $S$
13:    **else**
14:        **if** $S_k$ is non-accepting **then**
15:            **if** $S_k$ does not contain Rabin accepting states **then**
16:                $(U_+^N)_P^G := (U_+^N)_P^G \cup S_k$ and save any remaining action to $\widetilde{\mu}_\Psi^{up}$ for the states in $S_k$
17:            **else**
18:                For all sets of Rabin accepting state $A_i$ 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 $A_i$. Initialize $A_{bad} = \emptyset$
19:                For all states in $S_k$, maximize the lower bound probability of $\Diamond A_1^{non}$, and remove the set of actions leading a lower bound of zero. Repeat this process for all $A_i^{non}$ and restart from $A_1$ every time a new action is removed. If a state has an empty action set, add it to $A_{bad}$, and stop the process
20:                **if** $A_{bad} = \emptyset$ **then**
21:                    $(U_+^N)_P^G := (U_+^N)_P^G \cup S_k$ and save any of the actions remaining after the maximization steps to $(\widetilde{\mu}_\Psi^{up})_\otimes$ for the states in $S_k$
22:                **else**
23:                    For all $A_i$, compute the SCCs formed by the states in $S_k \setminus A_i$ with the remaining actions before the maximization steps and add them to $S$
24:                **end if**
25:            **end if**
26:        **else**
27:            Find the set $A$ of all unmatched Rabin accepting states in $S_k$. Compute the SCCs formed by the states in $S_k \setminus A$ with the remaining actions and add them to $S$
28:        **end if**
29:    **end if**
30: **end for**
31: **return** $(U_+^N)_P^G$, $(\widetilde{\mu}_\Psi^{up})_\otimes$ for states in $(U_+^N)_P^G$
```

that $(U^A)_L^G \subseteq (U_+^A)_L^G \subseteq (WC)_L^G$ and $(U^N)_L^G \subseteq (U_+^N)_L^G \subseteq (LC)_L^G$.

GREATEST PERMANENT COMPONENTS SEARCH ALGORITHMS

Next, we present an algorithm which constructs the greatest permanent winning and losing components $(WC)_P^G$ and $(LC)_P^G$ in a product BMDP $\mathcal{B} \otimes \mathcal{A}$ once extended greatest permanent BSCCs $(U_+^A)_P^G$ and $(U_+^N)_P^G$ have 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 or losing component of $\mathcal{I} \otimes \mathcal{A}$, as discussed in the second part of the proof of Lemma 4. 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 BSCC of the "wrong" acceptance 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 (or losing) 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 (respectively, losing) set of states and sometimes a part of a winning (respectively losing) component that reaches a non permanent accepting (respectively, non-accepting) BSCC with probability one. The examples below presents situations where these scenarios can occur.

Example 3. *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 $\breve{T}(Q_1, Q_2) = \breve{T}(Q_2, Q_1) = 1$. Furthermore, $\breve{T}(Q_3, Q_1) = \breve{T}(Q_3, Q_2) = 0.5$. 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 $\breve{T}(Q_1, Q_1) = 1$, $\breve{T}(Q_2, Q_1) = \breve{T}(Q_2, Q_2) = 0$ 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 9 and Algorithm 10 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 $(U_+^A)_P^G$ and $(U_+^N)_P^G$ respectively.

We explain the main features of these algorithms: first, the greatest permanent components $\big((WC)_P^G$ in Algorithm 9, $(LC)_P^G$ in Algorithm 10$\big)$ are initialized to the extended greatest permanent BSCCs in line 3. Then, in line 5, the lower bound probability of reaching these components is maximized in the product BMDP to reveal the states which can be rendered permanent sinks with respect to $(WC)_P^G$ in Algorithm 9 and with respect to $(LC)_P^G$ in Algorithm 10, as these states yield a lower bound of 1 of reaching the components. The

Algorithm 9 Find Greatest Permanent Winning Components

```
1: Input: Product BMDP $\mathcal{B} \otimes \mathcal{A}$, extended greatest permanent accepting BSCC $(U_+^A)_P^G$, extended greatest accepting BSCCs $(U_+^A)_L^G$
2: Output: Greatest permanent winning component $(WC)_P^G$ with corresponding policy $(\widehat{\mu}_\Psi^{low})_\otimes$ for the states in this set
3: Initialize: $(WC)_P^G := (U_+^A)_P^G$, $(U^A)_?^G := (U_+^A)_L^G \setminus (U_+^A)_P^G$, $(WC)_{P,prev}^G := (WC)_P^G$
4: repeat
5:     Maximize the lower bound probability of $\Diamond(WC)_P^G$ for all states $\langle Q_i, s_j\rangle$ in $\mathcal{B} \otimes \mathcal{A}$
6:     Construct the set $L$ of all states with a lower bound equal to 1 that are not in $(WC)_P^G$
7:     for $Q \in L$ do
8:         $(WC)_P^G := (WC)_P^G \cup Q$, save the action $(\widehat{\mu}_\Psi^{low})_\otimes(Q)$ computed during maximization step
9:     end for
10:    Find the greatest accepting BSCC of $(U^A)_?^G \setminus L$ using Algorithm 7 and set $(U^A)_?^G$ to this new set of states
11:    Construct the set $N$ of all accepting BSCCs constructed in $(U^A)_?^G$ under some policy
12:    for $S_k \in N$ do
13:        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 $a \in A(\langle Q_i, s_j\rangle)$
14:        $C_0 := \emptyset, i := 0$
15:        repeat
16:            $R_i := S_k \setminus \cup_{\ell=0}^{i} C_\ell$;  $Tr_i := V \setminus (R_i \cup (WC)_P^G)$;  $(C_{i+1}, Act(R_i)) := At_P(Tr_i, R_i, Act(R_i))$;  $i := i + 1$
17:        until $C_i = \emptyset$ and no action is removed from $Act(R_i)$
18:        if $i \neq 1$ then
19:            Find the greatest accepting BSCC of $R_i$ (with remaining actions) using Algorithm 7, enumerate all accepting BSCCs constructed in this set under some policy, and add them to $N$
20:        else
21:            Find the set $A$ of all unmatched Rabin accepting states of $S_k$
22:            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
23:            if $A_{bad} = \emptyset$ then
24:                $(WC)_P^G := (WC)_P^G \cup S_k$, save corresponding actions in $(\widehat{\mu}_\Psi^{low})_\otimes$ for the states in $S$
25:                $(U^A)_?^G := (U^A)_?^G \setminus S_k$
26:            else
27:                Compute the greatest accepting BSCC of $A_{bad}$ and $S_k \setminus A_{bad}$ using Algorithm 7, enumerate all accepting BSCCs constructed in this set under some policy, and add them to $N$
28:            end if
29:        end if
30:    end for
31:    $Y := (WC)_P^G \setminus (WC)_{P,prev}^G$
32:    $(WC)_{P,prev}^G := (WC)_P^G$
33: until $Y = \emptyset$
34: return $(WC)_P^G$, $(\widehat{\mu}_\Psi^{low})_\otimes$ for states in $(WC)_P^G$
```

sink states are added to the greatest permanent components in line 8.

Next, we define the greatest potential accepting and non-accepting BSCC $(U^A)_?^G$ and $(U^N)_?^G$ of a product BMDP, which are computed by taking the set difference between the greatest BSCC and the greatest permanent BSCC. States in $(U^A)_?^G$ and $(U^N)_?^G$ are those which could engender the second type of permanent components previously discussed. If $(U^A)_?^G$ and $(U^N)_?^G$ 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 of both algorithms to update $(U^A)_?^G$ and $(U^N)_?^G$.

Then, in lines 12 to 17, for all BSCCs S which can be created in $(U^A)_?^G$ in Algorithm 9 and in $(U^N)_?^G$ in Algorithm 10, we check whether there exists a policy such that no

Algorithm 10 Find Greatest Permanent Losing Components

```
1: Input: Product BMDP $\mathcal{B}\otimes\mathcal{A}$, extended greatest permanent non-accepting BSCCs $(U^N_+)^G_P$, extended greatest non-accepting BSCCs $(U^N_+)^G_L$
2: Output: Greatest permanent losing component $(LC)^G_P$ with corresponding policy $(\breve{\mu}^{up}_\Psi)_\otimes$ for the states in this set
3: Initialize: $(LC)^G_P := (U^N_+)^G_P$, $(U^N)^G_? := (U^N_+)^G_L \setminus (U^N_+)^G_P$, $(LC)^G_{P,prev} := (LC)^G_P$
4: repeat
5:     Maximize the lower bound probability of $\Diamond(LC)^G_P$ for all states $\langle Q_i, s_j\rangle$ in $\mathcal{B}\otimes\mathcal{A}$
6:     Construct the set $L$ of all states with a lower bound equal to 1 that are not in $(LC)^G_P$
7:     for $Q \in L$ do
8:         $(LC)^G_P := (LC)^G_P \cup Q$, save the action $(\breve{\mu}^{up}_\Psi)_\otimes(Q)$ computed during maximization step
9:     end for
10:    Find the greatest non-accepting BSCC of $(U^N)^G_? \setminus L$ using Algorithm 8 and set $(U^N)^G_?$ to this new set of states
11:    Construct the set $N$ of all non-accepting BSCCs constructed in $(U^A)^G_?$ under some policy
12:    for $S \in N$ do
13:        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 $a \in A(\langle Q_i, s_j\rangle)$
14:        $C_0 := \emptyset$, $i := 0$
15:        repeat
16:            $R_i := S_k \setminus \cup^i_{\ell=0} C_\ell$;  $Tr_i := V \setminus (R_i \cup (LC)^G_P)$;  $(C_{i+1}, Act(R_i)) := At_P(Tr_i, R_i, Act(R_i))$;  $i := i+1$
17:        until $C_i = \emptyset$ and no action is removed from $Act(R_i)$
18:        if $i \neq 1$ then
19:            Find the greatest non-accepting BSCC of $R_i$ (with remaining actions) using Algorithm 8, enumerate all non-accepting BSCCs constructed in this set under some policy, and add them to $N$
20:        else
21:            if $S_k$ does not contain Rabin accepting states then
22:                $(LC)^G_P := (LC)^G_P \cup S$, save corresponding actions in $(\breve{\mu}^{up}_\Psi)_\otimes$ for the states in $S_k$
23:                $(U^N)^G_? := (U^N)^G_? \setminus S_k$
24:            else
25:                For all sets of Rabin accepting state $A_i$ with respect to pair $i$ in $S_k$, find the set $A^{non}_i$ of all states in $S_k$ which are non-accepting with respect to the same pair as $A_i$. Initialize $A_{bad} = \emptyset$
26:                For all states in $S_k$, maximize the lower bound probability of $\Diamond A^{non}_1$, and remove the set of actions leading a lower bound of zero. Repeat this process for all $A^{non}_i$ and restart from $A_1$ every time a new action is removed. If a state has an empty action set, add it to $A_{bad}$, and stop the process
27:                if $A_{bad} = \emptyset$ then
28:                    $(LC)^G_P := (LC)^G_P \cup S_k$, save remaining actions in $(\breve{\mu}^{up}_\Psi)_\otimes$ for the states in $S_k$
29:                    $(U^N)^G_? := (U^N)^G_? \setminus S_k$
30:                else
31:                    For all $A_i$, compute the greatest non-accepting BSCC of $S_k \setminus A_i$ using Algorithm 8 and remaining actions before the maximization steps, enumerate all non-accepting BSCCs constructed in this set under some policy, and add them to $N$
32:                end if
33:            end if
34:        end if
35:    end for
36:    $Y := (LC)^G_P \setminus (LC)^G_{P,prev}$
37:    $(LC)^G_{P,prev} := (LC)^G_P$
38: until $Y = \emptyset$
39: return $(LC)^G_P$, $(\breve{\mu}^{up}_\Psi)_\otimes$ for states in $(LC)^G_P$
```

state of S can transition outside of the union of S and the current version of the greatest permanent 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 algorithms enter 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 BSCC of the opposite acceptance status from the desired one within S under any adversary.

This step is done similarly as in Algorithm 7 and Algorithm 8 by maximizing the lower bound probability of reaching the unmatched Rabin accepting states in S in 9 and the matched Rabin non-accepting states in 10, 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 a BSCC of the appropriate acceptance status for all adversaries, and the states of S are added to the greatest permanent component. This process is described in line 21 to 28 in Algorithm 9 and in line 21 to 31 in Algorithm 10.

In the case that new states were added to $(WC)_P^G$ or $(LC)_P^G$ upon execution of the reachability maximization step and the graph search, which is checked in line 31 to 33 in Algorithm 9 and line 36 to 38 of Algorithm 10, we return to the beginning of the while-loop and repeat this process with the augmented version of the greatest permanent components, as these could now allow previously discarded states to become permanently winning or losing. Otherwise, the loop is exited and the algorithms return the true sets $(WC)_P^G$ and $(LC)_P^G$ with their associated control actions.

A slight modification of Algorithm 9 and Algorithm 10 can be employed to compute the greatest sets $(WC)_L^G$ and $(LC)_L^G$ defined in Lemma 5. However, in this paper, we solely use the greatest BSCCs $(U_+^A)_L^G$ and $(U_+^N)_L^G$ as our target sets 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 6.1.1.

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 a BDMP and the DRA encoding the specification of interest. In Lemma 4, we remarked that a product BMDP always possesses a greatest permanent losing component and a greatest permanent winning component. In Algorithms 7 to 10, we devise graph-based techniques for determining these components as well as the corresponding control actions for the states composing them. Finally, we show in Theorem 7 that, for the remaining states in the product BMDP, the optimal policy is found by carrying out a lower bound reachability maximization computation on the greatest permanent components.

6.1.3 Refinement of the Domain Partition

OPTIMALITY 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 (6.1) with respect to a partition of the system's continuous domain. Therefore, as each state of the BMDP abstracts the behavior of an infinite number of continuous states of (6.1), the switching policy derived in the BMDP abstraction is likely to be suboptimal when mapped onto the original system.

Here, we provide a measure of the suboptimality of the control strategy computed in a BMDP abstraction with respect to the abstracted system. We first focus on the case when the objective is to maximize the probability of satisfying specification Ψ. The value iteration algorithm used to design the policies $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ and $(\widehat{\mu}_{\Psi}^{up})_{\otimes}$ discussed in Theorem 7 and Theorem 8 provides useful information amenable to a quantitative measure of the suboptimality 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 $\widecheck{p}_\ell$ and $\widehat{p}_\ell$ respectively for action a_ℓ, this is formally stated as

$$\widecheck{p}_{\ell,max} \leq \max_{\substack{\mu \in \mathcal{U}_{\otimes}^{\mathcal{A}} \\ s.t. \\ \mu(\langle Q_j, s_i \rangle) = a_{\ell,max}}} \widecheck{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[\mu]}(\langle Q_j, s_i \rangle \models \Diamond R) \, , \tag{6.10}$$

and, for all actions $a_\ell \in A(\langle Q_j, s_i \rangle)$,

$$\widehat{p}_\ell \geq \max_{\substack{\mu \in \mathcal{U}_{\otimes}^{\mathcal{A}} \\ s.t. \\ \mu(\langle Q_j, s_i \rangle) = a_\ell}} \widehat{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[\mu]}(\langle Q_j, s_i \rangle \models \Diamond R) \, , \tag{6.11}$$

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, when the objective is to maximize the probability of satisfying a specification Ψ, 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

$$\epsilon_{\langle Q_j, s_i \rangle} = \max_{\ell \neq \ell, max} \widehat{p}_\ell - \widecheck{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[(\widehat{\mu}_{\Psi}^{low})_{\otimes}]} \left(\langle Q_j, s_i \rangle \models \Diamond (WC)_P^G \right) \; . \tag{6.12}$$

The quantity $\epsilon_{\langle Q_j, s_i \rangle}$ represents an upper bound on the maximum improvement in the probability of satisfying Ψ 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 optimal 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. 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 4. *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. Note that the suboptimality factor of $\langle Q_j, s_i\rangle$ with respect to $(\widehat{\mu}_{\Psi}^{low})_{\otimes}$ in this case is $\epsilon_{\langle Q_j,s_i\rangle} = 0.7 - 0.5 = 0.2$.

Definition 32 (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 (respectively, minimum) 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 (respectively, minimum) 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 $\langle Q_j, s_i \rangle$. When the objective is to maximize (respectively, minimize) the probability of reaching an accepting BSCC, 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 $\widehat{p}_\ell < \check{p}_k$ (respectively, $\check{p}_\ell > \widehat{p}_k$). 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 $a_k \in A(\langle Q_j, s_i \rangle)$, $k \neq \ell$, $\check{p}_\ell \geq \widehat{p}_k$ (respectively, $\widehat{p}_\ell \leq \check{p}_k$).*

Definition 33 (Optimal/Suboptimal Mode)**.** *Let $\pi = x[0]x[1]x[2] \ldots x[k]$ be any finite path of* (6.1) *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* (6.1) *over all policies of* (6.1) *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* (6.1) *over all policies of* (6.1) *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.

For the case of minimization, for all states $\langle Q_j, s_i \rangle$, the value iteration algorithm used to compute $(\breve{\mu}_{\Psi}^{up})_{\otimes}$ and $(\breve{\mu}_{\Psi}^{low})_{\otimes}$ returns an upper bound on the minimum 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 the upper bound minimizing action $a_{\ell,min} = (\breve{\mu}_{\Psi}^{up})_{\otimes}(\langle Q_j, s_i \rangle)$ at state $\langle Q_j, s_i \rangle$, and a lower bound on the minimum 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 action a_ℓ at state $\langle Q_j, s_i \rangle$ for all actions $a_\ell \in A(\langle Q_j, s_i \rangle)$, that is,

$$\widehat{p}_{\ell,min} \geq \min_{\substack{\mu \in \mathcal{U}_{\otimes}^{\mathcal{A}} \\ s.t. \\ \mu(\langle Q_j, s_i \rangle) = a_{\ell,min}}} \widehat{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[\mu]}(\langle Q_j, s_i \rangle \models \Diamond R) \ , \tag{6.13}$$

and, for all actions $a_\ell \in A(\langle Q_j, s_i \rangle)$,

$$\breve{p}_{\ell} \leq \min_{\substack{\mu \in \mathcal{U}_{\otimes}^{\mathcal{A}} \\ s.t. \\ \mu(\langle Q_j, s_i \rangle) = a_{\ell}}} \breve{\mathcal{P}}_{(\mathcal{B} \otimes \mathcal{A})[\mu]}(\langle Q_j, s_i \rangle \models \Diamond R) \ , \tag{6.14}$$

with $\Diamond R$ denoting the objective of reaching an accepting BSCC in $\mathcal{B} \otimes \mathcal{A}$. Hence, for the objective of minimizing the probability of satisfying Ψ, the suboptimality factor $\epsilon_{\langle Q_j, s_i \rangle}$ with respect to the upper bound minimizing policy $(\breve{\mu}_{\Psi}^{up})_{\otimes}$ is instead given by

$$\epsilon_{\langle Q_j, s_i \rangle} = (1 - \min_{\ell \neq \ell,min} \breve{p}_{\ell}) - \breve{\mathcal{P}}_{\mathcal{B} \otimes \mathcal{A}[(\breve{\mu}_{\Psi}^{up})_{\otimes}]}\left(\langle Q_j, s_i \rangle \models \Diamond (LC)_P^G\right) \ . \tag{6.15}$$

The bounds computed by the value iteration algorithm also allow to identify control actions which are suboptimal or optimal at given state of $\mathcal{B} \otimes \mathcal{A}$ as detailed in Definition 32.

REFINEMENT PROCEDURE

Now that a quantitative measure for the optimality 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

$$\epsilon_{max} = \max_{\langle Q_j, s_i \rangle \in (Q \times S)} \epsilon_{\langle Q_j, s_i \rangle} \tag{6.16}$$

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 $\mathcal{B} \otimes \mathcal{A}$.

Formally, as defined in Definition 26, 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 more optimal controllers with respect to the abstracted system.

The proposed refinement procedure to achieve a target precision ϵ_{thr} is inspired by our technique in Section 5.2 where refinement was conducted for the purpose of verification in an IMC. This 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 11 and is summarized as follows: first, we take as input a “best-case” product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and a “worst-case” product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$. For the case of maximization, the worst-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ is the worst-case product MC induced by the IMC $(\mathcal{B} \otimes \mathcal{A})[(\widehat{\mu}_{\Psi}^{low})_{\otimes}]$ with respect to the objective of reaching an accepting BSCC, while the best-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ is the best-case product MC induced by the IMC $(\mathcal{B} \otimes \mathcal{A})[(\widehat{\mu}_{\Psi}^{up})_{\otimes}]$. Similarly, for the case of minimization, the worst-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ is the worst-case product MC induced by the IMC $(\mathcal{B} \otimes \mathcal{A})[(\widetilde{\mu}_{\Psi}^{low})_{\otimes}]$ with respect to the objective of reaching an accepting BSCC, while the best-case product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ is the best-case product MC induced by the IMC $(\mathcal{B} \otimes \mathcal{A})[\widetilde{\mu}_{\Psi}^{up}]$. Again, the aforementioned MCs are automatically constructed when applying the value iteration algorithm used for designing the two maximizing (or minimizing) policies.

Next, for all state $\langle Q_j, s_i \rangle$ of the product BMDP $\mathcal{B} \otimes \mathcal{A}$ whose suboptimality factor is greater than the target ϵ_{thr}, we compute the probability $p_{\langle j,i \rangle \rightarrow \langle j',i' \rangle}$ 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 [62]. 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 $p_{\langle j,i \rangle \rightarrow \langle j',i' \rangle} \cdot ||T^u_{\langle j',i' \rangle} - T^{\ell}_{\langle j',i' \rangle}||_2$ is added to the score $\sigma_{j'}$ of the partition state $Q_{j'}$ on line 9, 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 $p_{\langle j,i \rangle \rightarrow \langle j',i' \rangle}$ 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 control action.

Algorithm 11 Refinement Scoring Algorithm

```
1: 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 $\epsilon_{thr}$, suboptimality factors $\epsilon_{\langle Q_j, s_i \rangle}$ for all states $\langle Q_j, s_i \rangle$ of $\mathcal{B} \otimes \mathcal{A}$
2: Output: Refinement scores $\sigma = \left[\sigma_0, \sigma_1, \ldots, \sigma_{|Q|-1}\right]$ for all states of partition $P$
3: Initialize: $\sigma = \left[\sigma_0, \sigma_1, \ldots, \sigma_{|Q|-1}\right]$ where $\sigma_i = 0$
4: In $U^?$, 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
5: 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 $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$
6: for $\langle Q_j, s_i \rangle \in \mathcal{B} \otimes \mathcal{A}$ do
7:     if $\epsilon_{\langle Q_j, s_i \rangle} \geq \epsilon_{thr}$ then
8:         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 [62]
9:         for $\langle Q_{j'}, s_{i'} \rangle \in \mathcal{B} \otimes \mathcal{A}$ such that $\langle Q_{j'}, s_{i'} \rangle \notin G$ do
10:            $\sigma_{j'} = \sigma_{j'} + p_{\langle j,i \rangle \to \langle j',i' \rangle} \cdot ||T^u_{\langle j',i' \rangle} - T^\ell_{\langle j',i' \rangle}||_2$, 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
11:            if $\langle Q_{j'}, s_{i'} \rangle \in U^?$ then
12:                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
13:                    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 control action then
14:                        $\sigma_{j''} = \sigma_{j''} + p_{\langle j,i \rangle \to \langle j',i' \rangle} \cdot ||T^u_{\langle j',i' \rangle} - T^\ell_{\langle j',i' \rangle}||_2$
15:                    end if
16:                end for
17:            end if
18:        end for
19:    end if
20: end for
```

Once a score is attributed to each state of P via Algorithm 11, 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}.

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 previously performed on the states in P. First, as discussed in the previous subsection, 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 $\langle Q_{j'}^k, s_i \rangle$ and can be eliminated in the synthesis procedure applied to P'.

Proposition 3. *Let $\mathcal{B}$ be a BMDP abstraction constructed from a partition P of the domain D of* (6.1)*, $\mathcal{A}$ 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* (6.1) *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 (6.1), $\forall x \in Q_j$, 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 (6.1) 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 $s[k] = s_i$, satisfies Ψ over all the policies of (6.1) 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 $x \in Q_{j'}^k$, $k = 0, 1 \ldots, m_j$, since $Q_{j'}^k \subseteq Q_j$, 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 7 to 10. Indeed, all actions in $A(\langle Q_j, s_i \rangle)$ which were discarded during the graph search for $(WC)_L^G$ (or $(LC)_L^G$) could not, under any policy and adversary, generate a winning (or losing) component in $\mathcal{B} \otimes \mathcal{A}$. Therefore, based on this fact, we can define the set of actions $A_{qual}(\langle Q_{j'}^k, s_i \rangle) \subseteq A(\langle Q_{j'}^k, s_i \rangle)$ 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 $(WC)_L^G$ (or $(LC)_L^G$) with respect to the partition P.

Proposition 4. *Let $\mathcal{B}$ be a BMDP abstraction constructed from a partition P of the domain D of* (6.1)*, $\mathcal{A}$ be a DRA corresponding to specification Ψ, and P' be refinement of a partition P. If state $\langle Q_j, s_i \rangle$ is not a member of $(WC)_L^G$ (respectively, $(LC)_L^G$) in the product BMDP $\mathcal{B} \otimes \mathcal{A}$ under any memoryless policy μ of $\mathcal{B} \otimes \mathcal{A}$ 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 $\pi = x[0]x[1]x[2] \ldots x[k]$, $x[k] =: x$, 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 $\mathcal{A}$, satisfies Ψ is strictly less than 1 (respectively, strictly greater than 0) for all policies of* (6.1) *choosing the mode represented by action a at state x. In particular, this statement is true for all $x \in Q_{j'}^k$, $k = 0, 1 \ldots, m_j$, where $\{Q_{j'}^k\}_{k=0}^{m_j}$, $Q_{j'}^k \in P'$, 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 $(WC)_L^G$ 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 (6.1), 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 (6.1) 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 $(LC)^G_L$. $\square$

An analogous proposition can be established with respect to the greatest BSCCs $(U^A)^G_L$ and $(U^N)^G_L$ in order to further reduce $A_{qual}(\langle Q^k_{j'}, s_i\rangle)$ for Algorithm 7 and 8 specifically.

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

Proposition 5. *Let $\mathcal{B}$ be a BMDP abstraction constructed from a partition P of the domain D of* (6.1)*, $\mathcal{A}$ be a DRA corresponding to specification Ψ, and P' be refinement of a partition P. A policy μ of $\mathcal{B}$ induced by a policy in $\mathcal{B} \otimes \mathcal{A}$ generating the greatest permanent winning component $(WC)^G_P$ (respectively, the greatest permanent losing component $(LC)^G_P$ in the case of minimization) of $\mathcal{B} \otimes \mathcal{A}$ selects an optimal mode (with the appropriate mode/action correspondence) for all $x \in Q_j$ such that $\langle Q_j, s_i\rangle \in (WC)^G_P$ (respectively, $(LC)^G_P$) 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 $\check{\mathcal{P}}(\langle Q_j, s_i\rangle \models \Diamond (WC)^G_P) = 1$ 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

$L(x[0])L(x[1])L(x[2])$ ... $L(x[k])$ produces a run $s[0]s[1]s[2]\dots s[k]$, 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 (6.1). Therefore, μ selects an optimal mode for all such x. In particular, this statement is true for all $x \in Q_{j'}^k$, $k = 0, 1 \dots, m_j$, since $Q_{j'}^k \subseteq Q_j$, proving the proposition. Symmetric arguments prove the proposition with respect to $(LC)_P^G$. $\square$

Therefore, by pruning all states which were a member of $(WC)_P^G$ or $(LC)_P^G$ in an abstraction constructed P, since an action engendering a fixed probability of reaching an accepting BSCC equal to 1 or 0 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.

This iterative approach which removes suboptimal actions at each refinement step is promising in terms of scalability compared to single gridding tools such as StocHy [9] and FAUST^2 [8] where all possible actions and states have to be considered on very fine partition grids, potentially causing intractability issues when the action space is large. Here, the action space to be analyzed is likely to shrink for a lot of states as the partition is progressively rendered finer and finer.

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 $\widehat{T}(Q_1^k, Q_2^k) = 0$ for any $Q_1^k \subseteq Q_1$ and $Q_2^k \subseteq Q_2$. Finding other structural properties which are transmitted from one partition to its refined versions will be the focus of future research.

Our specification-guided, refinement-based synthesis procedure for finite-mode systems is summarized in Algorithm 12. We assume that states selected by the scoring scheme

Algorithm 12 Controller Synthesis for Finite-mode Systems

```
1: Input: Partition $P_0$ of domain $D$ of (6.1), $\omega$-regular property $\Psi$ and corresponding
   DRA $\Psi$, target controller precision $\epsilon_{thr}$
2: Output: Maximizing (minimizing) switching policy $\widehat{\mu}_{\Psi}^{low}$ ($\widecheck{\mu}_{\Psi}^{up}$), final partition $P_{fin}$
3: Initialize: $\epsilon_{max} := 1$, $i := 0$
4: while $\epsilon_{max} > \epsilon_{thr}$ do
5:   Compute the sets $(WC)_P^G$ and $(WC)_L^G$ $\big((LC)_P^G$ and $(LC)_L^G\big)$ of the product BMDP
     $\mathcal{B} \otimes \mathcal{A}$ constructed from $P_i$ using Algorithms 7 to 10
6:   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 6.1.1
7:   Compute $\epsilon_{max}$ using (6.16)
8:   if $\epsilon_{max} > \epsilon_{thr}$ then
9:     Compute the best-case and worst-case product MC $(\mathcal{M}_u)_{\otimes}^{\mathcal{A}}$ and $(\mathcal{M}_l)_{\otimes}^{\mathcal{A}}$ as dis-
       cussed in Subsection 6.1.3
10:    Apply the scoring procedure in Algorithm 11 and refine all states above a user-
       defined threshold score to produce $P_{i+1}$
11:    Update the set of actions of all states in $P_{i+1}$ for the component search and reach-
       ability problem as discussed in Subsection 6.1.3
12:    $i := i + 1$
13:  end if
14: end while
15: return $\widehat{\mu}_{\Psi}^{low}$ ($\widecheck{\mu}_{\Psi}^{up}$), $P_{fin} := P_i$
```

are split in half along their greatest dimension. In this case, the worst-case growth of the BDMP abstraction throughout this refinement-based synthesis 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.

MONOTONICITY AND CONVERGENCE OF SYNTHESIS PROCEDURE

As pointed out in [17], 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 (6.1), 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 sets $(LC)^G_L$ and $(WC)^G_L$ to increase and the sets $(LC)^G_P$ and $(WC)^G_P$ 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 $(LC)^G_L$ or $(WC)^G_L$ in $\mathcal{B}' \otimes \mathcal{A}$ while $\langle Q_j, s_i \rangle$ does not belong to these sets in $\mathcal{B} \otimes \mathcal{A}$, and it is possible for $\langle Q_j, s_i \rangle$ to belong to $(LC)^G_P$ or $(WC)^G_P$ in $\mathcal{B} \otimes \mathcal{A}$ while $\langle Q'_j, s_i \rangle$ does not belong to these sets in $\mathcal{B}' \otimes \mathcal{A}$,

- The lower bound probabilities of reaching $(WC)^G_P$ and $(LC)^G_P$ to decrease from some states of the product BMDP for a fixed policy, and the upper bound probability of reaching $(LC)^G_L$ and $(WC)^G_L$ 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 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 4 and 5; however, the second bullet point affects the monotonicity of the value iteration algorithm of [26] 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 12, as seen in related case studies using iterative refinement [26], [17] and the case study presented further. We conjecture that the scoring and refinement procedure applied in Algorithm 12 satisfies this condition and therefore ensures convergence; however, we leave a thorough investigation and potential formal proof of these facts for future work. Modifying the value iteration algorithm in [26] to exploit all information obtained from coarser partitions and enforce monotonicity of the overall procedure is another immediate research direction.

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 (6.12), (6.15) and (6.16) corresponds to an upper bound on the potential improvement any continuous state of the system could experience in the probability of satisfying the specification 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 3. Lastly, in Algorithm 12, we presented an iterative partition refinement scheme which selectively targets certain regions of the state-space by comparing these two extreme scenarios to achieve a user-defined precision threshold. Some structural properties transmitted from coarser abstractions to refined ones are identified in Proposition 4 and 5, allowing to reduce the number of required computations after each refinement step.

6.2 Synthesis for Stochastic Systems with Continuous Set of Inputs

We next investigate stochastic systems with a continuous set of inputs $U \subseteq \mathbb{R}^{\ell}$ of the form

$$x[k+1] = \mathcal{F}(x[k], u[k], w[k]) \tag{6.17}$$

as defined in equation (3.2) in Section 3.1.

The difficulty of establishing policies aiming to maximize or minimize the probability of satisfying a temporal property in (6.17) is highly dependent on the structure of the considered system. In this dissertation, we restrict our attention to systems which are affine in input and disturbance, that is

$$x[k+1] = \mathcal{F}(x[k]) + u[k] + w[k] \; . \tag{6.18}$$

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 from the initial states of system (6.18).

Problem 2: *Given a system of the form* (6.18)*, 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 CIMC abstraction of the system is constructed and analyzed. As formally defined in Definitions 6 and 7 in Subsection 3.4.2, CIMCs differ from BMDP in that the set of available actions U of a CIMC is uncountably infinite. Then, computing an optimal policy in a CIMC abstraction translates to computing a near-optimal policy when the former is applied to the original abstracted system.

Thus, for all possible finite paths in $\mathcal{C}$, the goal is to find the input in the uncountable set U that induces 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, ruling out the possibility of using an exhaustive search.

Subproblem 2.1: *Given a system of the form* (6.18)*, a partition P of its domain D, a CIMC abstraction $\mathcal{C}$ of* (6.18) *arising from P, any initial state $Q_j \in Q$ of $\mathcal{C}$ and an ω-regular property Ψ, compute the control policies $\widecheck{\mu}_{\Psi}^{up} \in \mathcal{U}_{\mathcal{C}}$ 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.,

$$\widecheck{\mu}_{\Psi}^{up} = \arg\min_{\mu \in \mathcal{U}_{\mathcal{C}}} \widehat{\mathcal{P}}_{\mathcal{C}[\mu]}(Q_j \models \Psi) \tag{6.19}$$

$$\widehat{\mu}_{\Psi}^{low} = \arg\max_{\mu \in \mathcal{U}_{\mathcal{C}}} \widecheck{\mathcal{P}}_{\mathcal{C}[\mu]}(Q_j \models \Psi)\ . \tag{6.20}$$

As our approach relies on an analysis of finite-state abstractions, finer partitions of the domain D generally yield more optimal control policies. Therefore, partition refinement for the continuous input set case is discussed as well.

Subproblem 2.2: *Given a system of the form* (6.18) *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 optimality with respect to the objective of minimizing or maximizing the probability of satisfying Ψ in* (6.18).

We note that the results presented in the lemmas and theorems of Section 6.1 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 34 (Product Controlled Interval-valued Markov Chain)**.** *Let* $\mathcal{C} = (Q, U, \check{T}, \widehat{T}, q_0, \Pi, L)$ *be a CIMC and* $\mathcal{A} = (S, 2^{\Pi}, \delta, s_0, Acc)$ *be a DRA. The* product $\mathcal{C} \otimes \mathcal{A} = (Q \times S, U, \check{T}', \widehat{T}', q_0^{\otimes}, Acc', L')$ *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 (6.18) corresponding to different choices of input u can be viewed as being uncountably infinite, the techniques established in Section 6.1, 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 first make the following additional assumptions on (6.18) 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 7. *The partition P of the domain D of system* (6.18) *is rectangular (see Definition 11), 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 8. *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 9. *The random disturbance $w[k]$ in* (6.18) *is of the form* $w[k] = \Big[w_1[k] \; w_2[k] \; \ldots \; w_n[k]\Big]^T$, *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.*

Assumption 8 is relevant for wide classes of systems, such as mixed monotone systems as seen in Chapter 4. We remark 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 $R_{Q_j} = [\check{r}^j_1, \hat{r}^j_1] \times [\check{r}^j_2, \hat{r}^j_2] \times \ldots \times [\check{r}^j_n, \hat{r}^j_n] \supseteq \{\mathcal{F}(x) : x \in Q_j\}$ 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, $R^u_{Q_j} = [\check{r}^j_1+u_1, \hat{r}^j_1+u_1]\times[\check{r}^j_2+u_2, \hat{r}^j_2+u_2]\times\ldots\times[\check{r}^j_n+u_n, \hat{r}^j_n+u_n] \supseteq \{F(x)+u : x \in Q_j\}$ is an over-approximation of the one-step reachable set from Q_j under the state update map $\mathcal{F}(x) + u$.*

From Theorem 1 in Section 4.1, it follows that under Assumptions 7 to 9 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 5. *For system* (6.18) *under Assumptions 7 to 9, 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*

$$\widehat{T}_{Q_j \xrightarrow{u} Q_\ell} = \prod_{i=1}^{n} \int_{a^\ell_i}^{b^\ell_i} f_{w_i}(x_i - s^{j\to\ell}_{i,max})\, dx_i, \tag{6.21}$$

$$= \prod_{i=1}^{n} \bigg(F_{w_i}(b^\ell_i - s^{j\to\ell}_{i,max}) - F_{w_i}(a^\ell_i - s^{j\to\ell}_{i,max}) \bigg), \tag{6.22}$$

$$\widecheck{T}_{Q_j \xrightarrow{u} Q_\ell} = \prod_{i=1}^{n} \int_{a^\ell_i}^{b^\ell_i} f_{w_i}(x_i - s^{j\to\ell}_{i,min})\, dx_i \tag{6.23}$$

$$= \prod_{i=1}^{n} \bigg(F_{w_i}(b^\ell_i - s^{j\to\ell}_{i,min}) - F_{w_i}(a^\ell_i - s^{j\to\ell}_{i,min}) \bigg) \tag{6.24}$$

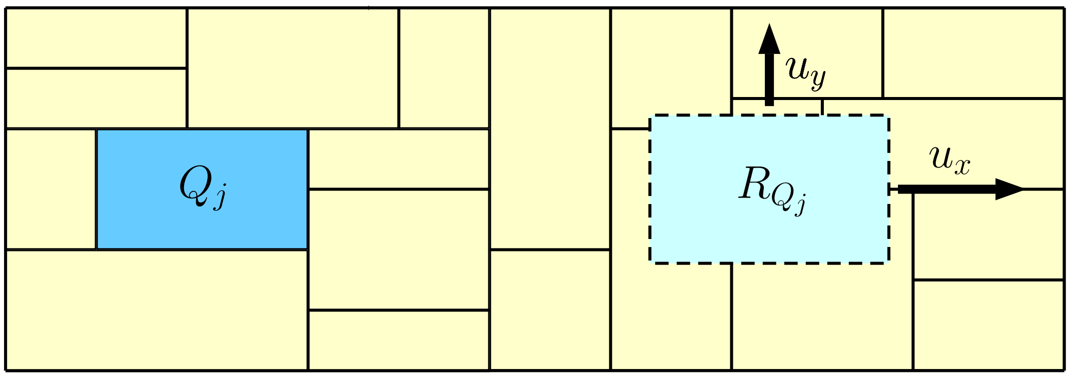


Figure 6.1: 2D depiction of the synthesis problem for system (6.18). Every state Q_j has a reachable set R_{Q_j} under $\mathcal{F}$ which is shifted when an input u is applied. The permanent component construction problem requires positioning R_{Q_j} such that all instances of noise inside R_{Q_j} ensures the satisfiability of the specification. If no input can achieve this, the lower bound reachability maximization problem amounts to finding a position for 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

$$s_{i,max}^{j\to\ell} = \begin{cases} s_{i,max}^{\ell}, & \textit{if } \; s_{i,max}^{\ell} \in [\widecheck{r}_i^j + u_i, \widehat{r}_i^j + u_i] \\ \widehat{r}_i^j + u_i, & \textit{if } \; s_{i,max}^{\ell} > \widehat{r}_i^j + u_i \\ \widecheck{r}_i^j + u_i, & \textit{if } \; s_{i,max}^{\ell} < \widecheck{r}_i^j + u_i, \end{cases} \tag{6.25}$$

$$s_{i,min}^{j\to\ell} = \begin{cases} \widecheck{r}_i^j + u_i, & \textit{if } \; s_{i,max}^{j\to\ell} > \frac{\widecheck{r}_i^j + \widehat{r}_i^j}{2} + u_i \\ \widehat{r}_i^j + u_i, & \textit{otherwise} \quad , \end{cases} \tag{6.26}$$

with $s_{i,max}^{\ell} = \frac{a_i^\ell + b_i^\ell}{2} - c_i$.

According to Remark 1, given a CIMC abstraction $\mathcal{C}$ of (6.18), for every state $\langle Q_j, s_i\rangle$ of the product CIMC $\mathcal{C} \otimes \mathcal{A}$, 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 $\langle Q_j, s_i\rangle$ (or a permanent losing component when the objective is to minimize the probability of satisfying Ψ), as illustrated in Figure 6.1.

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 components of $\mathcal{C} \otimes \mathcal{A}$ are 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.

6.2.1 Components Construction

In this subsection, we discuss the problem of generating the greatest permanent components $(WC)_P^G$ and $(LC)_P^G$ in a product CIMC $\mathcal{C} \otimes \mathcal{A}$ when $\mathcal{C}$ abstracts (6.18) under Assumptions 7 to 9, that is, the transition bounds between the states of $\mathcal{C}$ are given as in Fact 5.

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 finite interval support.

Assumption 10. *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* (6.18) *have a finite 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 $\widecheck{T}(Q_j, Q_i) > 0$,
- A transition which could be either "on" or "off" depending on the assumed transition values if $\widecheck{T}(Q_j, Q_i) = 0$ and $\widehat{T}(Q_j, Q_i) > 0$.

The connectivity properties of an IMC $\mathcal{I}$ dictate which states belong to a permanent winning and losing component or a largest winning and losing 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 6.1 in order to compute the greatest permanent components of the product CIMC $\mathcal{C} \otimes \mathcal{A}$, where $\mathcal{C}$ is the CIMC abstraction of (6.18) with domain partition P. The same procedure can be applied to find the greatest winning and losing components $(WC)_L^G$ and $(LC)_L^G$ of $\mathcal{C} \otimes \mathcal{A}$.

Fact 6. *The problem of computing the greatest permanent winning and losing components $(WC)_P^G$ and $(LC)_P^G$ as well as the greatest winning and losing components $(WC)_L^G$ and $(LC)_L^G$ 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 6.2. 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 ("on", "off" or either), formalized below as *trigger regions*.

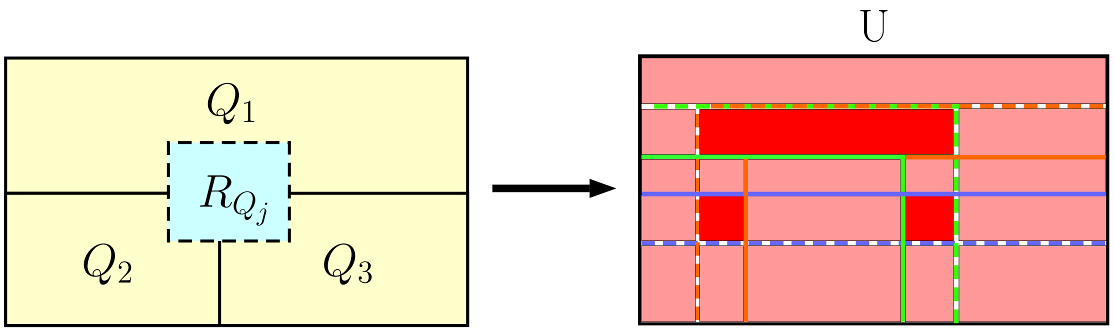


Figure 6.2: Sketch example of the component construction problem. The reachable set R_{Q_j} of state Q_j induces a partition of the input space U where each region produces a qualitatively different set of transitions. Dashed lines separate regions of U where the transition to some state is turned "on" 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.

Definition 35 (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 $U^?_{Q_j}(Q_i) \subseteq U$ is the set of inputs such that $\breve{T}(Q_j, u, Q_i) = 0$ and $\widehat{T}(Q_j, u, Q_i) > 0$, $\forall u \in U^?_{Q_j}(Q_i)$.*

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 (6.18) with Assumptions 7 to 10, 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 6. *The trigger regions of state $Q_j \in P$ with respect to state $Q_i \in P$ and input space U under dynamics* (6.18) *with partition P and satisfying Assumptions 7 to 10 are given by*

$$\begin{aligned} U^f_{Q_j}(Q_i) = \{&u \in \mathbb{R}^n : \exists k \ \widehat{r}^j_k + u_k + \widehat{w}_k \leq a^i_k \\ &or\ \widecheck{r}^j_k + u_k + \widecheck{w}_k \geq b^i_k\} \cap U \ , \end{aligned} \tag{6.27}$$

$$\begin{aligned} U^o_{Q_j}(Q_i) = \Big\{&u \in \mathbb{R}^n : \forall k \ \Big(\frac{\widehat{r}^j_k + \widecheck{r}^j_k}{2} + u_k \geq \frac{a^i_k + b^i_k}{2} - c_i \\ &and\ \widehat{r}^j_k + u_k + \widecheck{w}_k \leq b^i_k\Big) \ or \ \Big(\frac{\widehat{r}^j_k + \widecheck{r}^j_k}{2} + u_k \leq \frac{a^i_k + b^i_k}{2} - c_i \\ &and\ \widecheck{r}^j_k + u_k + \widehat{w}_k \geq a^i_k\Big)\Big\} \cap U \ , \end{aligned} \tag{6.28}$$

$$U^?_{Q_j}(Q_i) = \Big(\mathbb{R}^n \setminus (\ U^o_{Q_j}(Q_i) \cup U^f_{Q_j}(Q_i)\ )\Big) \cap U \ . \tag{6.29}$$

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

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

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

It should be noticed 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 demonstrated in the following example.

Example 5. *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 all three transitions are in the same categories in both cases, namely, undecided, 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 $\mathcal{H}^?_{Q_j}$, 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 $\mathcal{H}^?_{Q_j}$. Indeed, we observed in Example 5 that, for every overlap in the set $\mathcal{H}^?_{Q_j}$ 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 (6.18) under Assumption 7 to 10.

The input selection procedure is detailed in Algorithm 13. 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 $\mathcal{H}^?_{Q_j}$ 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 13 in its current state may select more actions than needed from the overlaps in $\mathcal{H}^?_{Q_j}$. This

Algorithm 13 Input Selection for State Q_j

```
1: Input: Sets of overlaps $\mathcal{H}^S_{Q_j}$ and $\mathcal{H}^?_{Q_j}$ 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:     $u^* := \min_{u \in \mathcal{H}_i} ||u||_2^2$
6:     $A(Q_j) \leftarrow u^*$
7: end for
8: for $\mathcal{H}_i \in \mathcal{H}^?_{Q_j}$ do
9:     $L := \emptyset, O := \emptyset, Y := \emptyset$
10:    For all states $Q_k$ such that $U^o_{Q_k} \cap \mathcal{H}_i \neq \emptyset$, $O \leftarrow Q_k$
11:    For all states $Q_k$ such that $U^?_{Q_k} \cap \mathcal{H}_i \neq \emptyset$, $Y \leftarrow Q_k$
12:    $L \leftarrow Y$
13:    for $S \in L$ do
14:        for $u \in A(Q_j)$ do
15:            Check if $\sum_{q \in O} \widehat{T}(Q_j, u, q) + \sum_{q \in Y \setminus S} \widehat{T}(Q_j, u, q) \geq 1$
16:        end for
17:        if Feasible for some $u \in A(Q_J)$ then
18:            Continue for-loop (Line 13)
19:        end if
20:        Solve $u^* = \min_{u \in \mathcal{H}_i} ||u||_2^2$ s.t. $\sum_{q \in O} \widehat{T}(Q_j, u, q) + \sum_{q \in Y \setminus S} \widehat{T}(Q_j, u, q) \geq 1$
21:        if Feasible then
22:            $A(Q_j) \leftarrow u^*$
23:        else
24:            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 $L$
25:        end if
26:    end for
27: end for
28: return $A(Q_j)$
```

is due to the fact that 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. A consequence is that 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 14 summarizes the component construction procedure and outputs the great-

```
Algorithm 14 Component Construction Method for (6.18)
1: Input: Domain Partition $P$, input Space $U$, DRA $\mathcal{A}$ of specification $\Psi$
2: Output: Greatest Permanent Components $(WC)_P^G$, $(LC)_P^G$ and $(WC)_L^G$ , $(LC)_L^G$ of
   product CIMC $\mathcal{C} \otimes \mathcal{A}$ constructed from $P$
3: Create a BMDP $\mathcal{B}$ with the same states as $P$ and with each action set $A(Q_j)$ initialized
   to the empty set
4: Compute the overlap sets for all $Q_j \in P$ using Proposition 6 and according to Definition 36
5: for $Q_j \in P$ do
6:     Compute the set of actions $A(Q_j)$ using Algorithm 13 as well as their corresponding
       transition profiles
7: end for
8: return $(WC)_P^G$, $(LC)_P^G$ $(WC)_L^G$ and $(LC)_L^G$ and their corresponding control actions
   by applying the component search in Algorithm 7, 8, 9 and 10 to $\mathcal{B} \otimes \mathcal{A}$
```

est permanent winning and losing component $(WC)_P^G$ and $(LC)_P^G$ of a product CIMC $\mathcal{C} \otimes \mathcal{A}$, as well as its greatest winning and losing component $(WC)_L^G$ and $(LC)_L^G$, where $\mathcal{C}$ serves as a CIMC abstraction of system (6.18).

6.2.2 Reachability Maximization

To devise an optimal control policy for system (6.18) 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 $(WC)_P^G$ or $(LC)_P^G$ in a product CIMC according to Theorem 7.

Our approach is inspired from the lower bound reachability maximization algorithm for BMDPs in [26]. 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 [63] 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

$$\max_{u\in U} \quad \sum_{\ell=1}^{m} p_\ell z_\ell \tag{6.30}$$

$$s.t. \quad z_\ell = \min\left\{ \widehat{T}\big(\langle Q_j, s_i\rangle, u, q_\ell\big),\ \ 1 - \sum_{k=1}^{\ell-1} z_k - \sum_{k=\ell+1}^{m} \widecheck{T}\big(\langle Q_j, s_i\rangle, u, q_k\big) \right\},$$

$$\ell = 1, 2, 3, \ldots, m\ ,$$

where the upper and lower bound terms are given by (6.21) and (6.23) for the specific case of system (6.18) under Assumption 7 to 9, rendering this problem non-convex in some instances. 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 manuscript, we tackle optimization problem (6.30) using numerical heuristics.

A more thorough analysis of this optimization problem as well as the implementation of an efficient solver, which could exploit spatial state correlation for enhanced computations and potentially ensuring global optimality, is left for future research.

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.

6.2.3 Refinement of the Domain Partition

Finally, we discuss partition refinement for system (6.18) to address Subproblem 2.2.

The optimality of the controller designed in the CIMC abstraction $\mathcal{C}$ with respect to continuous states of (6.18) can be assessed as in Section 6.1 for the finite-mode system case.

In light of Subsection 6.1.3, 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, the 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 (6.18) under the current partition P. When devising a minimizing control policy, the worst-case MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ is constructed by solving an upper bound reachability maximization problem on the greatest losing component $(LC)_L^G$ of the product CIMC $\mathcal{C} \otimes \mathcal{A}$, where $\mathcal{C}$ is the CIMC abstraction of (6.18). These upper bound reachability maximization problems are addressed using a similar procedure as in Subsection 6.2.2, with the difference that the ordering $\mathcal{O} = q_1 q_2 q_3 \dots q_m$ in the optimization program (6.30) is now descending with respect to the probability of reaching the target set G, that is $p_1 \geq p_2 \geq \dots \geq p_m$.

Propositions 3 to 5, 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 $(WC)_L^G$ (respectively, $(LC)_L^G$) 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, a lower bound on the minimum lower bound probability of reaching an accepting BSCC from$\langle Q_j, s_i \rangle$ which is greater than the upper bound produced by $(\widecheck{\mu}_{\Psi}^{up})_{\otimes}(\langle Q_j, s_i \rangle)$ 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 6.3.

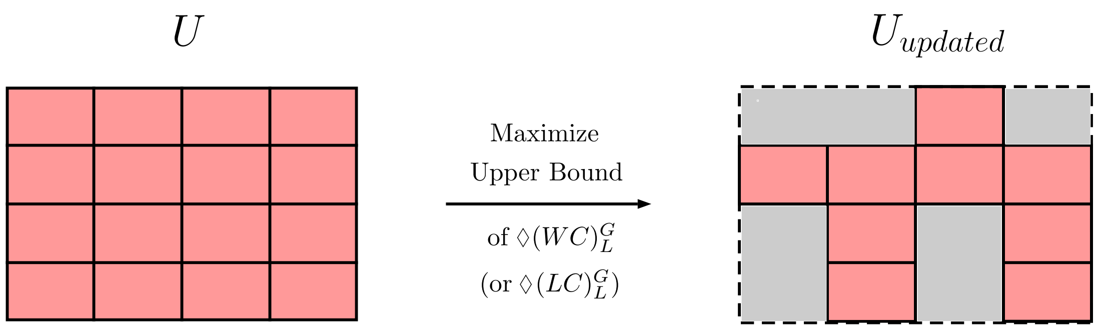


Figure 6.3: Sketch of an input space update before refinement of the domain partition. The original input space U of the considered state is gridded and the upper bound probability of reaching $(WC)_L^G$ (or $(LC)_L^G$) is maximized for all subsets of the grid. The subsets producing suboptimal bounds are shown in gray and are discarded.

Finally, once $(\mathcal{M}_{\otimes}^{\mathcal{A}})_u$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_l$ are generated and all input sets are updated, the scoring and refinement procedure are performed identical to the finite-mode case.

The controller synthesis algorithm for continuous input systems is summarized in Algorithm 15.

Future improvements of this procedure could aim to better exploit the common structures of the original CIMC abstraction and its refined versions so as to limit the state-space explosion phenomenon. For example, by saving which combinations of "off" states are achievable in the input selection scheme in Algorithm 13 for overlaps with 2 or more undecided trigger regions, one could drastically reduce the number of state combinations considered in the refined partitions and mitigate the combinatorial blowup affecting our current implementation.

Algorithm 15 Controller Synthesis for Continuous Input Systems

```
1: Input: Partition $P_0$ of domain $D$ of (6.1), $\omega$-regular property $\Psi$ and corresponding DRA $\mathcal{A}$, target controller precision $\epsilon_{thr}$
2: Output: Maximizing (minimizing) switching policy $\widehat{\mu}_{\Psi}^{low}$ ($\widetilde{\mu}_{\Psi}^{up}$), final partition $P_{fin}$
3: Initialize: $\epsilon_{max} := 1$, $i := 0$
4: while $\epsilon_{max} > \epsilon_{thr}$ do
5:    Compute the sets $(WC)_P^G$ and $(WC)_L^G$ ($(LC)_P^G$ and $(LC)_L^G$) of the product CIMC $\mathcal{C} \otimes \mathcal{A}$ constructed from $P_i$ using Algorithm 14
6:    Compute the policies $\widehat{\mu}_{\Psi}^{low}$ and $\widehat{\mu}_{\Psi}^{up}$ ($\widetilde{\mu}_{\Psi}^{up}$ and $\widetilde{\mu}_{\Psi}^{low}$) of the CIMC $\mathcal{C}$ according to Subsection 6.2.2
7:    Compute $\epsilon_{max}$ using (6.16)
8:    if $\epsilon_{max} > \epsilon_{thr}$ then
9:       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 6.2.3.
10:      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 $\mathcal{C} \otimes \mathcal{A}$
11:      for $U_n(\langle Q_j, s_m\rangle) \in U(\langle Q_j, s_m\rangle)$ do
12:         Maximize the upper bound probability of $\Diamond (WC)_L^G$ ($\Diamond (LC)_L^G$) from $\langle Q_j, s_m\rangle$ with the set of inputs $U_n(\langle Q_j, s_m\rangle)$
13:      end for
14:      Apply the scoring procedure in Algorithm 11 and refine all states in $P_i$ with a score above a user-defined threshold to produce $P_{i+1}$
15:      Update the set of inputs of all states in the product CIMC $\mathcal{C} \otimes \mathcal{A}$ constructed from $P_{i+1}$ as discussed in Subsection 6.2.3.
16:      $i := i + 1$
17:   end if
18: end while
19: return $\widehat{\mu}_{\Psi}^{low}$ ($\widetilde{\mu}_{\Psi}^{up}$), $P_{fin} := P_i$
```

CHAPTER 7
CASE STUDIES

In this chapter, we put the theoretical contributions established in Chapter 4, 5 and 6 into practical use through several case studies. In Section 7.1, we perform verification on a linear mixed monotone system with additive disturbance against two “simple” Probabilistic CTL (PCTL) specifications and compare the performance of our abstraction technique derived in Section 4.1 with previous works. In Section 7.2, verification against a PCTL specification is applied to a 3-dimensional model of a merging traffic junction, which is known to be mixed monotone, with a nonsymmetric additive disturbance. In Section 7.3, verification of a nonlinear mixed monotone system with additive disturbance is conducted for two probabilistic LTL specifications using the verification algorithm detailed in Chapter 5. In Section 7.4, we demonstrate the same verification algorithm on a stochastic polynomial system for which the abstraction method derived in Section 4.2 is applied. A synthesis example using the theory developed in Section 6.1 is presented as well. In Section 7.5, we apply the synthesis algorithms presented in Section 6.1 and Section 6.2 to a nonlinear mixed monotone system and demonstrate our refinement strategy to achieve a desired level of controller optimality. Conclusions regarding the strengths and potential improvements of our techniques are drawn from these illustrative examples.

We use Python 2.7 as our programming language for all case studies. The numerical examples shown in Section 7.1 to Section 7.4 were conducted on a OS X computer endowed with 8 GB of memory and a 3.3 GHz Intel Core i7 processor, while all computations in Section 7.5 were conducted on the Partnership for an Advanced Computing Environment (PACE) Georgia Tech cluster [64] which offered 120 GB of memory. Moreover, the examples in Subsection 7.5.1 were performed on a single core, while those in Subsection 7.5.2 were distributed over 4 cores.

7.1 Verification of Linear Mixed Monotone System with Probabilistic CTL Specifications

We investigate two case studies proposed in [26, Section VIII-A]. This section employs the IMC abstraction technique from Section 4.1, but uses the verification technique from [26] for Probabilistic CTL formulas and the naive partition refinement method in [14] to better assess the effect of the abstraction procedure on the runtimes. The refinement method in [14] systematically refines all undecided states in $Q^?$ and only looks at one-step transitions to assign a refinement score to each state in the partition instead of inspecting entire paths as done in Section 5.2. Selected states for refinement are split into two rectangles along their largest dimension to keep the new partition rectangular. In addition, the termination criterion for refinement in these examples, denoted by I_d, is the maximum size of the interval of satisfaction of all states in $Q^?$, with respect to the properties of interest.

Consider the system

$$x[k+1] = Ax[k] + w[k] \tag{7.1}$$

with $w[k] \in W$ where

$$A = \begin{bmatrix} 0.4 & 0.1 \\ 0 & 0.5 \end{bmatrix},$$
$$W = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -0.4 \\ -0.4 \end{bmatrix} \leq x \leq \begin{bmatrix} 0.4 \\ 0.4 \end{bmatrix} \right\}.$$

In addition, w is drawn from the truncated Normal distribution f_w given by

$$f_w(y) = \begin{cases} \frac{\mathcal{N}(y,0,0.09I)}{\int_W \mathcal{N}(z,0,0.09I)dz} & \text{if } y \in W \\ 0 & \text{Otherwise} \end{cases}$$

where I is the identity matrix and $\mathcal{N}(\cdot, 0, 0.09I)$ is the zero-mean Normal distribution with covariance matrix $0.09I$. We note that this system is monotone—a special case of mixed monotone systems—and the abstraction procedure derived in Section 4.1 can be applied to it. In particular, we take $g(x, y) = Ax$ as the decomposition function for $\mathcal{F}(x) = Ax$.

Our goal is to find a set of initial states satisfying some specification written as a Probabilistic CTL (PCTL) formula, and, following [26], we consider the two PCTL formulas

$$\phi_1 = \mathcal{P}_{<0.05}[\bigcirc \textbf{Obs}] \ ,$$
$$\phi_2 = \mathcal{P}_{\geq 0.90}[\neg \textbf{Obs} \ \mathcal{U} \ \textbf{Des}] \ ,$$

where $\neg$ denotes the ‘Not’ operator, $\bigcirc$ is the “Next” operator, $\mathcal{U}$ is the “Until” operator, **Obs** $\subset \mathbb{R}^2$ is the union of four rectangular “obstacle” regions, and **Des** $\subset \mathbb{R}^2$ is the union of two “destination” regions as shown in Figure 7.1. Thus, ϕ_1 states “the probability that the state of the system in the next time step is within the obstacle region is less than 0.05,” and ϕ_2 states “the probability that the system remains outside of the obstacle region until reaching the destination region is greater than or equal to 0.90”.

For both specifications, we initially perform model checking following the PCTL verification technique in [26] on an initial coarse partition P shown in Figure 7.1. The results for this step are displayed in the top plots of Figure 7.2 and Figure 7.3. Next, we execute the naive refinement algorithm of [14] on P until the interval of satisfaction for Ψ_i for all $Q^?$ states has size smaller than $I_d = 0.05$ where $\Psi_1 = \bigcirc\textbf{Obs}$ and $\Psi_2 = \neg\textbf{Obs} \ \mathcal{U} \ \textbf{Des}$ in accordance with ϕ_1 and ϕ_2; recall that $Q^?$ states are those partition regions for which we cannot conclude with certainty whether the specification is satisfied or not because the

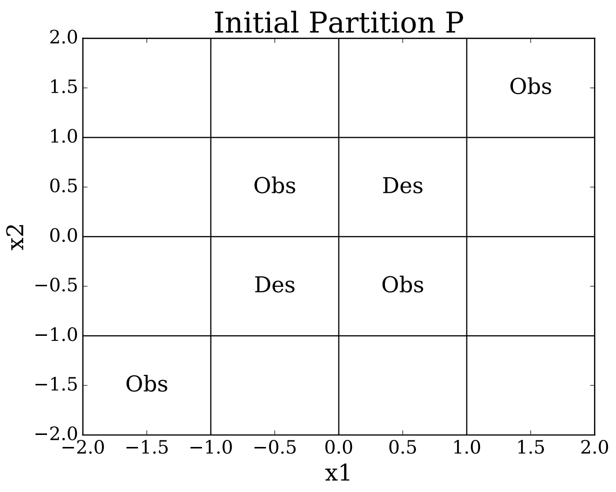


Figure 7.1: Initial partition P of the state space displaying the Obstacle (**Obs**) and Destination (**Des**) regions for the case studies in Section 7.1.

interval of satisfaction contains p_{sat} ($p_{sat} = 0.05$ in the case of ϕ_1 and $p_{sat} = 0.90$ in the case of ϕ_2).

The total computation times for the initial abstraction generation, verification and refinement all together were 1.14 and 14.3 seconds for ϕ_1 and ϕ_2 respectively. In [26], the authors employed a sampling-based technique to construct an IMC abstraction, requiring multiple expensive integral evaluations, and achieved the same level of precision in 4.8 and 51.4 hours respectively. Moreover, our refinement algorithm generated 210 states for ϕ_1 and 452 states for ϕ_2, while approximately twice as many states were produced in [26] for the same level of precision. These computational improvements were due to both a more efficient abstraction generation and a better targeted refinement.

In addition, we increase the precision of our results by a factor of 50 for the specification ϕ_1 by reducing the size of I_d to 0.001. For specification ϕ_2, we enhance the precision by a factor of 10 and choose $I_d = 0.005$. We show the final model-checked state-spaces in Figure 7.2 (Bottom) and Figure 7.3 (Bottom). The algorithm terminated in 33.15 minutes and produced 10388 states for the specification ϕ_1, while verifying against ϕ_2 took 15.5 hours to run and generated 15329 states.

We remark that our abstraction method is particularly powerful when the disturbance takes values from a compact set. Via the over-approximation of the reachable set, we can quickly check whether the affine disturbance can attain a given state or not, avoiding unnecessary integral evaluations.

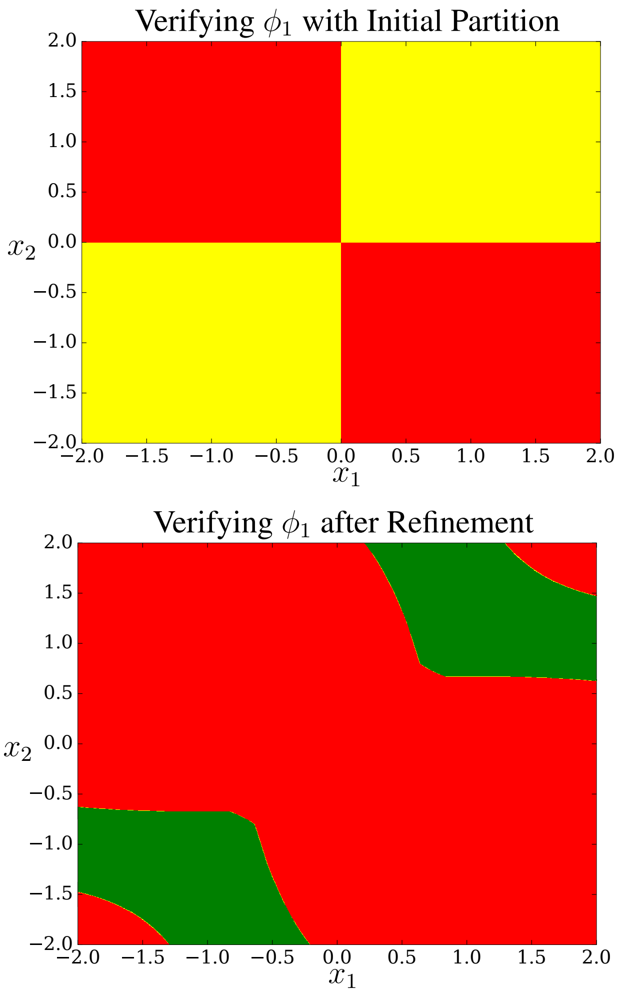


Figure 7.2: Results for specification ϕ_1 with the initial partition (Top) and the final partition after refinement when $I_d = 0.001$ (Bottom). Red states do not satisfy the specification, green states satisfy the specification, while yellow states are undecided.

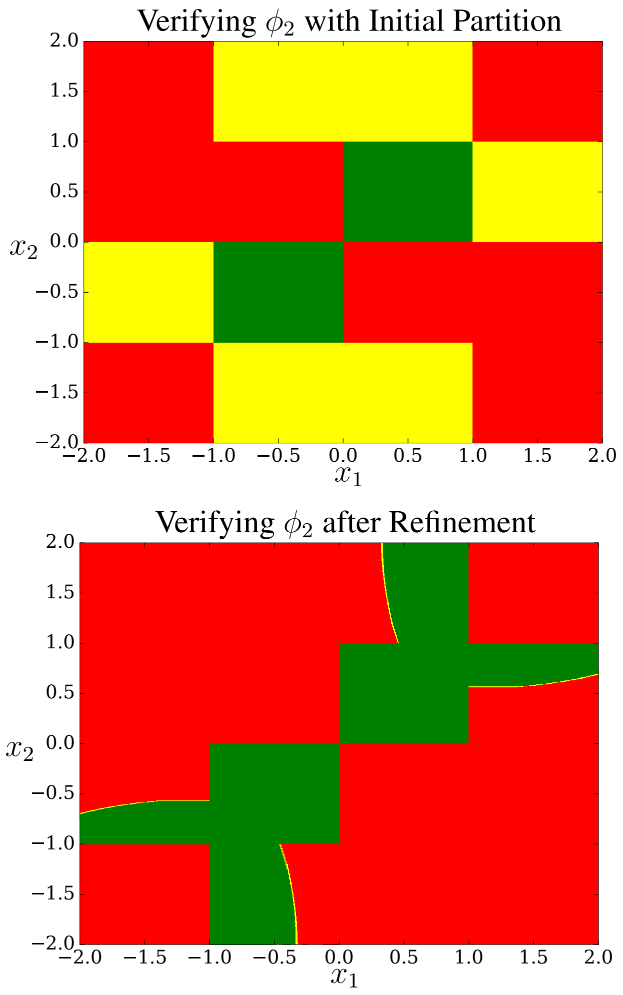


Figure 7.3: Results for specification ϕ_2 with the initial partition (Top) and the final partition after refinement when $I_d = 0.005$ (Bottom). Red states do not satisfy the specification, green states satisfy the specification, while yellow states are undecided.

7.2 Verification of Merging Traffic Junction with Nonsymmetric Disturbance

We now present a 3-dimensional case study for a model of a merging traffic junction as displayed in Figure 7.4. This example demonstrates the practical relevance of the derivations in Section 4.1. Traffic flow results in mixed monotone dynamics [65], and new vehicles entering traffic networks can readily be interpreted as affine disturbances. The following monotone discrete-time system describes the time evolution of the junction in Figure 7.4

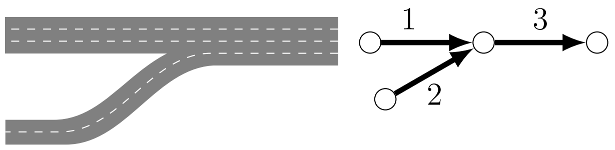


Figure 7.4: Sketch of a merging junction consisting of three links.

and is a slight modification of the model contained in [66]:

$$\begin{aligned}
x_1[k+1] &= x_1[k] - \min\left\{ D(x_1[k]), \frac{\alpha}{\beta} S(x_3[k]) \right\} + w_1 \qquad (7.2)\\
x_2[k+1] &= x_2[k] - \min\{D(x_2[k]), \bar{\alpha} S(x_3[k]), u[k]\} + w_2\\
x_3[k+1] &= x_3[k] + \min\{\beta D(x^1[k]), \alpha S(x_3[k])\}\\
&\quad + \min\{D(x_2[k]), \bar{\alpha} S(x_3[k]), u[k]\} - D(x_3[k]) - w_3
\end{aligned}$$

where $x_1[k]$, $x_2[k]$, $x_3[k]$ are the queue lengths of links 1, 2 and 3 respectively at time k; $D(x) = \min\{c, vx\}$ is a traffic demand function with c and v respectively denoting the capacity and free-flow speed; $S(x) = \bar{w}(\bar{x} - x)$ is a traffic supply function where $\bar{w}$ is a coefficient relating the available space on a given link and the supply on that link and $\bar{x}$ stands for the jam occupancy; α and $\bar{\alpha}$ denote supply weights for link 1 and 2 and respectively; β determines the fraction of vehicles leaving link 1 to enter link 3 at each time step; $u[k]$ is a parameter representing the maximum number of cars allowed to drive from link 2 to link 3 in one time step; w_1 and w_2 are disturbances corresponding to a random flow of cars entering the system through link 1 and 2 at each time step, while w_3 is a random number of cars exiting the system along link 3.

In reality, the arrival rates at Link 1 and 2, as well as the departure rate at Link 3, can only take integer values and are appropriately modeled by Poisson distributions. Although unimodal, Poisson distributions are not symmetric and the techniques developed in Theorem 1 do not directly apply. We thus choose to approximate each w_i by a unimodal, symmetric distribution v_i and use the facts highlighted in Theorem 2. We exploit the property that, for large λ, $Poisson(\lambda) \simeq \mathcal{N}(\lambda, \lambda)$. We denote by λ_i the mean arrival (or departure) rate of Link i and make the following approximations:

$$w_1 \sim Poisson(\lambda_1 = 100) \simeq v_1 \sim \mathcal{N}(100, 100)$$
$$w_2 \sim Poisson(\lambda_2 = 100) \simeq v_2 \sim \mathcal{N}(100, 100)$$
$$w_3 \sim Poisson(\lambda_3 = 60) \simeq v_3 \sim \mathcal{N}(60, 60).$$

We determine that $\delta_1 = 0.000895$, $\delta_2 = 0.000895$ and $\delta_3 = 0.0015$ satisfies $\delta_i \geq \max_{x_i \in \mathbb{R}} |v_i(x) - w_i(x)|$.
The initial partition P is shown in Figure 7.5 (Left). We aim to model-check system (7.2) against the specification

$$\phi = \mathcal{P}_{\geq 0.90}[true\, \mathcal{U}^{\leq 3}\, \textbf{Des}]$$

where **Des** $= \{x \in \mathbb{R}^3 : 0 \leq x_i < 400$ for $i = 1, 2, 3\}$ is the set of states that have all three queue lengths strictly smaller than 400. We interpret ϕ as *"What are the set of states that reach a queue length shorter than 400 for all 3 links, within 3 time steps, with probability greater than or equal to 0.90?"*.

We evaluate ϕ over the initial partition P using these approximations. The refinement strategy is the same as in the case study in Section 7.1. We stop the refinement process after the volume of the uncertain states $Q^?$ falls below 5 percent. The runtime was 15 hours and 35 minutes. The final partition is shown in Figure 7.5 (Right) and contains 16403 states. Green-colored states are certain to satisfy ϕ, red-colored stats are certain to not satisfy ϕ, and yellow-colored may or may not satisfy ϕ.

Parameter	Value
c	100
v	0.5
α	1
$\bar{\alpha}$	5
β	0.75
$\bar{x}$	800
$\bar{w}$	0.5/3
$u[k] = u$	60

Table 7.1: Parameter values for system (7.2)

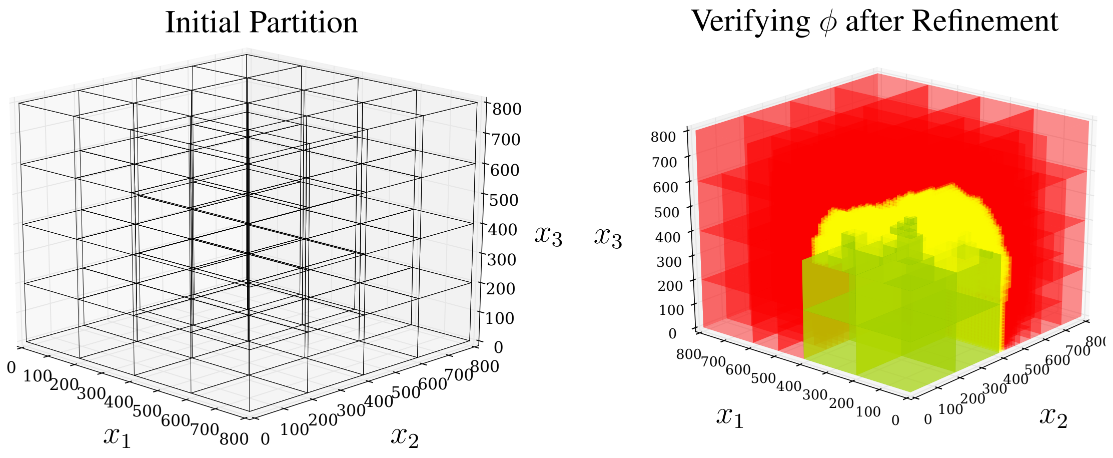


Figure 7.5: The initial partition P of the state-space for system (7.2) (Left) and the results of verification against ϕ after refinement (Right). Refinement was interrupted when the volume of $Q^?$ states reached 5 percent of the total state-space volume. Red states do not satisfy the specification, green states satisfy the specification, while yellow states are undecided.

7.3 Verification of Nonlinear Mixed Monotone System with LTL Specifications

We now apply our verification and refinement procedure presented in Chapter 5 in a case study. The code producing this case study is found at `https://github.gatech.edu/factslab/TacVerificationAlgorithm`. We consider a nonlinear, monotone bistable switch system with additive disturbance and governing equations

$$\begin{aligned} x_1[k+1] &= x_1[k] + (\ -ax_1[k] + x_2[k]\ ) \cdot \Delta T + w_1 \\ x_2[k+1] &= x_2[k] + \Big(\ \frac{(x_1[k])^2}{(x_1[k])^2+1} - bx_2[k]\ \Big) \cdot \Delta T + w_2\ , \end{aligned} \tag{7.3}$$

where we assume w_1 and w_2 to be 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 identical. 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. This assumption reflects the behavior of systems with bounded capacity where the state variables are restricted to some intervals. We choose $a = 1.3$, $b = 0.25$ and $\Delta T = 0.05$. The deterministic piece of the system has two stable equilibria at $(0, 0)$ and $(2.71, 3.52)$ and one unstable equilibrium. We seek to verify (7.3) on a domain D, with initial rectangular partition P depicted in Figure 7.6 (Top) and Figure 7.7 (Top), against the probabilistic LTL specifications

$$\phi_1 = \mathcal{P}_{\geq 0.80}[\Box((\neg A \wedge \bigcirc A) \rightarrow (\bigcirc \bigcirc A \wedge \bigcirc \bigcirc \bigcirc A))]\ , \tag{7.4}$$

$$\phi_2 = \mathcal{P}_{\leq 0.90}[(\Diamond \Box A \rightarrow \Diamond B) \wedge (\Diamond C \rightarrow \Box \neg B)]\ . \tag{7.5}$$

Specification ϕ_1 translates in natural language to "trajectories that have more than a 80% chance of remaining in an A state for at least 2 more time steps when entering an A state". Specification ϕ_2 translates to "trajectories that have less than a 90% chance of reaching a B state if it eventually always remain in A, and of always staying outside of B if it reaches

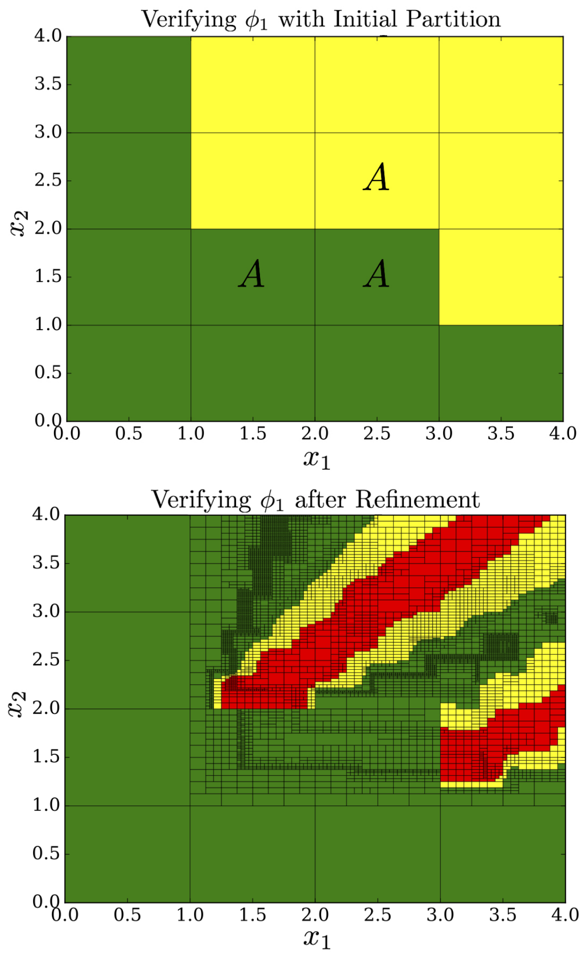
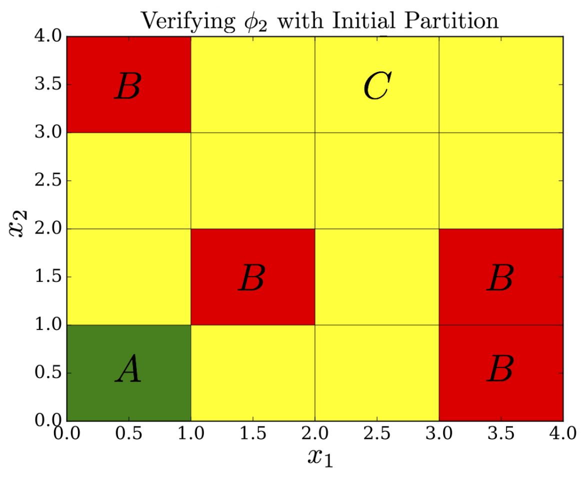


Figure 7.6: Initial verification of a partition of domain D for specification ϕ_1 (Top), and verification of final partition (Bottom). States satisfying ϕ_1 are in green, states violating ϕ_1 are in red, undecided states are yellow.

a C state". Their Rabin automaton representations contain 5 and 7 states respectively. We perform verification with stopping criterion $V_{stop} = 0.13$ for ϕ_1 and $V_{stop} = 0.1$ for ϕ_2. To construct IMC abstractions of this system, we use the technique shown in Section 4.1. Graph search is based on Section 5.1 and we compute reachability bounds applying the algorithm in [26]. Upon verification, we select states with an uncertainty score as defined in Section 5.2 that is greater than 10% of the highest score for refinement. Selected states are split into two rectangles along their largest dimension to keep the new partition rectangular.

For ϕ_1, the refinement algorithm produced 3531 states and terminated in 1h56min after 12 refinement steps. For ϕ_2, it generated 4845 states and terminated in 3h15min after 13



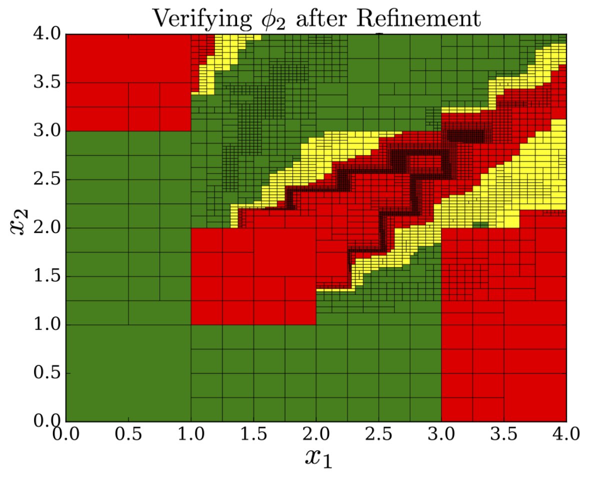


Figure 7.7: Initial verification of a partition of domain D for specification ϕ_2 (Top), and verification of final partition (Bottom). States satisfying ϕ_2 are in green, states violating ϕ_2 are in red, undecided states are yellow.

steps. The final partitions are shown in Figure 7.6 (Bottom) and Figure 7.7 (Bottom). Our new method outperforms the algorithm we propose in [14] which refines all undecided states at each refinement step: for instance, for ϕ_1, [14] achieves $V_? = 0.2137$ in 2 hours 58 min and 11 steps. Our algorithm non-uniformly refined the initial partition across the state-space. In the first example, the boundary between regions which can and cannot reach an A state are heavily targeted, as well as boundaries between regions which could keep the system in an A state for one and two time steps. In the second example, the edges of a region leading to A via B are refined the most, as this region is critical with respect to ϕ_2. Although these two examples share the same dynamics, our algorithm generates very different partitions depending on the specification. Therefore, specification-free gridding approaches as in FAUST^2 [8] are likely to perform conservatively for these examples.

7.4 Verification and Synthesis of Polynomial System with LTL Specification

The code used to generate the examples in this section is found at `https://github.com/gtfactslab/ACCBarrier`.

We consider the 2-dimensional polynomial system

$$\begin{aligned} x_1[k+1] &= 6.0x_1^3x_2 \\ x_2[k+1] &= 0.3x_1x_2 + w \ , \end{aligned} \tag{7.6}$$

with domain $D = [-0.5, 0.5] \times [-0.5, 0.5]$ and Gaussian additive noise $w \sim \mathcal{N}(\mu = 0,\ \sigma = 0.18)$. The probability of transition outside of D is negligible, thus we ignore the possibility of transitioning outside of D in order to keep the system self-contained[1]. We perform verification for these dynamics against the probabilistic specification

$$\phi = \mathcal{P}_{\geq 0.82}[\Box\neg B \wedge (\Diamond C \vee \bigcirc A \vee \bigcirc \bigcirc A)] \ ,$$

where the specification inside the probabilistic operator translates to "Never reach a B state and either eventually reach a C state or reach an A state in 2 time steps". The partition of the domain D is assumed to be as in Figure 7.8 and contains 160 states. The A states are located in $[-0.25, 0] \times [0.25, 0.5]$; the B states in $[-0.5, -0.25] \times [0.25, 0.5]$ and $[0.25, 0.5] \times [-0.5, -0.25]$; the C states in $[-0.5, -0.25] \times [0, 0.25]$ and $[0.25, 0.5] \times [-0.25, 0]$. We construct an IMC abstraction of system (7.6) using the procedure presented in Section 4.2. Given an IMC abstraction, formal techniques developed in Chapter 5 are applicable for verification with respect to ϕ. Note that no SOS barrier function can ensure a transition upper bound of exactly zero even if some states in the partition are unreachable from one other.

[1]Alternatively, a "sink" state can be used for all states outside the domain of interest D.

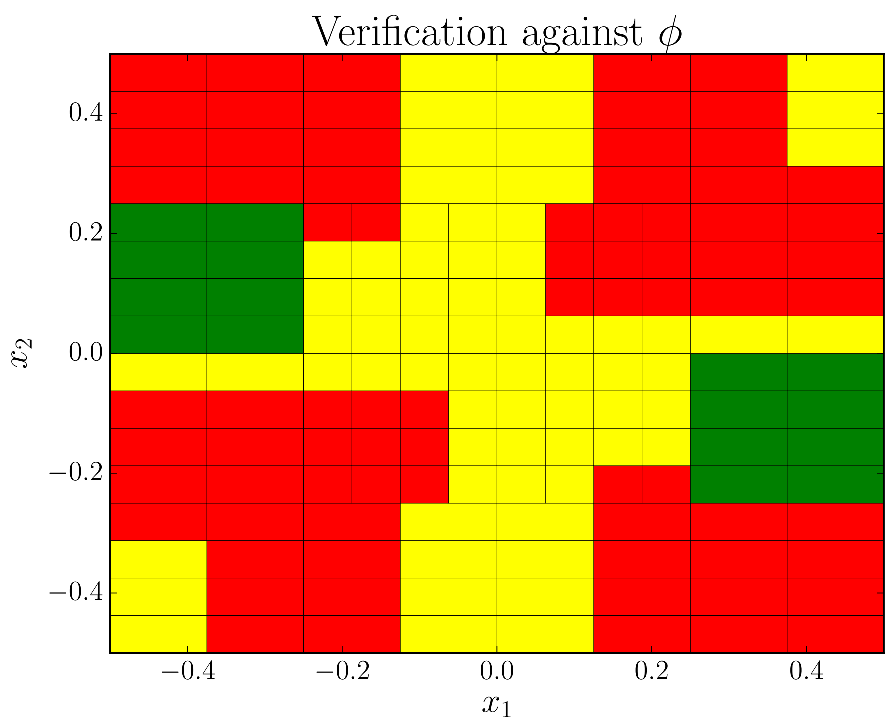


Figure 7.8: Verification of system (7.6) against specification ϕ on a 160-state partition of D. States in green satisfy ϕ, states in red violate ϕ, and states in yellow are undecided.

To address this issue, we apply a pre-processing step where states that are unreachable from one another have their upper bound transition probability set to 0. These transitions are identified by computing the range of reachable x_1 values for each discrete state, which can be done efficiently since the x_1 dynamics are locally monotone in the regions delimited by the partition, and by finding the states whose x_1 coordinates are entirely outside this range, as the disturbance only appears along the x_2 dimension. We search for SOS polynomials of degree 6 in the SOSP. To over and under-approximate the states in the domain partition with polynomial superlevel sets, we use shifted and scaled versions of 4th order polynomials approximating rectangular sets, as detailed in [15]. The result of verification is displayed in Figure 7.8. States in green satisfy ϕ, states in red violate ϕ, and states in yellow are undecided.

We now consider the two-mode system

$$
\begin{aligned}
x_1[k+1] &= a_i x_1^3 x_2 \\
x_2[k+1] &= b_i x_1 x_2 + w \ ,
\end{aligned}
\tag{7.7}
$$

for $i \in \{1, 2\}$, where $(a_1, b_1) = (6.0, 0.3)$ in the first mode, $(a_2, b_2) = (7.0, 0.2)$ in the second mode, and the domain D and noise term w are as in the verification case study.

Our goal is to find a switching policy minimizing the probability of satisfying the specification inside the probabilistic operator in ϕ. To compute a minimizing switching policy, we build a BMDP abstraction of system (7.7) by constructing an IMC abstraction for each mode using the tools from Section 4.2. Controller synthesis is performed according to Chapter 6. The partition is the same as in the previous subsection. We check our results against Monte-Carlo simulations with initial state $x_0 = [0.15, -0.2]$. The computed switching policy guarantees a probability of satisfying the specification between [0, 0.81] from x_0, which is confirmed in simulations with a probability of 0.1008.

The strength of our IMC and BMDP abstraction method for polynomials lies in its applicability to the wide class of discrete-time polynomial stochastic systems. Such abstractions allow us to perform verification and synthesis for these systems against all ω-regular specifications. On the other hand, the computational complexity of this method, which depends heavily on the hyperparameters of the SOSP, as well as the conservatism of the resulting bounds vary greatly with the dynamics of interest. As all transitions computations are parallelizable, the viability of this technique for verification and synthesis relies on the available parallel computing capabilities. For instance, building the abstraction for the verification case study on a 2-core machine took 14 hours.

7.5 Synthesis for Mixed Monotone System

We now present a numerical example to demonstrate the synthesis procedures derived in Chapter 6. The code used to generate this example is available at `https://github.`

com/gtfactslab/StochasticSynthesis.

We consider a stochastic model of a bistable switch with dynamics

$$\begin{aligned} x_1[k+1] &= x_1[k] + (\ -a x_1[k] + x_2[k]\ ) \cdot \Delta T + u_1 + w_1 \\ x_2[k+1] &= x_2[k] + \left(\ \frac{(x_1[k])^2}{(x_1[k])^2+1} - b x_2[k]\ \right) \cdot \Delta T + u_2 + w_2\ , \end{aligned} \tag{7.8}$$

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 (7.8) 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 (7.8) that maximizes the probability of satisfying the LTL specifications

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

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 eventually always remains in A, 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. 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.

7.5.1 Finite-mode Synthesis

First, we synthesize a switching policy for maximizing the probability of satisfying ϕ_1 and ϕ_2 in (7.8) using the finite set U_{fin}, where each input corresponds to one mode, and applying the synthesis Algorithm 12 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 (7.8) are known to be monotone. Therefore, BMDP abstractions of (7.8) for rectangular partitions of D are efficiently computed using the technique in [14] for each mode. The initial partition of the domain D for specification ϕ_1 is given in Figure 7.9 (Left), and the initial partition for specification ϕ_2 is in Figure 7.10 (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 12 until the set of potential accepting BSCCs $(U^A)^G_?$ 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 7.9 (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 7.10 (Right).

The cumulative execution time against the number of refinement steps is plotted in Figure 7.11 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 7.12 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 7.13 for specification ϕ_1 (Left) and specification ϕ_2 (Right).

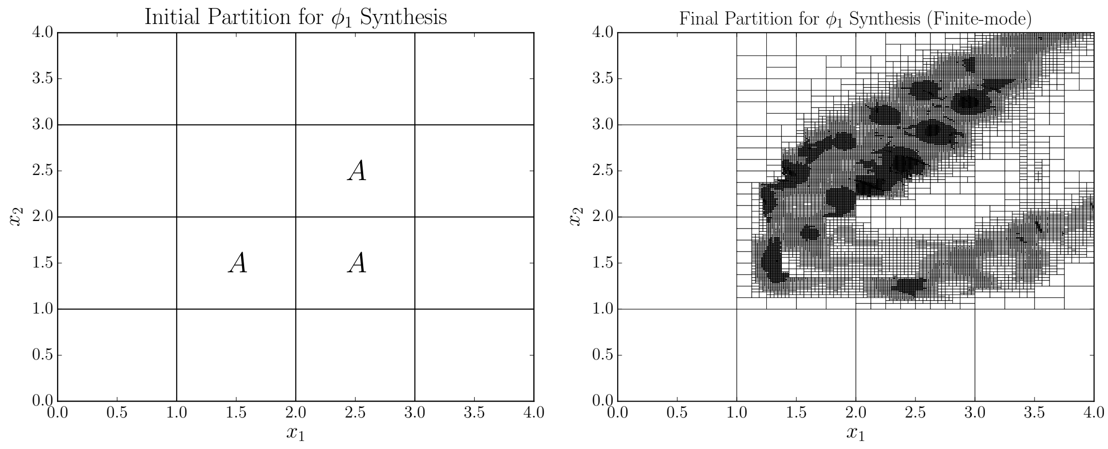
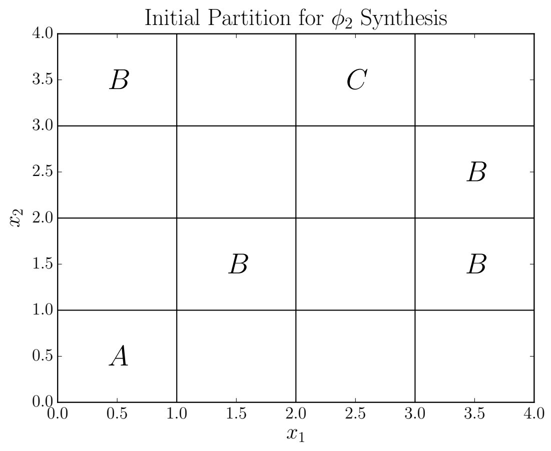


Figure 7.9: Initial domain partition with state labeling (Left) and final domain partition upon synthesis of a controller for maximizing the probability of satisfying ϕ_1 in (7.8) using the finite set of inputs U_{fin} after 18 refinement steps (Right). The final partition contains 18418 states, corresponding to 92090 states in the resulting product BMDP abstraction.



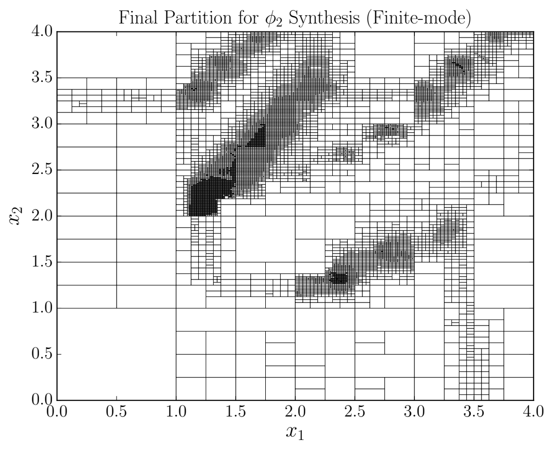
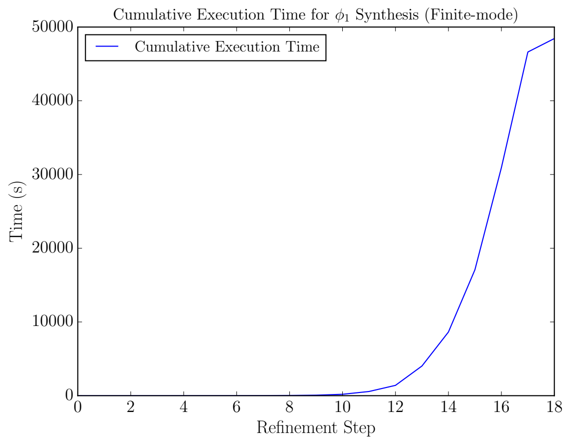


Figure 7.10: Initial domain partition with state labeling (Left) and final domain partition upon synthesis of a controller for maximizing the probability of satisfying ϕ_2 in (7.8) using the finite set of inputs U_{fin} after 15 refinement steps (Right). The final partition contains 7711 states, corresponding to 53977 states in the resulting product BMDP abstraction.



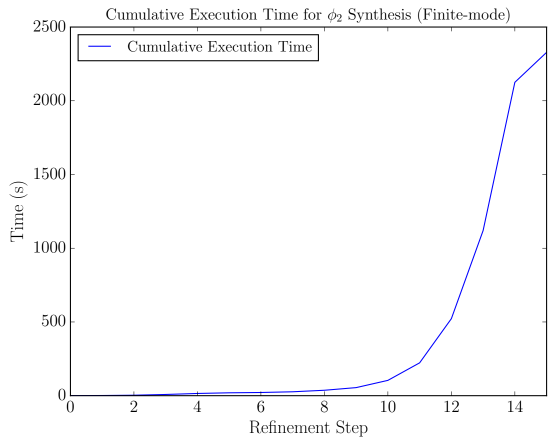


Figure 7.11: Cumulative execution time of the synthesis procedure with the finite input set U_{fin} as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right). The synthesis procedure for ϕ_1 terminated in 13 hours and 27 minutes; the synthesis procedure for ϕ_2 terminated in 38 minutes

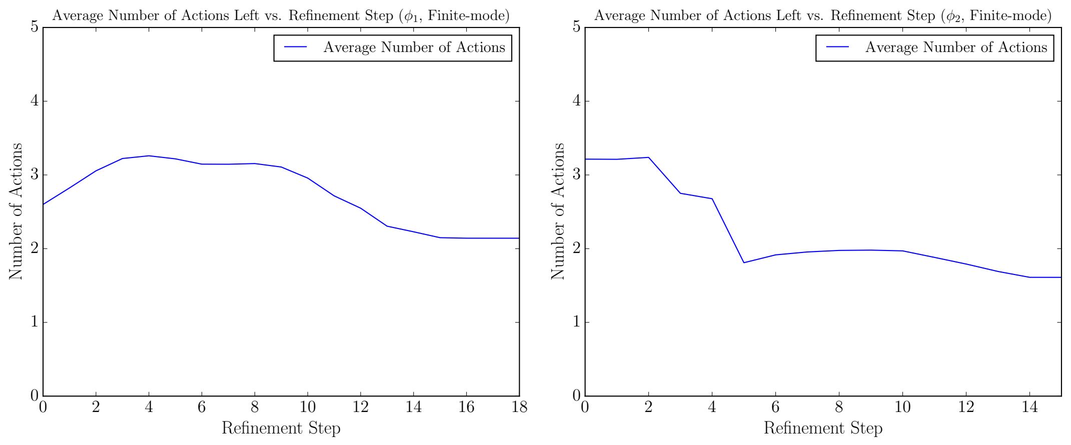


Figure 7.12: Average number of actions left at each state of the product BMDP as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right).

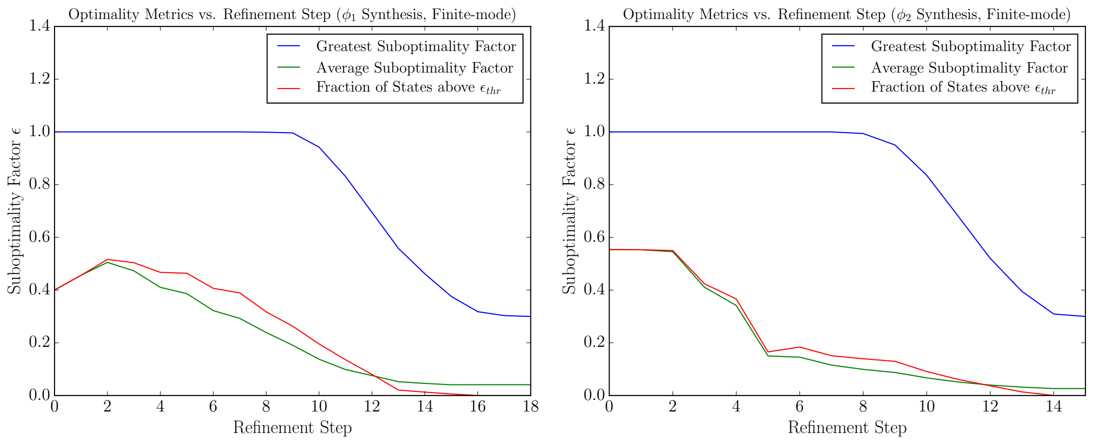


Figure 7.13: Different metrics of precision for the controller computed from the finite input set U_{fin} as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right). The synthesis algorithm reaches the target $\epsilon_{thr} = 0.30$ for both specifications. 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.

7.5.2 Continuous Input Set Synthesis

Next, we generate a control policy from the set of continuous inputs U by applying Algorithm 15. 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 (7.8) are obtained efficiently from the results in [13] 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 6.3, which facilitates the computation of the overlaps in Algorithm 14.

The possibly non-convex optimization problem in Algorithm 13, line 14, and the possibly non-convex optimization problem (6.30) 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 15 until the set of potential accepting BSCCs $(U^A)^G_?$ 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 7.14 (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 7.15 (Right).

The cumulative execution time against the number of refinement steps is plotted in Figure 7.17 for specification ϕ_1 (Left) and specification ϕ_2 (Right). The original input space for all states of the system is shown in Figure 7.16, 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 7.18 for specification ϕ_1 (Left) and specification ϕ_2 (Right).

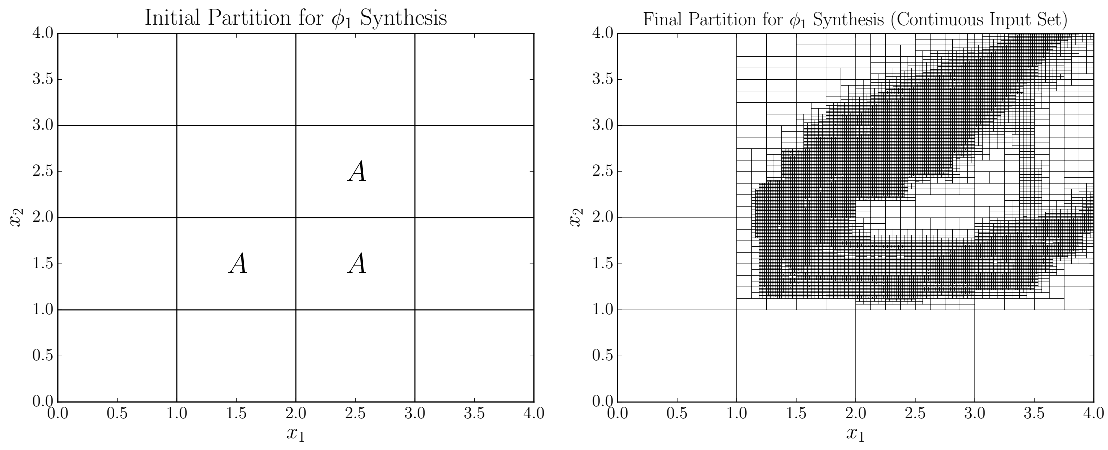

Initial Partition for ϕ_1 Synthesis
Final Partition for ϕ_1 Synthesis (Continuous Input Set)
A
A
A
x_1
x_2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

Figure 7.14: 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.

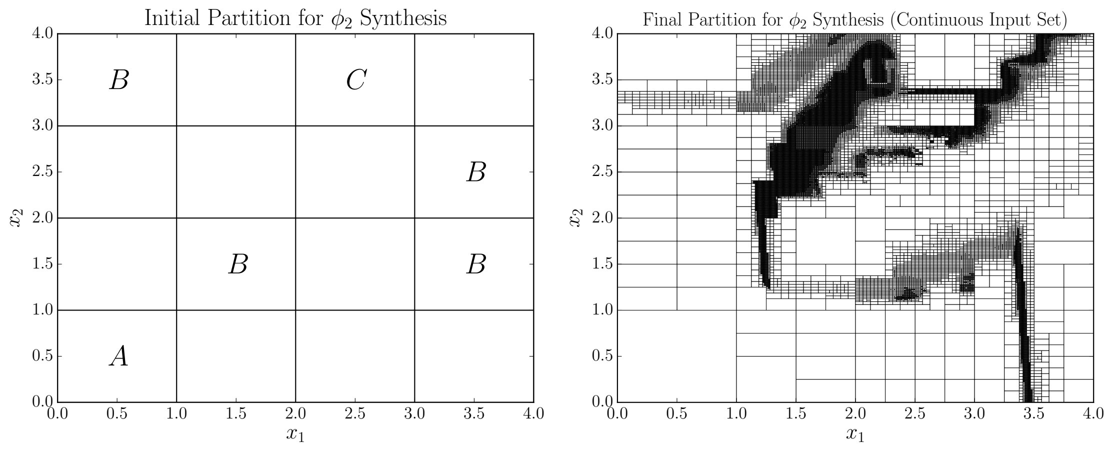


Figure 7.15: Initial domain partition with state labeling (Left) and final domain partition upon synthesis of a controller for maximizing the probability of satisfying ϕ_2 using the continuous set of inputs U after 14 refinement steps (Right). The final partition contains 24607 states, corresponding to 172249 states in the resulting product CIMC abstraction.

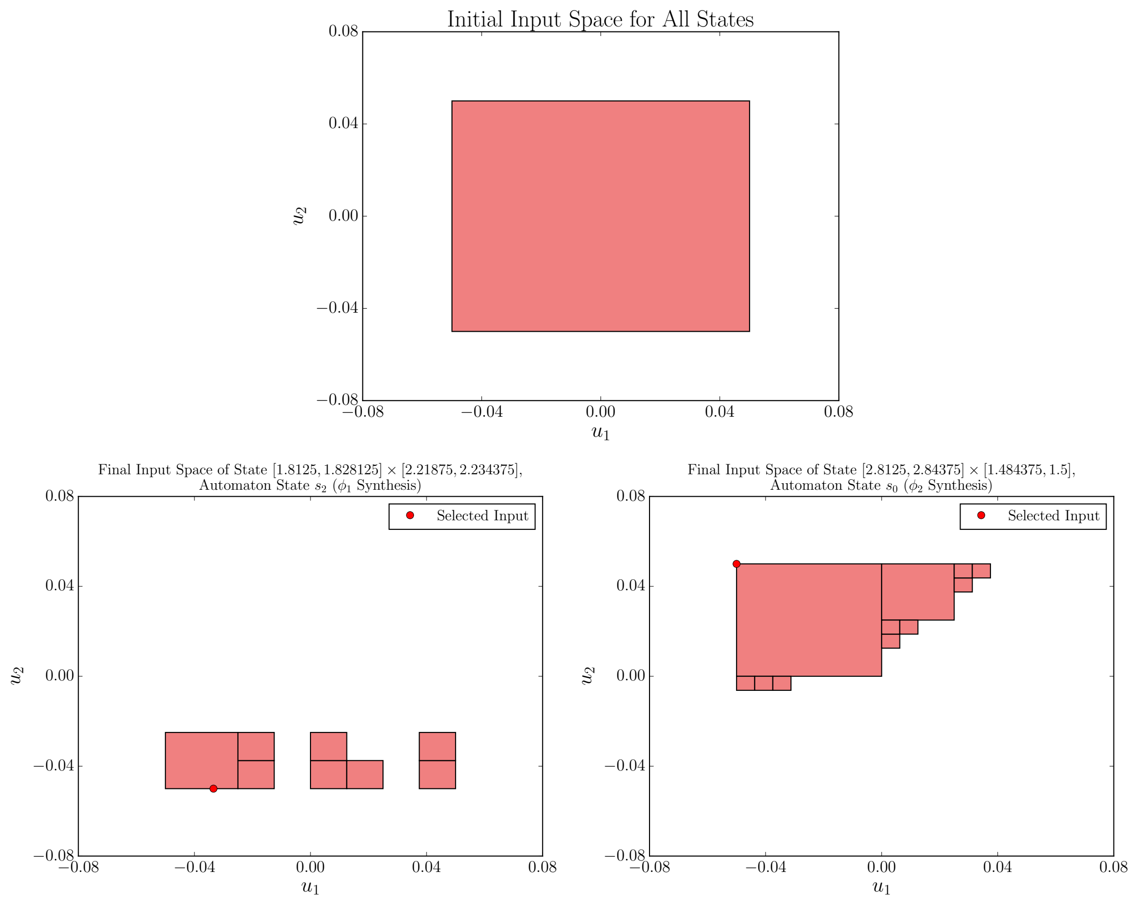
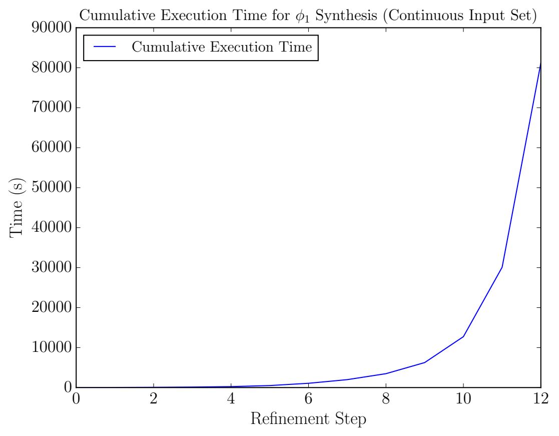


Figure 7.16: Plot of the initial input space U (Top) for all states of the state space. The reduced input space of state $[1.8125, 1.828125] \times [2.21875, 2.234375]$ with automaton state s_2 with respect to specification ϕ_1 upon refinement is shown in the bottom left plot. The reduced input space 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.



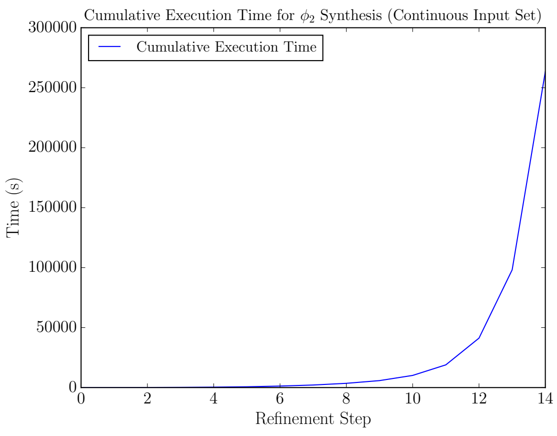
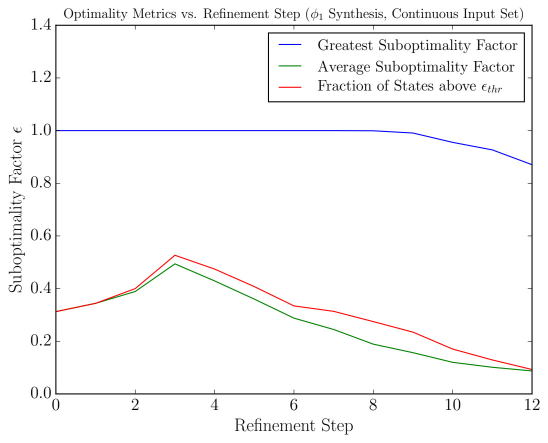


Figure 7.17: Cumulative execution time of the synthesis procedure with the continuous input set U as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right).



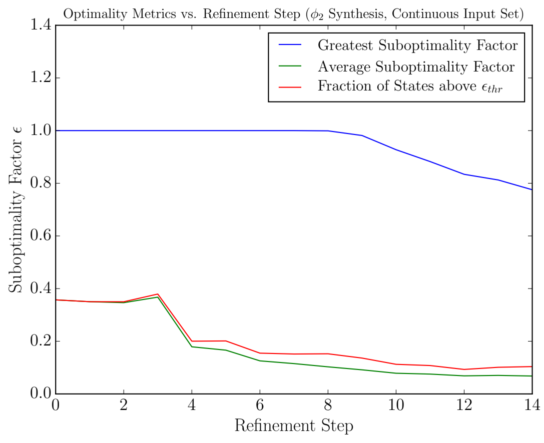


Figure 7.18: Different metrics of precision for the computed controller with the continuous input set as a function of the number of refinement steps for specification ϕ_1 (Left) and specification ϕ_2 (Right). The synthesis algorithm is manually terminated before reaching the target $\epsilon_{thr} = 0.30$ for both specifications.

7.5.3 Discussion

The synthesis algorithms presented in Chapter 6 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 [26], 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 7.19 (Top), we display the verification of system (7.8) against ϕ_1 without any available input with respect to a satisfaction threshold of 0.8 from Section 7.3, 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 (7.8) under the computed switching policy in the finite-mode section, and in the bottom right, we show the verification of system (7.8) 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.

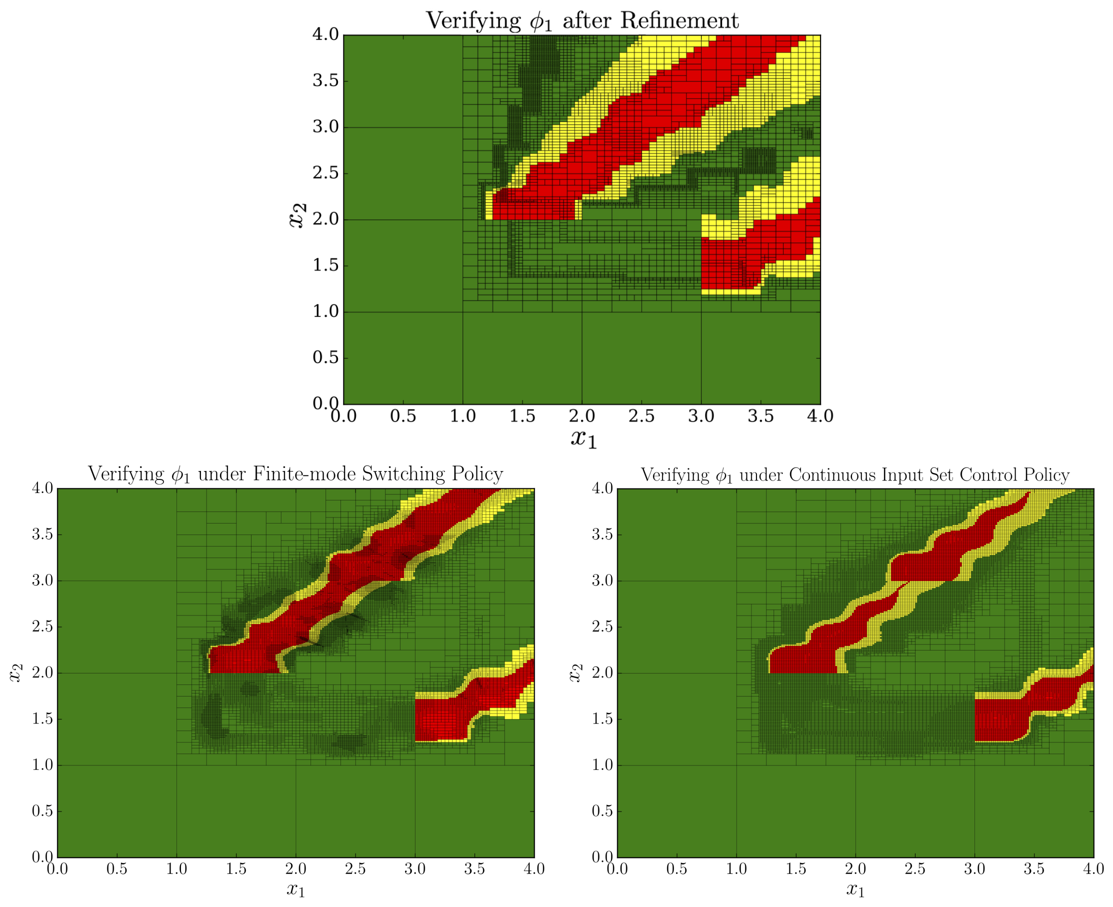


Figure 7.19: Verification of system (7.8) against ϕ_1 with respect to a satisfaction threshold of 0.8 without any input (Top), and under both the switching policy computed from the finite input set U_{fin} (Bottom Left) and the control policy computed from the continuous input space U (Bottom Right). 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 (7.8) 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.

Another observation is that a significant amount of time is spent on reaching the target optimality when very few states are still above the goal threshold, as seen in the finite-mode example. We first attribute this to the conservativeness of the scoring algorithm used in the numerical examples. Second, the value iteration scheme from [26] is reinitialized "from scratch" after each refinement step on the entire abstraction, which is inefficient as only a small subset of the state-space is refined in the later stages of the algorithm. Adapting this scheme to our targeted refinement-based method could tremendously reduce the run time of the synthesis procedure, especially when using a continuous set of inputs which incurs a computationally expensive value iteration step.

Third, the phenomenon described in Subsection 6.1.3 causing the algorithm to not be monotone is particularly prevalent when states that are reachable from one another have significantly different sizes. This situation occurs when the refinement technique selects very small and specific regions of the state-space, as it is the case when only a small number of states haven't attained the objective, causing a slowdown in the overall progress towards the target goal. All these facts should be considered in future implementations to improve the aforementioned algorithms.

Note that, in these case studies, we did not prune the product abstractions of the states which were not reachable from the initial states or had a very low probability of ever being reached. Therefore, the synthesis algorithms may attempt to find an optimal control for product states that the system will never actually reach in practice. A pre-processing step removing such states could be applied to the product constructions in order to decrease the computational complexity of the procedures.

Lastly, we impute a lot of the computation time to the code implementation itself which is still naive at this stage. In particular, a considerable amount of effort is spent on "book-keeping" when passing relevant information from coarser partitions to refined ones, and could be greatly reduced with the use of more adequate data structures tailored to this problem.

CHAPTER 8
CONCLUSIONS

The implementation of reliable formal tools tailored to the verification and control of systems experiencing random disturbances poses a unique set of challenges. This thesis provides results on the analysis of stochastic dynamical systems by means of finite-state abstractions. We study efficient techniques for abstracting large classes of stochastic systems, leverage the abstractions to conduct verification and controller synthesis for an expressive set of system properties, and propose scalable, specification-guided abstraction refinement techniques for reducing conservatism in the finite-state models.

8.1 Main Contributions

In Chapter 4, we described a procedure for constructing IMC abstractions of discrete-time, affine-in-disturbance mixed monotone systems from a rectangular partition of their domain. The discussed technique leverages mixed monotonicity to efficiently compute a rectangular over-approximation of the reachable set from every discrete state in the partition, and geometrically determines the minimum and maximum probability overlap with any state from these reachable sets under a symmetry and unimodality assumption on the disturbance. We also proposed an alternate method that relaxes the symmetry assumption on the disturbance. Furthermore, we presented an IMC abstraction algorithm for discrete-time polynomial systems from a partition of the domain. The proposed solution relies on a search of polynomial stochastic barrier functions for each pair of states of the partition to obtain one-step probabilistic guarantees. We showed that an upper and lower bound on the probability of transition between any two states can be computed by solving two SOSPs encoding the required properties of stochastic barrier functions as constraints.

In Chapter 5, we harnessed the IMC abstractions created in Chapter 4 to perform verification of discrete-time stochastic systems against probabilistic ω-regular specifications. We derived an algorithm for computing a range on the probability of satisfaction from any initial state in the IMC for any ω-regular property. The algorithm relies on an analysis of the Cartesian product between the IMC and a deterministic Rabin automaton encoding the specification. In this product, we first perform a graph search to find two sets of states known as the largest losing and winning components, and obtain the satisfaction intervals by solving a reachability problem on these components. The intervals translate to probability guarantees with respect to the abstracted continuous states. As probabilistic specifications query for the set of states whose probability of satisfying a property is below or above a fixed threshold, the verification procedure may yield a set of states which are undecided with respect to the specification of interest. In order to reduce conservatism in the IMC abstraction, we suggested a specification-guided refinement algorithm that targets the states of the domain partition which are likely to cause the most uncertainty in regard to the undecided states. The selection of states is carried out upon a quantitative and qualitative comparison of the paths generated by the best and worst-case adversary in the product IMC.

In Chapter 6, we designed an automated process for devising control strategies in stochastic controlled systems subject to ω-regular objectives. For systems with a finite number of modes, we employed BMDP abstractions constructed from a partition of the continuous system domain and presented algorithms for maximizing the lower bound probability or minimizing the upper bound probability of satisfying ω-regular specifications for any initial state in the abstraction. The propounded approach creates the greatest possible permanent winning or losing components in the Cartesian product of the BMDP abstraction with a deterministic Rabin automaton encoding the specification, and computes switching policies maximizing the lower bound probability of reaching these components. We introduced a measure of the optimality of the computed controllers with respect to the original continuous system states, and developed an iterative partition refinement strategy aimed at achieving a low suboptimality factor. We extended the theory developed for finite-mode systems to systems with a continuous set of possible inputs for which exhaustive searches cannot be performed, limiting ourselves to systems which are affine in disturbance and input. We showed that such systems are abstracted by CIMCs that are constructed from a finite partition of their domain and share similar properties with BMDPs. In particular, we demonstrated that the greatest permanent winning or losing components of the product between a CIMC and deterministic Rabin automaton can be constructed by carefully partitioning the continuous input space of the CIMC to generate a finite-mode BMDP and applying the component search algorithm to this BMDP. For the remaining states outside of these components, the lower bound probability of reaching the components is maximized by solving several optimization problems in an iterative fashion. A similar measure of the suboptimality of the designed controller was presented for this case, as well as a targeted partition refinement scheme.

In Chapter 7, we demonstrated our theoretical contributions in practical examples. We applied our verification and synthesis strategies on various classes of systems with both simple and complex specifications, and were able to highlight the strengths of our approach compared to existing techniques in terms of computational efficiency as well as to identify some of its current limitations.

8.2 Future Works

The results presented in this thesis confirm the potential of interval-valued abstractions as reliable and versatile tools for the verification and synthesis of stochastic systems subject to highly complex temporal tasks. While this dissertation proposes novel and promising ideas, many challenges are left to be overcome to render this approach universal and completely effective.

In particular, the iterative partition refinement procedures for both verification and synthesis are still in a nascent stage of development, and substantial work remains to be accomplished in this direction to achieve their full potential and attain higher levels of scalibility. Specifically, current implementations do not sufficiently leverage the common structure between a given abstraction and its refined versions. For example, by storing all computed information on a parent abstraction before refinement, one could drastically reduce the number of operations performed on the children abstractions were that information fully exploited. Capitalizing on previous computations from coarser abstractions could also help enforcing important properties such a monotonicity for faster convergence. However, such features would entail a significant modification of the graph searches and reachability algorithms currently employed in the presented verification and synthesis techniques, which consider all abstractions in isolation from one another. Other possible research directions on state-space refinement include the elaboration of more advanced heuristics accounting for additional factors such as spatial correlations or states volumes to better target the regions to be refined, and a more in-depth investigation of the refinement itself which could divide the selected states at specific locations according to the system's dynamics instead of naively splitting them in half, as done in this work. A formal analysis of the different conditions required on the system dynamics and the abstraction procedure to ensure the convergence of the refinement-based verification and synthesis algorithms should also be conducted. From a technical standpoint, it is critical that future implementations of the presented algorithms leverage the full capabilities of high-performance computing and parallelization, and also utilize optimized data structures customized to our refinement-based approach to maximize its practical potential.

Furthermore, supplemental effort has to be dedicated to the efficient computation of the interval-valued abstractions discussed in this dissertation. While abstraction techniques for mixed monotone systems appear to be mature and scalable, dependable approaches for other classes of systems remain to be developed. The barrier function-based abstraction algorithm for polynomial systems presented in this document provides reliable transition intervals between states but suffers from a prohibitive computational cost which still limits

its applicability towards high-dimensional systems. It appears that abstraction techniques exploiting rapid computations of reachable sets should be favored henceforth. Lastly, the synthesis algorithm for stochastic systems with a continuous set of inputs assumes a lot of structure on the dynamics in its current state; future works could attempt to make these assumptions less restrictive in order to extend the scope of this technique to a wider range of system models.

Appendices

APPENDIX A

LEMMAS FOR PROOF OF THEOREM 5

Lemma 6. *[40] For any infinite sequence of states $\pi = q_0 q_1 q_2 \ldots$ in a Markov Chain, there exists an index $i \geq 0$ such that q_i belongs to a BSCC.*

Corollary 2. *For any initial state $\langle Q_i, s_0\rangle$ in an induced product MC $\mathcal{M}_{\otimes}^{\mathcal{A}}$,*

$$\mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond U^A) + \mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond U^N) = 1\ . \tag{A.1}$$

Lemma 7. *For any initial state $\langle Q_i, s_0\rangle$ in an induced product Markov Chain $\mathcal{M}_{\otimes}^{\mathcal{A}}$,*

$$\mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond WC) + \mathcal{P}_{\mathcal{M}_{\otimes}^{\mathcal{A}}}(\langle Q_i, s_0\rangle \models \Diamond LC) = 1\ . \tag{A.2}$$

Proof. This lemma follows from Corollary 2 and the Definition 22 of winning and losing components. □

Lemma 8. *Let $\mathcal{I}$ be an IMC and let $\mathcal{M}_1$ and $\mathcal{M}_2$ be two MCs induced by $\mathcal{I}$ where the set B is a BSCC for both. If C_1 and C_2 are the sets of states such that $\mathcal{P}_{\mathcal{M}_1}(C_1 \models \Diamond B) = 1$ and $\mathcal{P}_{\mathcal{M}_2}(C_2 \models \Diamond B) = 1$, then there exists a MC $\mathcal{M}_3$ induced by $\mathcal{I}$ such that $\mathcal{P}_{\mathcal{M}_3}((C_1 \cup C_2) \models \Diamond B) = 1$.*

Proof. Let T_1 and T_2 denote the transition matrices of $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively, and Q denote the set of states in $\mathcal{I}$. Consider an induced MC $\mathcal{M}_3$ where $T_3(Q_i, Q_j) = T_1(Q_i, Q_j)\ \forall Q_i \in C_1$ and $\forall\ Q_j \in Q$, and $T_3(Q_i, Q_j) = T_2(Q_i, Q_j)\ \forall Q_i \in C_2 \setminus (C_1 \cap C_2)$

and $\forall\ Q_j \in Q$. By assumption, any state in C_1 reaches B with probability 1, while all states in $C_2 \setminus (C_1 \cap C_2)$ reach $B \cup (C_1 \cap C_2)$ with probability 1. Since $\mathcal{P}_{\mathcal{M}_3}((C_1 \cap C_2) \models \Diamond B) = 1$ by construction, we have $\mathcal{P}_{\mathcal{M}_3}((C_1 \cup C_2) \models \Diamond B) = 1$. $\square$

Lemma 9. *Let $\mathcal{I} \otimes \mathcal{A}$ be a product IMC and $(LC)_i$ be the losing components of any product MC $(\mathcal{M}^{\mathcal{A}}_{\otimes})_i$ induced by $\mathcal{I} \otimes \mathcal{A}$. There exists a set of product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ with losing components $(LC)_L$ and such that $(LC)_i \subseteq (LC)_L$.*

Proof. We proved in [67] that any product IMC induces a set of MCs with a largest set of non-accepting BSCCs. Lemma 9 is deduced from this fact and Lemma 8. $\square$

Lemma 10. *Let $\mathcal{I} \otimes \mathcal{A}$ be a product IMC. Let $(\mathcal{M}^{\mathcal{A}}_{\otimes})_1$ and $(\mathcal{M}^{\mathcal{A}}_{\otimes})_2$ be two product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ with sets of accepting BSCC U^A_1 and U^A_2 respectively. There exists a set of product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ with winning components $(WC)_3$ and such that $(U^A_1 \cup U^A_2) \subseteq (WC)_3$.*

Proof. Let T_1 and T_2 denote the transition matrices of $(\mathcal{M}^{\mathcal{A}}_{\otimes})_1$ and $(\mathcal{M}^{\mathcal{A}}_{\otimes})_2$ respectively, and Q denote the set of states in $\mathcal{I} \otimes \mathcal{A}$. Assume $U^A_1 \cap U^A_2 = \emptyset$. There exists a set of product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ such that $U^A_1 \cup U^A_2$ are accepting BSCCs (see [67]), and thus winning components. If $U^A_1 \cap U^A_2 \neq \emptyset$, consider the set of all product MCs $(\mathcal{M}^{\mathcal{A}}_{\otimes})_i$ induced by $\mathcal{I} \otimes \mathcal{A}$ such that, for all transition matrices T_i of the product MCs in this set, $T_i(Q_i, Q_j) = T_1(Q_i, Q_j)\ \forall Q_i \in U^A_1$ and $\forall\ Q_j \in Q$, and $T_i(Q_i, Q_j) = T_2(Q_i, Q_j)\ \forall Q_i \in U^A_2 \setminus (U^A_1 \cap U^A_2)$ and $\forall\ Q_j \in Q$. Clearly, U^A_1 is an accepting BSCC in all $(\mathcal{M}^{\mathcal{A}}_{\otimes})_i$. For all $(\mathcal{M}^{\mathcal{A}}_{\otimes})_i$, it holds that $\mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_i}((U^A_2 \setminus (U^A_1 \cap U^A_2)) \models \Diamond (U^A_1 \cap U^A_2)) = 1$, since U^A_2 is a BSCC for the same probability assignments in $(\mathcal{M}^{\mathcal{A}}_{\otimes})_2$. Thus, $U^A_1 \cup U^A_2$ are winning components with respect to all $(\mathcal{M}^{\mathcal{A}}_{\otimes})_i$. $\square$

Lemma 11. *Let $\mathcal{I} \otimes \mathcal{A}$ be a product IMC and $(WC)_i$ be the winning components of any product MC $(\mathcal{M}_{\otimes}^{\mathcal{A}})_i$ induced by $\mathcal{I} \otimes \mathcal{A}$. There exists a set of product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ with winning component $(WC)_L$ and such that $(WC)_i \subseteq (WC)_L$.*

Proof. This lemma follows from Lemmas 8 and 10. $\square$

Lemma 12. *Let $\mathcal{I} \otimes \mathcal{A}$ be a product IMC. Let $(\mathcal{M}_{\otimes}^{\mathcal{A}})_1$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_2$ be two product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ with winning components $(WC)_1$ and $(WC)_2$ respectively, and such that $(WC)_2 \subseteq (WC)_1$. Also, their losing components $(LC)_1$ and $(LC)_2$ are such that $(LC)_1 = (LC)_2 = LC$ and their respective transition matrices T_1 and T_2 satisfy $T_1(Q_i, Q_j) = T_2(Q_i, Q_j)$ $\forall Q_i \in (Q \times S) \setminus ((WC)_1 \cup LC)$ and $\forall Q_j \in (Q \times S)$. The sets of accepting BSCCs of $(\mathcal{M}_{\otimes}^{\mathcal{A}})_1$ and $(\mathcal{M}_{\otimes}^{\mathcal{A}})_2$ are denoted by U_1^A and U_2^A respectively. For any initial state $\langle Q_i, s_0 \rangle$, it holds that*

$$\mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_1}(\langle Q_i, s_0 \rangle \models \Diamond U_1^A) \geq \mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_2}(\langle Q_i, s_0 \rangle \models \Diamond U_2^A) .$$

Proof. For any initial state $\langle Q_i, s_0 \rangle \in LC$, it holds that $\mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_1}(\langle Q_i, s_0 \rangle \models \Diamond U_1^A) = \mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_2}(\langle Q_i, s_0 \rangle \models \Diamond U_2^A) = 0$. For any initial state $\langle Q_i, s_0 \rangle \in ((WC)_1 \cap (WC)_2)$, it holds that $\mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_1}(\langle Q_i, s_0 \rangle \models \Diamond U_1^A) = \mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_2}(\langle Q_i, s_0 \rangle \models \Diamond U_2^A) = 1$. For any initial state $\langle Q_i, s_0 \rangle \in ((WC)_1 \setminus (WC)_2)$, it holds that $\mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_1}(\langle Q_i, s_0 \rangle \models \Diamond U_1^A) = 1 \geq \mathcal{P}_{(\mathcal{M}_{\otimes}^{\mathcal{A}})_2}(\langle Q_i, s_0 \rangle \models \Diamond U_2^A)$. For any initial state $\langle Q_i, s_0 \rangle \in (Q \times S) \setminus ((WC)_1 \cup LC)$

(denoted by H for clarity), we have

$$
\begin{aligned}
&\mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_1}(\langle Q_i, s_0\rangle \models \Diamond U_1^A) = \\
&T_1(\langle Q_i, s_0\rangle, (WC)_1 \setminus (WC)_2) \cdot \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_1}(\, (WC)_1 \setminus (WC)_2 \models \Diamond U_1^A \,) \\
&+ T_1(\langle Q_i, s_0\rangle, (WC)_2) \cdot \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_1}(\, (WC)_2 \models \Diamond U_1^A) \\
&+ \sum_{Q_j \in H} T_1(\langle Q_i, s_0\rangle, Q_j) \cdot \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_2}(\, Q_j \models \Diamond U_2^A \,) \\
&\geq \\
&\sum_{Q_j \in (WC)_1 \setminus (WC)_2} T_2(\langle Q_i, s_0\rangle, Q_j) \cdot \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_2}(\, Q_j \models \Diamond U_2^A \,) \\
&+ T_2(\langle Q_i, s_0\rangle, (WC)_2) \cdot \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_2}(\, (WC)_2 \models \Diamond U_2^A) \\
&+ \sum_{Q_j \in H} T_2(\langle Q_i, s_0\rangle, Q_j) \cdot \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_2}(\, Q_j \models \Diamond U_2^A \,) \\
&= \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_2}(\langle Q_i, s_0\rangle \models \Diamond U_2^A)
\end{aligned}
$$

based on the transition matrices assumptions. □

Lemma 13. *Let $\mathcal{I} \otimes \mathcal{A}$ be a product IMC. Let $(\mathcal{M}^{\mathcal{A}}_{\otimes})_1$ and $(\mathcal{M}^{\mathcal{A}}_{\otimes})_2$ be two product MCs induced by $\mathcal{I} \otimes \mathcal{A}$ with losing components $(LC)_1$ and $(LC)_2$ respectively, and such that $(LC)_2 \subseteq (LC)_1$. Also, their winning components $(WC)_1$ and $(WC)_2$ are such that $(WC)_1 = (WC)_2 = WC$ and their respective transition matrices T_1 and T_2 satisfy $T_1(Q_i, Q_j) = T_2(Q_i, Q_j)$ $\forall Q_i \in (Q \times S) \setminus ((LC)_1 \cup WC)$ and $\forall Q_j \in (Q \times S)$. The sets of non-accepting BSCCs of $(\mathcal{M}^{\mathcal{A}}_{\otimes})_1$ and $(\mathcal{M}^{\mathcal{A}}_{\otimes})_2$ are denoted by U_1^N and U_2^N respectively. For any initial state $\langle Q_i, s_0\rangle$, it holds that*

$$\mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_1}(\langle Q_i, s_0\rangle \models \Diamond U_1^N) \geq \mathcal{P}_{(\mathcal{M}^{\mathcal{A}}_{\otimes})_2}(\langle Q_i, s_0\rangle \models \Diamond U_2^N)\ .$$

Proof. The proof is identical to the one of Lemma 12. □

Lemma 14. *Let $\mathcal{I} \otimes \mathcal{A}$ be a product IMC with permanent and largest sets $(WC)_P$, $(LC)_P$, $(WC)_L$ and $(LC)_L$ as previously defined. There exists a set of induced MCs of $\mathcal{I} \otimes \mathcal{A}$ whose sets of winning and losing components are $(WC)_P$ and $(LC)_L$, and a set of induced MCs whose sets of losing and winning components are $(LC)_P$ and $(WC)_L$.*

Proof. Consider the set $\mathcal{C}$ of all induced MCs of $\mathcal{I} \otimes \mathcal{A}$ whose set of losing components is $(LC)_L$. For any $\langle Q_i, s_0 \rangle \in ((WC)_? \setminus (LC)_L)$, consider an induced product MC $\mathcal{M} \in \mathcal{C}$ with transition matrix T such that $\langle Q_i, s_0 \rangle$ is not a winning component of $\mathcal{M}$. Such an induced product MC always exists by the definition of $(WC)_?$ and Lemma 8. Denote by $(WC)_?^{\mathcal{M}}$ the winning components of $\mathcal{M}$ which also belong to $(WC)_?$. There exists an induced product MC $\mathcal{M}'$ with transition matrix T' such that, for all $q_i \in (WC)_?^{\mathcal{M}}$, $\mathcal{P}_{\mathcal{M}'}(q_i \models \Diamond \neg((WC)_?^{\mathcal{M}} \cup (WC)_P)) > 0$, otherwise $q_i \in (WC)_P$, which is a contradiction. Consider the induced product MC $\mathcal{M}'' \in \mathcal{C}$ with transition matrix T'' such that $T''(q_i, q_j) = T'(q_i, q_j)$ for all $q_i \in (WC)_?^{\mathcal{M}}$ and $q_j \in (Q \times S)$, and $T'' = T$ for all other transitions. The sets of winning and losing components of $\mathcal{M}''$ are $(WC)_P$ and $(LC)_L$, proving the claim. The proof with respect to $(LC)_P$ and $(WC)_L$ is identical. $\square$

APPENDIX B
PROOF OF LEMMA 3

We provide a constructive proof for this lemma. Consider a product BMDP $\mathcal{B} \otimes \mathcal{A}$ with set of states $Q \times S$ and set of memoryless policies $(\mathcal{U})^{\mathcal{A}}_{\otimes}$. We define the greatest permanent BSCC $(U^A)^G_P \subseteq Q \times S$ as the set of all states of $\mathcal{B} \otimes \mathcal{A}$ such that, if $q \in (U)^G_P$, then there exists a policy in $(\mathcal{U})^{\mathcal{A}}_{\otimes}$ 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 policies $\mathcal{U}_{(U^A)^G_P} \subseteq (\mathcal{U})^{\mathcal{A}}_{\otimes}$ such that, under all product IMCs induced by a policy in $\mathcal{U}_{(U^A)^G_P}$, all states in $(U^A)^G_P$ belong to a permanent winning component simultaneously and, therefore, $(U^A)^G_P \subseteq (WC)^G_P$.

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, there exists a set of policies (which are a subset of $\mathcal{U}_{(U^A)^G_P}$), such that all these states are a permanent winning component simultaneously, proving the lemma.

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

First, we constructively show that, if there exists a policy $\mu_1 \in (\mathcal{U})^{\mathcal{A}}_{\otimes}$ 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 $\mu_2 \in (\mathcal{U})^{\mathcal{A}}_{\otimes}$ 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 policies in $(\mathcal{U})^{\mathcal{A}}_{\otimes}$ causing the set $B_1 \cup B_2$ to be a permanent winning component in $\mathcal{B} \otimes \mathcal{A}$. Consider a policy $\mu_3 \in (\mathcal{U})^{\mathcal{A}}_{\otimes}$ such that:

1) For all state $q \in B_1$, $\mu_3(q) = \mu_1(q)$, and for all state $q \in B_2 \setminus (B_1 \cap B_2)$, $\mu_3(q) = \mu_2(q)$,

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

By assumption, B_1 is a permanent accepting BSCC in $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$. Furthermore, because B_2 is a permanent accepting BSCC under policy μ_2, any state $q \in B_2 \setminus (B_1 \cap B_2)$ satisfies $\check{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Diamond(B_1 \cap B_2)) = 1$ under condition 1), since all states in a BSCC are reachable from one another with probability 1. Therefore, according to Definition 22, $B_1 \cup B_2$ has to belong to the permanent winning component in $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$.

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

II] Proof of existence of greatest permanent winning component

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

For a policy $\mu \in \mathcal{U}_{(U^A)_P^G}$, 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 $(U^A)_P^G$ — that is, $C \cup (U^A)_P^G = (WC)_P$ and $C \cap (U^A)_P^G = \emptyset$ — has to satisfy two conditions:

a) C does not allow a transition outside of $C \cup (U^A)_P^G$ under any adversary of $(\mathcal{B} \otimes \mathcal{A})[\mu]$, that is, $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu]}\Big(q \models \Diamond\Big((Q \times S) \setminus \big(C \cup (U^A)_P^G\big)\Big)\Big) = 0$ 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 $(U^A)^G_P$ 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 $\mu_1 \in \mathcal{U}_{(U^A)^G_P}$ 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 $C_1 \cup (U^A)^G_P = (WC^1)_P$ and $C_1 \cap (U^A)^G_P = \emptyset$, and if there exists a policy $\mu_2 \in \mathcal{U}_{(U^A)^G_P}$ 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 $C_2 \cup (U^A)^G_P = (WC^2)_P$ and $C_2 \cap (U^A)^G_P = \emptyset$, then there has to exist a policy $\mu_3 \in \mathcal{U}_{(U^A)^G_P}$ 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 $(C_1 \cup C_2) \cap (U^A)^G_P = \emptyset$. Consider a policy $\mu_3 \in \mathcal{U}_{(U^A)^G_P}$ such that:

1) For all state $q \in C_1$, $\mu_3(q) = \mu_1(q)$, and for all state $q \in C_2 \setminus (C_1 \cap C_2)$, $\mu_3(q) = \mu_2(q)$,

2) For all states $q \in (Q \times S) \setminus (C_1 \cup C_2)$, choose any action in $Act(q)$ as $\mu_3(q)$.

By construction, the set $C_1 \cup C_2$ satisfies condition b), as no subset of C_1 could form a losing component under the actions prescribed by μ_1 and no subset of $(C_2 \setminus (C_1 \cap C_2)$ could form a losing component under the actions prescribed by μ_2. Moreover, under policy μ_3, no adversary can generate a non-accepting BSCC A that has states in both C_1 and C_2, that is $A \cap C_1 \neq \emptyset$ and $A \cap C_2 \neq \emptyset$, because, by construction, for all state $q \in C_1$, $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}\Big(q \models \Diamond\big(C_2 \setminus (C_1 \cap C_2)\big)\Big) = 0$, violating the definition of a BSCC. Therefore, no adversary can generate a losing component in $C_1 \cup C_2$.

Now, for all state $q \in C_1$, $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}\Bigg(q \models \Diamond\Big((Q \times S) \setminus \big(C_1 \cup (U^A)^G_P\big)\Big)\Bigg) = 0$ by assumption, therefore $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}\Big(q \models \Diamond\Big((Q \times S) \setminus \big(C_1 \cup C_2 \cup (U^A)^G_P\big)\Big)\Big) = 0$, since $\Big((Q \times S) \setminus \big(C_1 \cup C_2 \cup (U^A)^G_P\big)\Big)$ is a subset of $\Big((Q \times S) \setminus \big(C_1 \cup (U^A)^G_P\big)\Big)$.

For all states $q \in \big(C_2 \setminus (C_1 \cap C_2)\big)$ such that $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Diamond(C_1 \cap C_2)) = 0$ (states of C_2 which cannot reach the intersection of C_1 and C_2), we have $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}\Big(q \models \Diamond\Big((Q \times S) \setminus \Big(\big(C_2 \setminus (C_1 \cap C_2)\big) \cup (U^A)^G_P\Big)\Big)\Big) = 0$ by construction, and therefore $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}\Big(q \models \Diamond\Big((Q \times S) \setminus \big(C_1 \cup C_2 \cup (U^A)^G_P\big)\Big)\Big) = 0$ since the latter set is a subset of the former.

For all states $q \in \big(C_2 \setminus (C_1 \cap C_2)\big)$ such that $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}(q \models \Diamond(C_1 \cap C_2)) > 0$ (states of C_2 which can reach the intersection of C_1 and C_2), we also have $\widehat{\mathcal{P}}_{(\mathcal{B}\otimes\mathcal{A})[\mu_3]}\Big(q \models \Diamond\Big((Q \times S) \setminus \big(C_1 \cup C_2 \cup (U^A)^G_P\big)\Big)\Big) = 0$ because this equality holds true for all states of C_1 as shown above.

Therefore, the set $C_1 \cup C_2$ satisfies conditions a) and b) and is a subset of the permanent winning component $(WC^3)_P$ of $(\mathcal{B} \otimes \mathcal{A})[\mu_3]$. Applying this process iteratively proves the existence of a set $(WC)^P_G$ satisfying the properties enunciated in the lemma and of a set of policies $\mathcal{U}_{(WC)^G_P}$ generating $(WC)^P_G$.

REFERENCES

[1] P. G. Neumann, *Computer-related risks*. Addison-Wesley Professional, 1994.

[2] T. Schlipf, T. Buechner, R. Fritz, M. Helms, and J. Koehl, “Formal verification made easy,” *IBM Journal of Research and Development*, vol. 41, no. 4.5, pp. 567–576, 1997.

[3] Y. Abarbanel-Vinov, N. Aizenbud-Reshef, I. Beer, C. Eisner, D. Geist, T. Heyman, I. Reuveni, E. Rippel, I. Shitsevalov, Y. Wolfsthal, *et al.*, “On the effective deployment of functional formal verification,” *Formal Methods in System Design*, vol. 19, no. 1, pp. 35–44, 2001.

[4] G. J. Holzmann, “Design and validation of protocols: A tutorial,” *Computer networks and ISDN systems*, vol. 25, no. 9, pp. 981–1017, 1993.

[5] W. Chan, R. J. Anderson, P. Beame, S. Burns, F. Modugno, D. Notkin, and J. D. Reese, “Model checking large software specifications,” *IEEE Transactions on software Engineering*, vol. 24, no. 7, pp. 498–520, 1998.

[6] G. Holzmann, E. Najm, and A. Serhrouchni, “Spin model checking: An introduction,” *International Journal on Software Tools for Technology Transfer (STTT)*, vol. 2, no. 4, pp. 321–327, 2000.

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

[8] S Soudjani, C. Gevaerts, and A. Abate, “Faust 2: Formal abstractions of uncountable-state stochastic processes,” *arXiv preprint arXiv:1403.3286*, 2014.

[9] N. Cauchi, K. Degiorgio, and A. Abate, “Stochy: Automated verification and synthesis of stochastic processes,” *arXiv preprint arXiv:1901.10287*, 2019.

[10] G. Gomes, R. Horowitz, A. A. Kurzhanskiy, P. Varaiya, and J. Kwon, “Behavior of the cell transmission model and effectiveness of ramp metering,” *Transportation Research Part C: Emerging Technologies*, vol. 16, no. 4, pp. 485–513, 2008.

[11] S. Coogan, M. Arcak, and A. A. Kurzhanskiy, “Mixed monotonicity of partial first-in-first-out traffic flow models,” in *IEEE Conference on Decision and Control*, 2016, pp. 7611–7616.

[12] E. D. Sontag, "Monotone and near-monotone biochemical networks," *Systems and Synthetic Biology*, vol. 1, no. 2, pp. 59–87, 2007.

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

[14] M. Dutreix and S. Coogan, "Efficient verification for stochastic mixed monotone systems," in *International Conference on Cyber-Physical Systems*, 2018.

[15] M. Dutreix, C. Santoyo, M. Abate, and S. Coogan, "Interval-valued Markov Chain Abstraction of Stochastic Systems using Barrier Functions," in *2020 American Control Conference*, IEEE, 2020.

[16] C. Santoyo, M. Dutreix, and S. Coogan, "A Barrier Function Approach to Finite-Time Stochastic System Verification and Control," *arXiv e-prints*, arXiv:1909.05109, arXiv:1909.05109, 2019. arXiv: `1909.05109`.

[17] M. Dutreix and S. Coogan, "Specification-Guided Verification and Abstraction Refinement of Mixed-Monotone Stochastic Systems," *arXiv e-prints*, arXiv:1903.02191, arXiv:1903.02191, 2019. arXiv: `1903.02191`.

[18] M. Dutreix, J. Huh, and S. Coogan, "Abstraction-based Synthesis for Stochastic Systems with Omega-Regular Objectives," 2020. arXiv: `2001.09236`.

[19] P. Florchinger, "Lyapunov-like techniques for stochastic stability," *SIAM Journal on Control and optimization*, vol. 33, no. 4, pp. 1151–1169, 1995.

[20] S. Prajna, A. Jadbabaie, and G. J. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1415–1428, 2007.

[21] J. Steinhardt and R. Tedrake, "Finite-time regional verification of stochastic nonlinear systems," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 901–923, 2012.

[22] P. Jagtap, S. Soudjani, and M. Zamani, "Temporal logic verification of stochastic systems using barrier certificates," in *International Symposium on Automated Technology for Verification and Analysis*, Springer, 2018, pp. 177–193.

[23] A. Abate, J.-P. Katoen, J. Lygeros, and M. Prandini, "Approximate model checking of stochastic hybrid systems," *European Journal of Control*, vol. 16, no. 6, pp. 624–641, 2010.

[24] A. Abate, A. D'Innocenzo, and M. D. Di Benedetto, "Approximate abstractions of stochastic hybrid systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 11, pp. 2688–2694, 2011.

[25] A. Abate, J.-P. Katoen, and A. Mereacre, "Quantitative automata model checking of autonomous stochastic hybrid systems," in *Proceedings of the 14th international conference on Hybrid systems: computation and control*, ACM, 2011, pp. 83–92.

[26] M. Lahijanian, S. B. Andersson, and C. Belta, "Formal verification and synthesis for discrete-time stochastic systems," *IEEE Transactions on Automatic Control*, vol. 60, no. 8, pp. 2031–2045, 2015.

[27] K. Y. Rozier, "Linear temporal logic symbolic model checking," *Computer Science Review*, vol. 5, no. 2, pp. 163–203, 2011.

[28] K. Chatterjee, K. Sen, and T. Henzinger, "Model-checking ω-regular properties of interval Markov chains," *Foundations of Software Science and Computational Structures*, pp. 302–317, 2008.

[29] A. Abate, A. D'Innocenzo, M. D. Di Benedetto, and 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.

[30] R. Luna, M. Lahijanian, M. Moll, and L. Kavraki, "Asymptotically optimal stochastic motion planning with temporal goals," in *Algorithmic Foundations of Robotics XI*, Springer, 2015, pp. 335–352.

[31] N. Cauchi, L. Laurenti, M. Lahijanian, A. Abate, M. Kwiatkowska, and L. Cardelli, "Efficiency through uncertainty: Scalable formal synthesis for stochastic hybrid systems," *arXiv preprint arXiv:1901.01576*, 2019.

[32] E. M. Wolff, U. Topcu, and 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.

[33] E. M. Hahn, V. Hashemi, H. Hermanns, M. Lahijanian, and A. Turrini, "Interval Markov decision processes with multiple objectives: From robust strategies to Pareto curves," *ACM Transactions on Modeling and Computer Simulation*, vol. 29, no. 4, pp. 1–31, 2019.

[34] M. Lahijanian, S. B. Andersson, and C. Belta, "A probabilistic approach for control of a stochastic system from LTL specifications," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, pp. 2236–2241.

[35] R. Majumdar, K. Mallik, and S. Soudjani, "Symbolic Controller Synthesis for Büchi Specifications on Stochastic Systems," *arXiv e-prints*, 2019. arXiv: `1910.12137`.

[36] S. Haesaert and S. Soudjani, "Robust dynamic programming for temporal logic control of stochastic systems," *arXiv preprint arXiv:1811.11445*, 2018.

[37] I. Tkachev, A. Mereacre, J.-P. Katoen, and A. Abate, "Quantitative model-checking of controlled discrete-time Markov processes," *Info. and Computation*, vol. 253, pp. 1–35, 2017.

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

[39] E. M. Clarke and E. A. Emerson, "Design and synthesis of synchronization skeletons using branching time temporal logic," in *Workshop on Logic of Programs*, Springer, 1981, pp. 52–71.

[40] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[41] S. Summers and J. Lygeros, "Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem," *Automatica*, vol. 46, no. 12, pp. 1951–1961, 2010.

[42] K. Sen, M. Viswanathan, and 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.

[43] R. Givan, S. Leach, and T. Dean, "Bounded-parameter Markov decision processes," *Artificial Intelligence*, vol. 122, no. 1-2, pp. 71–109, 2000.

[44] M. W. Hirsch, "Systems of differential equations that are competitive or cooperative II: Convergence almost everywhere," *SIAM Journal on Mathematical Analysis*, vol. 16, no. 3, pp. 423–439, 1985.

[45] H. L. Smith, *Monotone dynamical systems: An introduction to the theory of competitive and cooperative systems*. American Mathematical Society, 1995.

[46] D. Angeli and E. Sontag, "Monotone control systems," *IEEE Transactions on Automatic Control*, vol. 48, no. 10, pp. 1684–1698, 2003.

[47] E. Lovisari, G. Como, and K. Savla, "Stability of monotone dynamical flow networks," in *Proceedings of the 53rd Conference on Decision and Control*, 2014, pp. 2384–2389.

[48] H. Smith, "Global stability for mixed monotone systems," *Journal of Difference Equations and Applications*, vol. 14, no. 10-11, pp. 1159–1164, 2008.

[49] J. Nagumo, S. Arimoto, and S. Yoshizawa, "An active pulse transmission line simulating nerve axon," *Proceedings of the IRE*, vol. 50, no. 10, pp. 2061–2070, 1962.

[50] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of the atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.

[51] R. Brockett and L. Dai, "Non-holonomic kinematics and the role of elliptic functions in constructive controllability," in *Nonholonomic motion planning*, Springer, 1993, pp. 1–21.

[52] J. H. Cartwright, V. M. Eguíluz, E. Hernández-García, and O. Piro, "Dynamics of elastic excitable media," *International Journal of Bifurcation and Chaos*, vol. 9, no. 11, pp. 2197–2202, 1999.

[53] S. Chen and S. A. Billings, "Representations of non-linear systems: The narmax model," *International Journal of Control*, vol. 49, no. 3, pp. 1013–1032, 1989.

[54] G Palm, "On representation and approximation of nonlinear systems," *Biological Cybernetics*, vol. 34, no. 1, pp. 49–52, 1979.

[55] H. J. Kushner, "Stochastic stability and control," BROWN UNIV PROVIDENCE RI, Tech. Rep., 1967.

[56] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo, "SOSTOOLS: Sum of squares optimization toolbox for MATLAB," 2004.

[57] J. Klein and C. Baier, "Experiments with deterministic omega-automata for formulas of linear temporal logic," *Theoretical Computer Science*, vol. 363, no. 2, pp. 182–195, 2006.

[58] T. Babiak, F. Blahoudek, M. Křetínskỳ, and J. Strejček, "Effective translation of LTL to deterministic Rabin automata: Beyond the (F, G)-fragment," in *International Symposium on Automated Technology for Verification and Analysis*, Springer, 2013, pp. 24–39.

[59] T. Chen, T. Han, and M. Kwiatkowska, "On the complexity of model checking interval-valued discrete time Markov chains," *Info. Processing Letters*, vol. 113, no. 7, pp. 210–216, 2013.

[60] C. Baier, M. Größer, M. Leucker, B. Bollig, and F. Ciesinski, "Controller synthesis for probabilistic systems," in *Exploring New Frontiers of Theoretical Informatics*, Springer, 2004, pp. 493–506.

[61] L. De Alfaro, “Computing minimum and maximum reachability times in probabilistic systems,” in *International Conference on Concurrency Theory*, Springer, 1999, pp. 66–81.

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

[63] D. Wu and X. Koutsoukos, “Reachability analysis of uncertain systems using bounde d-parameter Markov decision processes,” *Artificial Intelligence*, vol. 172, no. 8-9, pp. 945–954, 2008.

[64] PACE, *Partnership for an Advanced Computing Environment (PACE)*, 2017.

[65] S. Coogan, E. A. Gol, M. Arcak, and C. Belta, “Traffic network control from temporal logic specifications,” *IEEE Transactions on Control of Network Systems*, vol. 3, no. 2, pp. 162–172, 2016.

[66] S. Coogan and M. Arcak, “Freeway traffic control from linear temporal logic specifications,” in *Proceedings of the 5th ACM/IEEE International Conference on Cyber-Physical Systems*, 2014, pp. 36–47.

[67] M. Dutreix and S. Coogan, “Satisfiability Bounds for ω-regular Properties in Interval-valued Markov Chains,” in *Proceedings of the 57th IEEE Conference on Decision and Control*, 2018.