Formal Abstractions for Automated Verification and Synthesis of Stochastic Systems

Sadegh Esmaeil Zadeh Soudjani

Cover illustration: The background presents the adaptive gridding approach, state-space truncation, and absorbing states. The foreground presents the concept of formal abstraction respect to the safety specification with application to energy domain.
Cover design: *Saleh Esmaeil Zadeh Soudjani*

FORMAL ABSTRACTIONS FOR AUTOMATED VERIFICATION AND SYNTHESIS OF STOCHASTIC SYSTEMS

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof.ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op

maandag 3 november 2014 om 15:00 uur

door

Sadegh ESMAEIL ZADEH SOUDJANI

Master of Science in Electrical Engineering – Control
geboren te Iran

Dit proefschrift is goedgekeurd door de promotoren:

Prof. Alessandro Abate
Prof. dr. ir. Hans Hellendoorn

Samenstelling promotiecommisie:

Rector Magnificus,	voorzitter
Prof. A. Abate,	University of Oxford, promotor
Prof. dr. ir. J. Hellendoorn,	Technische Universiteit Delft, promotor
Prof. dr. ir. J.H. van Schuppen,	Technische Universiteit Delft
Prof. dr. A.J. van der Schaft,	University of Groningen
Prof. dr. H. Hermanns,	Saarland University
Prof. G.E. Fainekos,	Arizona State University
Dr. M.T.J. Spaan,	Technische Universiteit Delft
Prof. dr. ir. C. Vuik,	Technische Universiteit Delft, reservelid

disc

This dissertation has been completed in fulfillment of the requirements of the Dutch Institute of Systems and Control (DISC) for graduate study.

The work presented in this thesis has been supported by the project *Modeling, Verification and Control of Complex Systems (MoVeS)*, funded by the European Union seventh framework FP7/2007-2013, under grant agreement n°257005.

The work presented in this thesis has been supported by the project *The Advanced Methods for Building Diagnostics and Maintenance (AMBI)*, funded by Marie-Curie call FP7-PEOPLE-2012-IAPP.

ISBN: 978-94-6203-683-3

Printed by CPI-Wöhrmann Print Service, The Netherlands.

To my wife for her support,
patience, and unconditional love

Acknowledgments

This thesis is the result of four years of research studies at the Delft Center for Systems and Control (DCSC) of Delft University of Technology. In this short note, I would like to acknowledge the help and support of the people who made the completion of this thesis possible.

First of all, I would like to express my sincere gratitude to Alessandro Abate for being my supervisor and giving me the possibility to achieve my goals and for the trust and support he gave me during these four years. I really appreciate his friendship and guidance, and the fact that none of my requests has ever been unheard. He is not only a supervisor, but also a great friend who cares about all aspects of his students' life, grows them scientifically, and matures their personal life.

Second, I must thank members of my research group in DCSC. Our biweekly group meetings were very beneficial in terms of brain storming and moving the research forward. I wish to thank Ilya for always discussing his outstanding abstract vision of research problems. A special thanks goes to Dieky, my four-year quiet roommate, for his methodological thinking and unique LaTeX 2_ε solutions. Thanks to Majid for his advices and for sharing his valuable academic experiences. Thanks also to Sofie, Katerina, and my other group-mates for their discussions and feedbacks during the presentations and group meetings. My collaboration with a couple of highly motivated master students of DCSC, Caspar Gevaerst and Pieter Verhoeven, was very nice and exciting.

Third, I would like to thank my collaborators in the MoVeS and AMBI projects for their contributions to my research. In particular thanks to Professors Joost-Pieter Katoen, Martin Fränzle, John Lygeros, and Maria Prandini for their constructive feedback. Also thanks to Sebastian, Christian, Nils, Maryam, Petr, Ondrej, and Karel for the scientific discussions during the project meetings which were very exciting to me.

Fourth, I am grateful to all the colleagues in DCSC. Working in close contact with you has been stimulating from an intellectual perspective and a lot of fun. I really enjoyed the time I spent with you in the annual social events and Christmas dinners of the department. The after-lunch foosball matches and summer soccer games were refreshing and energetic. I also want to thank Noor, Mohammad, Renshi, Subramanya, and Sachin with whom I have regularly played badminton

in the weekends. Furthermore, I want to thank the support staff and the secretaries of the department for being very professional and for always handling my requests with a kind smile.

I have greatly appreciated all my committee members for taking their valuable time to read my thesis and to provide constructive feedback. The improved quality of the thesis is the result of their effort. I also would like to thank Dr. Noortje Groot, Prof. Jan H. van Schuppen, and Prof.dr.ir. Hans Hellendoorn for their help in the translation of the summary and propositions in Dutch.

A special thanks goes to my beautiful amazing wife, Farzaneh, for her support and patience during these four years. Her consistent encouragement on top of my family support leads me to the successful completion of this step in my academic career. Last but not least, I would like to thank my Iranian friends, Hooman, Ali, Samira, Yashar, Mohammad Reza, Amir, Mohsen, Esmail, Mohammad, Mehdi, and Vahab who helped me in the adaptation to a new culture and a different environment through sharing their opinions and experiences.

Sadegh Esmaeil Zadeh Soudjani
Delft, September 2014

Contents

1 CHAPTER

Introduction

This thesis discusses formal abstractions for automated verification and synthesis of stochastic systems. In this chapter we introduce the processes under study, verification problems, and an application in power networks. We further briefly sketch our approach to solve these problems, which will be further elaborated throughout the thesis. The explanation of the organization of the thesis concludes this chapter.

1.1 Motivation

Stochastic processes were first studied rigorously in the late 19th century to aid in understanding Brownian motion [17] and financial markets. The first person to describe the mathematics behind Brownian motion was Thorvald N. Thiele in a paper on the method of least squares published in 1880 [73]. This was followed independently by Louis Bachelier in 1900 in his PhD thesis "The theory of speculation" [6], in which he presented a stochastic analysis of the stock and option markets. Albert Einstein (in his 1905 papers on photo-electric effect [27]) and Marian Smoluchowski (1906) [70] brought the solution of the problem to the attention of physicists, and presented it as a way to indirectly confirm the existence of atoms and molecules. Their equations describing Brownian motion were subsequently verified by the experimental work of Jean Baptiste Perrin in 1908 [24].

In probability theory, a *stochastic process* is a collection of random variables which is often used to represent the evolution of some random value, or system, over time. One approach to stochastic processes treats them as functions of one or several deterministic arguments (called indexes, in most cases regarded as time) whose values are random variables: non-deterministic quantities which have certain probability distributions. Random variables corresponding to various times may be completely different. In the simple case of discrete time, as opposed to continuous time, a stochastic process involves a sequence of random variables and the time series associated with these random variables.

	Countable or finite state space	Continuous or general state space
Discrete-time	Markov chain	Markov process on a general state space
Continuous-time	Continuous-time Markov process	Any continuous stochastic process with the Markov property, e.g. the Wiener process or solutions of stochastic differential equations

Table 1.1: Classes of Markov processes with their names used in this manuscript.

A stochastic process can be classified according to the cardinality of its index set (usually interpreted as time) and state space: the process is in discrete time if the index set is countable and has discrete space if its state space is countable. A *Markov process*, named after the Russian mathematician Andrey Markov, is a stochastic process that satisfies the Markov property: one can make predictions for the future of the process based solely on its present state just as well as one could knowing the process' full history. In other words, conditional on the present state of the process, its future and past are independent. A Markov process can be used to model a random system that changes states according to a transition rule that only depends on the current state. Table 1.1 gives an overview of the different instances of Markov processes.

In this thesis we address the investigation of complex properties over Markov processes evolving in discrete time and continuous (uncountable) state spaces [44, 63]. A discrete-time Markov Process (dtMP) is a tuple $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$, where $\mathcal{S}$ is a continuous state space and $T_{\mathfrak{s}}$ is a stochastic kernel that assigns to each state $s \in \mathcal{S}$ a probability measure $T_{\mathfrak{s}}(\cdot|s)$, so that $\mathbb{P}(s(k+1) \in A | s(k) = s) = T_{\mathfrak{s}}(A|s)$ for every $A \subset \mathcal{S}$. With regards to the probabilistic properties under investigation, we consider formulae expressed via a modal logic known as PCTL [7] and Linear Temporal Logic (LTL). PCTL encodes probabilistic specifications that can be equivalently expressed via value functions [68] and computed by recursive application of known operators or by solving integral equations, as typical in dynamic programming problems over continuous spaces [12].

To keep the discussion of the thesis clear, we zoom in on a particular PCTL specification expressing *probabilistic invariance*. Given a stochastic process evolving over a state space and a set of interest (known as invariance domain, or safe set) that is a subset of the state space, the probabilistic invariance problem is concerned with the computation of the probability that a realization of the process, started anywhere in the state space, remains within the invariance set over a given time horizon. Here we study the finite-time invariance problem and refer the reader to [74, 75, 76] for the results on infinite-time horizon properties.

Probabilistic invariance (or its dual, reachability) has been investigated for various models and with multiple techniques. Classical results on models with discrete state spaces are recapitulated in [7], whereas recent work deals with hybrid models in continuous- [18, 54] and discrete-time [4], respectively. The work in [2] has put forward a formal connection between the study of probabilistic invariance over the process and the computation of a related property over a discretized version of the model, namely a Markov chain. A Markov chain (MC) is a tuple $\mathfrak{P} = (\mathcal{P}, T_p)$ where $\mathcal{P} = \{z_1, z_2, \ldots, z_p\}$ is a finite set of states and

$T_p : \mathcal{P} \times \mathcal{P} \to [0,1]$ is a transition probability matrix, such that $T_p(z,z')$ characterizes the probability of transitioning from state z to state z'. The probabilistic invariance property over a Markov chain can be computed with a probabilistic model checker, such as PRISM [45] or MRMC [50]. The work in [3] has extended this approach to automata-based properties. Both contributions [2, 3] are formal in that they allow an exact computation of a bound on the formula-dependent approximation error.

The results of this thesis, presented for the probabilistic invariance problem, can be generalized to the reach-avoid property, i.e. computation of the probability that a realization of the process remains inside a safe set while hitting a target set in a finite time horizon. Moreover, other properties expressed as finite-state automaton [7] can be transformed to invariance properties of an auxiliary process which is the product of the automaton and the process.

We are in particular interested in Markov processes with state spaces displaying a *hybrid* structure, namely characterized by a finite collection of continuous domains (typically taken to be subsets of Euclidean spaces). These models are known in the literature as Stochastic Hybrid Systems (SHS) [14, 21]. We present our results for processes over continuous state spaces and discuss extensions to SHS at the end of each chapter.

A fascinating example of SHS is the model of thermostatically controlled loads (TCL), which captures the behavior of temperature inside a building. The evolution of the temperature of a TCL can be characterized by a stochastic difference equation. The temperature is regulated via a heating/cooling system and an ON/OFF switching strategy, where the dynamics depend on the mode of operation. Models for TCL have shown the potential to be employed in practical applications ranging from load balancing to demand-response programs. The abstraction methods of this thesis are employed to develop a model for the aggregation and control of population of TCL.

The class of dtMPs studied in this thesis is equivalent to the class of discrete-time dynamical systems that evolve according to $s(k+1) = f(s(k), w(k))$, where $s(k)$ is the state of the process at time k, f is any (possibly nonlinear, discontinuous) function, and $\{w(k), k = 0, 1, 2, \ldots\}$ are independent identically-distributed random vectors with known distribution [48]. In other words, any dtMP $\mathfrak{S} = (\mathcal{S}, T_s)$ admits such a representation as a dynamical system and vice versa. Throughout the thesis we use any of the two representations depending on the suitability for establishing the result.

1.2 Research Goals and Original Contributions

The broad aim of this PhD research is to develop a novel and general framework with efficient algorithmic tools for formal verification and synthesis of discrete-time Markov processes. The connection between the computation of a class of dynamical properties and the verification of related specifications in PCTL logic

has been investigated in [68] and extended in [3]. The contribution [4] has recently characterized the fundamental problem of probabilistic invariance and put forward an algorithm to compute this quantity. From a computational perspective, [2] has looked at the numerical evaluation of specifications discussed in [4]. This evaluation is possible by developing a formal abstraction that is based on the partitioning of the state space of the original continuous space model, which originates a discrete time, finite space Markov chain (MC) from the original process. The approach is formal in that it allows for the computation of explicit bounds on the error associated with the abstraction.

Adaptive and sequential gridding procedures. We extend the applicability of the technique developed in [2] by addressing its known bottleneck: the issue of dimensional scalability of the abstraction, which is limited by the "curse of dimensionality" related to the partitioning procedure and subsequent dynamic programming recursions. We propose adaptive and sequential gridding algorithms based on local computation of the abstraction error to make the approach applicable to larger dimensional processes.

Partially-degenerate stochastic processes. The computational approach of [2] hinges upon regularity of the conditional density function of the process, i.e. its Lipschitz continuity. Partially-degenerate stochastic processes do not satisfy such assumptions (due to the existence of Dirac-delta function in their associated density function) and require us to develop new techniques specialized for this class of processes. We have shown that the probabilistic invariance problem over such processes can be separated into two parts: a deterministic reachability analysis, and a probabilistic invariance problem that depends on the outcome of the first. This decomposition approach leads to computational improvements.

Higher-order approximations. The results [2, 3, 28, 30] have leveraged piece-wise constant interpolations of the kernels characterizing the discrete-time Markov process under study, which has direct consequences on the derived error bounds. In contrast, we provide approximation methods via higher-order interpolations of the value functions that are aimed at requiring less computational effort. Using higher-order interpolations (versus piece-wise constant ones) can be beneficial in terms of obtaining tighter bounds on the approximation error. Furthermore, since the approximation procedures depend on the partitioning of the state space, higher-order schemes display an interesting trade-off between more parsimonious representations versus more complex local computation.

Controlled discrete-time Markov processes. We provide an abstraction scheme to approximate a controlled discrete-time Markov process with a Markov decision process over a finite set of states. The approach enables us to solve the problem of obtaining maximally safe Markov policy for the Markov decision process and design a control policy for the original model. We quantify the total error made by the abstraction procedure and caused by exporting the result back to the original process.

Application to the aggregation of TCL. We propose a new, formal two-step abstraction procedure to generate a finite stochastic dynamical model as the aggregation of the dynamics of a population of TCL. The approach relaxes the limiting

assumptions employed in [53] by providing a model based on the natural probabilistic evolution of the single TCL temperature. We also describe a dynamical model for the time evolution of the abstraction, and develop a set-point control strategy aimed at reference tracking over the population total power consumption.

Implementation. The abstraction algorithms discussed in this thesis have been implemented as a MATLAB tool FAUST^2 (abbreviation for "Formal Abstractions of Uncountable-STate STochastic processes"). The first version of FAUST^2 is freely available for download at `http://sourceforge.net/projects/faust2/`.

1.3 Overview of the Thesis

This thesis discusses approaches to analysis that are based on finite-state abstractions of discrete-time Markov processes featuring general continuous or hybrid state spaces. We dedicate our analysis to the probabilistic invariance problem and apply the developed techniques to the problem of aggregation of TCL. This thesis is organized as follows:

- **Chapter 2** introduces the discrete-time Markov processes and the problem statement (computation of probabilistic invariance). An abstraction algorithm is proposed to relate a general state space model to a Markov chain and the error in the abstraction procedure is quantified. Furthermore, refinements of the error computation based on local properties are presented which results in the algorithmic generation of the abstraction in a sequential and adaptive scheme. The results are also adapted to the Stochastic Hybrid Systems model framework.

- **Chapter 3** introduces the model class of partially-degenerate stochastic systems and discusses properties of the value functions that characterize probabilistic invariance. Since the results of Chapter 2 are not applicable to this class of systems, this chapter puts forward a new approximation scheme for the computation of the probabilistic invariance over these systems. We show that the problem can be separated into two parts: a deterministic reachability analysis, and a probabilistic invariance problem on top of the first analysis. We explicitly characterize the approximation error and present a case study from Systems Biology.

- **Chapter 4** generalizes the result of Chapter 2 by introducing higher-order approximation schemes over the value functions of interest. The introduced error is first formulated for any general linear operator, employed for approximation, and then adapted to piece-wise polynomial functions obtained via interpolation. Extension of the results to SHS models is also discussed.

- **Chapter 5** generalizes the result of Chapters 2,4 to controlled discrete-time Markov processes. We present an approach based on partitioning both state and input spaces that abstracts the process to a Markov decision process.

Markov policies are defined in this chapter and the solution of finding the maximally safe policy for the process is formulated. Then we show that the proposed abstraction approach can be employed to find a sub-optimal policy and quantify the level of sub-optimality, i.e. the distance between the sub-optimal and optimal quantities.

- **Chapter 6** recapitulates the model of the single TCL dynamics as an SHS, describes its abstraction as a Markov Chain, and further discusses the aggregation of a homogeneous population of TCL. We quantify the errors introduced by both steps and discuss TCL models endowed with a control input. The synthesis of global (acting at the population level) controllers to achieve regulation of the total consumed power is achieved by two alternative schemes.

- **Chapter 7** presents the software tool FAUST^2 that implements the results of previous chapters on formal abstractions of discrete-time Markov processes defined over continuous state spaces. This chapter describes the Graphical User Interface that enhances the interaction of user with FAUST^2.

- **Chapter 8** summarizes the results of this thesis and outlines directions for future research.

The diagram in Figure 1.1 shows the relation of these chapters which can be used to select the chronological order of chapters to be studied by the reader.

1.4 Publications by the Author

Most of the material presented in Chapters 2,3,4, and 6 of this PhD thesis has appeared in international conference proceedings, both in the area of systems & control and in that of formal verification, or has been published in peer-reviewed journals. In addition to developing the theory, we have implemented most algorithms in this thesis as a MATLAB toolbox FAUST^2 (presented in Chapter 7). The connection between each chapter and the publications is as follows

- Chapter 2 is based on [28]. The interested reader may refer to [28, 31] for a more extensive discussion and mathematical details.

- Chapter 3 is based on [30]. The interested reader may refer to [30, 35] for more details and generalization of the developed methods to reach-avoid specifications.

- Chapter 4 adapted the results of [29] to the invariance problem. The mentioned paper presents the approach for reach-avoid specifications.

- Chapter 5 discusses formal abstraction of non-autonomous Markov processes to Markov Decision processes (with finite action and state spaces). This result has not been published yet.

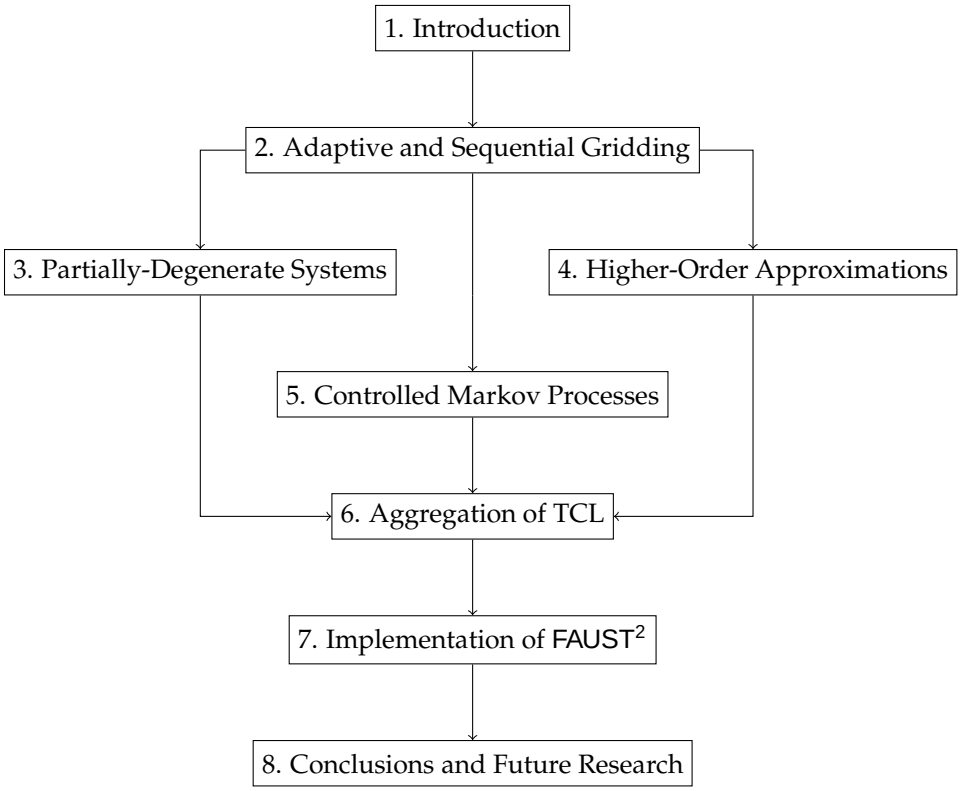


Figure 1.1: Relational structure of this thesis. Arrows indicate relations of interdependence.

- Chapter 6 is based on [32] which is devoted to the aggregation of homogeneous populations of TCL by formal abstraction. The reader may refer to [49, 33] for an extensive discussion on aggregation of heterogeneous populations of TCL and other modeling options.
- Chapter 7 is based on the manuscript [36].

2 CHAPTER

Adaptive and Sequential Gridding for the Abstraction and Verification of Stochastic Processes

This chapter is concerned with the generation of finite abstractions of general state-space processes, to be employed in the formal verification of probabilistic properties by means of automatic techniques such as probabilistic model checkers. We employ an abstraction procedure based on the partitioning of the state space, which generates a Markov chain as an approximation of the original process. We put forward a novel adaptive and sequential gridding algorithm that is expected to conform to the underlying dynamics of the model and thus to mitigate the curse of dimensionality unavoidably related to the partitioning procedure. The results are also extended to the general modeling framework known as Stochastic Hybrid Systems. While the technique is applicable to a wide arena of probabilistic properties, with focus on the study of a particular specification (probabilistic safety or invariance, over a finite horizon), the proposed adaptive algorithm is first benchmarked against a uniform gridding approach taken from the literature, and finally tested on an applicative case study in Biology.

2.1 Introduction

In this chapter we study the problem of computing probabilistic properties for discrete time Markov processes evolving over continuous (uncountable) state spaces. We interpret the analysis of a given property as the formal verification of a related specification expressed in a probabilistic modal logic [7]. Theoretically, the connection between the computation of a class of dynamical properties and the verification of related specifications in PCTL logic has been investigated in [68]

and extended in [3]. To keep the presentation focused, in this thesis we zoom in on the fundamental problem of probabilistic invariance, or safety – and on its related specification. This problem has been recently investigated in [4], which has characterized this concept and put forward an algorithm to compute this quantity.

From a computational perspective, [2] has looked at the numerical evaluation of specifications discussed in [4], among which probabilistic invariance. This evaluation is possible by developing a formal abstraction that is based on the partitioning of the state space of the original continuous space model, which derives a discrete time, finite space Markov chain (MC) from the original process. The approach is formal in that it allows for the computation of explicit bounds on the error associated with the abstraction. This technique enables considering classes of probabilistic specifications [3, 2] over continuous-space models and computing them over MC abstractions via available probabilistic model checkers [45, 50], with explicit bounds on the errors introduced with the abstraction procedure.

This chapter looks at extending the applicability of the technique developed in [2] by addressing its known bottleneck: the issue of dimensional scalability of the abstraction, which is limited by the "curse of dimensionality" related to the partitioning procedure and subsequent dynamic programming recursions. This new procedure is expected to adapt to the underlying dynamics of the model, which is characterized by (a set of) stochastic kernels. In contrast to the abstraction proposed in [2], which has leveraged a batch algorithm performing uniform partitioning based on the quantification of a global error, this chapter puts forward an adaptive and sequential procedure that exploits the knowledge of local quantities and performs the normalization of dynamics operating on multiple spatial scales. Furthermore, this chapter looks at the practical implementation of the adaptive procedure, which hinges on: the choice of the shape of partition sets (making up the states of the MC), the execution of the refinement step in the adaptive generation of the grid, as well as the generation of transition probabilities for the MC over the partition sets (which involves a marginalization procedure). Additionally, the issue of ill-conditioned dynamics (namely, widely separated dynamics operating over slow and fast scales) is tackled by considering a further refinement of the obtained errors based on state-space rescaling.

Owing to the explicit computation of the error bounds related to a given property, we provide an approach to abstraction that is effectively property-dependent. Furthermore, given the generality of the concepts of reachability and (dually) of invariance and due to their connections to more general properties [3], this abstraction technique allows a general approach for the study of these properties.

Most of the reviewed literature on the subject of formal verification of stochastic processes presents a penchant for models known as Stochastic Hybrid Systems (SHS), which are general dynamical models with interleaved discrete, continuous, and probabilistic dynamics. Fostered by their application in a number of diverse domains [14, 21], the study of SHS has recently flourished and has witnessed interesting advances at the intersection of the fields of Systems and Control [25] and of Formal Verification [7]. In this chapter we develop results over abstract state spaces and tailor them to SHS at a later stage (cf. Section 2.4 for the theory, and Section 2.6 for a case study).

From a different perspective and over classes of continuous time probabilistic hybrid models, [18] has formalized the notion of probabilistic reachability, [66] has put forward a computational technique based on convex optimization, and [38] has developed an approach based on satisfiability modulo theory, to attain the verification of similar probabilistic properties. Over models with similar semantics, [54, 67] have quantified the concept of probabilistic reachability as the solution of partial differential equations over the state-space, and put forward approximation techniques for its computation, which also leverage the use of discrete-time MC [56] – however, both approaches do not provide a quantification of the error made in the approximation step, which is a distinguishing factor of this chapter.

This chapter is structured as follows. Sections 2.2.1 and 2.2.2 introduce the model and the problem statement (computation of probabilistic invariance). Section 2.3.1 proposes an abstraction algorithm to relate a general state space model to a Markov chain. Furthermore, with focus on the probabilistic invariance problem, the quantification of the error in the abstraction procedure is presented in Section 2.3.2, whereas Section 2.3.3 puts forward refinements of the error computation based on local properties and state-space rescaling. Section 2.4 adapts the results to the Stochastic Hybrid Systems model framework. Section 2.5 deals with the algorithmic generation of the abstraction and elaborates on a number of choices leading to a sequential and adaptive scheme. Finally, Section 2.6 develops two numerical studies: a benchmark compares the adaptive and sequential approach versus the uniform procedure known from the literature [2], and tests the scalability of the adaptive approach. Also, Section 2.6 presents a case study drawn from Systems Biology – in particular Section 2.6.2 elucidates the results on a SHS model. Section 2.7 ends the chapter with conclusions and extensions.

2.2 Preliminaries

2.2.1 Model

We consider a discrete time Markov process $s(k), k \in \mathbb{N}_0 \doteq \{0, 1, 2, \ldots\}$ defined over a general state space. The model is denoted by $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$ and characterized by the following pair:

1. $\mathcal{S}$ is a continuous state space, which we assume to be endowed with a metric and to be Borel measurable. We denote by $(\mathcal{S}, \mathcal{B}(\mathcal{S}), \mathbb{P})$ the probability structure on $\mathcal{S}$, with $\mathcal{B}(\mathcal{S})$ the associated sigma algebra, and P a probability measure to be characterized shortly;

2. $T_{\mathfrak{s}}$ is a conditional stochastic kernel that assigns to each point $s \in \mathcal{S}$ a probability measure $T_{\mathfrak{s}}(\cdot|s)$, so that for any set $A \in \mathcal{B}(\mathcal{S})$,

$$\mathbb{P}(s(1) \in A | s(0) = s_0) = T_{\mathfrak{s}}(A|s_0) = \int_A T_{\mathfrak{s}}(ds|s_0).$$

The initial condition $s(0)$ for the model is sampled from $\pi : \mathcal{B}(\mathcal{S}) \to [0,1]$, a probability measure on $\mathcal{S}$. Over a finite horizon $\mathbb{Z}_{N+1} \doteq \{0,1,2,\ldots,N\}$, a Markov process $s(k), k \in \mathbb{Z}_{N+1}$ evolves over the product space $\Omega = (\mathcal{S})^{N+1}$, which is also endowed with a sigma algebra and thus allows computing the probability of events related to trajectories – we will use again $\mathbb{P}$ to denote such probability. Usually the state space is taken to be a finite-dimensional Euclidean domain, $\mathcal{S} = \mathbb{R}^n, n < \infty$. In Section 2.4, we tailor this setup to a specific "hybrid" state space, thus introducing a modeling framework known as Stochastic Hybrid Systems (see also Section 2.6.2 for a case study based on a SHS model).

2.2.2 Problem Statement

The problem of finite-horizon probabilistic invariance (alternatively referred to as probabilistic safety) can be formalized as follows.

Problem Description 2.1 *consider a bounded Borel set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$, representing a set of safe states. Characterize and compute the probability that an execution of $\mathfrak{S}$, associated with an initial condition $s_0 \in \mathcal{S}$ (sampled from π), remains within set $\mathcal{A}$ during the finite time horizon $\mathbb{Z}_{N+1}$:*

$$p_{s_0}(\mathcal{A}) \doteq \mathbb{P}\{s(k) \in \mathcal{A} \text{ for all } k \in \mathbb{Z}_{N+1} | s(0) = s_0\}. \tag{2.1}$$

This quantity allows to extend the result to a general initial probability distribution π as

$$p_\pi(\mathcal{A}) \doteq \mathbb{P}\{s(k) \in \mathcal{A} \text{ for all } k \in \mathbb{Z}_{N+1}\} = \int_{\mathcal{S}} p_{s_0}(\mathcal{A}) \pi(ds_0).$$

The following theorem provides a theoretical framework to study the probabilistic invariance problem [4] and is directly related to an algorithm for the determination of an invariant linear subspace [80].

Proposition 2.1 (Bellman recursion) *Consider value functions $V_k : \mathcal{S} \to [0,1]$, $k \in \mathbb{Z}_{N+1}$, computed by the following backward recursion:*

$$V_k(s) = \mathbf{1}_{\mathcal{A}}(s) \int_{\mathcal{S}} V_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s), \quad s \in \mathcal{S},$$

and initialized with:

$$V_N(s) = \mathbf{1}_{\mathcal{A}}(s) = \begin{cases} 1, & \text{if } s \in \mathcal{A}, \\ 0, & \text{else.} \end{cases}$$

Then $p_{s_0}(\mathcal{A}) = V_0(s_0)$.

This result characterizes the finite-horizon probabilistic invariance quantity as the solution of a dynamic programming problem in which the optimization domain has cardinality equal to one. However, since its explicit solution is in general

not available, the actual computation of the quantity $p_{s_0}(\mathcal{A})$ requires N numerical integrations over the whole set $\mathcal{A}$. This is usually performed with techniques based on state-space discretization [11], which leads to two major questions:

1. whether the numerical output can be precisely related to the actual solution; and
2. whether the approach is dimensionally scalable (e.g., as a function of n if $\mathcal{S} = \mathbb{R}^n$), particularly in comparison with alternative known approaches in the literature [2].

The goal of this chapter is to address these two issues. In the next section we answer the first question by introducing an abstraction of the original model via a numerical approximation, and by explicitly quantifying the error related to the computation of the finite-horizon probabilistic invariance with the abstraction. Furthermore, by focusing on the algorithmic implementation of the abstraction, in the remainder of this chapter we investigate the scalability properties of the proposed approach (computational complexity, memory usage), thus addressing the second question.

The overall approach, here presented over the problem of probabilistic invariance, can be directly extended to more general properties expressed in PCTL logic [68], as well as over specifications characterized as certain labeled automata [3] – both extensions can be reduced to computations of values functions related to that in Proposition 2.1 characterizing probabilistic invariance.

2.3 Model Abstraction

2.3.1 Algorithmic Abstraction as a Finite-State Markov Chain

We recall a procedure presented in [2] to approximate a model $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$, by a finite state Markov chain (MC) $\mathfrak{P} = (\mathcal{P}, T_p)$. Here $\mathcal{P} = \{z_1, z_2, \ldots, z_p\}$ is a finite set of states and $T_p : \mathcal{P} \times \mathcal{P} \to [0, 1]$ is a transition probability matrix, such that $T_p(z, z') = P(z'|z)$ characterizes the probability of transitioning from state z to state z' and thus induces a conditional discrete probability distribution over the finite space $\mathcal{P}$.

Consider the bounded safe set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$. Algorithm 1 provides a procedure to abstract model $\mathfrak{S}$ by a finite state MC $\mathfrak{P}$. In Algorithm 1, $\Xi : A_p \to 2^{\mathcal{A}}$ represents a set-valued map that associates to any point $z_i \in A_p$ the corresponding partition set $A_i \subset \mathcal{A}$. Furthermore, the map $\xi : \mathcal{A} \to A_p$ associates to any point $s \in \mathcal{A}$ of $\mathfrak{S}$ the corresponding discrete state in A_p. Additionally, notice that the absorbing set $\triangledown$ is added to the definition of the MC $\mathfrak{P}$ in order to render the transition probability matrix T_p stochastic.

Remark 2.1 *Notice that Algorithm 1 can be applied to abstract a general model by a finite state MC, regardless of the specifics of the probabilistic invariance problem studied*

Algorithm 1 Abstraction of model $\mathfrak{S}$ by MC $\mathfrak{P}$

Require: input model $\mathfrak{S}$, set $\mathcal{A}$

1: Select a finite partition of set $\mathcal{A}$ as $\mathcal{A} = \cup_{i=1}^{m} A_i$ (A_i are non-overlapping), where m represents the cardinality of the partition
2: For each A_i, select a single representative point $z_i \in A_i, \{z_i\} = \xi(A_i)$
3: Define $A_p = \{z_i | i \in \{1, 2, \ldots, m\}\}$ and take $\mathcal{P} = A_p \cup \{\triangledown\}$ as the finite state space of the MC $\mathfrak{P}$ ($\triangledown$ being a dummy variable as explained in the text)
4: Compute the transition probability matrix T_p for $\mathfrak{P}$ as:

$$T_p(z, z') = \begin{cases} T_{\mathfrak{s}}(\Xi(z')|z), & z' \in A_p, z \in A_p \\ 1 - \sum_{\bar{z} \in A_p} T_{\mathfrak{s}}(\Xi(\bar{z})|z), & z' = \triangledown, z \in A_p \\ 1, & z' = z = \triangledown \\ 0, & z' \in A_p, z = \triangledown \end{cases}$$

Ensure: output MC $\mathfrak{P}$

in this chapter (that is regardless of the given safe set $\mathcal{A}$), by assuming that $\mathcal{A} = \mathcal{S}$. The quantification of the abstraction error, to be carried out in Section 2.3.2, will however require that the set $\mathcal{A}$ (thus, as needed, the state space $\mathcal{S}$) is bounded.

Given a finite-state, discrete-time Markov Chain $\mathfrak{P} = (\mathcal{P}, T_p)$ and considering a safe set $A_p \subset \mathcal{P}$, the probabilistic invariance problem evaluates the probability that a finite execution associated with the initial condition $z_0 \in \mathcal{P}$ remains within the discrete safe set A_p during the finite time horizon $\mathbb{Z}_{N+1}$, and can be stated as follows:

$$p_{z_0}(A_p) \doteq \mathbb{P}\{p(k) \in A_p \text{ for all } k \in \mathbb{Z}_{N+1} | p(0) = z_0\}.$$

We now formulate the discrete version of Proposition 2.1.

Theorem 2.1 ([2]) *Consider value functions $V_k^p : \mathcal{P} \to [0, 1], k \in \mathbb{Z}_{N+1}$, computed by the backward recursion:*

$$V_k^p(z) = \mathbf{1}_{A_p}(z) \sum_{\bar{z} \in \mathcal{P}} V_{k+1}^p(\bar{z}) T_p(z, \bar{z}), \quad z \in \mathcal{P},$$

and initialized with:

$$V_N^p(z) = \mathbf{1}_{A_p}(z) = \begin{cases} 1, & \text{if } z \in A_p, \\ 0, & \text{if } z = \triangledown. \end{cases}$$

Then $p_{z_0}(A_p) = V_0^p(z_0)$.

It is of interest to provide a quantitative comparison between the discrete outcome obtained by Theorem 2.1 and the continuous solution that results from Proposition 2.1. The following section accomplishes this goal.

2.3.2 Quantification of the Abstraction Error

We first introduce a bound, inspired by [2, Theorem 1], on the distance between evaluations of the function $V_k, k \in \mathbb{Z}_{N+1}$ in Proposition 2.1. Consider a safe set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$. For any pair of points $s, s' \in \mathcal{A}$ and $k \in \mathbb{Z}_N$, notice that

$$\begin{aligned} |V_k(s) - V_k(s')| &= \left| \int_{\mathcal{A}} V_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s) - \int_{\mathcal{A}} V_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s') \right| \\ &\leq \int_{\mathcal{A}} |T_{\mathfrak{s}}(d\bar{s}|s) - T_{\mathfrak{s}}(d\bar{s}|s')| \,, \end{aligned} \tag{2.2}$$

since the value functions V_k are upper-bounded by the unity. Furthermore, for $k = N$ it holds trivially that $V_N(s) = V_N(s') = 1 \Rightarrow |V_N(s) - V_N(s')| = 0$.

The following Lipschitz continuity condition restricts the generality of the kernel $T_{\mathfrak{s}}$ characterizing the dynamics of model $\mathfrak{S}$.

Assumption 2.1 *Assume that the kernel $T_{\mathfrak{s}}$ admits density $t_{\mathfrak{s}}$, and that the following holds for a finite positive h:*

$$|t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \leq h \, \|s - s'\| \,, \quad \forall \bar{s}, s, s' \in \mathcal{A}.$$

Assumption 2.1 allows to derive the following bound on the abstraction error already obtained in [2] (notice the emphasis of the result on the time instance $k = 0$).

Theorem 2.2 ([2], Theorem 2) *Under Assumption 2.1, the invariance probability $p_{s_0}(\mathcal{A})$ for the model $\mathfrak{S}$ initialized at $s_0 \in \mathcal{A}$ satisfies:*

$$|p_{s_0}(\mathcal{A}) - p_{z_0}(A_p)| \leq \gamma\delta, \tag{2.3}$$

where $p_{z_0}(A_p)$ is the invariance probability for the MC $\mathfrak{P}$ obtained by Algorithm 1, and initialized at the discrete state $z_0 = \xi(s_0) \in A_p$. The constant γ is

$$\gamma = N\mathcal{K}, \qquad \textit{where} \qquad \mathcal{K} = h \, \mathscr{L}(\mathcal{A}),$$

and where δ is the largest diameter of the partition sets $A_i \subset \mathcal{A}$:

$$\delta = \max\{\|s - s'\| \, | s, s' \in A_i, i = 1, \ldots, m\},$$

h comes from Assumption 2.1, and $\mathscr{L}(B)$ denotes the Lebesgue measure of any set $B \in \mathcal{B}(\mathcal{S})$.

Theorem 2.2 allows for the synthesis of finite abstractions of continuous-space models with explicit, finite error bounds. The quality of the bounds is key in obtaining useful abstractions (that is, we are interested in bounds that are at least smaller than the unity). Furthermore, if a specific error is the objective of the study, then the quality of the error directly affects the cardinality (m) of the abstraction space, as well as the computational effort to obtain the abstraction – we shall explore this tradeoff later in this chapter.

In the next section we refine the abstraction error of Theorem 2.2 in three different ways: first, by computing a local version of the error; second, by leveraging continuity requirements that go beyond the Lipschitz condition raised in Assumption 2.1, and finally by normalizing possibly ill-conditioned dynamics operating on multiple spatial scales.

2.3.3 Refinement of the Abstraction Error

Local Computation of Abstraction Error

We relax Assumption 2.1 as follows.

Assumption 2.2 *Assume that the kernel $T_{\mathfrak{s}}$ admits density $t_{\mathfrak{s}}$, and that the following continuity assumption is valid:*

$$|t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \leq h(i,j) \left\| s - s' \right\|, \quad \forall \bar{s} \in A_j, \forall s, s' \in A_i,$$

where $i, j \in \mathbb{N}_m \doteq \{1, 2, \ldots, m\}$, the set A_i form a partition of $\mathcal{A}$ (as required for instance by Algorithm 1), and $h(\cdot, \cdot)$ are finite and positive constants.

Clearly, the global Lipschitz constant h in Assumption 2.1 represents an upper bound for the quantities $h(i,j)$ above. Equation (2.2) can be tailored to Assumption 2.2, which leads to the following result.

Theorem 2.3 *Suppose that the stochastic kernel of the model $\mathfrak{S}$ satisfies Assumption 2.2. Then the value functions $V_k : \mathcal{S} \to [0,1]$, characterizing the probabilistic invariance problem for $\mathfrak{S}$ over $\mathcal{A} \in \mathcal{B}(\mathcal{S})$, satisfy the following Lipschitz continuity, $k \in \mathbb{Z}_{N+1}$:*

$$|V_k(s) - V_k(s')| \leq \mathcal{K}_i \left\| s - s' \right\|,$$

$\forall s, s' \in A_i, i \in \mathbb{N}_m$, and where the constant $\mathcal{K}_i$ is given by: $\mathcal{K}_i = \sum_{j=1}^{m} h(i,j) \mathscr{L}(A_j)$.

Proof: Using Equation (2.2) together with the inequality in Assumption 2.2, leads directly to the following:

$$\begin{aligned} |V_k(s) - V_k(s')| &\leq \int_{\mathcal{A}} |T_{\mathfrak{s}}(d\bar{s}|s) - T_{\mathfrak{s}}(d\bar{s}|s')| = \sum_{j=1}^{m} \int_{A_j} |T_{\mathfrak{s}}(d\bar{s}|s) - T_{\mathfrak{s}}(d\bar{s}|s')| \\ &\leq \sum_{j=1}^{m} h(i,j) \left\| s - s' \right\| \mathscr{L}(A_j) = \mathcal{K}_i \left\| s - s' \right\|. \end{aligned}$$

Notice that the bound provided in this Theorem improves that derived from Equation (2.2) and Assumption 2.1, since $h \geq \max\{h(i,j) | i, j \in \mathbb{N}_m\}$. □

The result in Theorem 2.3 can be employed to quantify the error between the value $p_{z_0}(A_p)$ and $p_{s_0}(\mathcal{A})$, which leads to a refinement of Theorem 2.2.

Theorem 2.4 *Assume that Assumption 2.2 holds. Then the invariance probability $p_{s_0}(\mathcal{A})$ for model $\mathfrak{S}$, initialized at $s_0 \in \mathcal{A}$, satisfies:*

$$|p_{s_0}(\mathcal{A}) - p_{z_0}(A_p)| \leq \max\{\gamma_i \delta_i | i \in \mathbb{N}_m\}, \tag{2.4}$$

where $p_{z_0}(A_p)$ is the invariance probability for the MC $\mathfrak{P}$, initialized at the discrete state $z_0 = \xi(s_0) \in A_p$, where δ_i is the diameter of the set $A_i \subset \mathcal{A}$, namely

$$\delta_i = \max\{\|s - s'\| \,|\, s, s' \in A_i\},$$

and the constants γ_i are specified as $\gamma_i = N\mathcal{K}_i$, as per Theorem 2.3.

Proof: Let us recall that the function $\xi : \mathcal{A} \to A_p$ maps any point $s \in \mathcal{A}$ to the corresponding discrete state $z \in A_p$ via a representative point $\xi(s)$, and that $\Xi : A_p \to 2^{\mathcal{A}}$ associates a continuous partition set to a discrete (representative) point in A_p. Let us define a piecewise constant function $\hat{V}_k^p : \mathcal{A} \to [0, 1]$ with $\hat{V}_k^p(s) = V_k^p(\xi(s))$ for all $s \in \mathcal{A}$. Next we show that

$$|V_k(s) - \hat{V}_k^p(s)| \leq (N - k) \max\{\mathcal{K}_i \delta_i | i \in \mathbb{N}_m\}. \tag{2.5}$$

For $k = N$ the inequality is trivial, since $|V_k(s) - \hat{V}_k^p(s)| = |V_N(s) - \hat{V}_N^p(s)| = 1 - 1 = 0$. Suppose now that the inequality holds for $k + 1$, then at time step k we have:

$$\begin{aligned} |V_k(s) - \hat{V}_k^p(s)| = |V_k(s) - \hat{V}_k^p(\xi(s))| &\leq |V_k(s) - V_k(\xi(s))| + \left|V_k(\xi(s)) - \hat{V}_k^p(\xi(s))\right| \\ &\leq \mathcal{K}_{\underline{i}} \delta_{\underline{i}} + \left|V_k(\xi(s)) - \hat{V}_k^p(\xi(s))\right|, \end{aligned}$$

where the index $\underline{i} \in \mathbb{N}_m$ corresponds to the set $A_{\underline{i}} = \Xi(\xi(s))$. On the other hand, by exploiting the discrete feature of the function $\hat{V}_k^p$ evaluated at $\xi(s)$ and its piecewise constant structure, we can observe that

$$\begin{aligned} \hat{V}_k^p(\xi(s)) = \sum_{z \in A_p} \hat{V}_{k+1}^p(z) T_p(\xi(s), z) &= \sum_{z \in A_p} \hat{V}_{k+1}^p(z) \int_{\Xi(z)} T_{\mathfrak{s}}(d\omega | \xi(s)) \\ &= \int_{\mathcal{A}} \hat{V}_{k+1}^p(\omega) T_{\mathfrak{s}}(d\omega | \xi(s)), \end{aligned}$$

which results in the following inequality:

$$\begin{aligned} \left|V_k(\xi(s)) - \hat{V}_k^p(\xi(s))\right| &= \left| \int_{\mathcal{A}} V_{k+1}(\omega) T_{\mathfrak{s}}(d\omega | \xi(s)) - \sum_{z \in A_p} \hat{V}_{k+1}^p(z) T_p(\xi(s), z) \right| \\ &= \left| \int_{\mathcal{A}} V_{k+1}(\omega) T_{\mathfrak{s}}(d\omega | \xi(s)) - \int_{\mathcal{A}} \hat{V}_{k+1}^p(\omega) T_{\mathfrak{s}}(d\omega | \xi(s)) \right| \\ &\leq \int_{\mathcal{A}} \left|V_{k+1}(\omega) - \hat{V}_{k+1}^p(\omega)\right| T_{\mathfrak{s}}(d\omega | \xi(s)). \end{aligned}$$

We then obtain:

$$|V_k(s)-\hat{V}_k^p(s)| \leq \mathcal{K}_{\underline{i}}\delta_{\underline{i}} + \int_{\mathcal{A}} \left|V_{k+1}(\omega) - \hat{V}_{k+1}^p(\omega)\right| T_{\mathfrak{s}}(d\omega|\xi(s))$$
$$\leq \mathcal{K}_{\underline{i}}\delta_{\underline{i}} + (N-k-1)\max_i\{\mathcal{K}_i\delta_i\} \underbrace{\int_{\mathcal{A}} T_{\mathfrak{s}}(d\omega|\xi(s))}_{\leq 1} \leq (N-k)\max_i\{\mathcal{K}_i\delta_i\}.$$

The results in Theorems 2.1, 2.1 and inequality (2.5) applied at $k = 0$ yield the following bound:

$$|p_{s_0}(\mathcal{A}) - p_{z_0}(A_p)| = |V_0(s_0) - V_0^p(\xi(s_0))| = |V_0(s_0) - \hat{V}_0^p(s_0)|$$
$$\leq N\max\{\mathcal{K}_i\delta_i | i \in \mathbb{N}_m\} = \max\{\gamma_i\delta_i | i \in \mathbb{N}_m\},$$

which concludes the proof of the statement. □

Notice that often in practice the (global or local) Lipschitz constants need to be numerically computed or over-approximated, which relates to a computational cost. This leads to propose a simplification of Assumption 2.2 and an adaptation of Theorems 2.3, 2.4 accordingly. The new requirement is computationally less demanding, however as expected the related error bounds will be more conservative (less tight).

Assumption 2.3 *Assume that the kernel $T_{\mathfrak{s}}$ admits density $t_{\mathfrak{s}}$, and that the following holds for a choice of a finite positive $h(\cdot)$:*

$$|t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \leq h(i)\,\|s - s'\|, \quad \forall \bar{s} \in \mathcal{A}, \forall s, s' \in A_i,$$

where $i \in \mathbb{N}_m$ and A_i form a partition of $\mathcal{A}$ (as obtained for instance from Algorithm 1).

Theorem 2.5 *Suppose the stochastic kernel of the model $\mathfrak{S}$ satisfies Assumption 2.3. Then the value functions $V_k : \mathcal{S} \to [0,1]$, characterizing the probabilistic invariance problem for the model $\mathfrak{S}$ over $\mathcal{A} \in \mathcal{B}(\mathcal{S})$, satisfy the following Lipschitz continuity, $k \in \mathbb{Z}_{N+1}$:*

$$|V_k(s) - V_k(s')| \leq \mathcal{K}_i\,\|s - s'\|,$$

$\forall s, s' \in A_i, i \in \mathbb{N}_m$, where the constant $\mathcal{K}_i$ is given by:

$$\mathcal{K}_i = h(i)\mathscr{L}(\mathcal{A}),$$

and where $\mathscr{L}(B)$ denotes the Lebesgue measure of any set $B \in \mathcal{B}(\mathcal{S})$.

Proof: The proof can be directly adapted from that of Theorem 2.3, in particular noticing that $|V_k(s) - V_k(s')| \leq \int_{\mathcal{A}} |T_{\mathfrak{s}}(d\bar{s}|s) - T_{\mathfrak{s}}(d\bar{s}|s')| \leq h(i)\,\|s - s'\|\,\mathscr{L}(\mathcal{A})$. □

Theorem 2.6 *Under Assumption 2.3 the invariance probability $p_{s_0}(\mathcal{A})$ for the model $\mathfrak{S}$, initialized at $s_0 \in \mathcal{A}$, satisfies:*

$$|p_{s_0}(\mathcal{A}) - p_{z_0}(A_p)| \leq \max\{\gamma_i\delta_i | i \in \mathbb{N}_m\}, \tag{2.6}$$

where $p_{z_0}(A_p)$ is the invariance probability for the MC $\mathfrak{P}$ initialized at the discrete state $z_0 = \xi(s_0) \in A_p$, the constants $\gamma_i = N\mathcal{K}_i$, as per Theorem 2.5, and where δ_i is the diameter of the set $A_i \subset \mathcal{A}$: $\delta_i = \max\{\|s - s'\| \,|s, s' \in A_i\}$.

Proof: The proof can be directly adapted from that of Theorem 2.4. □

Variable rescaling and direct computation

We are interested in the application of the abstraction bounds on models with kernels that present ill-conditioned dynamics, that is dynamics operating on multiple spatial scales or characterized by both slow and fast variables. This goal will be further clarified in the light of the algorithmic procedures discussed in Section 2.5. We start by investigating whether a rescaling of the dynamics affects the abstracted Markov Chain and the associated computation of the local error, according to Assumption 2.2 (as needed the results can be easily tailored to the other two presented assumptions). Let us consider a stochastic kernel endowed with a density function $t_{\mathfrak{s}}(\bar{s}|s)$, and let us transform the state space by applying a linear map $s = Pr$, where P is an invertible matrix.[1]

A generic set $A_i \in \mathcal{B}(\mathcal{S})$ is mapped into $\tilde{A}_i = \{r \in \mathcal{S} : s = Pr, s \in A_i\}$, which is such that $\mathscr{L}(\tilde{A}_i) = |\det(P^{-1})|\mathscr{L}(A_i)$. Furthermore, the new density function $t_{\mathfrak{r}}(\bar{r}|r)$ is related to the original one $t_{\mathfrak{s}}(\bar{s}|s)$ by the equality

$$t_{\mathfrak{r}}(\bar{r}|r) = |J(r)|\, t_{\mathfrak{s}}\left(P\bar{r}|Pr\right), \tag{2.7}$$

where $|J(r)|$ denotes absolute value of the determinant of the Jacobian

$$J(r) = \left|\frac{\partial(s_1, \ldots, s_n)}{\partial(r_1, \ldots, r_n)}\right| = \det \begin{bmatrix} \frac{\partial s_1}{\partial r_1} & \cdots & \frac{\partial s_1}{\partial r_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial s_n}{\partial r_1} & \cdots & \frac{\partial s_n}{\partial r_n} \end{bmatrix} = \det(P).$$

Suppose that the representative points $z_i, i \in \mathbb{N}_m$, of the abstracted Markov Chain are also mapped to points $v_i : z_i = Pv_i$, which leads to the entries of a new transition probability matrix T_p specified, for any $j \in \mathbb{N}_m$, by

$$\begin{aligned} T_p(v_i, \Xi(v_j)) &= \int_{\tilde{A}_j} t_{\mathfrak{r}}(\bar{r}|v_i)d\bar{r} = \int_{\tilde{A}_j} |\det(P)|\, t_{\mathfrak{s}}\left(P\bar{r}|Pv_i\right) d\bar{r} \\ &= \int_{A_j} t_{\mathfrak{s}}(\bar{s}|z_i)d\bar{s} = T_{\mathfrak{s}}(z_i, \Xi(z_j)). \end{aligned}$$

This equality shows that the Markov Chains obtained from the original and from the rescaled Markov processes are equivalent.

[1] We leave to the reader the extension to an affine transformation, namely $s = Pr + Q$, where Q is properly sized. It is easy to verify that the properties discussed below are shift invariant, and to adapt them to the affine case accordingly.

With focus on Assumption 2.2, we compute the local Lipschitz constants of the new conditional distribution. Notice that the Lipschitz constant of a function is not uniquely defined, since any finite upper bound is also a legitimate Lipschitz constant. As a result, the abstraction error depends on the method employed to compute local Lipschitz constants. A common method for computation of the Lipschitz constant is maximization of the Euclidean norm of the function gradient. Recall from Assumption 2.2 that $|t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \leq h(i,j)\|s - s'\|$, and assume that the following method is used to compute $h(i,j)$:[2]

$$h(i,j) = \max_{s \in A_i, \bar{s} \in A_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|s) \right\|.$$

Then, in the new coordinates, we have that $|t_{\mathfrak{r}}(\bar{r}|r) - t_{\mathfrak{r}}(\bar{r}|r')| \leq \tilde{h}(i,j)\|r - r'\|$, with the following Lipschitz constant:

$$\tilde{h}(i,j) = \max_{r \in \tilde{A}_i, \bar{r} \in \tilde{A}_j} \left\| \frac{\partial t_{\mathfrak{r}}}{\partial r}(\bar{r}|r) \right\|.$$

Let us relate these two Lipschitz constants using Equation (2.7) and applying the chain rule in the computation of partial derivatives:

$$\begin{aligned}\tilde{h}(i,j) &= \max_{r \in \tilde{A}_i, \bar{r} \in \tilde{A}_j} \left\| \frac{\partial t_{\mathfrak{r}}}{\partial r}(\bar{r}|r) \right\| = |\det(P)| \max_{r \in \tilde{A}_i, \bar{r} \in \tilde{A}_j} \left\| \frac{\partial}{\partial r} t_{\mathfrak{s}}\left(P\bar{r}|Pr\right) \right\| \\ &= |\det(P)| \max_{r \in \tilde{A}_i, \bar{r} \in \tilde{A}_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}\left(P\bar{r}|Pr\right) P \right\| = |\det(P)| \max_{s \in A_i, \bar{s} \in A_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}\left(\bar{s}|s\right) P \right\|.\end{aligned}$$

Then $\tilde{h}$ differs from h over two terms:

- The constant term $|\det(P)|$. This constant has no effect on the computation of the abstraction error (cf. terms $\mathcal{K}_i$ in Theorem 2.4), since $\mathscr{L}(\tilde{A}_j) = |\det(P^{-1})|\mathscr{L}(A_j)$. Without loss of generality we can then restrict the attention to matrices with determinant that is equal to one.
- The matrix P within the norm. It provides a weighted sum of the partial derivatives. We can exploit this matrix in order to balance the partial derivatives over different directions. In particular, this scaling matrix can be useful in the presence of ill-conditioned dynamics.

With the above discussion we have argued that the Lipschitz constant depends on the coordinates where the distribution function is defined. Since we are interested in the value of the Lipschitz constant as part of the approximation error formula (as per Theorem 2.4), rescaling provides a degree of freedom in the error computation. This is discussed in the following theorem, which emphasizes improvements of the approximation error bounds, again focusing on Assumption 2.2.

[2] In the following, we assume that all the optimization problems have been computed over the closure of the corresponding optimization domain. However for the sake of notation simplification, we simply refer to the optimization domains as they are given.

Theorem 2.7 *Consider the conditional distribution $t_{\mathfrak{s}}$ of $\mathfrak{S}$, any set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$, a partition $\cup_{i=1}^{m} A_i$ of $\mathcal{A}$, and a properly-sized, square invertible matrix P. Then for all points $s, s' \in A_i, \bar{s} \in A_j$, it holds that*

$$\max_{s\in A_i,\bar{s}\in A_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|s) \right\| \max_{s,s'\in A_i} \|s - s'\| \geq \tag{2.8}$$

$$\geq \min_{P} \left(\max_{s\in A_i,\bar{s}\in A_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|s)P \right\| \max_{s,s'\in A_i} \left\| P^{-1}(s - s') \right\| \right) \tag{2.9}$$

$$\geq \max_{\bar{s}\in A_j} \max_{s,s',\zeta\in A_i} \left| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|\zeta)(s - s') \right| \tag{2.10}$$

$$\geq |t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')|. \tag{2.11}$$

Proof: The inequality (2.8)≥(2.11) is employed in the bound discussed in Theorem 2.4, and is based on the maximum norm of the partial derivatives. The expression in (2.9)≥(2.11) is based on the idea of rescaling the state space as follows: transform the inequality $|t_{\mathfrak{r}}(\bar{r}|r) - t_{\mathfrak{r}}(\bar{r}|r')| \leq \tilde{h}(i,j)\|r - r'\|$ into

$$|\det(P)|\,|t_{\mathfrak{s}}\,(P\bar{r}|Pr) - t_{\mathfrak{s}}\,(P\bar{r}|Pr')\,| \leq \tilde{h}(i,j)\|r - r'\|,$$

which leads to

$$|t_{\mathfrak{s}}\,(\bar{s}|s) - t_{\mathfrak{s}}\,(\bar{s}|s')\,| \leq \frac{\tilde{h}(i,j)}{|\det(P)|}\|r - r'\| = \max_{s\in A_i,\bar{s}\in A_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}\,(\bar{s}|s)\,P \right\| \|P^{-1}(s - s')\|.$$

Selecting the matrix P to be equal to the identity matrix leads to the inequality (2.8)≥(2.9). The mean value theorem for scalar fields implies the last inequality, namely (2.10)≥(2.11). Let us finally relate (2.9) to (2.10) by using the Cauchy-Schwartz inequality:

$$\begin{aligned}
\min_{P} \left(\max_{s\in A_i,\bar{s}\in A_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|s)P \right\| \max_{s,s'\in A_i} \left\| P^{-1}(s - s') \right\| \right) = \\
= \min_{P} \left(\max_{s,s',\zeta\in A_i,\bar{s}\in A_j} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|\zeta)P \right\| \left\| P^{-1}(s - s') \right\| \right) \\
\geq \max_{s,s',\zeta\in A_i,\bar{s}\in A_j} \left(\min_{P} \left\| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|\zeta)P \right\| \left\| P^{-1}(s - s') \right\| \right) \\
\geq \max_{s,s',\zeta\in A_i,\bar{s}\in A_j} \left(\min_{P} \left| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|\zeta)PP^{-1}(s - s') \right| \right) \\
= \max_{s,s',\zeta\in A_i,\bar{s}\in A_j} \left| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|\zeta)(s - s') \right|.
\end{aligned}$$

This concludes the proof. □

The above theorem does not pose any restriction on the choice of the invertible matrix P. Notice that the bound in (2.9) is invariant under constant multiplications of matrix P: we can then reduce the optimization domain to the set of square matrices with $|\det(P)| = 1$. As an alternative to the above bounds, which hinge

on the computation of quantities related to the Lipschitz constant, we put forward the next result.

Corollary 2.1 *Consider the conditional distribution $t_{\mathfrak{s}}$ of $\mathfrak{S}$, any set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$, and a partition $\cup_{i=1}^{m} A_i$ of $\mathcal{A}$. The continuous conditional distribution $t_{\mathfrak{s}}(\bar{s}|s)$ satisfies the inequality*

$$|t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \leq \max_{\bar{s}\in A_j} \left[\max_{s\in A_i} t_{\mathfrak{s}}(\bar{s}|s) - \min_{s\in A_i} t_{\mathfrak{s}}(\bar{s}|s) \right] \quad \forall s, s' \in A_i, \forall \bar{s} \in A_j. \tag{2.12}$$

Proof: The distribution is assumed to be continuous over the closure of A_i, hence it admits finite maximum and minimum, which leads to the following:

$$\begin{aligned} \max_{s,s'\in A_i, \bar{s}\in A_j} |t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| &= \max_{\bar{s}\in A_j} \left[\max_{s,s'\in A_i} |t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \right] \\ &= \max_{\bar{s}\in A_j} \left[\max_{s\in A_i} t_{\mathfrak{s}}(\bar{s}|s) - \min_{s\in A_i} t_{\mathfrak{s}}(\bar{s}|s) \right]. \end{aligned}$$

□

Notice that the quantity in (2.12) provides the optimal (lowest) upper bound over (2.8)-(2.10), since (2.12) represents a particular instantiation of (2.11) and we have shown that (2.11)≤(2.10).

Owing to the emphasis of this chapter on numerics, let us focus on the overhead associated to the computation of the presented bounds. Assume that we are given a Cartesian partition of the safe set $\mathcal{A}$, which will be the underlying assumption for the Algorithms developed in Section 2.5. This enables an analytic expression of the distance between points in (2.8)-(2.10). Therefore the upper bounds (2.8) and (2.10) are clearly related to the same computational cost (maximization over the variables appearing in the partial derivatives). With regards to the bound based on (2.9), the cost is also the same if a specific matrix P is selected – on the contrary, the optimization over this matrix increases the computational overhead. In general, matrix P in (2.9) can be treated either as an optimization variable or, as discussed, as a transformation matrix for improving the effect of widely separated dynamics. The additional bound in (2.12), which does not depend on the computation of the Lipschitz constant, requires an optimization over three variables and as such it is computationally heavier than (2.8) and (2.10); however it can be matched to their complexity – at the expense of loss of tightness – by the following simplification:

$$\max_{\bar{s}\in A_j} \left[\max_{s\in A_i} t_{\mathfrak{s}}(\bar{s}|s) - \min_{s\in A_i} t_{\mathfrak{s}}(\bar{s}|s) \right] \leq \max_{s\in A_i, \bar{s}\in A_j} t_{\mathfrak{s}}(\bar{s}|s) - \min_{s\in A_i, \bar{s}\in A_j} t_{\mathfrak{s}}(\bar{s}|s). \tag{2.13}$$

Finally notice that, while the quantities defined in (2.8)-(2.10) are proportional to the size of the partition sets, that in (2.12) is not. In the following, either of the bounds (2.8)-(2.10) will be used to construct an adaptive partition, thereafter employing the improved bound in (2.12) as an a-posteriori analysis of the abstraction error.

Let us summarize the results in Theorem 2.7 and Corollary 2.1: we have provided four different methods for computing a local upper bound, call it $k(i,j)$, as $|t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \leq k(i,j)$, for any $s, s' \in A_i, \bar{s} \in A_j, i,j \in \mathbb{N}_m$. The upper bound $k(i,j)$, in whatever form (2.8)-(2.10) or (2.12), can be directly used to quantify the abstraction errors in Theorems 2.3 and 2.4 as:

$$E = \max_{i \in \mathbb{N}_m} \left\{ \sum_{j=1}^{m} N k(i,j) \mathscr{L}(A_j) \right\}. \tag{2.14}$$

Notice the difference between the bound $k(i,j)$ and the quantity $h(i,j)$ (Lipschitz constant), as used in Theorems 2.3 and 2.4. Similar to $h(i,j)$, which can be relaxed to $h(i)$ as discussed in Assumption 2.3, the formulas for $k(i,j)$ can also be relaxed: for instance, the inequality (2.11)$\leq$(2.10) would become

$$|t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| \leq \max_{\bar{s} \in \mathcal{A}} \max_{s,s',\zeta \in A_i} \left| \frac{\partial t_{\mathfrak{s}}}{\partial s}(\bar{s}|\zeta)(s - s') \right|.$$

A similar adaptation can be applied over global bounds based on Assumption 2.1.

Let us remark that the Assumptions 2.2,2.3 based on local Lipschitz continuity not only yield error bounds that are tighter than their global counterpart, but are also practically less conservative. Discontinuous density functions are in fact not globally Lipschitz continuous and thus do not satisfy Assumption 2.1, however they can satisfy Assumptions 2.2,2.3 if the discontinuity points lie on the boundaries of the partition sets: this requirement can then be satisfied by a proper selection of these sets. While we do not further focus on discontinuous kernels in the rest of the manuscript, this discussion hints at the application of the abstraction procedure to a wider range of models, for instance models endowed with kernels derived from data. Of course this comes at the expense of a more elaborated (and likely slower) partitioning procedure. Along these lines, the Lipschitz continuity assumptions over the densities can be generalized by looking at Lipschitz continuity over the kernels instead: more precisely, in the case of Assumption 2.3 we would obtain

$$\int_{\mathcal{A}} |t_{\mathfrak{s}}(\bar{s}|s) - t_{\mathfrak{s}}(\bar{s}|s')| d\bar{s} \leq H(i), \quad \forall s, s' \in A_i,\ i \in \mathbb{N}_m. \tag{2.15}$$

The global error becomes then $E = \max\{NH(i), i \in \mathbb{N}_m\}$. This assumption is practically less conservative since it allows dealing with discontinuous conditional density functions, regardless of the chosen partitioning procedure. In contrast to (2.14), the error bound based on (2.15) does not explicitly depend on the Lebesgue measure of the partition sets, and then provides a tighter upper bound for the error. On the other hand, the computation of the parameters $H(i)$ in (2.15), requires an increased effort: the maximization needs to be performed over two variables (s, s') and each function evaluation requires a numerical integration. As we shall see in the experiments of Section 2.6.2, the numerical integration makes the computation much more time consuming than the other methods developed above. In conclusion, the bound in (2.15) provides tighter error bound, can lead

to memory savings, but also to more time consuming algorithms.

2.4 Application to Stochastic Hybrid Systems

In this section we tailor the presented results on error bounds for the abstraction around models endowed with a particular state space that is "hybrid" in nature [4]. Stochastic Hybrid Systems are Markov processes defined over a hybrid state space $\mathcal{S}$ made up of a finite, disjoint union of continuous domains over a finite, discrete set of locations (or modes) $\mathcal{Q} = \{q_1, q_2, \ldots, q_{\mathfrak{m}}\}$, namely

$$\mathcal{S} = \cup_{q\in\mathcal{Q}}\{q\} \times \mathbb{R}^{n(q)}.$$

The continuous domains have a dimension $n(q)$ that is mode dependent and characterized by a bounded function $n : \mathcal{Q} \to \mathbb{N}$. The conditional stochastic kernel $T_{\mathfrak{s}} : \mathcal{B}(\mathcal{S})\times\mathcal{S} \to [0,1]$ on $\mathcal{S}$ is fully characterized by three kernels T_q, T_x, T_r, dealing respectively with the discrete evolution over locations, the continuous evolution in the domain of a given location, and the continuous reset between domains of different locations.

Given a hybrid point $s = (q, x) \in \mathcal{S}$ and a Borel measurable set $\mathcal{A} = \cup_{q\in\mathcal{Q}}\{q\} \times A_q, \mathcal{A} \in \mathcal{B}(\mathcal{S})$, the stochastic kernel $T_{\mathfrak{s}}$ is further specified as follows [4]:

$$T_{\mathfrak{s}}(\{\bar{q}\} \times A_{\bar{q}}|(q,x)) = T_q(\bar{q}|(q,x)) \times \begin{cases} T_x(A_{\bar{q}}|(q,x)), & \text{if } \bar{q} = q, \\ T_r(A_{\bar{q}}|(q,x),\bar{q}), & \text{if } \bar{q} \neq q. \end{cases} \tag{2.16}$$

Here $T_{\mathfrak{s}}$ is made up of three distinct conditional kernels. $T_q : \mathcal{Q} \times \mathcal{S} \to [0,1]$ assigns to each $s \in \mathcal{S}$ a discrete probability distribution $T_q(\cdot|s)$ over $\mathcal{Q}$. Based on a sample of T_q, if the selected location $\bar{q}$ coincides with the current mode q, then $T_x : \mathcal{B}(\mathbb{R}^{n(\cdot)}) \times \mathcal{S} \to [0,1]$ assigns to each $s \in \mathcal{S}$ a probability measure $T_x(\cdot|s)$ over the continuous domain associated with $q \in \mathcal{Q}$. On the other hand, if $\bar{q} \neq q$, then $T_r : \mathcal{B}(\mathbb{R}^{n(\cdot)}) \times \mathcal{S} \times \mathcal{Q} \to [0,1]$ assigns to each $s \in \mathcal{S}$ and $\bar{q} \in \mathcal{Q}$ a probability measure $T_r(\cdot|s,\bar{q})$ over the continuous domain associated with $\bar{q} \in \mathcal{Q}$.

We shall denote such a discrete-time stochastic hybrid model $\mathfrak{S} = (\mathcal{Q}, n, T_q, T_x, T_r)$, and refer the reader to [4] for technical details on its topological and measurability properties and for an algorithmic definition of its execution. Section 2.6.2 develops a case study based on a SHS model.

2.4.1 Abstraction and Error Computation

The abstraction of a SHS as a MC follows the same lines as in Section 2.3.1. Consider the hybrid safe set $\mathcal{A} \in \mathcal{B}(\mathcal{S}), \mathcal{A} = \cup_{q\in\mathcal{Q}}\{q\} \times A_q$, with $A_q \in \mathcal{B}(\mathbb{R}^{n(q)})$. For all $q \in \mathcal{Q}$, select a finite (m_q-dimensional) partition of the local set A_q as $A_q = \cup_{i=1}^{m_q} A_{q,i}$ ($A_{q,i}$ are non-overlapping). For each $A_{q,i}$, select a single representative point $(q, z_{q,i}) \in A_{q,i}$, and redefine $A_p = \{(q, z_{q,i})|i \in \mathbb{N}_{m_q}, q \in \mathcal{Q}\}$. Focusing on bounds based on the Lipschitz constant of densities, the following is an extension of Assumption 2.2 to the SHS framework.

Assumption 2.4 *Assume that the kernels T_x, T_r admit densities t_x, t_r respectively, and that the following continuity assumptions are valid:*

$$|T_q(\bar{q}|(q,x)) - T_q(\bar{q}|(q,x'))| \leq h_q(q,\bar{q},i)\,\|x-x'\| \quad \forall x,x' \in A_{q,i}$$
$$|t_x(\bar{x}|(q,x)) - t_x(\bar{x}|(q,x'))| \leq h_x(q,i,j)\,\|x-x'\| \quad \forall \bar{x} \in A_{q,j},\ x,x' \in A_{q,i}$$
$$|t_r(\bar{x}|(q,x),\bar{q}) - t_r(\bar{x}|(q,x'),\bar{q})| \leq h_r(q,\bar{q},i,k)\,\|x-x'\|\ \forall \bar{x} \in A_{\bar{q},k},\ x,x' \in A_{q,i}, \bar{q} \neq q,$$

where $q,\bar{q} \in \mathcal{Q}$; $i,j \in \mathbb{N}_{m_q}$; $k \in \mathbb{N}_{m_{\bar{q}}}$; and $h_q(\cdot), h_x(\cdot), h_r(\cdot)$ are finite positive constants.

Assumption 2.4 is the local Lipschitz continuity of the conditional density functions T_q, t_x, t_r respect to the current state. This assumption can as well be applied to SHS models with piecewise continuous density functions by proper selection of partition sets (cf. the TCL model of Chapter 6).

Let us consider a SHS model, a (hybrid) invariant set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$, a finite time horizon $\mathbb{Z}_{N+1}$, a point $s_0 \in \mathcal{S}$, and an abstraction procedure over $\mathfrak{S}$. The error between the value $p_{z_0}(A_p)$ for the MC from $p_{s_0}(\mathcal{A})$ for the SHS can be quantified as follows.

Theorem 2.8 *Assume that Assumption 2.4 holds. Then the invariance probability $p_{s_0}(\mathcal{A})$ for the SHS $\mathfrak{S}$, initialized at $s_0 \in \mathcal{A}$, satisfies:*

$$|p_{s_0}(\mathcal{A}) - p_{z_0}(A_p)| \leq \max\{\gamma_{q,i}\delta_{q,i} | i \in \mathbb{N}_{m_q}, q \in \mathcal{Q}\}, \tag{2.17}$$

where $p_{z_0}(A_p)$ is the invariance probability for the MC $\mathfrak{P}$, initialized at the discrete state $z_0 = \xi(s_0) \in A_p$, where $\delta_{q,i}$ is the diameter of the set $A_{q,i} \subset A_q$, namely

$$\delta_{q,i} = \max\{\|x-x'\| \,|\, x,x' \in A_{q,i}\},$$

and the constants $\gamma_{q,i}$ are specified as $\gamma_{q,i} = N\mathcal{K}_{q,i}$, where

$$\mathcal{K}_{q,i} = \sum_{j=1}^{m_q} h_x(q,i,j)\mathscr{L}(A_{q,j}) + \sum_{\bar{q}\in Q} h_q(q,\bar{q},i) + \sum_{\bar{q}\neq q}\sum_{k=1}^{m_{\bar{q}}} h_r(q,\bar{q},i,k)\mathscr{L}(A_{\bar{q},k}),$$

and where $\mathscr{L}(B)$ denotes the Lebesgue measure of any set $B \in \mathcal{B}(\mathcal{S})$.

Proof: Inequality (2.2) corresponds to the following:

$$|V_k(q,x) - V_k(q,x')| \leq \int_{A_q} |T_q(q|(q,x))t_x(\bar{x}|(q,x)) - T_q(q|(q,x'))t_x(\bar{x}|(q,x'))|\,d\bar{x}$$
$$+ \sum_{\bar{q}\neq q}\int_{A_{\bar{q}}} |T_q(\bar{q}|(q,x))t_r(\bar{x}|(q,x),\bar{q}) - T_q(\bar{q}|(q,x'))t_r(\bar{x}|(q,x'),\bar{q})|\,d\bar{x}.$$

As in Theorem 2.3, the local Lipschitz continuity of the value functions is established by Assumption 2.4: for all $x,x' \in A_{q,i}$,

$$|V_k(q,x) - V_k(q,x')| \leq \int_{A_q} T_q(q|(q,x))\,|t_x(\bar{x}|(q,x)) - t_x(\bar{x}|(q,x'))|\,d\bar{x}$$

$$
\begin{aligned}
&+ \int_{A_q} t_x(\bar{x}|(q,x'))\,|T_q(q|(q,x)) - T_q(q|(q,x'))|\,d\bar{x} \\
&+ \sum_{\bar{q}\neq q} \int_{A_{\bar{q}}} T_q(\bar{q}|(q,x))\,|t_r(\bar{x}|(q,x),\bar{q}) - t_r(\bar{x}|(q,x'),\bar{q})|\,d\bar{x} \\
&+ \sum_{\bar{q}\neq q} \int_{A_{\bar{q}}} t_r(\bar{x}|(q,x'),\bar{q})\,|T_q(\bar{q}|(q,x)) - T_q(\bar{q}|(q,x'))|\,d\bar{x} \\
&\leq \sum_{j=1}^{m_q} h_x(q,i,j)\mathscr{L}(A_{q,j}) + h_q(q,q,i) \\
&+ \sum_{\bar{q}\neq q}\sum_{k=1}^{m_{\bar{q}}} h_r(q,\bar{q},i,k)\mathscr{L}(A_{\bar{q},k}) + \sum_{\bar{q}\neq q} h_q(q,\bar{q},i) = \mathcal{K}_{q,i}.
\end{aligned}
$$

The rest of the proof follows the same lines of Theorem 2.4. □

2.5 Algorithms for Abstraction

In the previous sections we considered arbitrary partitions of the state space and, with focus on the problem of finite-time probabilistic invariance over a given set $\mathcal{A}$, we derived bounds between the exact value $p_{s_0}(\mathcal{A})$ and the approximation $p_{z_0}(A_p)$, based respectively on the model $\mathfrak{S}$ and on its MC abstraction $\mathfrak{P}$. In this section we focus on a few alternative techniques for the generation of the abstraction $\mathfrak{P} = (\mathcal{P}, T_p)$ from $\mathfrak{S} = (\mathcal{S}, T_s)$. We explicitly exploit the knowledge of the (local) error to adapt the abstraction to the underlying dynamics of $\mathfrak{S}$, as well as to the invariance problem of interest. Since the approach can be extended to more general specifications, expressed as formulas in a particular probabilistic modal logic [3], the approach effectively allows for a formula-based abstraction of stochastic models.

In order to maintain focus and keep the notation light, we present the procedures in the case where no rescaling of the state space has been performed. The abstraction procedure consists of two main steps (see Algorithm 1):

1. grid generation, namely the partitioning of $\mathcal{S}$ that yields $\mathcal{P}$; and
2. marginalization of T_s, which leads to T_p.

We proceed with the analysis of these two successive items.

2.5.1 Grid Generation

Let us first focus on the state space partitioning, which involves the generation of a grid. The grid can be either uniform and generated instantaneously [2], or be variable and generated adaptively. More precisely, for the problem at hand the

generation of a uniform grid leverages the explicit knowledge of the global error of Theorem 2.2 and is thus instantaneous. On the other hand, the adaptive partitioning requires the knowledge of errors that are local to the existing partition sets (see Theorems 2.4 and 2.6) and proceeds via a progressive refinement of the grid. We will thus sequentially perform adaptive gridding either under Assumption 2.2 or under Assumption 2.3 (which give errors that are less tight) over the existing partition sets, whereas Assumption 2.1 will be associated to the generation of a uniform gridding [2]. Comparing Assumption 2.2 against Assumption 2.3, we will argue that the first ensures tighter error bounds (which leads to smaller cardinality of the partition), but requires error updates for possibly all the cells during each refinement step (whereas the second will perform just local updates) and is thus computationally more complex.

Let us discuss a few details about the adaptive grid generation. Consider for the sake of discussion an n-dimensional model. There are two main options over the shape of the cells of a grid [65, 78]: n-dimensional simplices, or Cartesian hyper-rectangles. The first option leads to the known Kuhn triangulation [78] and is widely used in numerical solution of partial differential equations. The second approach generates hyper-rectangular cells aligned with the main axes which, for our problem at hand, appears to be advantageous. Cartesian cells in fact better accommodate the subsequent step that involves the marginalization of probability laws, which generates the transition probability matrix $T_{\mathfrak{s}}$. Marginalization over general convex polygons (in particular simplices) is known to be a computationally expensive problem [71].

With focus on the refinement step, consider a single Cartesian cell. We are again presented with two options for its further partitioning: to replace the cell with 2^n smaller cells by splitting it along its centroid; or to replace the cell with 2 smaller cells by partitioning along one axis. The second approach is also known as variable resolution approach [65]. While the first approach decreases the error (which depends on the cell diameter, see Theorems 2.4 and 2.6) faster than the second, it is also associated with the generation of partitions with larger cardinality. Since we aim at economizing over the memory usage, we opt for the second option. Based on this choice, the convergence speed of the procedure is optimized by selecting the longest axis for the partitioning. This leads to the following result.

Proposition 2.2 *For an n-dimensional model, the convergence rate of the computed error bound for a partitioning procedure based on a Cartesian grid that proceeds by splitting the longest axis, is lower bounded by the factor* $\sqrt{1-\frac{3}{4n}}$.

The grid generation procedures are formally presented in Algorithm 2 for the uniform error, and in Algorithms 3 and 4 for the local ones. In the first case, the union of the partitioning sets is supposed to include the space $\mathcal{S}$. In the latter case, the initial partition can be any, and in particular it can coincide with the state space $\mathcal{S}$. Furthermore, notice the differences in step 4:, which leads to conclude that Algorithm 3 is geared towards an abstraction with the least number of states, whereas Algorithm 4 aims at faster generation time. More precisely, note than when we split a cell A_i along its main axis the related local error is reduced firstly because

of the decrease in its diameter δ_i, and secondly due to the possible reduction in the local Lipschitz constants $h(i,j)$ (other local errors may also be decreased because of the update of local Lipschitz constants). Hence, if we split a group of cells, as suggested in Algorithm 4, we possibly obtain a larger decrease of the error bound. The actual computation of the errors in the Algorithms can be performed based on any of the bounds in Section 2.3.3.

Algorithm 2 Generation of the uniform grid

Require: model $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$ under Assumption 2.1; error threshold ϵ
1: pick a partition diameter δ based on bound (2.3) in Theorem 2.2 and on the threshold ϵ
2: perform partitioning of $\mathcal{S}$ with uniformly-packed hypercubes
Ensure: $\mathcal{P}$, error $E = \epsilon$

Algorithm 3 Generation of the adaptive grid 1

Require: model $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$ under Assumption 2.2 over initial partition; error threshold ϵ
1: set initial partition over the hybrid state space $\mathcal{S}$
2: compute the error E according to (2.4) in Theorem 2.4
3: **if** $E > \epsilon$ **then**
4: refine the partition by splitting the single cell with maximum local error along its main axis
5: go to step 2
6: **end if**
Ensure: $\mathcal{P}$, error $E \leq \epsilon$

2.5.2 Marginalization

The generation of a grid and the choice of representative points for each of the resulting partition sets (let us recall that the choice of representative points is arbitrary), fully characterizes the state space $\mathcal{P}$ of the MC $\mathfrak{P}$. The second step in the generation of the abstraction involves the computation of the transition probability matrix T_p. This computation necessitates the marginalization of the stochastic kernel $T_{\mathfrak{s}}$, evaluated at the representative points, over the partition sets. While the complexity of the procedure highly depends on the shape of the kernels $T_{\mathfrak{s}}$, we have attempted to alleviate it 1) by working with hyper-rectangular partitions, 2) by exploiting vectorial representations of the quantities of interest, and 3) by leveraging as much as possible the sparsity of the manipulated matrices.

The sparsity of the generated transition probability matrix (number of its non-zero entries) depends on the kernels underlying $T_{\mathfrak{s}}$, particularly on their variance

Algorithm 4 Generation of the adaptive grid 2

Require: model $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$ under Assumption 2.3 over initial partition; error threshold ϵ

1: set initial partition over the hybrid state space $\mathcal{S}$
2: compute the error E according to (2.6) in Theorem 2.6
3: **if** $E > \epsilon$ **then**
4: refine the partition by splitting all the cells with error greater than threshold ϵ along the main axis
5: go to step 2
6: **end if**

Ensure: $\mathcal{P}$, error $E \leq \epsilon$

terms. Intuitively, a higher variance relates to a less sparse matrix, since the related probability law is more "spread out". More interestingly, there is a tradeoff between the sparsity of the transition probability matrix and its size, as a function of the variance terms in the underlying dynamics: indeed, both are increased by small variance terms, which are related both to dynamics that are spatially "concentrated" (and thus sparser), as well as to higher error bounds via the Lipschitz constants.

It is possible to use and to tune a tolerance threshold in the marginalization step, below which the transition probabilities are approximated with zero terms. As a last remark, notice that in the uniform partitioning case the marginalization procedure is greatly simplified, given the regular arrangement of the partition cells.

2.6 Experiments

This section develops a numerical computational benchmark to compare the presented algorithms for abstraction, in particular with focus on grid generation and marginalization steps. Additionally, a case study selects a SHS model and reflects on the choice of the error bounds and on the role of rescaling (cf. Section 2.3.3).

2.6.1 Computational Benchmark

Let us consider an n-dimensional linear, controlled stochastic difference equation

$$x(k+1) = Ax(k) + Bu(k) + w(k), \quad k \in \mathbb{N}_0,$$

where $w(k), k \geq 0$, is the process noise, taken to be Gaussian i.i.d. with zero mean and covariance W: $w(k) \sim \mathcal{N}(0, W)$. The initial condition $x(0)$ is independent of $w(k), k \geq 0$, and is Gaussian with zero mean and covariance X : $x(0) \sim \mathcal{N}(0, X)$. The input $u(k) \in \mathbb{R}^m, k \geq 0$, is designed according to a state feedback law mini-

mizing the following quadratic cost function of the state and of the input:

$$J = \lim_{N\to\infty} \frac{1}{N}\mathbb{E}\left(\sum_{k=0}^{N-1}\left(x^T(k)Qx(k) + u^T(k)Ru(k)\right)\right),$$

with properly-sized, positive (semi-)definite weighting matrices $Q \succeq 0$ and $R \succ 0$. The optimal control law for this stochastic control problem (also known as stochastic linear quadratic regulator) is given as a stationary linear state feedback $u(k) = Kx(k)$, where K represents the steady-state feedback gain matrix $K = -\left(R + B^T P_s B\right)^{-1} B^T P_s A$, and P_s is the solution of the following matrix equation:

$$P_s = Q + A^T P_s A - A^T P_s B \left(R + B^T P_s B\right)^{-1} B^T P_s A.$$

The closed loop system can be represented as

$$x(k+1) = (A + BK)x(k) + w(k), \quad k \in \mathbb{N},$$

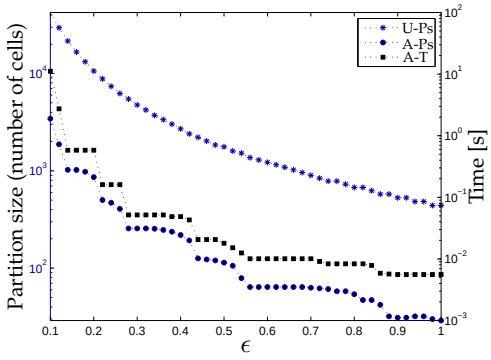
which is a stochastic difference equation evolving over $\mathbb{R}^n$. Given any point $x \in \mathbb{R}^n$ at any time, the distribution at the next time can be characterized by a transition probability kernel $T_x(\cdot|x) \sim \mathcal{N}((A + BK)x, W)$. The computation of the Lipschitz constant of this kernel can be adapted from [2] and involves the calculation of partial derivatives of the density.

With focus on the closed loop model, let us consider the probabilistic invariance problem on a safe set defined as $\mathcal{A} = [-1, 1]^n$, namely on a hypercube pointed at the origin, and over a time horizon $\mathbb{Z}_{N+1}$. For the cost function, we have selected the weighting matrices $Q = \mathbb{I}_n, R = \mathbb{I}_m$ (henceforth, $\mathbb{I}_l, l \in \mathbb{N}$, will denote the l-dimensional identity matrix). The control dimension has been chosen to be $m = 1$ and the time horizon has been fixed to $N = 10$. The state and control matrices A and B have been randomly generated for each experiment, and A has been further scaled so that $\max\{|\lambda_i(A)|, i \in \mathbb{N}_n\} = 1$, where $\lambda_i(A)$ denotes the i-th eigenvalue of matrix A. The variance of the initial condition has been selected to be $X = 10\,\mathbb{I}_n$.

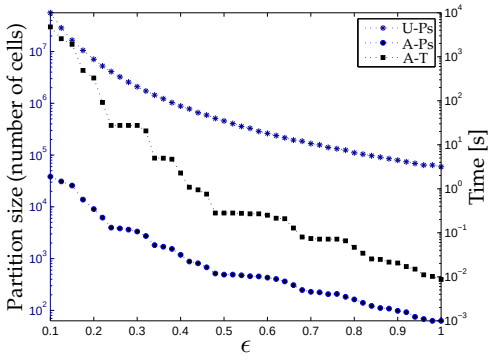
Grid Generation

Let us select a noise variance $W = 0.5\,\mathbb{I}_n$. Figure 2.1 compares the partition size (i.e., the number of grid cells) generated by Algorithm 3 for the adaptive gridding, and by Algorithm 2 for the uniform one, *given an (upper bound on the) abstraction error* ϵ for all the methods. The horizontal axis represents the threshold ϵ. The error is based on, respectively, Equation (2.4) in Theorem 2.4 and Equation (2.3) in Theorem 2.2. The local Lipschitz constants are computed based on (2.8) in Theorem 2.7. This batch of computations is performed for dimensions $n = 2, 3, 4$. As expected, for the adaptive algorithm the number of generated cells is always less than that for the uniform procedure. Furthermore, the number of cells becomes larger for smaller threshold values ϵ.

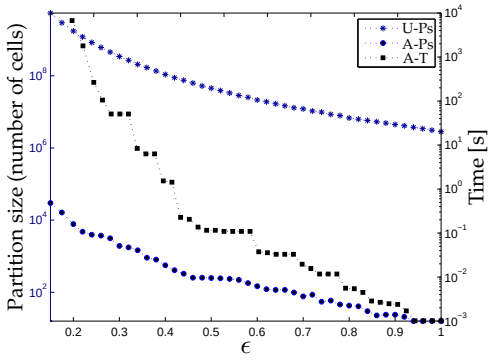
Figure 2.1 also plots the time required to generate the grid according to Algorithm 3 for the adaptive partitioning. The horizontal axis represents again the threshold



(a) $n = 2$

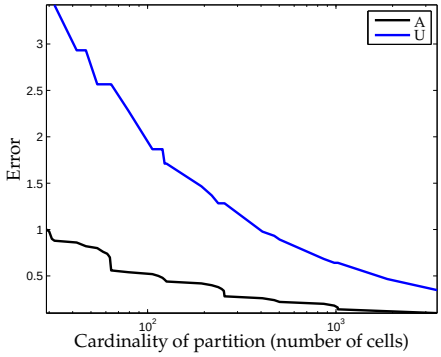


(b) $n = 3$

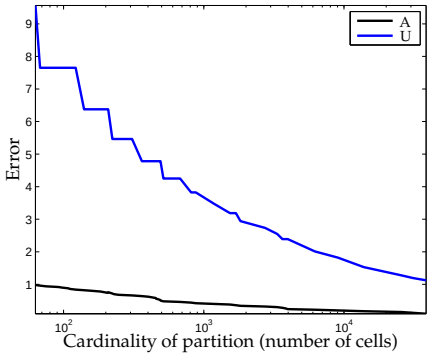


(c) $n = 4$

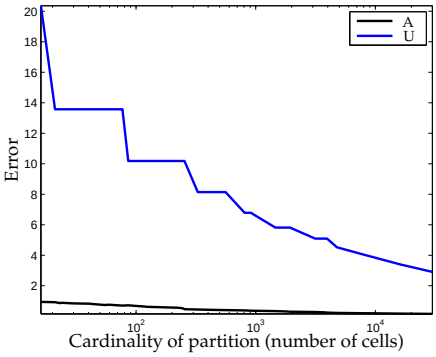
Figure 2.1: Numerical benchmark. For dimensions $n = 2$ (a), $n = 3$ (b), and $n = 4$ (c) and for different levels of the error threshold ϵ (horizontal axis), the plots display partition size (number of cells) generated by adaptive (Algorithm 3, labeled A-Ps) vs. uniform gridding (Algorithm 2, labeled U-Ps), as well as time required to generate the adaptive partitioning (Algorithm 3, labeled A-T). The results represent an average over 30 independent runs.



(a) $n = 2$

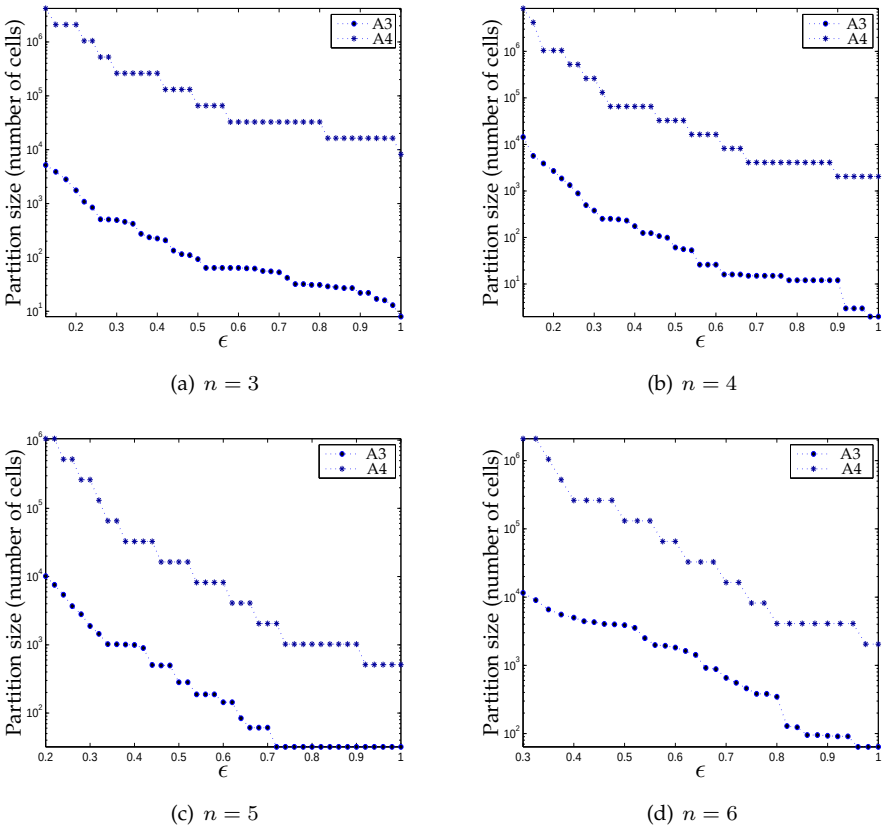


(b) $n = 3$



(c) $n = 4$

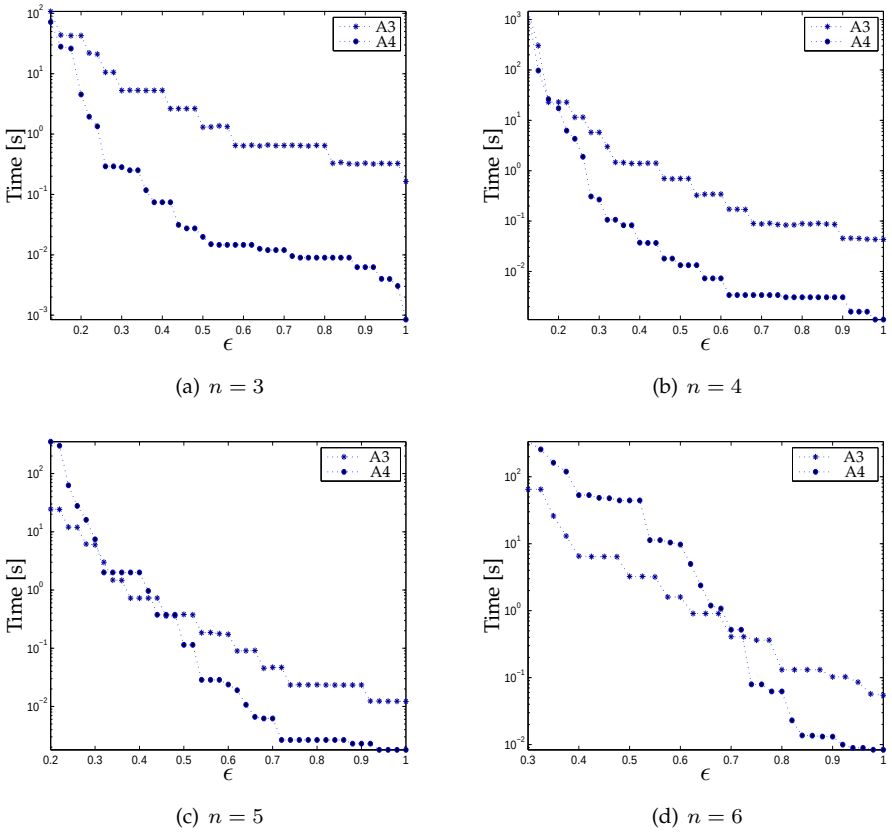
Figure 2.2: Numerical benchmark. Errors obtained selecting the same number of cells (same partition size), for dimensions $n = 2$ (a), $n = 3$ (b), $n = 4$ (c), for the adaptive gridding of Algorithm 3 (labeled A) vs. the uniform gridding of Algorithm 2 (labeled U).



(a) $n = 3$ (b) $n = 4$

(c) $n = 5$ (d) $n = 6$

Figure 2.3: Numerical benchmark. Partition size (number of cells), for dimensions $n = 3$ (a), $n = 4$ (b), $n = 5$ (c), and $n = 6$ (d), generated by the adaptive gridding of Algorithm 3 (labeled A3) vs. the adaptive gridding of Algorithm 4 (labeled A4), for different levels of error threshold ϵ.



(a) $n = 3$ (b) $n = 4$

(c) $n = 5$ (d) $n = 6$

Figure 2.4: Numerical benchmark. Computation time, for dimensions $n = 3$ (a), $n = 4$ (b), $n = 5$ (c), and $n = 6$ (d), required to generate the adaptive partitioning of Algorithm 3 (labeled A3) and the adaptive gridding of Algorithm 4 (labeled A4), for different levels of error threshold ϵ. The outcomes are obtained as the average over 30 independent runs.

ϵ on the error. This batch of computations is performed for dimensions $n = 2, 3, 4$ and the results are averaged over 30 runs. The discontinuities discernible in the plots are intrinsic to the implemented refinement algorithm for the adaptive partitioning. Notice that, as expected, the time is larger for smaller thresholds. Recall that for the uniform gridding the grid generation is a one-shot procedure and, as such, independent of the choice of ϵ.

Figure 2.2 compares the error obtained by generating the adaptive gridding with Algorithm 3 (see Theorem 2.4) against that obtained by generating the uniform gridding of Algorithm 2 (see Theorem 2.2), *given a fixed number of cells* for both methods (these values are represented on the horizontal axis). The experiments are again performed for dimensions $n = 2, 3, 4$. The local Lipschitz constants are computed based on (2.8). It is easily observed that the error associated to the uniform gridding approach is always higher than that associated to the adaptive method. (Notice that, for the probabilistic invariance problem under study, an error greater than one as obtained in the uniform case is not practically useful.)

Let us now select a noise variance $W = \mathbb{I}_n$ and benchmark the two adaptive gridding approaches. Figure 2.3 compares the number of cells generated by the adaptive gridding of Algorithm 3 vs. the adaptive gridding of Algorithm 4. This batch of experiments is performed for dimensions $n = 3, 4, 5, 6$. Similarly, Figure 2.4 compares the run time required for generating the adaptive partitioning of Algorithm 3 and the adaptive gridding of Algorithm 4. The outcomes of this batch of experiments are averages over 30 runs. Figure 2.3 confirms that, since the continuity bounds related to Assumption 2.3 are less tight, Algorithm 4 ends up requiring a larger number of cells, given any threshold ϵ. However (cf. Figure 2.4), Algorithm 4 works faster than Algorithm 3 in the partition refinement step, since it requires a local error update for the partitions with error greater than the given threshold, whereas Algorithm 3 requires in the worst case a global update of the error of each cell based on the largest obtained error. Thus, for smaller accuracy threshold ϵ and larger dimensions (and large number of generated cells) the method based on Algorithm 4 ends being the faster (Figure 2.4). Algorithm 3 can alternatively be made faster by substituting its refinement step (4:) with that of Algorithm 4 (notice that this, however, will not mitigate the possible global update of the error).

Marginalization

The time requirements for the marginalization procedure are recapitulated by the data on Figure 2.5. These figures are are obtained by taking the average over 100 independent runs, and display the marginalization time in relative terms versus the time required for the partitioning procedure, which has been discussed in the previous section. More specifically, we have focused on the adaptive gridding obtained according to Algorithm 3, and compared the time spent generating the grid to that needed in performing the marginalization step. The data display that the marginalization step requires more time, relative to the partitioning procedure, as the error level ϵ decreases (that is, as the abstraction precision increases). This trend is consistent regardless of the model size (n).

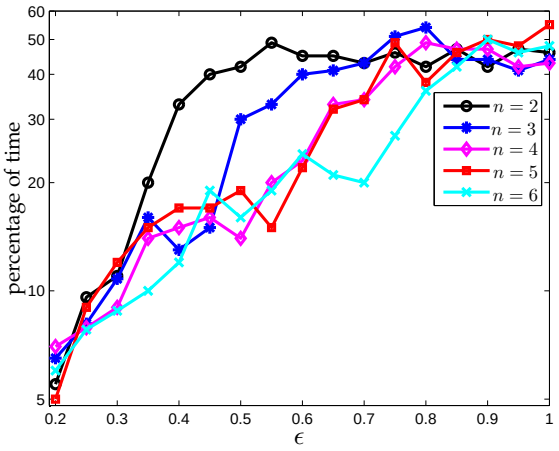


Figure 2.5: Time spent on the gridding procedure (according to Algorithm 3) as a percentage of the total time needed for grid generation and marginalization. The figure presents the results for various dimensions ($n = 2, 3, \ldots, 6$) as a function of threshold ϵ for the error. The required data are obtained by running averages over 100 independent runs.

Parameter	$k_a = k_d$	k_r	γ_r	k_p	γ_p
Value	0.001	0.0078	0.0039	$b\gamma_r, b = 11$	0.0007

Table 2.1: Parameters for the case study, taken from [19], and expressed in $[s^{-1}]$.

2.6.2 Case Study

This section applies the abstraction approach developed in this chapter to the study of a probabilistic invariance problem over a template chemical reaction network. We introduce a model for eukaryotic gene regulation. The stoichiometry (set of chemical reactions) underlying the system is the following:

$$\begin{array}{lll} (1) & D \xrightarrow{k_a} D^\star & D^\star \xrightarrow{k_d} D \\ (2) & D^\star \xrightarrow{k_r} M + D^\star & M \xrightarrow{k_p} P + M \\ (3) & M \xrightarrow{\gamma_r} \emptyset & P \xrightarrow{\gamma_p} \emptyset \end{array} \tag{2.18}$$

The reactants represent respectively the number of an inactive and active gene (D and $D^\star$ respectively), of m-RNA (M), and of a protein (P). There are three kinds of reactions: (1) conversion (between inactive and active state of the gene), (2) catalytic production (transcription of m-RNA and translation into a protein), and (3) degradation (of m-RNA and protein). The reaction and degradation rates (appearing above the arrows) are directly taken from [19] and summarized in Table 2.1.

The dynamics of chemically reacting environments can be described by the gen-

eral Chemical Master Equation (CME) [40], which has seldom an analytical solution and is usually quite hard to integrate. Alternatively, species dynamics in time are studied via the Stochastic Simulation Algorithm (SSA) [39, 40], a computational scheme that has recently attracted much research [8, 69]. Among the various approaches employed to approximate the SSA and thus expedite its running time, the works in [41, 51] have investigated a technique based on the use of second-order approximations, which assigns probabilistic dynamics (stochastic differential equations) to species concentrations. We leverage this latter approach below.

Global Stochastic Approximation

Let us introduce the state vector $x = [D, D^\star, M, P]^T$ describing the concentration of the reactants present in (2.18). Since the new variables are indeed concentrations, they are non-negative reals, rather than natural numbers as in (2.18). We can associate to this state continuous dynamics over time, which can be characterized by a stochastic differential equation of the form [41, 51]

$$dx = f(x)dt + \sigma(x)dw.$$

Time is discretized with constant sampling interval Δ, according to a Euler-Maruyama, first-order scheme [52], obtaining:

$$x(k+1) = x(k) + f(x(k))\Delta + \sigma(x(k))\sqrt{\Delta}w(k),$$

where $f(x) = Ax$ and

$$A = \begin{bmatrix} -k_a & k_d & 0 & 0 \\ k_a & -k_d & 0 & 0 \\ 0 & k_r & -\gamma_r & 0 \\ 0 & 0 & k_p & -\gamma_p \end{bmatrix},$$

and

$$\sigma(x) = \begin{bmatrix} \sqrt{k_a D + k_d D^\star} & 0 & 0 \\ -\sqrt{k_a D + k_d D^\star} & 0 & 0 \\ 0 & \sqrt{k_r D^\star + \gamma_r M} & 0 \\ 0 & 0 & \sqrt{k_p M + \gamma_p P} \end{bmatrix}.$$

The noise term is given by $w(k) = [w_1(k), w_2(k), w_3(k)]^T$, where $w_i(k), i = 1, 2, 3$ and $k \in \mathbb{N}_0$, are independent standard Gaussian random variables, which are also independent of the initial condition of the process. The steady-state values for the dynamics are directly computed as in [51]:

- $P_{ss} = 65\,[nM]$,
- $M_{ss} = \frac{\gamma_p}{k_p} P_{ss} = 1.0606\,[nM]$,
- $D_{ss} = D^\star_{ss} = \frac{\gamma_r}{k_r} M_{ss} = \frac{\gamma_r}{k_r}\frac{\gamma_p}{k_p} P_{ss} = \frac{\gamma_p}{bk_r} P_{ss} = 0.5303\,[nM]$.

Notice that, given the parameters choice in Table 2.1, the steady-state concentrations assume values that span different dimensions. This will later motivate the use of state-space rescaling.

Since the dynamics of D and $D^\star$ are coupled, it is possible to eliminate either of the variables – here we remove D. The equality $k_a = k_d$ leads to the following discrete-time probabilistic dynamical system:

$$\begin{cases} x_1(k+1) = (1-2k_d\Delta)x_1(k) + 2k_d\Delta D^\star_{ss} + \sqrt{2k_d\Delta D^\star_{ss}}w_1(k) \\ x_2(k+1) = k_r\Delta x_1(k) + (1-\gamma_r\Delta)x_2(k) + \sqrt{k_r\Delta x_1(k) + \gamma_r\Delta x_2(k)}w_2(k) \\ x_3(k+1) = k_p\Delta x_2(k) + (1-\gamma_p\Delta)x_3(k) + \sqrt{k_p\Delta x_2(k) + \gamma_p\Delta x_3(k)}w_3(k), \end{cases} \tag{2.19}$$

where we have denoted $[x_1, x_2, x_3]^T = [D^\star, M, P]^T$.

Based on the recursive expression in (2.19), the associated conditional probability density function can be defined as:

$$t_x(\bar{x}|x) = t_x(\bar{x}_1|x_1)t_x(\bar{x}_2|x_1, x_2)t_x(\bar{x}_3|x_2, x_3), \tag{2.20}$$

where

$$\begin{aligned} &t_x(\bar{x}_1|x_1) \sim \mathcal{N}(\mu_1(x_1), \sigma_1^2), \\ &t_x(\bar{x}_2|x_1, x_2) \sim \mathcal{N}(\mu_2(x_1, x_2), \sigma_2^2(x_1, x_2)), \\ &t_x(\bar{x}_3|x_2, x_3) \sim \mathcal{N}(\mu_3(x_2, x_3), \sigma_3^2(x_2, x_3)), \end{aligned}$$

and

$$\begin{aligned} \mu_1(x_1) &= (1-2k_d\Delta)x_1 + 2k_d\Delta D^\star_{ss}, & \sigma_1^2 &= 2k_d\Delta D^\star_{ss}, \\ \mu_2(x_1, x_2) &= k_r\Delta x_1 + (1-\gamma_r\Delta)x_2, & \sigma_2^2(x_1, x_2) &= k_r\Delta x_1 + \gamma_r\Delta x_2, \\ \mu_3(x_2, x_3) &= k_p\Delta x_2 + (1-\gamma_p\Delta)x_3, & \sigma_3^2(x_2, x_3) &= k_p\Delta x_2 + \gamma_p\Delta x_3. \end{aligned}$$

It can be observed that, due to differences in variables and parameters ranges, the domain of the density function in (2.20) is compact along the first two variables, while being stretched along the third one. Such an asymmetric shape of the probability density calls for the use of a rescaling by coordinate transformation. The quantities in (2.19)-(2.20) characterize the model of reference for the remainder of the case study.

Probabilistic Invariance for Global Stochastic Approximation: In order to introduce a probabilistic invariance problem for the model of interest, we select a hyper-box around the above steady state values for the variables x_1, x_2, x_3. We plan to assess the probabilistic invariance of the process therein, over a finite time horizon. The hyper-box is parameterized by the quantities r_1, r_2, and r_3:

$$\left|\frac{x_1 - D^\star_{ss}}{D^\star_{ss}}\right| \leq r_1, \quad \left|\frac{x_2 - M_{ss}}{M_{ss}}\right| \leq r_2, \quad \left|\frac{x_3 - P_{ss}}{P_{ss}}\right| \leq r_3.$$

This box is set to show 10% variations around the steady state values, i.e. $r_i = 0.1, i = 1, 2, 3$.

Let us define the abstraction errors computed based on global and local version of (2.14) under Assumptions 2.1 and 2.2, respectively. More precisely, let $E_i, i = 1, 2, 3$, represent the abstraction error using upper bounds (2.8)-(2.10), respectively. Furthermore, let us consider the upper bounds E_4, E_5 obtained via (2.12), (2.15). In order to provide a comparison for these different bounds, two sets of experiments have been set up. Both employ uniform Cartesian gridding based on hyper-rectangular partition sets. In the first set of experiments the edges of the partition sets have been selected to be proportional to the length of the edges of the safe set, whereas in the second set of experiments the partition sets have been chosen to be close to cubic cells.

The upper bounds in (2.8)-(2.10) are tailored to Cartesian partitions as follows. Suppose the uniform grid is made up of partition sets characterized by a vector $\boldsymbol{\delta}$ containing its edges. The grid size δ equals to the Euclidean norm of $\boldsymbol{\delta}$. Assuming that P is a diagonal matrix, whose elements are proportional to the entries of $\boldsymbol{\delta}$, the upper bounds expressed in (2.8)-(2.10) are simplified as follows for the local form (Assumption 2.2):

$$\begin{cases} k_1(i,j) = \delta \max_{s\in A_i, \bar{s}\in A_j} \left\| \frac{\partial t_{\bar{s}}}{\partial s}(\bar{s}|s) \right\|, \\ k_2(i,j) = \left\| P^{-1}\boldsymbol{\delta} \right\| \max_{s\in A_i, \bar{s}\in A_j} \left\| \frac{\partial t_{\bar{s}}}{\partial s}(\bar{s}|s) P \right\|, \\ k_3(i,j) = \max_{s\in A_i, \bar{s}\in A_j} \left| \frac{\partial t_{\bar{s}}}{\partial s}(\bar{s}|s) \right| \boldsymbol{\delta}; \end{cases} \tag{2.21}$$

and as follows for the global form (Assumption 2.1):

$$\begin{cases} k_1 = \delta \max_{s,\bar{s}\in \mathcal{A}} \left\| \frac{\partial t_{\bar{s}}}{\partial s}(\bar{s}|s) \right\|, \\ k_2 = \left\| P^{-1}\boldsymbol{\delta} \right\| \max_{s,\bar{s}\in A} \left\| \frac{\partial t_{\bar{s}}}{\partial s}(\bar{s}|s) P \right\|, \\ k_3 = \max_{s,\bar{s}\in \mathcal{A}} \left| \frac{\partial t_{\bar{s}}}{\partial s}(\bar{s}|s) \right| \boldsymbol{\delta}. \end{cases} \tag{2.22}$$

In order to elucidate the outcomes in Tables 2.2 and following, let us discuss the computational overhead related to the different bounds. For the bounds (2.8)-(2.10), the number of optimizations in the local form of (2.21) and in the global form of (2.22) are m^2 and one, respectively. The computation of abstraction error (2.12) in local form can be simplified by using (2.13). Both (2.12) and (2.13) have the same complexity order as the first three upper bounds. Finally, the computation of E_5 based on the local and global form of (2.15) requires respectively m optimizations and a single one, however it also needs an integration step: this will lead to higher computational times compared to the errors $E_1 - E_4$.

The time horizon N has been set to be equal to 10 for both experiments. The first set of runs (edges of the partition sets are proportional to the length of the edges of the safe set) is performed with the following specifications:

- length of edges of the (three dimensional) safe set: $(0.1061, 0.2121, 13)$;

Error bound	E_1	E_2	E_3	E_4	E_5
Inequality used for bound	(2.8)	(2.9)	(2.10)	(2.12)	(2.15)
Global form (Assumption 2.1)	7095.1	1376.5	799.3	230.9	13.6
Local form (Assumption 2.2)	1577.3	283.4	167.3	48.6	13.5
Number of optimizations (local form)	4096	4096	4096	8192	64
Computation time	3 (m)	3 (m)	3 (m)	6 (m)	17.9 (h)

Table 2.2: Global stochastic approximation: error comparison for the first set of experiments, using a uniform grid with partition sets that are proportional to the length of the edges of the safe set. The computational overhead for the bounds in local form is also reported.

- length of edges of the partition cells: $\boldsymbol{\delta} = (0.0265, 0.0530, 3.25)$;
- resulting number of bins per dimension: $(4, 4, 4)$;
- resulting total number of cells: $m = 4^3 = 64$;
- resulting partitions diameter: $\delta = 3.2505$.

Table 2.2 summarizes abstraction errors for this set of parameters. The number of optimization steps, as well as the optimization time, has been reported as a measure of the complexity in the error computation.

The second set of runs (partition sets are close to cubic cells) is performed with the following parameters:

- length of edges of the (three dimensional) safe set: $(0.1061, 0.2121, 13)$;
- length of edges of the partition cells: $\boldsymbol{\delta} = (0.1061, 0.2101, 0.3171)$;
- resulting number of bins per dimension: $(1, 1, 41)$;
- resulting total number of cells: $m = 41$;
- resulting partitions size: $\delta = 0.3949$.

Table 2.3 summarizes abstraction errors for this second set of parameters. Note that in this second case we have used a lower number of cells, and at the same time obtained lower abstraction errors and shorter run times. This is due to the underlying ill-conditioned dynamics and to a safe set that is stretched along one axis. This outcome shows the importance of rescaling and of the selection of cubic partition cells for the uniform grid.

For both batches of experiments we have considered relatively coarse partitions in order to clearly highlight differences in the computed error bounds. As such, the errors are not practically useful since, being larger than 1, they cannot be used in the approximation of probabilistic quantities. Of course, since they monotonically converge to zero as the partition size δ goes to zero, the error bounds can be simply reduced by considering finer partitions, at the expense of longer optimization times.

Error bound	E_1	E_2	E_3	E_4	E_5
Global form (Assumption 2.1)	864.3	402	300	219.2	13.6
Local form (Assumption 2.2)	90.3	42.4	33.5	24.1	7.8
Number of optimizations (local form)	1681	1681	1681	3362	41
Computation time	1.5 (m)	1.5 (m)	1.5 (m)	3 (m)	16.7 (h)

Table 2.3: Global stochastic approximation: error comparison for the second set of experiments, using a uniform grid with partition sets that are chosen to be cubic. The computational overhead for the bounds in local form is also reported.

We now test the adaptive partitioning approach under rescaling. The direct implementation of Algorithms 3 and 4 leads to some computational issues: if the algorithms are initialized over a uniform grid as in the first setup above, they proceed splitting the partitions along the longest edge in order to try obtaining cubic-shaped cells. On the other hand, if initialization of the algorithm is set over cubic partition cells, the system dynamics along the shortest direction tend to be lost. To cope these difficulties we have performed a rescaling of the state space, so that all the dynamics evolve in a comparable range. More precisely, consider the affine map $x = Py + Q$ with matrices

$$P = \begin{bmatrix} r_1 & 0 & 0 \\ 0 & r_2 & 0 \\ 0 & 0 & r_3 \end{bmatrix}, \qquad Q = \begin{bmatrix} D^\star_{ss} \\ M_{ss} \\ P_{ss} \end{bmatrix},$$

which projects the safe set $\mathcal{A}$ to the cube $[-1, 1]^3$. The dynamics in the new state space become:

$$\begin{cases} y_1(k+1) = (1 - 2k_d\Delta)y_1(k) + \frac{\sqrt{\Delta}}{r_1}\sqrt{2k_d D^\star_{ss}}w_1(k) \\ y_2(k+1) = \frac{r_1}{r_2}\gamma_r\Delta y_1(k) + (1-\gamma_r\Delta)y_2(k) + \frac{\sqrt{\Delta}}{r_2}\sqrt{k_r r_1 y_1 + \gamma_r r_2 y_2 + 2\gamma_r M_{ss}}w_2(k) \\ y_3(k+1) = \frac{r_2}{r_3}\gamma_p\Delta y_2(k) + (1-\gamma_p\Delta)y_3(k) + \frac{\sqrt{\Delta}}{r_3}\sqrt{k_p r_2 y_2 + \gamma_p r_3 y_3 + 2\gamma_p P_{ss}}w_3(k). \end{cases}$$

Over the new coordinates we have implemented Algorithm 3, which hinges on Assumption 2.2. As an outcome of this Algorithm, Figures 2.6(a), 2.6(b), and 2.6(c) present the three level sets $p_{x_0}(\mathcal{A}) = 0.0015$, $p_{x_0}(\mathcal{A}) = 0.0013$, and $p_{x_0}(\mathcal{A}) = 0.0011$ respectively, for any $x_0 \in \mathcal{A}$ (these figures are in accord with the uniform results obtained above). The obtained number of cells is 15236 for an error $E_1 = 7.68$ (based on (2.8)), which is as expected lower than that in Tables 2.2 and 2.3. An a-posteriori computation of the error bound based on (2.12) results in the quantity $E_4 = 1.94$.

Stochastic Hybrid Approximation

We next present a simplification of the probabilistic dynamics in (2.19)-(2.20) as a stochastic hybrid model, as defined in Section 2.4. Recall that the conditional

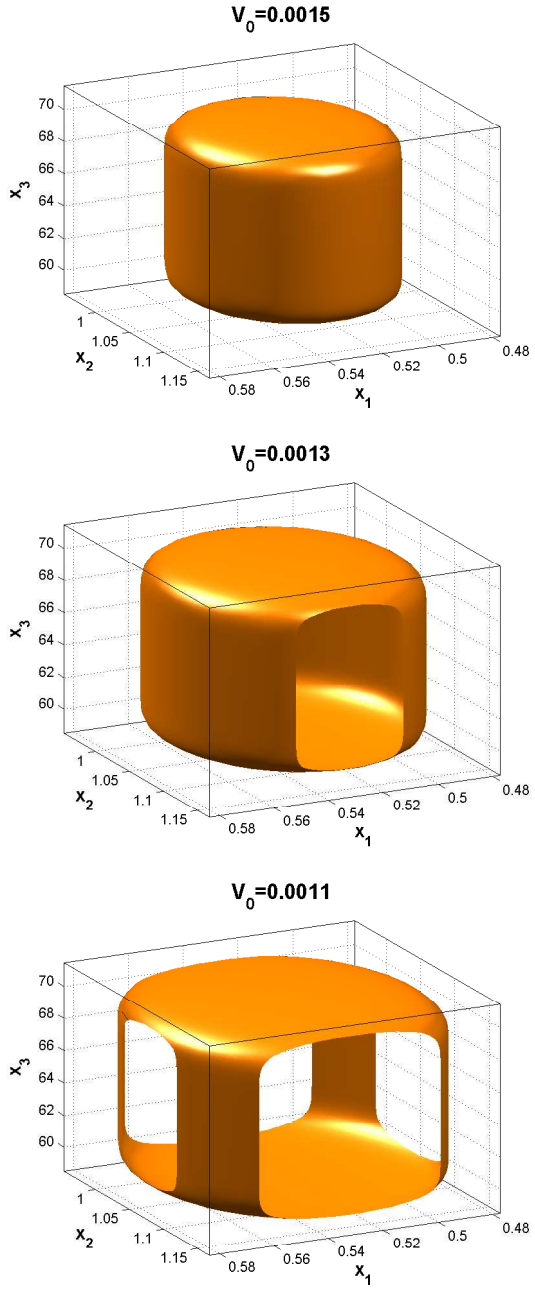


Figure 2.6: Level set $p_{x_0}(\mathcal{A}) = 0.0015$, $p_{x_0}(\mathcal{A}) = 0.0013$, and $p_{x_0}(\mathcal{A}) = 0.0011$ for original (non-hybrid) model, performed after rescaling under affine mapping, and with an adaptive approach based on Algorithm 3.

density of x_1 is Gaussian as

$$t_x(\bar{x}_1|x_1) \sim \mathcal{N}(\mu_1(x_1), \sigma_1^2),$$

where the mean is an affine function exclusively of the conditional variable x_1, and the variance is a constant, namely:

$$\mu_1(x_1) = (1 - 2k_d\Delta)x_1 + 2k_d\Delta D_{ss}^\star, \quad \sigma_1^2 = 2k_d\Delta D_{ss}^\star.$$

As such, the conditional density t_x is independent of x_2, x_3. This suggests performing a simplification of the dynamics over the sole variable x_1. Let us introduce two sets that partition $\mathbb{R}$, the domain of x_1:

$$q_1 = \{x_1 \leq D_{ss}^\star\}, \quad q_2 = \{x_1 > D_{ss}^\star\}.$$

The first set q_1 indicates that the concentration of active genes is lower than its steady-state (we call this the "inactive" mode), whereas the second set q_2 refers to a concentration of active genes that is higher than its steady-state (this is named the "active" mode). These two occurrences make up the discrete modes as $\mathcal{Q} = \{q_1, q_2\}$. Let us additionally select two generic points, one for each of the two modes $l_1 \in q_1, l_2 \in q_2$. We associate to both modes the continuous domain $\mathbb{R}^2$, as needed for the dynamics of the two variables x_2, x_3.

We characterize the discrete probability matrix for the transitions between modes, namely $T_q(q_j|(q_i, x)), x = (x_2, x_3)$, as follows:

$$T_q(q_1|(q_i, x)) = \mathbb{P}\{\bar{x}_1 \leq D_{ss}^\star|q_i\} = \phi_{\sigma_i}(D_{ss}^\star - \mu_i(l_i)), \tag{2.23}$$

where $\phi_\sigma(\alpha) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\alpha^2}{2\sigma^2}\right)$ is the Gaussian density function with zero mean and standard deviation σ. Then we explicitly obtain:

$$\begin{aligned}
&T_q(q_1|(q_1, x)) = \phi_{\sigma_1}(D_{ss}^\star - \mu_1(l_1)) = \phi_{\sigma_1}((1 - 2k_d\Delta)(D_{ss}^\star - l_1)) \doteq p_1,\\
&T_q(q_1|(q_2, x)) = \phi_{\sigma_2}(D_{ss}^\star - \mu_2(l_2)) = \phi_{\sigma_2}((1 - 2k_d\Delta)(D_{ss}^\star - l_2)) \doteq p_2,\\
&T_q(q_2|(q_1, x)) = \mathbb{P}\{\bar{x}_1 > D_{ss}^\star|q_1\} = 1 - p_1,\\
&T_q(q_2|(q_2, x)) = \mathbb{P}\{\bar{x}_1 > D_{ss}^\star|q_2\} = 1 - p_2.
\end{aligned}$$

Notice that the probabilities in above equations (see in particular the second equality in (2.23)) depend on the arbitrary choice of the points $l_i \in q_i, i = 1, 2$. We select such points l_1, l_2 so that the variable x_1 has the same conditional expectation before and after the introduction of the partition sets q_1, q_2. More precisely, with reference to equations (2.19)-(2.20), given an $x_1 \in \mathbb{R}$, for any $\bar{x}_1 \in \mathbb{R}$,

$$\mathbb{E}\{\bar{x}_1|x_1\} = \mu(x_1) = (1 - 2k_d\Delta)x_1 + 2k_d\Delta D_{ss}^\star,$$

whereas for any $\bar{q} \in \mathcal{Q}$, and given the selected $l_i \in q_i$,

$$\begin{aligned}
&\mathbb{E}\{\bar{q}|q_1\} = \mathbb{E}\{\bar{q}|l_1\} = p_1 l_1 + (1 - p_1)l_2,\\
&\mathbb{E}\{\bar{q}|q_2\} = \mathbb{E}\{\bar{q}|l_2\} = p_2 l_1 + (1 - p_2)l_2.
\end{aligned}$$

This leads to the following two equations, which are nonlinear since p_1, p_2 are nonlinear functions of l_1, l_2:

$$\begin{cases} p_1 l_1 + (1-p_1) l_2 = (1 - 2k_d\Delta) l_1 + 2k_d \Delta D^\star_{ss} \\ p_2 l_1 + (1-p_2) l_2 = (1 - 2k_d\Delta) l_2 + 2k_d \Delta D^\star_{ss}. \end{cases}$$

Applying the following variable transformation:

$$\begin{cases} \ell_1 = (1 - 2k_d\Delta)(D^\star_{ss} - l_1)/\sigma \\ \ell_2 = (1 - 2k_d\Delta)(D^\star_{ss} - l_2)/\sigma, \end{cases}$$

we obtain

$$\begin{cases} \ell_1 \phi_1(\ell_1) + \ell_2 (1 - \phi_1(\ell_1)) = (1 - 2k_d\Delta)\ell_1 \\ \ell_1 \phi_1(\ell_2) + \ell_2 (1 - \phi_1(\ell_2)) = (1 - 2k_d\Delta)\ell_2. \end{cases}$$

This set of nonlinear equations has a trivial solution $\ell_1 = \ell_2 = 0$, which is not interesting. Its second solution can be computed by the following recursive scheme:

$$\begin{bmatrix} \ell_1^{k+1} \\ \\ \ell_2^{k+1} \end{bmatrix} = (1 - 2k_d\Delta) \begin{bmatrix} \phi_1(\ell_1^k) & 1 - \phi_1(\ell_1^k) \\ \\ \phi_1(\ell_2^k) & 1 - \phi_1(\ell_2^k) \end{bmatrix}^{-1} \begin{bmatrix} \ell_1^k \\ \\ \ell_2^k \end{bmatrix},$$

which, for the above nominal values, leads to the quantities $\ell_1 = 3.0902, \ell_2 = -3.0902$. These correspond to the points $l_1 = 0.4295, l_2 = 0.6311$, and finally to the probabilities $p_1 = 0.9990, p_2 = 0.0010$. These values fully characterize the discrete kernel T_q. Further, the continuous kernels $T_x = T_r$ can be directly derived, as done for (2.19)-(2.20), from the following system of stochastic difference equations:

$$\begin{cases} x_2(k+1) = k_r \Delta q(k) + (1 - \gamma_r \Delta) x_2(k) + \sqrt{k_r \Delta q(k) + \gamma_r \Delta x_2(k)}\, w_2(k) \\ x_3(k+1) = k_p \Delta x_2(k) + (1 - \gamma_p \Delta) x_3(k) + \sqrt{k_p \Delta x_2(k) + \gamma_p \Delta x_3(k)}\, w_3(k). \end{cases} \tag{2.24}$$

Probabilistic Invariance for Stochastic Hybrid Approximation: We implement a uniform gridding with partition cells that are proportional to the edges of the safe set. Note that the safe set in the hybrid state space is made up of two identical sets A_{q_1}, A_{q_2} for modes q_1, q_2, respectively. As discussed before, both sets coincide over $\mathbb{R}^2$ and are defined as rectangles spanning a 10% variation from the steady state values of the variables x_2, x_3. The analysis run is performed with the following parameters, defined for each of the two modes q_1, q_2:

- length of edges of the (two dimensional) safe set: $(0.2121, 13)$;
- length of edges of the partition cells: $\boldsymbol{\delta} = (0.0424, 2.60)$;
- resulting number of bins per dimension: $(5, 5)$;
- resulting total number of cells: $2 \times 5^2 = 50$;

Error bound	E_1	E_2	E_3	E_4	E_5
Global form (Assumption 2.1)	3352	1420.1	1003.8	364.7	15.59
Local form (Assumption 2.2)	614.89	250.18	178.95	66.04	15.58
Computation time (sec.)	37	38	42	89	478

Table 2.4: Stochastic hybrid approximation: error comparison for the first set of experiments, based on a uniform grid. The computational overhead for the bounds in local form is also reported.

- resulting partition size $\delta = 2.6003$.

Notice that the safe set and the partition sets are mode invariant. Let us denote the safe set by $\mathcal{A}$ and its partition by $\cup_{i=1}^{m} A_i$. Since the probability distribution T_q does not depend on the state x, we have that $k_q(q, \bar{q}, i) = 0$. Furthermore, since $t_r(\bar{x}|(q, x), \bar{q}) = t_x(\bar{x}|(q, x))$, then

$$k_r(q, \bar{q}, i, j) = k_x(q, i, j) \geq |t_x(\bar{x}|(q, x)) - t_x(\bar{x}|(q, x'))|, \quad \forall x, x' \in A_i, \bar{x} \in A_j.$$

These observations simplify local error computations to:

$$E_{q,i} = 2N \sum_{j=1}^{m} k_x(q, i, j)\mathscr{L}(A_j), \quad q \in \{q_1, q_2\}, i \in \mathbb{N}_n.$$

The local form of the abstraction error is $\max\{E_{q,i} | q \in \{q_1, q_2\}, i \in \mathbb{N}_n\}$, while its global form is $E = 2N\mathscr{L}(\mathcal{A})k_x$. The upper bound $k_x(q, i, j)$ is computable as done for the non-hybrid case. The computation of E_5 is simplified to $E_5 = \max_{q,i} 2NH(q, i)$ where

$$\int_{\mathcal{A}} |t_x(\bar{x}|(q, x)) - t_x(\bar{x}|(q, x'))| d\bar{x} \leq H(q, i), \quad \forall x, x' \in A_i,\ q \in \{q_1, q_2\}.$$

Table 2.4 reports the abstraction error for the chosen set of parameters. For the bounds in the global form, two maximization problems (one per mode) need to be solved. The optimization time for the local error computations is also reported in the table.

Again with the uniform discretization approach, let us increase number of bins per dimension (from 5) to 30 and compute a more accurate approximation for the safety problem. The resulting total number of cells is thus $2 \times 30^2 = 1800$, which is dimensionally higher than the previous instance, as well as than the experiments in the global case presented in Tables 2.2 and 2.3. This of course comes at a computational cost (cf. with optimization time in Table 2.4). The output of the safety invariance problem is presented in Figures 2.7(a) and 2.7(c).

As a second step, we have implemented Algorithm 3 to generate an adaptive grid. The associated errors have been computed based on local Lipschitz constants of the distribution, using the error quantification of Theorem 2.8. Notice that the reset kernel does not depend on the next mode and coincides with t_x, which

Error bound	E_1	E_2	E_3	E_4	E_5
Global form (Assumption 2.1)	558.67	236.68	167.30	111.56	15.59
Local form (Assumption 2.2)	53.67	28.58	20.69	20.04	8.36
Computation time (hours)	5.84	6.42	8.93	15.98	16.73

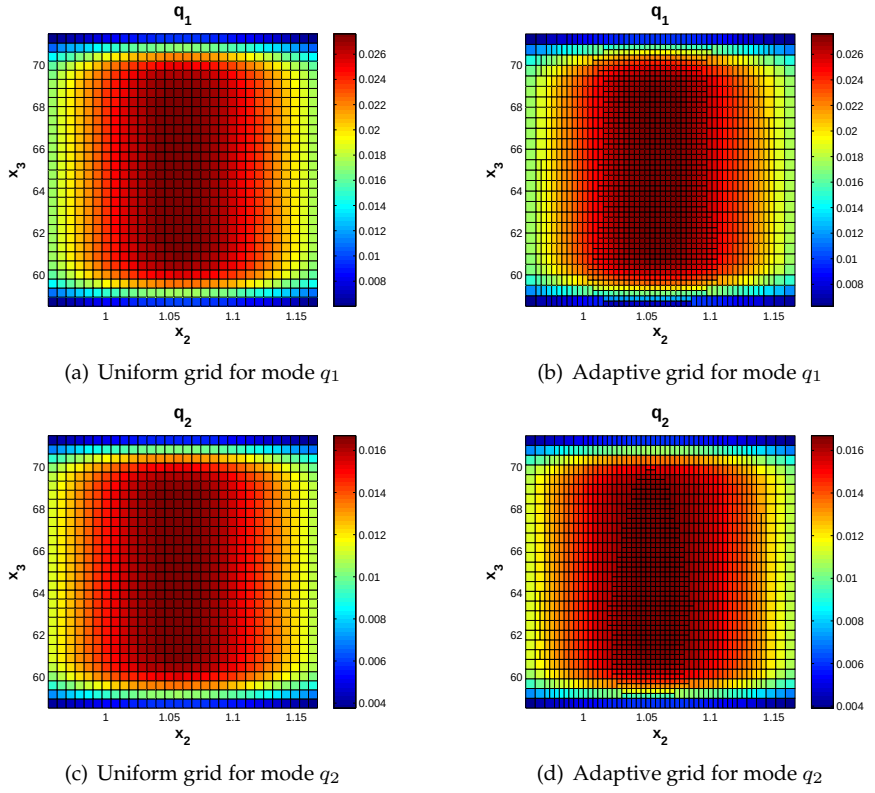
Table 2.5: Stochastic hybrid approximation: error comparison for the second set of experiments, which again use a uniform grid but with higher precision. The computational overhead for the bounds in local form is also reported.

does not imply $h_r(q, \bar{q}, i, k) = h_x(q, i, k)$ in general. Consequently, the adaptive grid is in general mode dependent. Figures 2.7(b) and 2.7(d) present the adaptive grid together with the invariance probability for points over the state space of the stochastic hybrid system. The grid has been generated for an error E_1 (based on (2.8)) equal to 14.47 and has resulted in a total of 3504 cells. The run time has amounted to 87 seconds. An a-posteriori analysis of the adaptive grid, based on (2.15), insures an improved abstraction error equal to $E_5 = 5.21$.

2.7 Conclusions

In this chapter we have presented an abstraction procedure based on a partitioning of the state space, and discussed an adaptive gridding generation technique exploiting a local formula-based error computation and a state-space rescaling. By conforming to the underlying dynamics of the model, the method alleviates the "curse of dimensionality" that is in general related to partitioning procedures. While the focus of this chapter has been on the study of probabilistic safety over a finite horizon, the technique can be employed in the formal abstraction and verification of stochastic models over more general probabilistic properties, by means of model checkers.

In the next chapter we extend the results towards more general dynamics, i.e. partially degenerate stochastic systems which do not satisfy continuity assumptions proposed in this chapter. Chapter 5 presents extension of the current approach to controlled Markov processes. In Chapter 7 we discuss the software tool FAUST^2 which is developed based on the presented theoretical results of this chapter.



(a) Uniform grid for mode q_1

(b) Adaptive grid for mode q_1

(c) Uniform grid for mode q_2

(d) Adaptive grid for mode q_2

Figure 2.7: Visualization of the quantity $p_{x_0}(\mathcal{A})$ (probabilistic invariance), over the grid points of the two discrete modes of the stochastic hybrid model. On the left, outcomes obtained using uniform grids, whereas on the right, outcomes obtained using adaptive grids.

3 CHAPTER

Probabilistic Invariance of Partially-Degenerate Stochastic Systems

This chapter is concerned with the computation of probabilistic invariance (safety) over a finite horizon for partially-degenerate stochastic (that is, mixed deterministic-stochastic) processes evolving in discrete time over a continuous state space. The models of interest consist of two fully coupled dynamical parts: the first part is described by deterministic maps (vector fields), whereas the second depends on probabilistic dynamics that are characterized by stochastic kernels. In contrast with a fully probabilistic approach (which is possible since the two dynamical components are coupled), we show in this chapter that the probabilistic invariance problem can be characterized – and thus computed – in two sequential steps: the first is a simple deterministic reachability analysis, which is then followed by a probabilistic invariance problem depending on the outcome of the first step. This characterization leads to implementation advantages over a fully probabilistic approach and allows synthesizing a computational algorithm with explicit error bounds.

3.1 Introduction

In this chapter we deal with models with explicit mixed deterministic-stochastic dynamics, which naturally arise in a number of situations or application domains. For instance, this feature is expected in models with variables that take values within ranges that are dimensionally different.

Mixed deterministic-stochastic models are composed of two complementary sets of variables, possibly coupled between each other. The first set of variables has

associated dynamics that depend on deterministic maps, namely vector fields. The complement set has dynamics characterized by a stochastic kernel.

A naïve approach to the probabilistic invariance problem for mixed deterministic-stochastic models would merely tackle it as a safety verification instance over degenerate systems (by degenerate systems we refer to probabilistic laws that are concentrated deterministically, i.e. whose support consists of a single point). This would not only be a computationally expensive solution, but also lead to the inability to leverage computational techniques that apply exclusively to non-degenerate systems, e.g. the techniques developed in the previous chapter.

This chapter originally shows that the probabilistic invariance problem can be separated into two parts: a deterministic reachability analysis, and a probabilistic invariance problem that depends on the outcome of the first. Deterministic reachability analysis is a rather mature field of research with ample software tool support, whereas the second problem can harvest recent developments [4, 18, 54]. We argue that this decomposition approach can lead to computational improvements – for instance, whenever the first deterministic problem yields a "false" outcome (i.e., no states are deterministically safe over the given time horizon), no further probabilistic invariance calculation is necessary. This advantage of the proposed approach also leads to an approximation algorithm to compute the quantity of interest with explicit error bounds.

This chapter is structured as follows. Section 3.2 introduces the model class and quickly reviews the invariance problem from previous chapter. Section 3.3 focuses on the properties of the value functions that characterize probabilistic invariance. Section 3.4 puts forward an approximation scheme for the computation of the desired quantities based on the discretization of the state space, and explicitly characterizes its error. Section 3.5 specializes the error to affine deterministic dynamics with polytopic invariant sets. Section 3.6 applies the results to a case study from Systems Biology.

3.2 Preliminaries

We consider discrete time Markov processes, described in the previous chapter (Section 2.2.1), which are characterized by the state space $\mathcal{S}$ and stochastic kernel $T_{\mathfrak{s}}$. The class of Markov processes is equivalent to the class of discrete-time dynamical systems (Kallenberg [48]) characterized by the dynamical equation

$$s(k+1) = f(s(k), w(k)), \quad k \in \mathbb{N}_0, \tag{3.1}$$

where $s(k)$ is the state of the system at time k, f is any vector field, and $\{w(k), k \in \mathbb{N}_0\}$ are independent identically-distributed (i.i.d.) random vectors with known distribution. In other words, there is a representation (3.1) for any stochastic kernel $T_{\mathfrak{s}}$ and vice versa. In this chapter we study those processes driven by the

following mixed deterministic-stochastic dynamics:

$$\begin{cases} s_1(k+1) = f_1(s_1(k), s_2(k), w(k)) \\ s_2(k+1) = f_2(s_1(k), s_2(k)). \end{cases} \tag{3.2}$$

In model (3.2),

- $\{w(k) : \Omega \to \mathbb{R}^{n_1}, k \in \mathbb{N}_0\}$ is an i.i.d. random sequence on a sample space Ω with known distribution;
- $\{s_1(k) : \Omega \to \mathbb{R}^{n_1}, k \in \mathbb{N}_0\}$ is a vector-valued random sequence on Ω with dynamics that are directly affected by the random variable $w(\cdot)$ at a given time through the vector field $f_1 : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_1} \to \mathbb{R}^{n_1}$;
- $\{s_2(k) : \Omega \to \mathbb{R}^{n_2}, k \in \mathbb{N}_0\}$ is a vector-valued random sequence on Ω with dynamics characterized by a given vector field $f_2 : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^{n_2}$.

Denote by

$$s(k) = \begin{bmatrix} s_1(k) \\ s_2(k) \end{bmatrix} \in \mathbb{R}^n = \mathcal{S}, \quad n = n_1 + n_2,$$

the state variable of the whole model in (3.2). The knowledge of the distribution of random vector $w(\cdot)$ at a given time allows us to characterize a stochastic kernel $T_{\mathfrak{s}}(\cdot|s)$ for the Markov process. The special structure of model (3.2) leads us to express the conditional density function of the stochastic kernel $T_{\mathfrak{s}}$ as follows:

$$t_{\mathfrak{s}}(\bar{s}|s) = t_{\mathfrak{s}_1}(\bar{s}_1|s_1, s_2)\delta(\bar{s}_2 - f_2(s_1, s_2)), \tag{3.3}$$

for $s = [s_1^T, s_2^T]^T$ and where $\delta(s - a)$ is the continuous Dirac delta function located at the point a. The first term $t_{\mathfrak{s}}(\bar{s}_1|s_1, s_2)$ depends on the stochastic part of the dynamical model, whereas the second term $\delta(\bar{s}_2 - f_2(s_1, s_2))$ hinges on the deterministic vector field.

Over such models we are interested to solve the probabilistic invariance problem which was defined in Section 2.2.2 and its solution was characterized by Bellman recursion in Proposition 2.1. Let us recall this backward recursion for the invariance problem over the bounded set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$ as the safe set and the finite time horizon $\mathbb{Z}_{N+1}$:

$$V_k(s) = \mathbf{1}_{\mathcal{A}}(s) \int_{\mathcal{S}} V_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s), \quad V_N(s) = \mathbf{1}_{\mathcal{A}}(s), \tag{3.4}$$

and remind that the solution of the invariance problem is $p_{s_0}(\mathcal{A}) = V_0(s_0)$.

In Chapter 2 we presented a discretization approach with proven error bounds, under continuity conditions of the stochastic kernel $T_{\mathfrak{s}}$. We also refined the error bounds by leveraging an adaptive partitioning approach with improved (local) error computations.

The goal of this chapter is first to tailor problem (2.1) to the structure of model (3.2), then to provide a technique to compute the solution of (2.1) by a numerical scheme with associated errors.

3.3 Properties of the Value Functions

3.3.1 On the Support of the Value Functions

With focus on the recursion step in Equation (3.4), let us define the support of function V_k as:

$$supp(V_k) = \{s \in \mathcal{S} | V_k(s) \neq 0\}, \quad k \in \mathbb{Z}_N,$$

and $supp(V_N) = \mathcal{A}$. The support of the value functions V_k plays an important role in the problem definition, as elaborated in the following observations:

- since $V_k(s) = 0$ for all $s \notin \mathcal{A}$, then $supp(V_k) \subseteq \mathcal{A}$ for all $k \in \mathbb{Z}_{N+1}$;
- by direct inductive argument, it can be shown that $0 \leq V_k(s) \leq V_{k+1}(s)$ for all $s \in \mathcal{A}, k \in \mathbb{Z}_N$, which leads to conclude that $supp(V_k) \subseteq supp(V_{k+1})$.

Notice that, because of the constant value of the cost function on the complement of the set $\mathcal{A}$, the integral in (3.4) is effectively computed only over $\mathcal{A}$ (rather than on $\mathcal{S}$). Furthermore, the observations above suggest that it is possible to adapt the integration domain in (3.4) to the actual support of the value functions, as follows:

$$V_k(s) = V_k(s_1, s_2) = \int_{supp(V_{k+1})} V_{k+1}(\bar{s}_1, \bar{s}_2) t_{\mathfrak{s}}(\bar{s}_1 | s_1, s_2) \delta(\bar{s}_2 - f_2(s_1, s_2)) d\bar{s}_2 d\bar{s}_1, \quad (3.5)$$

where we have used the expression in (3.3). Characterizing the sets $supp(V_k), k \in \mathbb{Z}_n$, becomes thus critical for the optimization of the original recursion in (3.4). However, in general it is complicated to exactly determine the sets $supp(V_k)$, in particular due to the need to characterize $supp(t_{\mathfrak{s}}(\cdot|s))$ as a function of s.

To mitigate this complication, let us introduce two projection maps as follows:

$$\Pi_i : \mathbb{R}^n \to \mathbb{R}^{n_i}, \quad \Pi_i \left(\begin{bmatrix} s_1 \\ s_2 \end{bmatrix} \right) = s_i, \quad i = 1, 2.$$

We can determine an over-approximation of the sets $supp(V_k)$ as follows:

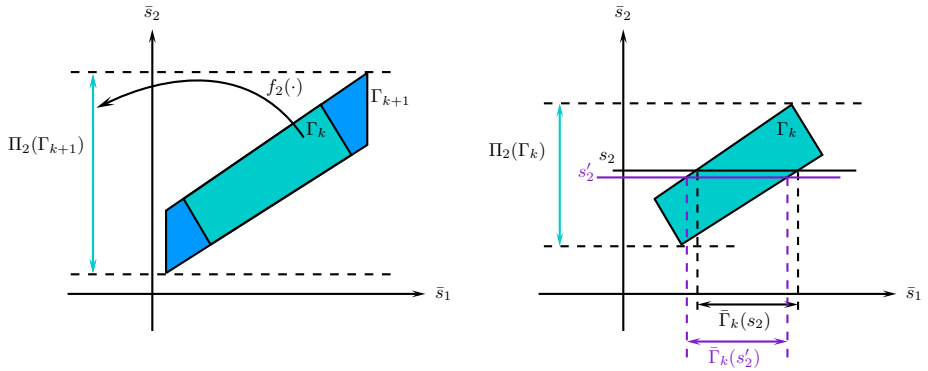
$$supp(V_k) \subseteq \{(s_1, s_2) \in supp(V_{k+1}) | f_2(s_1, s_2) \in \Pi_2(supp(V_{k+1}))\}.$$

Notice that in general the above inclusion is strict. This suggests to over-approximate the sets $supp(V_k)$ by Γ_k, as defined by the following recursive procedure:

$$\Gamma_N = \mathcal{A}, \quad \Gamma_k = \{(s_1, s_2) \in \Gamma_{k+1} | f_2(s_1, s_2) \in \Pi_2(\Gamma_{k+1})\}, \quad k \in \mathbb{Z}_N. \quad (3.6)$$

The sequence $\{\Gamma_k, k \in \mathbb{Z}_{N+1}\}$ is endowed with the following facts:

- $supp(V_k) \subseteq \Gamma_k$, then $p_{s_0}(\mathcal{A}) = 0$ for all $s_0 \notin \Gamma_0$;
- $\mathcal{A} = \Gamma_N \supseteq \Gamma_{N-1} \supseteq \Gamma_{N-2} \supseteq ... \supseteq \Gamma_0$;
- if there exists a positive integer $k_0 \in \mathbb{Z}_N$ such that $\Gamma_{k_0} = \Gamma_{k_0+1}$, then for all $0 \leq k \leq k_0, \Gamma_k = \Gamma_{k_0}$;



(a) Visualization of backward reachability (3.6) for the computation of support sets Γ_k

(b) Visualization of the set-valued function $\bar{\Gamma}_k(\cdot)$ defined in (3.7)

Figure 3.1: Illustration of the backward recursion over the support sets Γ_k and over the set-valued function $\bar{\Gamma}_k$ (cf. Section 3.6).

- more generally, if there exists a positive integer $k_0 \in \mathbb{Z}_N$ such that $\Pi_2(\Gamma_{k_0}) = \Pi_2(\Gamma_{k_0+1})$, then for all $0 \leq k \leq k_0, \Gamma_k = \Gamma_{k_0}$.

These properties highlight the dependence of the sets Γ_k (we will denote them simply as *support sets*) on the deterministic vector field f_2, particularly over the points that are mapped by f_2 outside of the support sets.

3.3.2 Simplifying the Bellman Recursion

With focus on the support sets introduced in (3.6), define additionally the following set-valued function $\bar{\Gamma}_k : \Pi_2(\Gamma_k) \to 2^{\Pi_1(\Gamma_k)}$. for any $k \in \mathbb{N}_N$, as

$$\bar{\Gamma}_k(s_2) = \{s_1 \in \Pi_1(\Gamma_k) | (s_1, s_2) \in \Gamma_k\}, \quad \forall s_2 \in \Pi_2(\Gamma_k). \tag{3.7}$$

Recall the recursive formula in (3.5) for V_k. By definition of Γ_k, we know that V_k is equal to zero outside of the set Γ_k. We can then simplify the recursive formula to the following:

$$V_k(s) = \int_{\bar{\Gamma}_{k+1}(f_2(s))} V_{k+1}(\bar{s}_1, f_2(s)) t_{\mathbf{s}}(\bar{s}_1|s) d\bar{s}_1, \quad \forall s \in \Gamma_k. \tag{3.8}$$

This formulation characterizes the value functions V_k in terms of the sets Γ_k. The computation of sets Γ_k based on (3.6) is a known deterministic backward reachability procedure over the map f_2. Deterministic reachability analysis is a rather mature field of research [42] with ample software tool support [1]. Figure 3.1 illustrates the backward recursion over the support sets Γ_k for a two dimensional system (cf. case study in Section 3.6), and displays the construction of the set-valued function $\bar{\Gamma}_k$.

3.3.3 Continuity Properties of the Value Functions

We are interested in establishing continuity properties of the value functions over their support, which will be key for the computational schemes that will be introduced later. To achieve this, the following set of assumptions is needed.

Assumption 3.1 *Suppose that the kernel $T_{\mathfrak{s}}$ admits a density function $t_{\mathfrak{s}}$ as in (3.3). Furthermore, suppose that the density function $t_{\mathfrak{s}_1}$, the vector field f_2, and the parametrized sets $\bar{\Gamma}_k$ satisfy the following conditions:*

1. $|t_{\mathfrak{s}_1}(\bar{s}_1|s) - t_{\mathfrak{s}_1}(\bar{s}_1|s')| \leq h_k\|s - s'\|$, *for any* $\bar{s}_1 \in \Pi_1(\Gamma_{k+1})$ *and* $s, s' \in \Gamma_k$;
2. $\|f_2(s) - f_2(s')\| \leq \bar{h}_k\|s - s'\|$, *for any* $s, s' \in \Gamma_k$;
3. $\mathscr{L}(\bar{\Gamma}_k(s_2) \triangle \bar{\Gamma}_k(s_2')) \leq \theta_k\|s_2 - s_2'\|$, *for any* $s_2, s_2' \in \Pi_2(\Gamma_k)$,

where $h_k, \bar{h}_k, \theta_k$ are finite constants, for $k \in \mathbb{Z}_{N+1}$. Here $\mathscr{L}$ is the Lebesgue measure over $\mathbb{R}^{n_1}$, whereas $\triangle$ denotes the symmetric difference of two sets ($A \triangle B = (A \setminus B) \cup (B \setminus A)$).

The first two are continuity assumptions on the probabilistic density and on the vector field, whereas the third is a regularity requirement on the variation of the (projection along the s_1 variables of the) support sets, as a function of the s_2 coordinates. This last assumption depends on the actual shape of the support sets Γ_k and on f_2, as displayed in Figure 3.1.

Theorem 3.1 *If Assumption 3.1 is valid, then the value functions V_k are Lipschitz continuous over Γ_k,*

$$|V_k(s) - V_k(s')| \leq \lambda_k\|s - s'\|, \quad \forall s, s' \in \Gamma_k, \tag{3.9}$$

where the finite Lipschitz constant λ_k satisfies the recursive formula:

$$\lambda_k = (h_k L_{k+1} + M_k \bar{h}_k \theta_{k+1}) + \bar{h}_k M_k^\star \lambda_{k+1}, \qquad k \in \mathbb{Z}_N,$$

initialized with $\lambda_N = 0$, and where:

$$L_k \doteq \mathscr{L}\left(\Pi_1(\Gamma_k)\right), \quad M_k^\star \doteq \sup\left\{ \int_{\Pi_1(\Gamma_{k+1})} t_{\mathfrak{s}_1}(\bar{s}_1|s) d\bar{s}_1 \,\middle|\, s \in \Gamma_k \right\}$$
$$M_k \doteq \sup\left\{ t_{\mathfrak{s}_1}(\bar{s}_1|s) | s \in \Gamma_k, \bar{s}_1 \in \Pi_1(\Gamma_{k+1}) \right\}.$$

Remark 3.1 *For the sake of clarity, let us simplify Assumptions 3.1 and express the bounds in Theorem 3.1 anew. Notice that $0 \leq M_k^\star \leq 1$ and that, because $\Gamma_k \subseteq \Gamma_{k+1}$,*

$$h_k \leq h_{k+1}, \bar{h}_k \leq \bar{h}_{k+1}, L_k \leq L_{k+1}, M_k \leq M_{k+1}, \forall k \in \mathbb{Z}_N.$$

Then setting the following (global) continuity assumptions for all $\bar{s}_1 \in \Pi_1(\mathcal{A}), s, s' \in \mathcal{A}$,

$$\|f_2(s) - f_2(s')\| \leq \bar{h}\|s - s'\|, \quad |t_{\mathfrak{s}_1}(\bar{s}_1|s) - t_{\mathfrak{s}_1}(\bar{s}_1|s')| \leq h\|s - s'\|,$$

and defining the time-independent constants $L, M, M^\star$ as

$$L_k \leq L \doteq \mathscr{L}(\mathcal{A}), \quad M_k^\star \leq M^\star \doteq sup\left\{\int_{\Pi_1(\mathcal{A})} t_{\mathfrak{s}_1}(\bar{s}_1|s)d\bar{s}_1 \middle| s \in \mathcal{A}\right\}$$
$$M_k \leq M \doteq \sup\{t_{\mathfrak{s}_1}(\bar{s}_1|s), x \in \mathcal{A}, \bar{s}_1 \in \Pi_1(\mathcal{A})\},$$

we can express the bound in (3.9) with the constant $\lambda_k = hL + \bar{h}(M\theta_{k+1} + M^\star\lambda_{k+1})$, for all $k \in \mathbb{Z}_N$, and $\lambda_N = 0$.

3.4 Approximation Scheme and Error Quantification

We propose an approximation scheme to do the computations in (3.8). In order to keep the notations light, we replace the generic integration domain $\bar{\Gamma}_{k+1}(f_2(s))$, $k \in \mathbb{Z}_N$, by $\Pi_1(\mathcal{A})$, and comment on how the procedure applies similarly to the more general case.

We adapt Algorithm 1 from the previous chapter to our model at hand. Select an arbitrary partition of the support set $\mathcal{A} = \cup_{i=1}^m A_i$, $A_{i_1} \cap A_{i_2} = \emptyset$, $i_1, i_2 = \mathbb{N}_m$, $i_1 \neq i_2$, where m represents the cardinality of the partition. A partition of the whole state space $\mathcal{S}$ is obtained by adding the complement set $A_{m+1} = \mathcal{S}\backslash\mathcal{A}$. Pick any point $z_i = (z_1^i, z_2^i) \in A_i, i \in \mathbb{N}_{m+1}$. Notice that $\Pi_1(\mathcal{A}) = \Pi_1(\cup_{i=1}^m A_i) = \cup_{i=1}^m \Pi_1(A_i)$, however the sets $\Pi_1(A_i)$ produce in general a cover (not necessarily a partition) of the set $\Pi_1(\mathcal{A})$. To make up for this, let us additionally select an arbitrary (q dimensional) partition $\Pi_1(\mathcal{A}) = \cup_{j=1}^q X_j$ for the projection of the safe set along the first variable. This allows to express, for all $s \in \mathcal{A}$,

$$V_k(s) = \int_{\Pi_1(\mathcal{A})} V_{k+1}(\bar{s}_1, f_2(s))t_{\mathfrak{s}_1}(\bar{s}_1|s)d\bar{s}_1 = \sum_{j=1}^q \int_{X_j} V_{k+1}(\bar{s}_1, f_2(s))t_{\mathfrak{s}_1}(\bar{s}_1|s)d\bar{s}_1.$$

Let us now approximate the value functions V_k by piecewise constant functions $\bar{V}_k$, which are computed over the selected points $\{z_i \in A_i, i \in \mathbb{N}_{m+1}\}$, as follows:

$$\bar{V}_k(s) = \sum_{i=1}^{m+1} \bar{V}_k(z_i)\mathbf{1}_{A_i}(s), \quad \forall s \in \mathcal{A}, \quad k \in \mathbb{Z}_N,$$

initialized as $\bar{V}_N(z_i) = 1, i \in \mathbb{N}_m$, and $\bar{V}_N(z_{m+1}) = 0$. Introduce the simplified notation $V_k^i \doteq \bar{V}_k(z_i)$. These functions are recursively computed as follows:

$$V_k^i = \sum_{j=1}^q \int_{X_j} \bar{V}_{k+1}(\bar{s}_1, f_2(z_i))t_{\mathfrak{s}_1}(\bar{s}_1|z_i)d\bar{s}_1. \tag{3.10}$$

In this formulation the values of $\bar{V}_{k+1}$ over the hyperplane $X_j \times \{f_2(z_i)\}$ are needed. Thus, in order to implement the procedure discretely, the function $\bar{V}_{k+1}$ should be constant over this hyperplane. This feature is achieved by raising the

following assumption on the partition sets X_j of $\Pi_1(\mathcal{A})$:

$$\forall i \in \mathbb{N}_m, \forall j \in \mathbb{N}_q, \exists i' \mathbb{N}_m : \; X_j \times \{f_2(z_i)\} \subseteq A_{i'}.$$

Notice that this assumption does not depend on step k, and is immediately satisfiable by selecting two partitions $\Pi_1(\mathcal{A}) = \cup_j X_j$ and $\Pi_2(\mathcal{A}) = \cup_r Y_r$, and then constructing the partition for $\mathcal{A}$ as a subset of the cross product of these two partitions: $\mathcal{A} \subseteq \Pi_1(\mathcal{A}) \times \Pi_2(\mathcal{A}) = \cup_{j,r} X_j \times Y_r$.

Consider a map $i' = R(i,j)$, which assigns to each partition set X_j and value $f_2^i \doteq f_2(z_i)$ the corresponding partition set $A_{i'}$ containing $X_j \times f_2^i$. Finally, starting from the recursive procedure (3.8), the discrete version of the (continuous) step (3.10) can be formulated as:

$$V_k^i = \sum_{j=1}^{q} V_{k+1}^{i'} \int_{X_j} t_{\mathfrak{s}_1}(\bar{s}_1 | z_i) d\bar{s}_1. \tag{3.11}$$

Let us again emphasize that the above steps, developed for set $\mathcal{A}$, can be tailored to the integration domain based on the knowledge of Γ_k. Furthermore, notice that in the above procedure we allow for additional approximation error, since there may exist partition sets that cross the boundaries of the support sets, and which are not contained in neither Γ_k nor $\mathcal{S} \backslash \Gamma_k$. In order to avoid this error, we select a partition for the smallest support set Γ_0 and, iteratively, extend the partition of set $\Gamma_k \subseteq \Gamma_{k+1}$ to obtain a proper partition of Γ_{k+1}.

The approximation scheme is summarized in Algorithm 5, and its error can be explicitly quantified as follows.

Algorithm 5 Approximation scheme for probabilistic invariance of $(T_{\mathfrak{s}}, \mathcal{S})$

Require: mixed deterministic-stochastic system $(T_{\mathfrak{s}}, \mathcal{S})$, safe set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$, finite time horizon N; sequence of support sets $\Gamma_k, k \in \mathbb{Z}_{N+1}, \Gamma_N = \mathcal{A}$

1: Select partitions $\cup_j X_j$ of $\Pi_1(\Gamma_k)$ and associated partitions $\cup_i A_i$ of set Γ_k, $k \in \mathbb{Z}_{N+1}$
2: Compute the map $i' = R(i,j)$ based on the chosen partition sets
3: Compute marginalization matrix P, with entries $P_{ij} = \int_{X_j} t_{\mathfrak{s}_1}(\bar{s}_1 | z_i) d\bar{s}_1$
4: At step $k \in \mathbb{Z}_{N+1}$, use support set Γ_k to set entries equal to zero, namely $V_k^i = 0$ for all i such that $A_i \subset \mathcal{S} \backslash \Gamma_k$
5: Compute recursively value functions $V_k^i = \sum_{j=1}^{q} P_{ij} V_{k+1}^{i'}$ as in (3.11), where $V_N^i = 1, i \in \mathbb{N}_m, V_N^{m+1} = 0$

Ensure: $V_0^i, i \in \mathbb{N}_{m+1}$, approximate solution of the safety problem over set $\mathcal{A}$

Theorem 3.2 *Suppose that the value functions V_k are approximated by the piecewise constant functions $\bar{V}_k$, as described above. Then the approximation error is upper bounded by the quantity*

$$|V_k(s) - \bar{V}_k(s)| \leq E_k, \quad \forall s \in \Gamma_k,$$

where $E_k = \lambda_k \delta + M_k^\star E_{k+1}$, initialized by $E_N = 0$, and where δ is the partition size of $\cup_{i=1}^m A_i$ (namely, $\delta = \max_{i=1}^m \delta_i$, where δ_i is the diameter of A_i), λ_k is the Lipschitz constant of the value function V_k, and $M_k^\star$ is defined as in Theorem 3.1.

Remark 3.2 *Notice that the error critically depends on the quantities $\lambda_k, M_k^\star$ from Theorem 3.1, which are computed as function of the support sets Γ_k. Leaving the question of continuity of the value functions aside, had we not characterized and computed the sets Γ_k via deterministic reachability and instead used the original safe set $\mathcal{A}$, we would have obtained a larger error, and attained a discretization in time with higher cardinality. In other words, computing the sets Γ_k via deterministic reachability leads to better approximations and faster computations of the probabilistic quantities.*

3.5 Affine Deterministic Dynamics on Polytopic Invariant Sets

It is in general difficult to find an explicit or an exactly computable bound for Condition 3 in Assumption 3.1, which depends on the shape of the sets Γ_k and on the map f_2. However, such a bound can be derived in the relevant instance of models in (3.2) with deterministic dynamics that are affine and of an invariant set $\mathcal{A}$ that is a bounded convex polytope [16]. Under these conditions, the following lemma gives an explicit representation for the invariant sets Γ_k, which is later used to derive the desired error bounds.

Lemma 3.1 *Suppose that the deterministic dynamics in (3.2) are characterized by affine functions, namely:*

$$f_2(s_1, s_2) = F_1 s_1 + F_2 s_2 + F_3,$$

where $F_1 \in \mathbb{R}^{n_2 \times n_1}, F_2 \in \mathbb{R}^{n_2 \times n_2}, F_3 \in \mathbb{R}^{n_2 \times 1}$. Furthermore, suppose that the invariant set $\mathcal{A}$ is a bounded convex polytope, described by the following set of linear inequalities:

$$\mathcal{A} = \left\{(s_1, s_2) \in \mathbb{R}^n | A_N^1 s_1 + A_N^2 s_2 \leq B_N\right\}.$$

Then the support sets $\Gamma_k, k \in \mathbb{Z}_N$, are also bounded convex polytopes as in (3.12).

Proof: Based on Equation (3.6), we can compute the sets $\Gamma_k, k \in \mathbb{Z}_N$, as:

$$\Gamma_k = f_2^{-1}(\Pi_2(\Gamma_{k+1})) \cap \Gamma_{k+1}.$$

Suppose Γ_{k+1} is compact and convex then $\Pi_2(\Gamma_{k+1})$ is also a compact and convex set since the operator Π_2 is linear. Additionally, as the function f_2 is linear (and continuous), then $f_2^{-1}(\Pi_2(\Gamma_{k+1}))$ is also compact and convex. Suppose now that set Γ_{k+1} is a polytope in $\mathbb{R}^n$, characterized by the following set of linear inequalities:

$$\Gamma_{k+1} = \left\{(s_1, s_2) \in \mathbb{R}^n | A_{k+1}^1 s_1 + A_{k+1}^2 s_2 \leq B_{k+1}\right\}.$$

Then $\Pi_2(\Gamma_{k+1})$ is also a polytope in $\mathbb{R}^{n_2}$ and is characterized by

$$\Pi_2(\Gamma_{k+1}) = \left\{s_2 \in \mathbb{R}^{n_2} | C_{k+1} s_2 \leq D_{k+1}\right\}.$$

The matrices C_{k+1}, D_{k+1} in the definition of $\Pi_2(\Gamma_{k+1})$ can be directly obtained from Γ_{k+1} by taking the perpendicular projection of bounded polytopes: [47] proved that the polyhedral projection is equivalent to the feasibility of a parametric linear programming problem. Computationally, the MPT toolbox [57] performs this operation first constructing a vertex representation of Γ_{k+1}, having its half-space representation (vertex enumeration problem); it then projects these vertices based on the Π_2 operator; and finally it obtains a half-space representation of $\Pi_2(\Gamma_{k+1})$ from its vertex representation (facet enumeration problem).

Having matrices C_{k+1}, D_{k+1} from the expression of $\Pi_2(\Gamma_{k+1})$, set Γ_k can be found as

$$\Gamma_k = \{(s_1, s_2) \in \Gamma_{k+1} | C_{k+1}F_1 s_1 + C_{k+1}F_2 s_2 \leq D_{k+1} - C_{k+1}F_3\}.$$

Then Γ_k is a convex and bounded polytope with the following half-space representation

$$\Gamma_k = \left\{(s_1, s_2) \in \mathbb{R}^n | A_k^1 s_1 + A_k^2 s_2 \leq B_k\right\}, \tag{3.12}$$

where

$$A_k^1 = \begin{bmatrix} C_{k+1}F_1 \\ A_{k+1}^1 \end{bmatrix}, A_k^2 = \begin{bmatrix} C_{k+1}F_2 \\ A_{k+1}^2 \end{bmatrix}, B_k = \begin{bmatrix} D_{k+1} - C_{k+1}F_3 \\ B_{k+1} \end{bmatrix}.$$

Note that this representation is not unique: it is in particular possible to eliminate redundant half-spaces in the representation of Γ_k at each step k. □

The following theorem derives an explicit bound for Condition 3 in Assumption 3.1.

Theorem 3.3 *Suppose Γ_k is a bounded convex polytope with the representation in (3.12). Then the sets $\bar{\Gamma}_k(s_2)$ are polytopes in $\mathbb{R}^{n_1}$, which satisfy the Condition 3 in Assumption 3.1 with the following constant:*

$$\theta_k = \sum_{i=1, A_k^1(i) \neq 0}^{m_k} c_k(i) \frac{\|A_k^2(i)\|}{\|A_k^1(i)\|}.$$

In the formula, vectors $A_k^1(i)$ and $A_k^2(i)$ represent the i^{th} row of A_k^1 and A_k^2, respectively. The constant m_k accounts for the number of inequalities in the half-space representation of Γ_k, that is m_k is equal to the number of rows of A_k^1 (we do not account for the rows of A_k^1 that are identically equal to zero). The constant $c_k(i)$ is computed as follows:

1. *if $n_1 = 1$ then $c_k(i) = 1$ for any $i \in \mathbb{N}_{m_k}$;*
2. *if $n_1 \geq 2$, project $\Pi_1(\Gamma_k)$ along the normal to the i^{th} hyperplane, i.e. along vector $A_k^1(i)$, resulting in $\Pi^{\perp}(\Pi_1(\Gamma_k))$, which is a polytope in $\mathbb{R}^{n_1-1}$. Then $c_k(i) = \mathscr{L}(\Pi^{\perp}(\Pi_1(\Gamma_k)))$ or any upper bound for the given Lebesgue measure.*

For the sake of completeness, let us explicitly derive the Lipschitz constant required for Condition 2 in Assumption 3.1, for a map f_2 with affine deterministic dynamics. With reference to the affine function $f_2(s_1, s_2) = F_1 s_1 + F_2 s_2 + F_3$, we can express $\|f_2(s_1, s_2) - f_2(s_1', s_2')\| \leq \|[F_1, F_2]\|_2 \|(s_1, s_2) - (s_1', s_2')\|$.

3.6 Case Study

Consider the case study introduced in Section 2.6.2 of previous chapter. We eliminate the process noises w_2, w_3 in (2.19) due to the fact that the related variables are dimensionally different from x_1. Then we have the following dynamical equations

$$\begin{cases} x_1(k+1) = (1-2k_d\Delta)x_1(k) + 2k_d\Delta D^\star_{ss} + \sqrt{2k_d\Delta D^\star_{ss}}w_1(k) \\ x_2(k+1) = k_r\Delta x_1(k) + (1-\gamma_r\Delta)x_2(k) \\ x_3(k+1) = k_p\Delta x_2(k) + (1-\gamma_p\Delta)x_3(k), \end{cases}$$

where $\{w_1(k), k \in \mathbb{N}_0\}$ are independent standard Gaussian random variables. Notice that the model is now mixed deterministic-stochastic: namely, deterministic over the dynamics of x_2 (concentration of m-RNA M) and of x_3 (protein P), whereas stochastic for x_1 (active genes $D^\star$). Notice that in (3.2) we would have variables $s_1 = x_1$ and $s_2 = (x_2, x_3)$. Further, to connect the model with the representation in (3.3), the kernel for the dynamics in s_1 is Gaussian and admits a density $t_{s_1}(\bar{s}_1|s_1) \sim \mathcal{N}(\mu_1(s_1), \sigma_1^2)$, where the mean is an affine function of the conditional variable s_1, whereas the variance is constant:

$$\mu_1(s_1) = (1-2k_d\Delta)s_1 + 2k_d\Delta D^\star_{ss}, \quad \sigma_1^2 = 2k_d\Delta D^\star_{ss}.$$

3.6.1 Computation in a Two Dimensional System

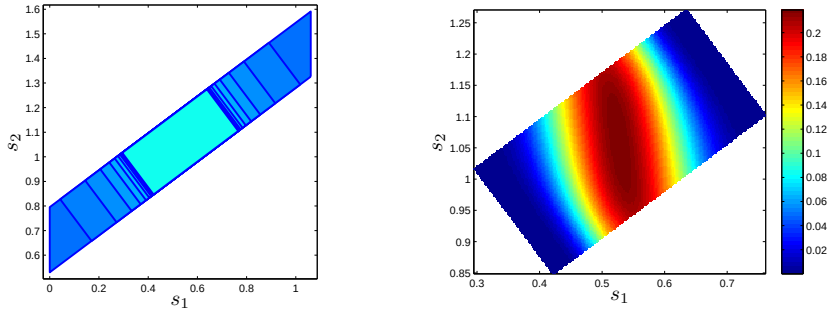
Observe that the dynamics over the variables s_1, s_2 are decoupled from that of variabe s_3. In order to visually elucidate the concepts introduced above, this section sets up the computation of probabilistic invariance for the mixed deterministic-stochastic system comprising variables s_1 ($D^\star$, with stochastic dynamics), and s_2 (M, with deterministic dynamics). The safe set is selected to be the following bounded polygon, centered at the steady-state point for the model:

$$\mathcal{A} = \left\{ (s_1, s_2) \in \mathbb{R}^2, \left| \frac{s_1 - D^\star_{ss}}{D^\star_{ss}} \right| \leq r_1, \left| \frac{\bar{f}_2(s_1, s_2) - \bar{f}_2(D^\star_{ss}, M_{ss})}{\bar{f}_2(D^\star_{ss}, M_{ss})} \right| \leq r_2 \right\}.$$

where $\bar{f}_2(s_1, s_2) = (\gamma_r\Delta - 1)\, s_1 + k_r\Delta s_2$ is selected such that functions f_2 and $\bar{f}_2$ are orthogonal. We consider the parameter values $r_1 = 1, r_2 = 0.2$ for the size of the polygon, and the time horizon $N = 10$. The deterministic backward reachability procedure over set $\mathcal{A}$ shows that the sets Γ_k are all convex polytopes, and in particular rectangles, with the following descriptions:

$$\Pi_2(\Gamma_{N-k}) = \left\{ s_2 \in \mathbb{R} : \left| \frac{s_2 - M_{ss}}{M_{ss}} \right| \leq 0.2 + 0.3 \times 0.6^k \right\},$$

$$\Gamma_{N-k-1} = \left\{ (s_1, s_2) \in \mathbb{R}^2 : \left| \frac{f_2(s_1, s_2) - M_{ss}}{M_{ss}} \right| \leq 0.2 + 0.3 \times 0.6^k, \left| \frac{\bar{f}_2(s_1, s_2) - D^\star_{ss}}{D^\star_{ss}} \right| \leq 0.2 \right\},$$



(a) Support sets Γ_k, shrinking backwards in time. (b) probabilistic invariance $p_s(\mathcal{A}), \forall s \in \Gamma_0$.

Figure 3.2: Representation of the support sets of the value functions in time, and of the solution of the probabilistic invariance problem, for a two dimensional case study.

$$\Pi_1(\Gamma_{N-k-1}) = \left\{ s_1 \in \mathbb{R} : \left| \frac{s_1 - D^\star_{ss}}{D^\star_{ss}} \right| \leq 0.44 + 0.48 \times 0.6^k \right\}.$$

Figure 3.2(a) shows the dynamics of the invariance sets Γ_k, starting from $\mathcal{A} = \Gamma_n$ (external polygon). Notice that the sequence of support sets shrinks, converging as the time horizon grows (this can possibly lead to the study of the problem over an infinite horizon). The Lipschitz constants h_k are computed based on the maximum norm of the partial derivative of the density function with respect to the conditional variable s_1,

$$h_k = \max \left\{ \left| \frac{\partial t_{\mathfrak{s}_1}}{\partial s_1}(\bar{s}_1|s_1) \right| : s_1 \in \Pi_1(\Gamma_k), \bar{s}_1 \in \Pi_1(\Gamma_{k+1}) \right\} = \frac{(1 - k_a - k_d)e^{-1/2}}{\sigma^2\sqrt{2\pi}}.$$

which leads to $h_k = 56.58$ that is independent of the time step. Constants M_k and $M^\star_k$ have been considered independent of the step k and take the following values:

$$M_k = \frac{1}{\sigma\sqrt{2\pi}}, \quad M^\star_k = 2\int_0^{\frac{r_2}{0.6\sigma} D^\star_{ss}} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{u^2}{2}\right] du = \text{erf}\left(\frac{r_2}{0.6\sigma\sqrt{2}} D^\star_{ss}\right),$$

which leads to $M_k = 6.125$ and to $M^\star_k = 0.9934$ (here erf is the error function associated to a Gaussian distribution). Finally, the following constants are required for the computation of the error: $\bar{h}_k = 1, \theta_k = 8/3, \mathscr{L}(\Gamma_N) = D^{\star 2}_{ss}, \mathscr{L}(\Gamma_{N-k}) = 0.8 D^{\star 2}_{ss}(0.4 + 0.6^k)$. The accrued error, as in Theorem 3.2, is equal to $E_0 = 1371.1\delta$: the linear dependence over the discretization parameter δ allows tuning it as desired. The partitioning sets have been introduced as tight over-approximations of the Γ_k by a uniform grid aligned with the coordinates. The approximate solution of the probabilistic safety is presented in Figure 3.2(b). The probability is plotted within set Γ_0, since it is constantly equal to zero in the complement of this set. Notice that as expected the output is maximal in a neighborhood of the equilibrium

point for the dynamics.

3.6.2 Computation in a Three Dimensional System

This section applies the results derived in this chapter to the whole three dimensional model introduced in Section 3.6. Let us select the safe set $\mathcal{A}$ as a box around the steady state values defined above, and compute probabilistic invariance over $\mathcal{A}$, for a time horizon $N = 10$. The size of the box is characterized by the parameters $r_1 = 0.20, r_2 = 0.10, r_3 = 0.05$, as:

$$\mathcal{A} = \left\{ (s_1, s_2, s_3) \in \mathbb{R}^3 : \left| \frac{s_1 - D^\star_{ss}}{D^\star_{ss}} \right| \leq r_1, \left| \frac{s_2 - M_{ss}}{M_{ss}} \right| \leq r_2, \left| \frac{s_3 - P_{ss}}{P_{ss}} \right| \leq r_3 \right\}.$$

The chosen parameters lead to a variance $\sigma = 0.32$ and to the following constants: $h_1 = 1.82, h_2 = 4.43, M = 12.25, M^\star = 0.99$. The deterministic procedure for computing the time-varying support sets Γ_k yields sets Γ_N, Γ_{N-1} and $\Gamma_{N-2} = \Gamma_k$, for all $k \in \mathbb{Z}_{N-2}$. Notice also that $\Pi_1(\Gamma_k) = \Pi_1(\mathcal{A})$ for any k, which leads to a constant $L = L_k = \mathscr{L}(\Pi_1(\mathcal{A})) = 0.21$.

Figure 3.3 displays the support sets Γ_N, Γ_{N-1} and Γ_0. Notice that the sets shrink as time decreases, as expected. The computations have been performed with a partition size $\delta = 0.03$. The probabilistic invariance is computed over the support sets Γ_k.

Figure 3.4 displays the level set $V_k(s) = 0.1$, for varying time instants $k = 2, 4, 6, 8$. As intuitive, the level sets shrink backwards in time. Figure 3.5 displays the level sets of $V_0(s) = p_s(\mathcal{A})$, for varying invariance levels: $0, 0.02, 0.04, 0.06, 0.08, 0.1$. Notice that the set of points $V_0 = 0$ cover a region that is the complement of Γ_0 in $\mathcal{A}$ (cf. the top left plot in Figure 3.5 with the bottom plot in Figure 3.3). As intuitive, the level sets shrink as the invariance level grows. Notice that the last (bottom-right) plot of Figure 3.5 would also fit the sequence of Figure 3.5, had we additionally considered $k = 0$.

3.7 Conclusions

In this chapter we have presented a novel approach to compute probabilistic invariance over a finite horizon for mixed deterministic-stochastic, discrete-time processes. The computational technique, based on state-space discretization, has been associated to an explicit error bound. On the theoretical side, we have showed that the problem under study can be separated into a deterministic reachability problem, and a probabilistic invariance one that depends on the outcome of the first. The technique has been tested on the case study modeling a chemical reaction network.

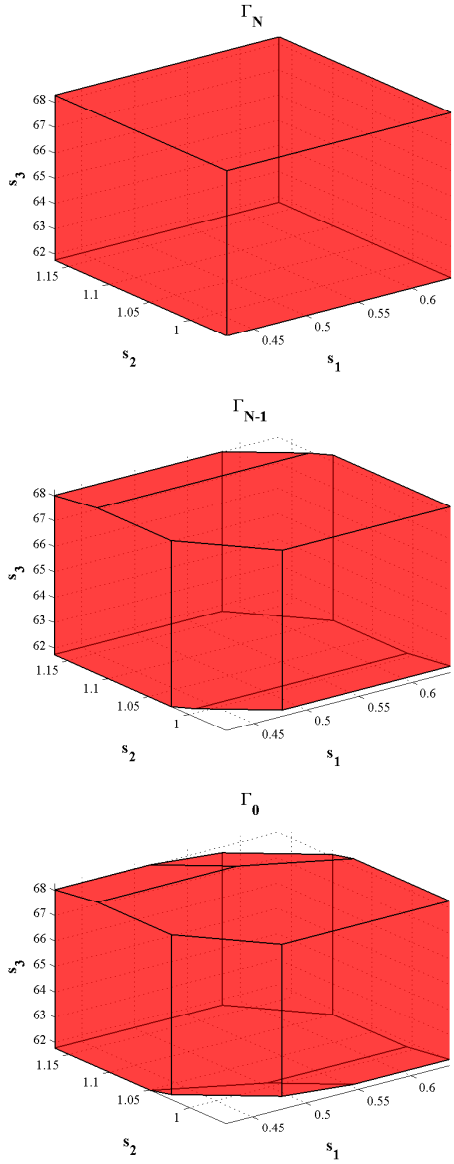


Figure 3.3: Representation of the support sets Γ_N, Γ_{N-1}, and Γ_0, where $\Gamma_k = \Gamma_{k-1}$, for all $k \in \mathbb{N}_{N-2}$ and where $N = 10$.

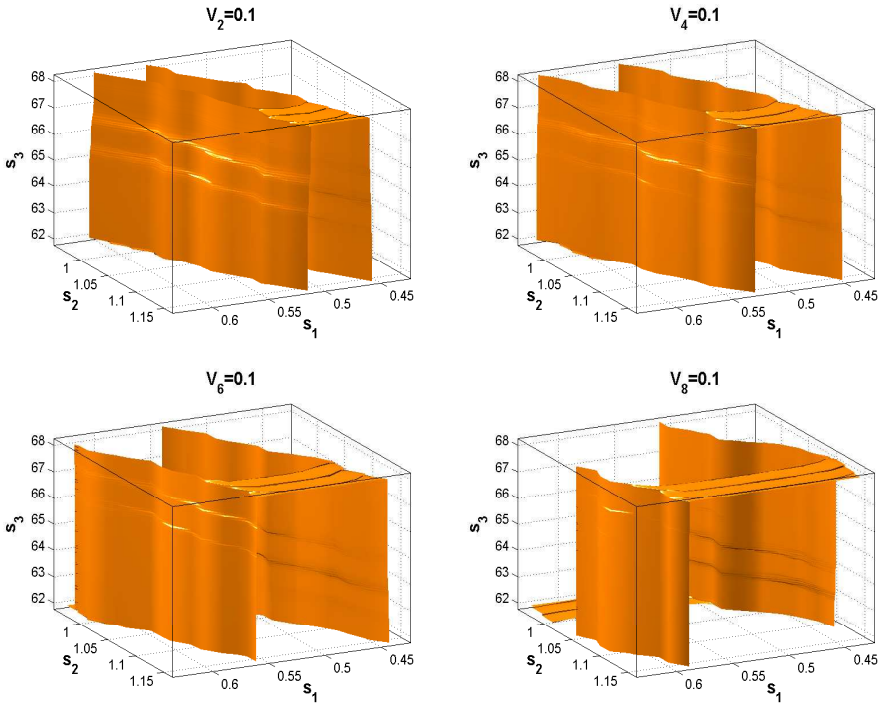


Figure 3.4: Representation of the level set $V_k(s) = 0.1$, for varying time instants $k = 2, 4, 6, 8$. The curves within the plots contain points that are invariant with a probability greater than or equal to 0.1, for the residual time span $\{k, k+1, \ldots, N\}$.

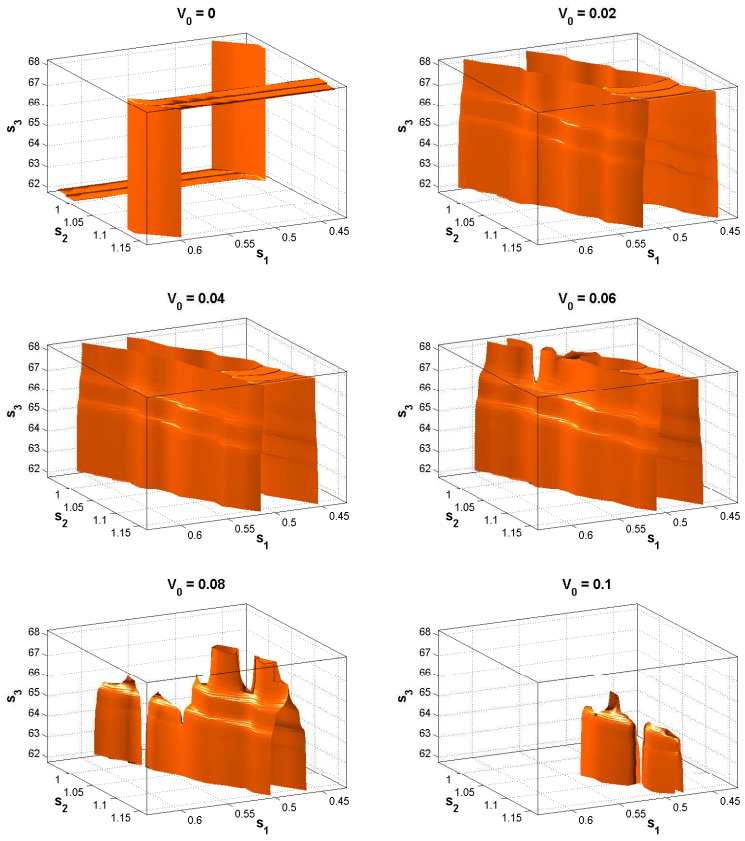


Figure 3.5: Representation of the level sets of $V_0(s) = p_s(\mathcal{A})$, for varying probability levels $\{0, 0.02, 0.04, 0.06, 0.08, 0.1\}$. The curves within the plots contain points that are invariant with a probability greater than or equal to the specific probability level, over the complete time span $\mathbb{Z}_{N+1}$.

4 CHAPTER

Higher-Order Approximations for Verification of Stochastic Systems

This chapter presents new algorithms, based on higher-order function approximation, for the efficient computation of approximate solutions of probabilistic invariance problem. The approach provides explicit bounds on the error which can be substantially lower than those for piecewise constant (zeroth-order) approximations presented in Chapter 2 and, unlike the latter, can display convergence in time to a finite value. Furthermore, higher-order approximation procedures, which depend on the partitioning of the state space, can lead to lower partition cardinality than the related piecewise constant ones. The result is first presented for Markov processes over Euclidean spaces and thereafter extended to hybrid spaces characterizing Stochastic Hybrid Systems.

4.1 Introduction

We investigated the development of enhanced computational approaches with tightened bounds on the error in Chapter 2, to translate a Markov process into a MC with the end goal of model checking the safety property. In approximating the Markov process as MC, the surveyed results of [2, 3] and Chapters 2, 3 have leveraged piecewise constant interpolations of the kernels characterizing the model under study, which has direct consequences on the derived error bounds. In contrast, this chapter provides approximation methods via higher-order interpolations of the value functions that are aimed at requiring less computational effort.

More precisely, drawing on the expression of reachability properties as value functions [2, 3, 68], this chapter builds on the premises in Chapter 2 and puts forward higher-order approximation methods, obtained via interpolation procedures, in order to express the value functions under study as compactly as possible. The

claim is that using higher-order interpolations (versus piecewise constant ones) can be beneficial in terms of obtaining tighter bounds on the approximation error. Furthermore, since the approximation procedures depend on the partitioning of the state space, higher-order schemes display an interesting tradeoff between more parsimonious representations versus more complex local computation – this chapter explores the computational compromise between partition size and local interpolation. In assessing the computability of the results, an underlying tenet is that the total number of integrations required in the interpolation is a proxy for total computational time. An additional advantage of the present study over previous chapters is that in some cases the approximation error converges over time, which allows the applicability of the method to the approximate solution of infinite-horizon specifications [75].

This chapter is structured as follows: Section 4.2 introduces a function operation view to the computation of value functions characterizing the solution of probabilistic invariance problem. Section 4.3 considers higher-order approximation schemes over the value functions, and quantifies explicitly the introduced approximation error over the formula (or problem). Section 4.4 tailors the results to the abstraction algorithm of Chapter 2, and specializes the proposed approach to explicit schemes for low-dimensional models and known interpolation bases. Section 4.5 extends the results to SHS models. Finally, Section 4.6 discusses a few numerical case studies to test and benchmark the proposed schemes.

4.2 Function Operation View of Probabilistic Invariance

We consider discrete time Markov processes, described in Section 2.2.1, which are characterized by a general state space $\mathcal{S}$ and a stochastic kernel $T_{\mathfrak{s}}$. We are interested to solve the probabilistic invariance problem over such models. Recall that solution of this problem can be expressed by introducing time-dependent value functions $V_k : \mathcal{S} \to [0,1], k \in \mathbb{Z}_{N+1}$. Proposition 2.1 characterized these value functions by Bellman recursion: for the invariance problem over the bounded set $\mathcal{A} \in \mathcal{B}(\mathcal{S})$ as the safe set and the finite time horizon $\mathbb{Z}_{N+1}$,

$$V_k(s) = \mathbf{1}_{\mathcal{A}}(s) \int_{\mathcal{S}} V_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s), \quad V_N(s) = \mathbf{1}_{\mathcal{A}}(s), \tag{4.1}$$

which leads to the expression $P_s(\mathcal{A}) = V_0(s)$ for the solution.

The Bellman recursion in (4.1) indicates that the supports of value functions are contained in the set $\mathcal{A}$ (namely, they are equal to zero over the complement of $\mathcal{A}$). We are thus only interested in computing the value functions over the set $\mathcal{A}$, which allows simplifying the recursion in (4.1) as follows

$$V_k(s) = \int_{\mathcal{A}} V_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s), \quad V_N(s) = 1, \quad \forall s \in \mathcal{A}, k \in \mathbb{Z}_N. \tag{4.2}$$

Let us denote with $\mathbb{B}(\mathcal{A})$ the space of bounded and measurable functions $f : \mathcal{A} \to \mathbb{R}$, and let us assign to this space the infinity norm to this space:

$$\|f\|_\infty = \sup\{|f(s)|,\, s \in \mathcal{A}\}, \quad \forall f \in \mathbb{B}(\mathcal{A}).$$

The linear operator $\mathcal{R}_\mathcal{A}$, defined over $\mathbb{B}(\mathcal{A})$ by

$$\mathcal{R}_\mathcal{A} f(s) = \int_\mathcal{A} f(\bar{s}) T_\mathfrak{s}(d\bar{s}|s), \quad \forall s \in \mathcal{A}, f \in \mathbb{B}(\mathcal{A}), \tag{4.3}$$

characterizes the solution of the recursion in (4.2) as $V_k(s) = \mathcal{R}_\mathcal{A}^{N-k}(V_N)(s)$, for any $k \in \mathbb{Z}_{N+1}$. Moreover, we suppose that the transition kernel $T_\mathfrak{s}(d\bar{s}|s)$ of the Markov process admits a density function $t_\mathfrak{s}(\bar{s}|s)$, such that $T_\mathfrak{s}(d\bar{s}|s) = t_\mathfrak{s}(\bar{s}|s)d\bar{s}$.

4.3 Approximation Schemes and Error Quantification

The goal of this section is to propose numerical schemes for approximating the value functions $V_k, k \in \mathbb{Z}_N$, with an explicit quantification of the approximation error. While Chapters 2 and 3 proposed approximations of the value functions V_k by piecewise constant functions, in this chapter we are interested in considering approximations via higher-order interpolations.

4.3.1 Error Quantification of a Projection Over a Function Space

Consider a function space $\Phi = span\{\phi_1(s), \phi_2(s), \ldots, \phi_n(s)\}$ as a subset of $\mathbb{B}(\mathcal{A})$, and a projection operator $\Pi_\mathcal{A} : \mathbb{B}(\mathcal{A}) \to \Phi$ that satisfies the inequality

$$\|\Pi_\mathcal{A}(f) - f\|_\infty \leq \mathcal{E}(f) \tag{4.4}$$

under some regularity conditions on f (beyond $f \in \mathbb{B}(\mathcal{A})$, see assumptions in Theorem 4.1), and where the bound $\mathcal{E}$ depends on the properties of the function f. With focus on a linear projection operator, the next result provides a useful tool for approximating the solution of the invariance problem.

Theorem 4.1 *Assume that a linear operator $\Pi_\mathcal{A}$ satisfies the inequality*

$$\|\Pi_\mathcal{A}(t_\mathfrak{s}(\bar{s}|\cdot)) - t_\mathfrak{s}(\bar{s}|\cdot)\|_\infty \leq \epsilon, \quad \forall \bar{s} \in \mathcal{A}, \tag{4.5}$$

and that there exists a finite constant $\mathcal{M}$, such that

$$\int_\mathcal{A} |\Pi_\mathcal{A}(t_\mathfrak{s}(\bar{s}|s))|\, d\bar{s} \leq \mathcal{M}, \quad \forall s \in \mathcal{A}. \tag{4.6}$$

Define the value functions $\bar{V}_k$ as approximations of the value functions V_k (cf. (4.3)), by

$$\bar{V}_k = (\Pi_\mathcal{A}\mathcal{R}_\mathcal{A})^{N-k}(V_N), \quad k \in \mathbb{Z}_{N+1}. \tag{4.7}$$

Then it holds that

$$\|V_k - \bar{V}_k\|_\infty \leq E_k, \quad k \in \mathbb{Z}_{N+1}, \tag{4.8}$$

where the error E_k satisfies the difference equation

$$E_k = \mathcal{M} E_{k+1} + \mathscr{L}(\mathcal{A})\epsilon,$$

initialized by $E_N = 0$, and where $\mathscr{L}(\mathcal{A})$ denotes the Lebesgue measure of the set $\mathcal{A}$.

Corollary 4.1 *Under the assumptions raised in (4.5)-(4.6), the error E_k can be alternatively expressed explicitly as*

$$E_k = \epsilon \mathscr{L}(\mathcal{A}) \frac{1 - \mathcal{M}^{N-k}}{1 - \mathcal{M}}, \text{ for } \mathcal{M} \neq 1, \quad \text{and} \quad E_k = \epsilon \mathscr{L}(\mathcal{A})(N-k), \text{ for } \mathcal{M} = 1.$$

One possible general choice for the constant $\mathcal{M}$ is $\mathcal{M} = 1 + \epsilon \mathscr{L}(\mathcal{A})$.

Notice that the above error converges if $\mathcal{M} < 1$ as N goes to infinity, which makes the result applicable to the approximate computation of the infinite-horizon safety property with a finite approximation error.

4.3.2 Construction of the Projection Operator

In the ensuing sections we focus, for the sake of simplicity, on a state space that is Euclidean, namely $\mathcal{S} = \mathbb{R}^d$, where d is its finite dimension. In Section 4.5 we extend the upcoming results to be valid over Stochastic Hybrid Systems.

We discuss a general form for the interpolation operator. Let $\phi_j : \mathcal{D} \subset \mathbb{R}^d \to \mathbb{R}, j \in \mathbb{N}_n$ be independent functions defined over a generic set $\mathcal{D}$. The interpolation operator $\Pi_\mathcal{D}$ is defined as a projection map into the function space $\Phi = span\{\phi_1(s), \phi_2(s), \ldots, \phi_n(s)\}$, which projects any function $f : \mathcal{D} \to \mathbb{R}$ to a unique function $\Pi_\mathcal{D}(f) = \sum_{j=1}^n \alpha_j \phi_j$, using a finite set of data $\{(z_j, f(z_j)) | z_j \in \mathcal{D}, j \in \mathbb{N}_n\}$ and such that $\Pi_\mathcal{D}(f)(z_j) = f(z_j)$. The operator $\Pi_\mathcal{D}$ is guaranteed to verify the inequality in (4.4), namely $\|\Pi_\mathcal{D}(f) - f\|_\infty \leq \mathcal{E}_\mathcal{D}(f)$, under some regularity assumptions on its argument function f (cf. Corollary 4.2).

With focus on the invariance problem described in Section 2.2.2, let us select a partition $\{A_i\}_{i=1}^m$ for the set $\mathcal{A}$, with finite cardinality m. Using a basis $\{\phi_{ij}\}_{j=1}^n$, let us introduce the interpolation operators Π_{A_i} for the projection over each partition set A_i, which is done as described above by replacing the domain $\mathcal{D}$ with A_i. Finally, let us introduce the (global) linear operator $\Pi_\mathcal{A}$ on a function $f : \mathcal{A} \to \mathbb{R}$ by

$$\Pi_\mathcal{A}(f) = \sum_{i=1}^m \mathbf{1}_{A_i} \Pi_{A_i}(f|_{A_i}), \tag{4.9}$$

where $f|_{A_i}$ represents the restriction of the function f over the partition set A_i. The following result holds:

Theorem 4.2 *The operator in (4.9) satisfies the inequality in (4.4) with constant $\mathcal{E}(f) = \max\{\mathcal{E}_{A_i}(f|_{A_i}), i \in \mathbb{N}_m\}$, and where $\|\Pi_{A_i}(f|_{A_i}) - f|_{A_i}\|_\infty \leq \mathcal{E}_{A_i}(f|_{A_i})$.*

Corollary 4.2 *The result in Theorem 4.1 can be tailored to the operator in (4.9) and applied to the density $t_{\mathfrak{s}} = f$, under the assumptions (4.5)-(4.6) on $t_{\mathfrak{s}}$ and using the following two quantities:*

$$\epsilon = \max_i \epsilon_i, \text{ where } \|\Pi_{A_i}(t_{\mathfrak{s}}(\bar{s}|\cdot)|_{A_i}) - t_{\mathfrak{s}}(\bar{s}|\cdot)|_{A_i}\|_\infty \leq \epsilon_i, \text{ for all } \bar{s} \in \mathcal{A};$$

$$\mathcal{M} = \max_i \mathcal{M}_i, \text{ where } \int_{\mathcal{A}} |\Pi_{A_i}(t_{\mathfrak{s}}(\bar{s}|s))|\, d\bar{s} \leq \mathcal{M}_i, \text{ for all } s \in A_i.$$

Here ϵ_i represents the interpolation error on the density function over the partition set A_i.

4.3.3 Approximation Algorithm

An advantage of the interpolation operator in (4.9) is that $\Pi_{\mathcal{A}}(f)$ is fully characterized by the interpolation coefficients α_{ij}, such that

$$\Pi_{\mathcal{A}}(f) = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij}\phi_{ij}\mathbf{1}_{A_i}.$$

The set of interpolation coefficients α_{ij} are computable by matrix multiplication based on the data $\{(z_{ij}, f(z_{ij})), i \in \mathbb{N}_m, j \in \mathbb{N}_n\}$, where the matrix depends on the interpolation points z_{ij} and on the basis functions ϕ_{ij} and can be computed off-line (see step 4 in Algorithm 6).

Let us now focus on the recursion in (4.7), $\bar{V}_k = \Pi_{\mathcal{A}}\mathcal{R}_{\mathcal{A}}(\bar{V}_{k+1})$, given the initialization $\bar{V}_N = 0$, for the approximate computation of the value functions. This recursion indicates that the approximate value functions $\bar{V}_k, k \in \mathbb{Z}_N$, belong to the image of the operator $\Pi_{\mathcal{A}}$. Let us express these value functions by

$$\bar{V}_k = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij}^k \phi_{ij}\mathbf{1}_{A_i},$$

where α_{ij}^k denote the interpolation coefficients referring to $\bar{V}_k$ (at step k). This suggests that we need to store and update the coefficients α_{ij}^k for each iteration in (4.7). Writing the recursion in the form $\bar{V}_k = \Pi_{\mathcal{A}}\left(\mathcal{R}_{\mathcal{A}}(\bar{V}_{k+1})\right)$ indicates that it is sufficient to evaluate the function $\mathcal{R}_{\mathcal{A}}(\bar{V}_{k+1})$ over the interpolation points in order to compute the coefficients α_{ij}^k. In the following, the pair i, r indicate the indices of the related partition sets, namely A_i, A_r, whereas the pair of indices j, u show the ordering positions within partition sets. For an arbitrary interpolation point z_{ru} we have:

$$\mathcal{R}_{\mathcal{A}}(\bar{V}_{k+1})(z_{ru}) = \int_{\mathcal{A}} \bar{V}_{k+1}(\bar{s})t_{\mathfrak{s}}(\bar{s}|z_{ru})d\bar{s} = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij}^{k+1} \int_{A_i} \phi_{ij}(\bar{s})t_{\mathfrak{s}}(\bar{s}|z_{ru})d\bar{s}.$$

Introducing the following quantities $Q_{ij}(r,u) = \int_{A_i} \phi_{ij}(\bar{s})t_{\mathfrak{s}}(\bar{s}|z_{ru})d\bar{s}$, we have

that

$$\bar{V}_k(r,u) = \mathcal{R}_{\mathcal{A}}(\bar{V}_{k+1})(z_{ru}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij}^{k+1} Q_{ij}(r,u).$$

Algorithm 6 provides a general procedure for the discrete computation of the interpolation coefficients and of the approximate value functions.

Algorithm 6 Approximate computation of the value functions $\bar{V}_k$

Require: Density function $t_{\mathfrak{s}}(\bar{s}|s)$, safe set $\mathcal{A}$

1: Select a finite m-dimensional partition of the set $\mathcal{A} = \cup_{i=1}^{m} A_i$ (A_i are non-overlapping)
2: For each A_i, select interpolation basis functions ϕ_{ij} and points $z_{ij} \in A_i, j \in \mathbb{N}_n$
3: Compute $Q_{ij}(r,u) = \int_{A_i} \phi_{ij}(\bar{s}) t_{\mathfrak{s}}(\bar{s}|z_{ru}) d\bar{s}$, where $i, r \in \mathbb{N}_m$ and $j, u \in \mathbb{N}_n$
4: Compute matrix representation of operators Π_{A_i}
5: Set $k = N-1$ and $\bar{V}_N(i,j) = 1$ for all i, j
6: **if** $k \geq 0$ **then**
7: Compute interpolation coefficients α_{ij}^{k+1} given $\bar{V}_{k+1}(i,j)$, using matrices in step 4
8: Compute values $\bar{V}_k(r,u)$ based on $\bar{V}_k(r,u) = \sum_i \sum_j \alpha_{ij}^{k+1} Q_{ij}(r,u)$
9: $k = k-1$
10: **end if**

Ensure: Approximate value functions $\bar{V}_k, k \in \mathbb{Z}_{N+1}$

Next, we provide a condition on the selection of the basis functions and of the interpolation points, leading to a simplification of Algorithm 6.

Theorem 4.3 ([59]) *Assume that there exists a choice of interpolation points z_{ij} and of basis functions ϕ_{ij} such that*

$$\det \begin{bmatrix} \phi_{i1}(z_{i1}) & \cdots & \phi_{in}(z_{i1}) \\ \vdots & \ddots & \vdots \\ \phi_{i1}(z_{in}) & \cdots & \phi_{in}(z_{in}) \end{bmatrix} \neq 0, \quad \forall i \in \mathbb{N}_m.$$

Then, there additionally exists an equivalent basis made up of functions ψ_{ij} such that

$$span\{\psi_{i1}, \psi_{i2}, \ldots, \psi_{in}\} = span\{\phi_{i1}, \phi_{i2}, \ldots, \phi_{in}\}$$

for all $i \in \mathbb{N}_m$, and which is related to the interpolation coefficients $\alpha_{ij}^k = \bar{V}_k(i,j)$.

Theorem 4.3 ensures that by utilizing the basis functions ψ_{ij} step 4 in Algorithm 6 can be skipped, and that the main update (steps 7 and 8) can be simplified as follows:

$$\bar{V}_k(r,u) = \sum_{i=1}^{m}\sum_{j=1}^{n} \bar{V}_{k+1}(i,j) Q_{ij}(r,u), \quad \bar{V}_N(i,j) = 1.$$

A sufficient condition for the satisfaction of the assumption in Theorem 4.3 is the selection of a basis $\{\phi_{i1}, \ldots, \phi_{in}\}$ as a Chebyshev (or Haar) system [59], for all $i \in \mathbb{N}_m$. In this case, the choice of the distinct interpolation points z_{ij} can be made freely, for each partition set A_i (instances of this selection will be given below).

In Algorithm 6, the interpolation points z_{ij} are in general pair-wise distinct. Extending the domain of interpolation A_i to its closure $\bar{A}_i$, it is legitimate to use boundary points as interpolation points, which can lead to a reduction of the number of integrations required in Algorithm 6. However, special care should be taken, since the interpolation operator should produce a continuous output over the boundaries of the neighboring partition sets. In the ensuing sections, we will exploit this feature upon selecting equally spaced points.

4.4 Special Forms of the Projection Operator

In this section we leverage known interpolation theorems for the construction of the projection operator $\Pi_{\mathcal{A}}$. These theorems are presented over a general domain $\mathcal{D}$ and are then used to derive specific error bounds for the problem of interest presented in Section 4.2. [1]

4.4.1 Piecewise Constant Approximations

We focus on the approximation of a function by a piecewise constant one, which has inspired the chapters 2 and 3. For this purpose we select $n = 1$ and the basis functions $\phi_{i1}(s) = \psi_{i1}(s) = 1$, for all $s \in A_i, i \in \mathbb{N}_m$. The procedure is detailed in Algorithm 7 which is essentially Algorithm 1 (presented in Chapter 2) combined with the approximate solution of the safety problem over the abstracted Markov chain. As we discussed in Chapter 2, the associated error is equal to $N\mathscr{L}(\mathcal{A})h\delta$ where h is the Lipschitz constant of the conditional density function (cf. Theorem 2.2).

For piecewise constant approximation of this section, the constant $\mathcal{M}$ of Theorem 4.1 is simplified to

$$\mathcal{M} = \max\left\{\int_{\mathcal{A}} t_{\mathbf{s}}(\bar{s}|s)d\bar{s}, s \in \mathcal{A}\right\},$$

which is upper bounded by one. This constant might be less than one in some cases [75], depending on the conditional density function and the safe set, which leads to an error (cf. Corollary 4.1) that converges as time horizon N grows.

Let us compare Algorithms 6 and 7 in terms of their computational complexity. Algorithm 6 requires $mn(mn+1)$ integrations in the marginalization steps (3 and 4), whereas $m(m+1)$ integrations are required in Algorithm 7. Furthermore, steps 5 and 8 in Algorithm 6 can be skipped only if a Chebyshev (Haar) system can be

[1] In the rest of this chapter, we employ normal typeset for bounds derived from general interpolation theorems, whereas calligraphic letters are used for theorems developed specifically for this chapter.

Algorithm 7 Piecewise constant computation of the value functions $\bar{V}_k$

Require: Density function $t_s(\bar{s}|s)$, safe set $\mathcal{A}$
1: Select a finite partition of the set $\mathcal{A} = \cup_{i=1}^{m} A_i$ (A_i are non-overlapping), where m represents the cardinality of the partition
2: For each A_i, select one representative point $z_i \in A_i$
3: Compute matrix P with entries $P(r,i) = Q_i(r) = \int_{A_i} t_s(\bar{s}|z_r)d\bar{s}$, where $i, r \in \mathbb{N}_m$
4: Set $k = N-1$ and $\bar{V}_N(i) = 1$ for all $i \in \mathbb{N}_m$
5: **if** $k \geq 0$ **then**
6: Compute the column vector $\bar{V}_k$ based on $\bar{V}_k = P\bar{V}_{k+1}$
7: $k = k-1$
8: **end if**
Ensure: Approximate value functions $\bar{V}_k, k \in \mathbb{Z}_{N+1}$

selected, whereas these steps are not needed at all in Algorithm 7. As a bottom line, higher interpolation orders increase the computational complexity of the approximation procedure, however this can as well lead to a lower global approximation error. Since the global approximation error depends on the local partitioning sets (their diameter, size, and the local continuity of the density function), for a given error higher interpolation procedures may require partitions with lower cardinality.

4.4.2 Higher-Order Approximations For 1-Dimensional Systems

In this section we study higher-order interpolations over the real axis, where the partition sets A_i are real intervals. We use this simple setting to quantify the error related to the approximate solution of the invariance problem. In order to assess the effect of the choice of the interpolation points on the approximation error and on the computational complexity of the method, we compare two different sets of interpolation points: equally spaced points and Chebyshev nodes.

Theorem 4.4 ([59]) *Let f be a real $(n+1)$-times continuously differentiable function on the bounded (one-dimensional) interval $\mathcal{D} = [\alpha, \beta]$. For the interpolation polynomial $\Pi_{\mathcal{D}}(f) \in span\{1, s, s^2, \ldots, s^n\}$, with $(n+1)$ pair-wise distinct points $\{z_0, z_1, \ldots, z_n\} \subset \mathcal{D}$, and condition $\Pi_{\mathcal{D}}(f)(z_j) = f(z_j), j \in \mathbb{Z}_{n+1}$, there exist a $\xi \in \mathcal{D}$ such that*

$$f(s) - \Pi_{\mathcal{D}}(f)(s) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{j=0}^{n} (s - z_j), \quad \forall s \in \mathcal{D}.$$

Equally-Spaced Interpolation Points. The following result can be adapted from [59].

Theorem 4.5 *Consider equally spaced interpolation points* $\{z_j = \alpha + j\frac{\beta-\alpha}{n}, j \in \mathbb{Z}_{n+1}\}$. *The interpolation error is upper bounded by*

$$\|f - \Pi_{\mathcal{D}}(f)\|_\infty \leq \frac{M_n}{4(n+1)} \left(\frac{\beta-\alpha}{n}\right)^{n+1}, \quad M_n = \max\left\{|f^{(n+1)}(s)|, s \in \mathcal{D}\right\}.$$

Application to the Invariance Problem. Consider a one dimensional invariance problem with a partitioning of $\mathcal{A} = \cup_{i=1}^m A_i$ which is such that $A_i = [\alpha_i, \beta_i]$. Define the interpolation operator $\Pi_{A_i}\left(t_{\mathfrak{s}}(\bar{s}|\cdot)|_{A_i}\right)$ over the basis $\Phi = span\{1, s, s^2, \ldots, s^n\}$ using equally spaced interpolation points $\{z_{ij} \in A_i, j \in \mathbb{Z}_{n+1}\}$. Then we can easily derive the following constants:

$$\mathcal{M}_n = \max\left\{\left|\frac{\partial^{n+1} t_{\mathfrak{s}}(\bar{s}|s)}{\partial s^{n+1}}\right|, s, \bar{s} \in \mathcal{A}\right\}, \qquad \epsilon = \frac{\mathcal{M}_n}{4(n+1)}\left(\frac{\delta}{n}\right)^{n+1},$$

and $\delta_i = \beta_i - \alpha_i, i \in \mathbb{N}_m, \delta = \max_i \delta_i$. Changing the basis of interpolation gives us the opportunity to obtain another value for $\mathcal{M}$ to be used in the error computation. Let us select the interpolation basis functions to be Lagrange polynomials:

$$L_{ij}(s) = \prod_{u=0, u\neq j}^{n} \frac{s - z_{iu}}{z_{ij} - z_{iu}}.$$

This leads to a projection with a special form, namely

$$\Pi_{A_i}\left(t_{\mathfrak{s}}(\bar{s}|s)|_{A_i}\right) = \sum_{j=0}^{n} \alpha_{ij} s^j = \sum_{j=0}^{n} t_{\mathfrak{s}}(\bar{s}|z_{ij}) L_{ij}(s).$$

Computing the constants $\kappa_i = \max\left\{\sum_{j=0}^n |L_{ij}(s)|, s \in A_i\right\}$ yields the following choice of $\mathcal{M}$:

$$\int_{\mathcal{A}} |\Pi_{A_i}(t_{\mathfrak{s}}(\bar{s}|s)|_{A_i})|\, d\bar{s} \leq \kappa_i \int_{\mathcal{A}} t_{\mathfrak{s}}(\bar{s}|z_{ij}) d\bar{s} \leq \kappa_i, \text{ and } \mathcal{M} = \max_i \kappa_i.$$

Having the values of ϵ and $\mathcal{M}$ we are ready to implement Algorithm 6 for equally spaced points and polynomial basis functions of degree at most n, with the pre-specified error of Theorem 4.1.

Chebyshev Nodes. The following statement can be adapted from [59].

Theorem 4.6 *Let* f *be a real* $(n+1)$*-times continuously differentiable function on the interval* $\mathcal{D} = [\alpha, \beta]$. *For the interpolation polynomial* $\Pi_{\mathcal{D}}(f) \in span\{1, s, s^2, \ldots, s^n\}$ *with Chebyshev nodes*

$$z_j = \frac{\alpha+\beta}{2} + \frac{\beta-\alpha}{2}\cos\left(\frac{2j+1}{2(n+1)\pi}\right) \quad j \in \mathbb{Z}_{n+1},$$

n	1	2	3	4	5	6	7	8	9	10	11	12
$\frac{\epsilon_2}{\epsilon_1}$	0.50	0.50	0.42	0.33	0.25	0.19	0.14	0.10	0.07	0.05	0.04	0.03

Table 4.1: Ratio between equally spaced pints (ϵ_1) versus Chebyschev nodes (ϵ_2), expressed with double digit precision, for different interpolation orders (n).

and values $\Pi_{\mathcal{D}}(f)(z_j) = f(z_j)$*, we have*

$$\|f - \Pi_{\mathcal{D}}(f)\|_\infty \leq \frac{M_n}{2^n(n+1)!}\left(\frac{\beta-\alpha}{2}\right)^{n+1}, \quad M_n = \max\{|f^{n+1}(s)|, s \in \mathcal{D}\}.$$

Application to the Reach-Avoid Problem. We can implement Algorithm 6 for Chebyshev nodes and Chebyshev polynomials of degree n, given a pre-specified error in Theorem 4.1, and with the following value of ϵ:

$$\epsilon = \frac{\mathcal{M}_n}{2^n(n+1)!}\left(\frac{\delta}{2}\right)^{n+1},$$

where the quantity $\mathcal{M}_n$ is that defined for equally spaced points. The only difference between the selection of equally spaced points and of Chebyshev nodes is the value of ϵ. The ratio of ϵ for these two cases (denoted respectively ϵ_1 and ϵ_2) is presented in Table 4.1 as a function of n (interpolation order). The advantage gained by using Chebyshev nodes is distinctive over larger values of the interpolation order.

It is worth mentioning that, unlike the piecewise constant case of Chapters 2 and 3, the global error of higher-order approximation approaches is a nonlinear function of the partition size δ, namely it depends on a power of the partition size contingent on the order of the selected interpolation operator.

4.4.3 Bilinear Interpolation For 2-Dimensional Systems

We directly tailor the results above to a general 2-dimensional system.

Theorem 4.7 *Consider a partially differentiable function $f(s_1, s_2)$, defined (for simplicity) over the unit square $\mathcal{D} = [0,1]^2$. For the interpolation operator*

$$\begin{aligned}\Pi_{\mathcal{D}}(f)(s_1,s_2) =& a_1 + a_2 s_1 + a_3 s_2 + a_4 s_1 s_2 \\ =& s_1(1-s_2)f(1,0) + s_1 s_2 f(1,1) \\ &+ (1-s_1)(1-s_2)f(0,0) + (1-s_1)s_2 f(0,1),\end{aligned}$$

the error is upper bounded by

$$\|f - \Pi_{\mathcal{D}}(f)\|_\infty \leq \frac{1}{8}\left[M_{s_1^2} + M_{s_2^2} + 2M_{s_1^2 s_2} + 2M_{s_2^2 s_1}\right],$$

where $\left|\frac{\partial^2 f}{\partial s_i^2}\right| \leq M_{s_i^2}, \left|\frac{\partial^3 f}{\partial s_i^2 s_{3-i}}\right| \leq M_{s_i^2 s_{3-i}}, i = 1, 2,$ *for all* $(s_1, s_2) \in \mathcal{D}$.

Application to the Invariance Problem.

With focus on a 2-dimensional invariance problem, consider a uniform partition (using squared partition sets) of diameter δ for the set $\mathcal{A}$. We employ a bilinear interpolation within each partition set $A_i = [\alpha_{i1}, \alpha_{i2}] \times [\beta_{i1}, \beta_{i2}]$ with basis $\{\phi_1(s) = 1, \phi_2(s) = s_1, \phi_3(s) = s_2, \phi_4(s) = s_1 s_2\}$, or with Lagrange polynomials

$$\psi_{i1}(s) = \frac{(\alpha_{i2} - s_1)(\beta_{i2} - s_2)}{(\alpha_{i2} - \alpha_{i1})(\beta_{i2} - \beta_{i1})}, \qquad \psi_{i2}(s) = \frac{(\alpha_{i2} - s_1)(s_2 - \beta_{i1})}{(\alpha_{i2} - \alpha_{i1})(\beta_{i2} - \beta_{i1})},$$
$$\psi_{i3}(s) = \frac{(s_1 - \alpha_{i1})(\beta_{i2} - s_2)}{(\alpha_{i2} - \alpha_{i1})(\beta_{i2} - \beta_{i1})}, \qquad \psi_{i4}(s) = \frac{(s_1 - \alpha_{i1})(s_2 - \beta_{i1})}{(\alpha_{i2} - \alpha_{i1})(\beta_{i2} - \beta_{i1})},$$

and compute the associated error, given the following value for ϵ:

$$\epsilon = \frac{\delta^2}{16}\left[\mathcal{M}_{s_1^2} + \mathcal{M}_{s_2^2} + \delta\sqrt{2}\mathcal{M}_{s_1^2 s_2} + \delta\sqrt{2}\mathcal{M}_{s_2^2 s_1}\right],$$

where

$$\left|\frac{\partial^2 t_{\mathfrak{s}}}{\partial s_i^2}(\bar{s}|s)\right| \leq \mathcal{M}_{s_i^2}, \left|\frac{\partial^3 t_{\mathfrak{s}}}{\partial s_i^2 s_{3-i}}(\bar{s}|s)\right| \leq \mathcal{M}_{s_i^2 s_{3-i}}, i = 1, 2, \forall s, \bar{s} \in A.$$

Note that the basis function ψ_{ij} is non-negative on the partition set A_i and that $\sum_{j=1}^4 \psi_{ij}(s) = 1$, which leads to a constant $\mathcal{M} = \max_{s \in A} \int_A t_{\mathfrak{s}}(\bar{s}|s) d\bar{s} \leq 1$.

4.4.4 Trilinear Interpolation For 3-Dimensional Systems

We now apply the results above to a general 3-dimensional system.

Theorem 4.8 *Consider a partially differentiable function* $f(s_1, s_2, s_3)$*, defined (for simplicity) over the unit cube* $\mathcal{D} = [0, 1]^3$*. For the interpolation operator*

$$\begin{aligned}\Pi_{\mathcal{D}}(f)(s_1, s_2, s_3) =& a_1 + a_2 s_1 + a_3 s_2 + a_4 s_3 \\ &+ a_5 s_1 s_2 + a_6 s_1 s_3 + a_7 s_2 s_3 + a_8 s_1 s_2 s_3 \\ =& (1 - s_1)(1 - s_2)(1 - s_3) f(0, 0, 0) + s_1 s_2 s_3 f(1, 1, 1) \\ &+ s_1(1 - s_2)(1 - s_3) f(1, 0, 0) + (1 - s_1) s_2 s_3 f(0, 1, 1) \\ &+ (1 - s_1) s_2 (1 - s_3) f(0, 1, 0) + s_1 (1 - s_2) s_3 f(1, 0, 1) \\ &+ (1 - s_1)(1 - s_2) s_3 f(0, 0, 1) + s_1 s_2 (1 - s_3) f(1, 1, 0),\end{aligned}$$

the error is upper bounded by the expression

$$\begin{aligned}\|f - \Pi_{\mathcal{D}}(f)\|_\infty \leq \frac{1}{8}[&M_{s_1^2} + M_{s_2^2} + M_{s_3^2} + 2M_{s_1^2 s_2} + 2M_{s_2^2 s_1} + 2M_{s_1^2 s_3} \\ &+ 2M_{s_3^2 s_1} + 2M_{s_2^2 s_3} + 2M_{s_3^2 s_2} + 6M_{s_1 s_2 s_3}],\end{aligned}$$

where $\left|\frac{\partial^2 f}{\partial s_i^2}\right| \leq M_{s_i^2}, \left|\frac{\partial^3 f}{\partial s_i^2 s_j}\right| \leq M_{s_i^2 s_j}, \left|\frac{\partial^3 f}{\partial s_1^2 s_2 s_3}\right| \leq M_{s_1 s_2 s_3}, \forall s = (s_1, s_2, s_3) \in \mathcal{D}$.

Application to the Invariance Problem. With focus on a 3-dimensional invariance problem, consider a uniform partition (using cubic sets) of diameter δ for the set $\mathcal{A}$. We employ a trilinear interpolation within each partition set and compute the associated error, given the following value for ϵ:

$$\epsilon = \frac{\delta^2}{24}\left[\mathcal{M}_{s_1^2} + \mathcal{M}_{s_2^2} + \mathcal{M}_{s_3^2}\right] + \frac{\delta^3}{12\sqrt{3}}\left[\mathcal{M}_{s_1^2 s_2} + \mathcal{M}_{s_2^2 s_1} + \mathcal{M}_{s_2^2 s_3} + \mathcal{M}_{s_3^2 s_2} + \mathcal{M}_{s_1^2 s_3} + \mathcal{M}_{s_3^2 s_1} + 3\mathcal{M}_{s_1 s_2 s_3}\right],$$

where $\left|\frac{\partial^2 t_{\mathfrak{s}}}{\partial s_i^2}(\bar{s}|s)\right| \leq M_{s_i^2}$, $\left|\frac{\partial^3 t_{\mathfrak{s}}}{\partial s_i^2 s_j}(\bar{s}|s)\right| \leq M_{s_i^2 s_j}$, and $\left|\frac{\partial^3 t_{\mathfrak{s}}}{\partial s_1^2 s_2 s_3}(\bar{s}|s)\right| \leq M_{s_1 s_2 s_3}$, for all $s = (s_1, s_2, s_3), \bar{s} = (\bar{s}_1, \bar{s}_2, \bar{s}_3) \in D$. Similar to the bilinear interpolation case, the function ψ_{ij} is non-negative on the partition set A_i and $\sum_{j=1}^{8} \psi_{ij}(s) = 1$, which leads to a constant $\mathcal{M} = \max_{s\in\mathcal{A}} \int_{\mathcal{A}} t_{\mathfrak{s}}(\bar{s}|s)d\bar{s} \leq 1$.

4.5 Extension to Stochastic Hybrid Systems

Consider the stochastic hybrid model $\mathfrak{S} = (\mathcal{Q}, n, T_q, T_x, T_r)$ defined in Chapter 2. As we discussed in Section 2.4, this model is a Markov process with the hybrid state space $\mathcal{S} = \cup_{q\in\mathcal{Q}}\{q\} \times \mathbb{R}^{n(q)}$ and the stochastic kernel $T_{\mathfrak{s}}$ described in (2.16).

Suppose the conditional kernels T_x, T_r admit density functions t_x, t_r. Consider a safe set $\mathcal{A} = \cup_{q\in\mathcal{Q}}\{q\} \times A_q$ with $A_q \in \mathcal{B}(\mathbb{R}^{n(q)})$ and define the operator $\mathcal{R}_{\mathcal{A}}$ acting on $f \in \mathbb{B}(\mathcal{A})$ as

$$\mathcal{R}_{\mathcal{A}} f(q,x) = T_q(q|(q,x)) \int_{A_q} f(q,\bar{x}) t_x(\bar{x}|(q,x))d\bar{x} + \sum_{\bar{q}\neq q} T_q(\bar{q}|(q,x)) \int_{A_{\bar{q}}} f(\bar{q},\bar{x}) t_r(\bar{x}|(q,x),\bar{q})d\bar{x}, \quad \forall x \in A_q, q \in \mathcal{Q}.$$

Given a partition $A_q = \cup_{i=1}^{m_q} A_{q,i}$ and a basis of interpolation functions $\{\psi_{q,ij}(x)\}$, we can construct the projection operator $\Pi_{\mathcal{A}}$ on $\mathbb{B}(\mathcal{A})$ by separately interpolating over the continuous domains associated to each discrete location. The following holds:

Theorem 4.9 *Suppose the conditional kernels of the SHS model satisfy the following inequalities*

$$\|\Pi_{\mathcal{A}}(T_q(q|(q,\cdot))t_x(\bar{x}|(q,\cdot))) - T_q(q|(q,\cdot))t_x(\bar{x}|(q,\cdot))\|_\infty \leq \mathcal{E}_x, \quad \forall q \in \mathcal{Q}, \forall \bar{x} \in A_q,$$
$$\|\Pi_{\mathcal{A}}(T_q(\bar{q}|(q,\cdot))t_r(\bar{x}|(q,\cdot),\bar{q})) - T_q(\bar{q}|(q,\cdot))t_r(\bar{x}|(q,\cdot),\bar{q})\|_\infty \leq \mathcal{E}_r, \quad \forall q, \bar{q} \in \mathcal{Q}, \bar{q} \neq q, \forall \bar{x} \in A_{\bar{q}},$$

then the following error bound can be established:

$$\|\mathcal{R}_{\mathcal{A}}^{k}(V_N) - (\Pi_{\mathcal{A}}\mathcal{R}_{\mathcal{A}})^{k}(V_N)\|_{\infty} \leq E_k, \quad V_N = \mathbf{1}_{\mathcal{A}},$$
$$E_k = \lambda(\mathcal{E}_x + (\mathfrak{m}-1)\mathcal{E}_r) + \kappa E_{k+1}, \quad E_N = 0,$$

where $\lambda = \max_q \mathscr{L}(A_q)$, $\kappa = \max\left\{\sum_j |\psi_{q,ij}(x)|, x \in A_{q,i}, i \in \mathbb{N}_{m_q}, q \in \mathcal{Q}\right\}$, *and* $\mathfrak{m}$ *is the cardinality of the set of discrete locations.*

4.6 Case Studies

We develop a few case studies in different dimensions to investigate the proposed higher-order approximation approach.

4.6.1 A 1-Dimensional Case Study

Consider a probabilistic safety problem with the safe set $\mathcal{A} = [0, 2]$ and the time horizon $N = 10$, over a model characterized by the kernel $T_{\mathfrak{s}}(d\bar{s}|s) = g(s+c-\bar{s})d\bar{s}$, where $c = 1.3035$, and the function g is defined as:

$$g(u) = \begin{cases} 3.57485\dfrac{1}{u^2}\exp\left(-u - \dfrac{1}{u}\right), & u > 0, \\ 0, & u \leq 0. \end{cases}$$

Selecting an approximation error $E_0 = 0.01$, we compute the required number of partition sets to abide by such figure. Using piecewise constant approximations based on a global Lipschitz constant (cf. Section 4.4.1) yields a value $h = 6.90$ and the error function $E_0 = N\mathscr{L}(\mathcal{A})h\delta$. This leads to a required number of partition sets $m = 27616$ and a total number of integrations $m(m+1) = 7.6 \times 10^8$ (the number of integrations is here conceived as a proxy for computational complexity).

Now consider algorithms and error bounds developed for higher-order approximations. The constants $\mathcal{M}_n$ are

$$\mathcal{M}_1 = 88.93, \quad \mathcal{M}_2 = 2063.65, \quad \mathcal{M}_3 = 79064.41, \quad \mathcal{M}_4 = 5428040,$$

whereas $\mathcal{M}$ is computed based on the following optimization problem:

$$\mathcal{M} = \max_{s\in\mathcal{A}} \int_{\mathcal{A}} t_{\mathfrak{s}}(\bar{s}|s)d\bar{s} = \max_{s\in\mathcal{A}} \int_0^2 g(s+c-\bar{s})d\bar{s} = \max_{s\in\mathcal{A}} \int_{s+c-2}^{s+c} g(u)du,$$

which leads to $s_{opt} = 0.82$ and $\mathcal{M} = 0.96$.

Table 4.2 compares the number of partition sets and the number of integrations required to reach an approximation error $E_0 = 0.01$, using equally spaced points and Chebyshev nodes. Notice that the two methods coincide for $n = 0$. The formulas for the number of integrations are an adaptation of the corresponding

uniform partitioning	total number of partitions		number of integrations	
$E_0 = 0.01$	m_1	m_2	$m_1(n+1)(m_1 n+1)$	$m_2^2(n+1)^2$
piecewise constant, $n=0$	23357	23357	$5.5 \cdot 10^8$	$5.5 \cdot 10^8$
piecewise linear, $n=1$	275	194	$1.5 \cdot 10^5$	94864
piecewise quadratic, $n=2$	67	53	27135	25281
third-order, $n=3$	36	29	15696	13456
fourth-order, $n=4$	28	22	15820	12100
uniform partitioning	total number of partitions		number of integrations	
$E_0 = 0.001$	m_1	m_2	$m_1(n+1)(m_1 n+1)$	$m_2^2(n+1)^2$
piecewise constant, $n=0$	233563	233563	$5.5 \cdot 10^{10}$	$5.5 \cdot 10^{10}$
piecewise linear, $n=1$	868	614	1508584	1507984
piecewise quadratic, $n=2$	143	114	123123	116964
third-order, $n=3$	64	52	49408	43264
fourth-order, $n=4$	43	35	37195	30625

Table 4.2: Number of partition sets and integrations for equally spaced points (indexed by 1) and for Chebyshev nodes (indexed by 2), given two error bounds $E_0 = 0.01, 0.001$.

ones developed to assess Algorithm 6 (this case deals with invariance, rather than the more general reach-avoid for Algorithm 6). Similar outcomes, performed for an experiment with error $E_0 = 0.001$, are also reported. These results show that Chebyshev nodes require in general a lower number of partition sets and therefore fewer integrations. The values are comparable since the ratio ϵ_2/ϵ_1 is smaller for larger values of n, as per Table 4.1. Notice further that equally spaced points give the opportunity to select common boundary points over adjacent partition sets as interpolation points, which can lead to a reduction on the associated number of integrations. However, interestingly the complexity is in general not monotonically decreasing with the order.

4.6.2 A 2-Dimensional Case Study

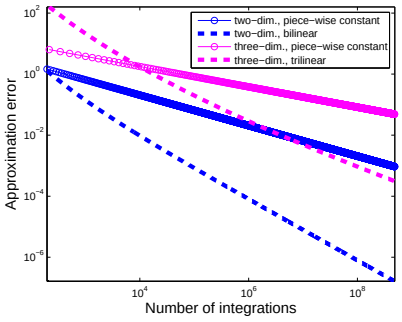
Consider the computational benchmark of Section 2.6.1 with a 2-dimensional state space. Let us select the covariance matrix $W = 0.5\,\mathbb{I}_2$ for the process noise and discuss the probabilistic invariance problem over the safe set $\mathcal{A} = [-1, 1]^2$ and the time horizon $\mathbb{Z}_{N+1}$. The following constants are needed to compute the error:

$$\mathcal{M} = 0.71, \quad \mathcal{M}_{s_1^2} = 2.23, \quad \mathcal{M}_{s_2^2} = 0.72, \quad \mathcal{M}_{s_1^2 s_2} = 3.80, \quad \mathcal{M}_{s_1 s_2^2} = 2.17.$$

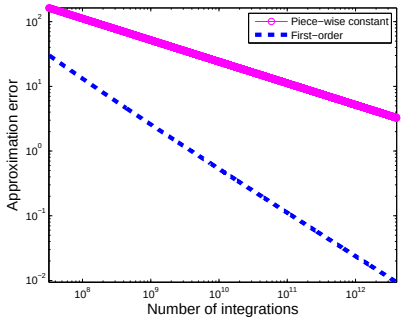
Table 4.3 compares the complexity of piecewise constant and bilinear approximations, for different values of the global error E_0. Similarly, Figure 4.1(a) (on the left) compares the two approximations (blue lines). The vertical axis represents the global approximation error, whereas the horizontal axis indicates the corresponding number of integrations, pointing to the computational complexity of each method. For a given computational complexity, bilinear interpolations approximate the solution with less error and their performance is dimensionally better in compared to the piecewise constant approximations. Similarly, for a given error threshold, less computations are required when using bilinear interpolations.

	piecewise constant		bilinear	
error	number of partitions per dimension	number of integrations	number of partitions per dimension	number of integrations
E_0	m_1	m_1^2	m_2	$4(m_2+1)^2$
0.1	206	$4.2 \cdot 10^4$	18	1444
0.01	2053	$4.2 \cdot 10^6$	49	10^4
0.001	20525	$4.2 \cdot 10^8$	145	$8.5 \cdot 10^4$
0.0001	205241	$4.2 \cdot 10^{10}$	448	$8.1 \cdot 10^5$

Table 4.3: Piecewise constant versus bilinear approximations.



(a) 2- and 3-dimensional models, Secs. 4.6.2, 4.6.3



(b) Hybrid model, Sec. 4.6.4

Figure 4.1: Error comparison between piecewise constant versus higher-order approximations, as a function of their computational complexity, for three case studies.

4.6.3 A 3-Dimensional Case Study

Consider the computational benchmark of Section 2.6.1 with three dimensional state space, covariance matrix $W = 0.5\,\mathbb{I}_3$, and the safe set $\mathcal{A} = [-1,1]^3$. The following constants are needed to compute the error:

$$\mathcal{M} = 0.60, \mathcal{M}_{s_1^2} = 2.66, \mathcal{M}_{s_2^2} = 0.33, \mathcal{M}_{s_3^2} = 1.50, \mathcal{M}_{s_1^2 s_2} = 3.47, \mathcal{M}_{s_2^2 s_1} = 1.28,$$
$$\mathcal{M}_{s_2^2 s_3} = 0.95, \mathcal{M}_{s_3^2 s_2} = 1.92, \mathcal{M}_{s_1^2 s_3} = 8.37, \mathcal{M}_{s_3^2 s_1} = 6.27, \mathcal{M}_{s_1 s_2 s_3} = 2.56.$$

Table 4.4 compares piecewise constant and trilinear approximations, for different values of the global error E_0. Similarly, Figure 4.1(a) (on the left) compares the two approximations (magenta lines). Recall that there is a tradeoff between local computations and global error for higher-order interpolations. Thus, if we consider a large global error, piecewise approximations may be computationally favorable (left of the crossing in the magenta curves). However, for small error bounds the performance of trilinear interpolations is much better in comparison with that of piecewise constant approximations.

	piecewise constant		trilinear	
error	number of partitions per dimension	number of integrations	number of partitions per dimension	number of integrations
E_0	m_1	m_1^3	m_2	$8(m_2+1)^3$
0.1	383	$5.6 \cdot 10^7$	30	$2.4 \cdot 10^5$
0.01	3825	$5.6 \cdot 10^{10}$	78	$3.9 \cdot 10^6$
0.001	38250	$5.6 \cdot 10^{13}$	220	$8.6 \cdot 10^7$
0.0001	382498	$5.6 \cdot 10^{16}$	681	$2.5 \cdot 10^9$

Table 4.4: Piecewise constant versus trilinear approximations.

4.6.4 Case Study For a Hybrid Model

Consider the stochastic hybrid model of the chemical reaction network described in Section 2.6.2. The model has two locations $\mathcal{Q} = \{q_1, q_2\}$ indicating a gene in active or inactive mode and the continuous dynamics described in (2.24). Again we select the safe set $\mathcal{A}$ to cover an interval of 10% variation around the steady state of the model and $N = 10$.

Figure 4.1(b) compares the approximation errors of piecewise constant and first-order approximations. The total number of integrations differ roughly only by a factor of two. Furthermore, considering for instance 1000 bins per dimension, the piecewise constant (zeroth-order) approximation has a global error equal to 32.64, whereas the first-order approximation leads to an error equal to 0.62, with only twice as many integrations involved in the procedure.

4.7 Conclusions

In this chapter we presented new algorithms, based on higher-order function approximation, for the efficient computation of approximate solutions of probabilistic invariance problem. The approach extends the results of Chapters 2, 3 using interpolation theory and is applicable to processes with hybrid state spaces. In the next chapter we extend our abstraction techniques to controlled Markov processes.

5 CHAPTER

Abstraction of Controlled Discrete-Time Markov Processes

This chapter is concerned with the generation of finite abstractions of Controlled Discrete-Time Markov Processes with general state and input spaces, to be employed for the formal verification of probabilistic properties by means of automatic techniques such as probabilistic model checking (with related software tools). As an extension of the work presented in earlier sections, we employ an abstraction procedure based on the partitioning of the *state and input spaces*, which generates a Markov Decision Process as an approximation of the original model. While the technique is applicable to a wide arena of probabilistic properties, with focus on the study of a particular specification (maximal invariance probability, over a finite horizon), maximally safe Markov policies of the abstracted model are used to design sub-optimal policies for the original continuous model. As before, we further provide upper bounds on the distance of such policies to the optimal solutions. The proposed algorithm is tested on a SHS model of a thermal load.

5.1 Introduction

In this chapter we study the problem of computing probabilistic properties for *controlled* discrete time Markov processes (dtMPs) evolving over continuous (uncountable) state and input spaces. We focus on the fundamental problem of computing the maximal invariance probability. This problem has been first studied in [4], which has characterized this concept and put forward an algorithm to compute this quantity. From a computational perspective, [2] has looked at the numerical evaluation of this specification for *autonomous* Markov processes. In Chapter 2 we have proposed adaptive and sequential procedures to extend the result of [2] and make it applicable to a wider class of models.

From a different perspective, approximation of controlled dtMP models for the problem of maximizing a reward function has been studied in [26]. The authors approximated the dtMP with a Markov Decision Process (MDP) and formulated the error associated with the dynamic programming characterization of the optimal solution, which hinges on the continuity properties of the stochastic kernel operator of the process and of the reward function. This result is not applicable to the maximal invariance probability, since the reward function is assumed to be *additive* while the reward function associated to the maximal invariance problem is *multiplicative*. Moreover, the cost-per-stage is assumed to be continuous, while it is specifically discontinuous in our setting.

MDP approximation of controlled Markov processes has also been investigated in the book by Kushner and Dupuis [56]. Their study lacks explicit quantification of the error made in the approximation which necessitates the study of formal MDP approximation methods for this class of models. The current chapter discusses formal MDP approximation of controlled dtMPs to address this weakness.

This chapter extends the result of Chapter 2 to controlled dtMPs by employing a partitioning procedures for the state and the input spaces of the process. The dtMP model is abstracted to an MDP and its invariance property is translated to the same property over the MDP. The maximal invariance probability is computed over the MDP and is exported to the original model with guaranteed error bounds: the approximation error of [26] grows exponentially with the time horizon, while in our case the error is a linear function of time horizon. To keep the discussion more focused, we first assume that the sets of feasible inputs are independent of the current state. In section 5.6 we show how the approach can be applied to state-dependent input spaces.

This chapter is structured as follows. Section 5.2 introduces the controlled dtMP models and the problem statement (computation of the maximal invariance probability and maximally safe policies). Section 5.3 discusses the sensitivity of the maximal invariance probability to the conditional density function of the dtMP. Section 5.4 proposes an abstraction algorithm based on the analysis of Section 5.3, to relate the general space controlled dtMP model to an MDP. Furthermore, with focus on the maximal invariance probability, the quantification of the error in the abstraction procedure is presented in Section 5.4. Section 5.5 deals with the algorithmic generation of the abstraction and the construction of policies which are within a given distance from the maximally safe policy. Section 5.6 adapts the results to the controlled dtMP models featuring state-dependent input spaces. Finally, Section 5.7 develops a numerical study and uses the results of this chapter to solve the optimal temperature control problem for the SHS model of a thermal load. Section 5.8 completes the chapter with conclusions and extensions.

5.2 Controlled Discrete-Time Markov Process

A controlled discrete-time Markov process is a tuple

$$\mathfrak{S}_c \doteq (\mathcal{S}, \mathcal{U}, \{\mathcal{U}(s) | s \in \mathcal{S}\}, T_{\mathfrak{s}}),$$

where $\mathcal{S} = \mathbb{R}^n$ is the continuous state space and $\mathcal{U}$ is a bounded input space. The set $\{\mathcal{U}(s)|s \in \mathcal{S}\}$ is a family of non-empty measurable subsets of $\mathcal{U}$, where $\mathcal{U}(s)$ is the set of feasible inputs when the system is at state $s \in \mathcal{S}$. The evolution of the Markov process is characterized by a Borel-measurable stochastic kernel $T_{\mathfrak{s}} : \mathcal{B}(\mathbb{R}^n) \times \mathcal{S} \times \mathcal{U} \to [0,1]$, which assign to any $s \in \mathcal{S}$ and $u \in \mathcal{U}(s)$ a probability measure on the Borel space $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$. For the sake of simplicity we assume that the set of feasible inputs are state-independent, i.e. $\mathcal{U}(s) = \mathcal{U}$ for all $s \in \mathcal{S}$, and postpone the discussion on the state-dependent input sets to Section 5.6. In the rest of this chapter we assume that the stochastic kernel of the process admits a conditional density function, namely $T_{\mathfrak{s}}(d\bar{s}|s,u) = t_{\mathfrak{s}}(\bar{s}|s,u)d\bar{s}$.

A Markov policy $\boldsymbol{\mu}$ for the Markov process $\mathfrak{S}_c$ is a sequence $\boldsymbol{\mu} = (\mu_0, \mu_1, \mu_2, \ldots)$ of universally measurable [1] maps [12] $\mu_k : \mathcal{S} \to \mathcal{U}, k \in \mathbb{N}_0$. The set of all Markov policies over the pair $(\mathcal{S},\mathcal{U})$ is denoted by $\mathcal{M}_c$. The invariance problem was defined in Chapter 2 for autonomous Markov processes. In this chapter we provide the definition on controlled processes for a given policy and study the maximal solution of the invariance problem.

Definition 5.1 *Let $\mathcal{A} \in \mathcal{B}(\mathcal{S})$ be a bounded Borel set as the safe set. We define the invariance problem under the Markov policy $\boldsymbol{\mu}$ over the time horizon $\mathbb{Z}_{N+1}$ as follows:*

$$p^{\boldsymbol{\mu}}_{s_0}(\mathcal{A}) \doteq \mathbb{P}^{\boldsymbol{\mu}}_{s_0}\{s(k) \in \mathcal{A} \text{ for all } k \in \mathbb{Z}_{N+1} | s(0) = s_0\}.$$

The maximal invariance probability is defined as

$$p^*_{s_0}(\mathcal{A}) = \sup\left\{p^{\boldsymbol{\mu}}_{s_0}(\mathcal{A}),\ \boldsymbol{\mu} \in \mathcal{M}_c\right\}, \quad \text{for all } s_0 \in \mathcal{A}.$$

A Markov policy $\boldsymbol{\mu}^*$ is maximally safe *if $p^{\boldsymbol{\mu}^*}_{s_0}(\mathcal{A}) = p^*_{s_0}(\mathcal{A})$ for all $s_0 \in \mathcal{A}$.*

It is worth mentioning that the above invariance problem can be defined over a more general set of policies, namely the sequences of universally measurable stochastic kernels [12], which are conditioned on the history of the process and account for randomization. The work [77] has proved that the maximally safe policies belong to the class of Markov policies, which motivated us to restrict our attention to this class of policies.

The next theorem characterizes the quantity $p^*_{s_0}(\mathcal{A})$ and the maximally safe Markov policy using the value functions V_k, V^*_k which are computed by the Bellman recursion.

Theorem 5.1 *Define the value functions $V_k : \mathcal{S} \times \mathcal{U} \to [0,1]$ and $V^*_k : \mathcal{S} \to [0,1]$ by the following recursion:*

$$V_k(s,u) = \boldsymbol{1}_{\mathcal{A}}(s) \int_{\mathcal{S}} V^*_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s,u) \quad \forall s \in \mathcal{S}, u \in \mathcal{U},$$
$$V^*_k(s) = \sup\{V_k(s,u),\ u \in \mathcal{U}\} \quad \forall s \in \mathcal{S},$$

[1] Consider Borel spaces X, Y. A set $A \subset X$ is called universally measurable if it is $\mu-$measurable for every finite measure μ on X. The function $f : X \to Y$ is called universally measurable if the set $f^{-1}(B)$ is universally measurable in X for every B in the Borel $\sigma-$algebra of Y.

initialized with $V_N^*(s) = \mathbf{1}_{\mathcal{A}}(s), s \in \mathcal{S}$. *Then* $V_0^*(s_0) = p_{s_0}^*(\mathcal{A})$. *If* $\mu_k^* : \mathcal{S} \to \mathcal{U}$ *is such that* $\mu_k^*(s) \in \arg\sup_{u \in \mathcal{U}} V_k(s,u)$, *then* $\boldsymbol{\mu}^* = (\mu_0^*, \mu_1^*, \mu_2^*, \ldots)$ *is a maximally safe Markov policy.*

The maximally safe policy of a dtMP does not admit a closed form in general. We are thus interested in computing policies that are close to the maximally safe policy. The next definition formally introduces this notion of approximation.

Definition 5.2 *A Markov policy* $\boldsymbol{\mu}$ *is* ε*–maximally safe if* $|p_{s_0}^*(\mathcal{A}) - p_{s_0}^{\boldsymbol{\mu}}(\mathcal{A})| \leq \varepsilon$ *for all* $s_0 \in \mathcal{A}$.

In the next section we study the dependence of the maximal invariance probability from the stochastic kernel of the process. Such an analysis is essential in developing abstraction methods for the computation of ε–maximally safe policies with any given ε.

5.3 Sensitivity Analysis of the Invariance Probability

In this section we study the sensitivity of the maximal invariance probability to the conditional density function of the process. In other words, assume that two Markov processes are defined over the same state and input spaces but with different density functions: what is the distance between their maximal invariance probability as a function of the distance between their density functions? The next theorem quantifies the effect of perturbations on the conditional density function, which answers this question.

Theorem 5.2 *Let* $\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$ *and* $\mathfrak{P}_c = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{p}})$ *be two Markov processes with the same state and input spaces. Assume that* $T_{\mathfrak{s}}, T_{\mathfrak{p}}$ *admit density functions* $t_{\mathfrak{s}}, t_{\mathfrak{p}}$ *such that*

$$|t_{\mathfrak{s}}(\bar{s}|s,u) - t_{\mathfrak{p}}(\bar{s}|s,u)| \leq \epsilon \quad \forall s, \bar{s} \in \mathcal{A}, u \in \mathcal{U}.$$

If we implement the recursion of theorem 5.1 for Markov processes $\mathfrak{S}_c, \mathfrak{P}_c$ *with value functions* V_k, V_k^*, *and* W_k, W_k^* *respectively, then*

$$|V_k^*(s) - W_k^*(s)| \leq (N-k)\epsilon \mathscr{L}(\mathcal{A}) \quad \forall s \in \mathcal{S}, k \in \mathbb{Z}_{N+1}.$$

Theorem 5.2 enables us to replace the density function of the given process with a second density function that may have a simpler structure; to efficiently compute the maximal invariance probability on it; and then to provide a guarantee over the maximal probability of the original process. There are two main questions to be answered at this point: how to select the density function $t_{\mathfrak{p}}(\cdot|s,u)$ and how to compute distance ϵ for this specific selection.

Construction of the density function $t_{\mathfrak{p}}(\cdot|s,u)$**.** We construct the density function $t_{\mathfrak{p}}(\cdot|s,u)$ based on the partitioning approach used in the previous chapters:

select a partition for the bounded input space $\mathcal{U} = \cup_{j=1}^{m_{\mathfrak{u}}} U_j$, choose a partition for the safe set $\mathcal{A} = \cup_{i=1}^{m_{\mathfrak{a}}} A_i$, and construct a partition for the state space $\mathcal{S} = \cup_{i=1}^{m_{\mathfrak{s}}} A_i$ with $m_{\mathfrak{s}} = m_{\mathfrak{a}} + 1$ and $A_{m_{\mathfrak{s}}} = \mathcal{S} \backslash \mathcal{A}$. We choose representative points for these partition sets $\{z_i \in A_i,\, i \in \mathbb{N}_{m_{\mathfrak{s}}}\}$ and $\{v_j \in U_j,\, j \in \mathbb{N}_{m_{\mathfrak{u}}}\}$, and define the conditional density function $t_{\mathfrak{p}}$ as

$$t_{\mathfrak{p}}(\bar{s}|s,u) = \sum_{j=1}^{m_{\mathfrak{u}}} \sum_{i=1}^{m_{\mathfrak{s}}} t_{\mathfrak{s}}(\bar{s}|z_i, v_j) \mathbf{1}_{U_j}(u) \mathbf{1}_{A_i}(s). \tag{5.1}$$

Note that $t_{\mathfrak{p}}$ is piecewise constant respect to the conditioned state and input pair (s,u).

Computation of the approximation quantity ϵ. We raise a continuity assumption on the density function of the process in order to establish Lipschitz continuity of the value functions as in Theorem 5.1 for the quantification of the term ϵ in Theorem 5.2.

Assumption 5.1 *Assume that the kernel $T_{\mathfrak{s}}$ admits density $t_{\mathfrak{s}}$, and that the following inequalities hold for finite positive constants $h_{\mathfrak{s}}, h_{\mathfrak{u}}$,*

$$\begin{aligned} |t_{\mathfrak{s}}(\bar{s}|s,u) - t_{\mathfrak{s}}(\bar{s}|s',u)| &\leq h_{\mathfrak{s}} \|s - s'\| \quad \forall s, s', \bar{s} \in \mathcal{A}, u \in \mathcal{U}, \\ |t_{\mathfrak{s}}(\bar{s}|s,u) - t_{\mathfrak{s}}(\bar{s}|s,u')| &\leq h_{\mathfrak{u}} \|u - u'\| \quad \forall s, \bar{s} \in \mathcal{A}, u, u' \in \mathcal{U}. \end{aligned}$$

Assumption 5.1 allows us to derive the following bound on ϵ which in turn can be used in Theorem 5.2.

Lemma 5.1 *Under Assumption 5.1, the density function $t_{\mathfrak{p}}$ of (5.1) satisfies the inequality*

$$|t_{\mathfrak{s}}(\bar{s}|s,u) - t_{\mathfrak{p}}(\bar{s}|s,u)| \leq h_{\mathfrak{s}} \delta_{\mathfrak{s}} + h_{\mathfrak{u}} \delta_{\mathfrak{u}} \quad \forall s, \bar{s} \in \mathcal{A}, u \in \mathcal{U},$$

where $h_{\mathfrak{s}}, h_{\mathfrak{u}}$ are defined in Assumption 5.1 and $\delta_{\mathfrak{s}}, \delta_{\mathfrak{u}}$ are the max diameters of the partitions $\{A_i,\, i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$, $\{A_j,\, j \in \mathbb{N}_{m_{\mathfrak{u}}}\}$, formally defined as

$$\begin{aligned} \delta_{\mathfrak{s}} &= \sup \left\{ \|s - s'\| \,\middle|\, s, s' \in A_i,\, i \in \mathbb{N}_{m_{\mathfrak{a}}} \right\}, \\ \delta_{\mathfrak{u}} &= \sup \left\{ \|u - u'\| \,\middle|\, u, u' \in U_j,\, j \in \mathbb{N}_{m_{\mathfrak{u}}} \right\}. \end{aligned}$$

Employing the quantity $\epsilon = h_{\mathfrak{s}} \delta_{\mathfrak{s}} + h_{\mathfrak{u}} \delta_{\mathfrak{u}}$ in Theorem 5.2 enables us to quantify the error introduced by using $t_{\mathfrak{p}}$ of (5.1) for the computation of the maximal invariance probability.

Corollary 5.1 *Suppose that we implement the recursion in Theorem 5.1 for Markov process $\mathfrak{S}_c, \mathfrak{P}_c$, characterized by the density function $t_{\mathfrak{s}}(\cdot|s,u)$ and its piecewise constant approximation $t_{\mathfrak{p}}(\cdot|s,u)$ of (5.1), respectively. Then*

$$|V_k^*(s) - W_k^*(s)| \leq (N-k)(h_{\mathfrak{s}} \delta_{\mathfrak{s}} + h_{\mathfrak{u}} \delta_{\mathfrak{u}}) \mathscr{L}(\mathcal{A}),$$

for all $s \in \mathcal{S}$ and $k \in \mathbb{Z}_{N+1}$.

Notice that the case of $k = 0$ gives an upper bound on the distance of the solutions of maximal invariance probability over the two processes. Corollary 5.1 can be seen as an extension of Theorem 2.2 in Chapter 2 for autonomous Markov processes: the additional term $h_{\mathfrak{u}}\delta_{\mathfrak{u}}$ appears in the error due to the discretization of the input space.

In Section 5.4 we provide an algorithm to abstract the process $\mathfrak{S}_c$ to an MDP. We prove that solution of the maximal invariance problem on the constructed finite-state MDP is equivalent to the same problem for $\mathfrak{P}_c$.

5.4 Algorithmic Abstraction of a Controlled dtMP as an MDP

In this section we propose an approach to abstract the controlled dtMP into an MDP, with the goal of approximate computation of $p^*_{s_0}(\mathcal{A})$ with guaranteed error bounds on the approximation.

The MDP $\mathfrak{D} = (\mathcal{S}_{\mathfrak{d}}, \mathcal{U}_{\mathfrak{d}}, T_{\mathfrak{d}})$ is constructed according to Algorithm 8, which is a generalization of the earlier Algorithm 1 proposed for the abstraction of Markov chains. Here $\mathcal{S}_{\mathfrak{d}} = \{z_i,\, i \in \mathbb{N}_{m_{\mathfrak{s}}}\}$ is the finite set of states and $\mathcal{U}_{\mathfrak{d}} = \{v_j,\, j \in \mathbb{N}_{m_{\mathfrak{u}}}\}$ is the finite set of inputs. $T_{\mathfrak{d}} : \mathcal{S}_{\mathfrak{d}} \times \mathcal{U}_{\mathfrak{d}} \times \mathcal{S}_{\mathfrak{d}} \to [0,1]$ is the state transition matrix such that $T_{\mathfrak{d}}(z, v, z')$ is the probability that the MDP $\mathfrak{D}$ moves into its new state z' when the input v is applied at state z, and thus induces the following conditional discrete probability distribution over the finite space $\mathcal{S}_{\mathfrak{d}}$:

$$T_{\mathfrak{d}}(z, v, z') = \mathbb{P}\left(z(k+1) = z' \middle| z(k) = z, v(k) = v\right).$$

In Algorithm 8, $\Xi_{\mathfrak{s}} : A_{\mathfrak{d}} \to 2^{\mathcal{A}}$ represents a set-valued map that associates to any point $z_i \in A_{\mathfrak{d}}$ the corresponding partition set $A_i \subset \mathcal{A}$. We also define $\Xi_{\mathfrak{u}} : \mathcal{U}_{\mathfrak{d}} \to 2^{\mathcal{U}}$ as the set-valued map that associates to any point $v_j \in \mathcal{U}_{\mathfrak{d}}$ the corresponding partition set $U_j \subset \mathcal{U}$ (this map will be used in Section 5.5). Furthermore, the map $\xi : \mathcal{A} \to A_{\mathfrak{d}}$ associates to any point $s \in \mathcal{A}$ of $\mathfrak{S}_c$ the corresponding discrete state in $A_{\mathfrak{d}}$. Additionally, notice that the absorbing set $\triangledown$ is added to the definition of the MDP $\mathfrak{D}$ in order to render the state transition matrix $T_{\mathfrak{d}}$ stochastic.

Remark 5.1 *Notice that Algorithm 8 can be applied to abstract a general controlled dtMP by an MDP with finite state and input spaces, regardless of the specifics of the maximal probabilistic invariance problem studied in this chapter (that is regardless of the given safe set $\mathcal{A}$), by assuming that $\mathcal{A} = \mathcal{S}$. The quantification of the abstraction error, to be carried out in Theorem 5.4, will however require that the set $\mathcal{A}$ (thus, as needed, the state space $\mathcal{S}$) is bounded.*

Given an MDP $\mathfrak{D} = (\mathcal{S}_{\mathfrak{d}}, \mathcal{U}_{\mathfrak{d}}, T_{\mathfrak{d}})$ with finite state and input spaces, and considering a safe set $A_{\mathfrak{d}}$, the invariance problem under the Markov policy $\boldsymbol{\vartheta} = (\vartheta_0, \vartheta_1, \vartheta_2, \ldots)$, $\vartheta_k : \mathcal{S}_{\mathfrak{d}} \to \mathcal{U}_{\mathfrak{d}}$, over the time horizon $\mathbb{Z}_{N+1}$, is defined as follows:

$$p^{\boldsymbol{\vartheta}}_{z_0}(A_{\mathfrak{d}}) \doteq \mathbb{P}^{\boldsymbol{\vartheta}}_{z_0}\{z(k) \in A_{\mathfrak{d}} \text{ for all } k \in \mathbb{Z}_{N+1} | z(0) = z_0\}.$$

Algorithm 8 Abstraction of model $\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$ by MDP $\mathfrak{D}$

Require: input model $\mathfrak{S}_c$, set $\mathcal{A}$

1: Select a finite partition of set $\mathcal{A}$ as $\mathcal{A} = \cup_{i=1}^{m_{\mathfrak{a}}} A_i$ (A_i are non-overlapping), where $m_{\mathfrak{a}}$ represents the cardinality of the partition
2: Select a finite partition of the input space $\mathcal{U}$ as $\mathcal{U} = \cup_{j=1}^{m_{\mathfrak{u}}} U_j$ (U_j are non-overlapping), where $m_{\mathfrak{u}}$ represents the cardinality of the partition
3: For each A_i, select single representative point $z_i \in A_i$, $\{z_i\} = \xi(A_i)$
4: For each U_j, select single representative point $v_j \in U_j$
5: Define $A_{\mathfrak{d}} = \{z_i,\, i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$ and take $\mathcal{S}_{\mathfrak{d}} = A_{\mathfrak{d}} \cup \{\triangledown\}$ as the finite state space of the MDP $\mathfrak{D}$ ($\triangledown$ being a dummy variable as explained in the text)
6: Define $\mathcal{U}_{\mathfrak{d}} = \{v_j,\, j \in \mathbb{N}_{m_{\mathfrak{u}}}\}$ as the finite input space of the MDP $\mathfrak{D}$
7: Compute the state transition matrix $T_{\mathfrak{d}}$ for $\mathfrak{D}$ as:

$$T_{\mathfrak{d}}(z, v, z') = \begin{cases} T_{\mathfrak{s}}(\Xi_{\mathfrak{s}}(z')|z, v), & z' \in A_{\mathfrak{d}}, z \in A_{\mathfrak{d}}, v \in \mathcal{U}_{\mathfrak{d}} \\ 1 - \sum_{\bar{z} \in A_{\mathfrak{d}}} T_{\mathfrak{s}}(\Xi_{\mathfrak{s}}(\bar{z})|z, v), & z' = \triangledown, z \in A_{\mathfrak{d}}, v \in \mathcal{U}_{\mathfrak{d}} \\ 1, & z' = z = \triangledown, v \in \mathcal{U}_{\mathfrak{d}} \\ 0, & z' \in A_{\mathfrak{d}}, z = \triangledown, v \in \mathcal{U}_{\mathfrak{d}} \end{cases}$$

Ensure: output MDP $\mathfrak{D} = (\mathcal{S}_{\mathfrak{d}}, \mathcal{U}_{\mathfrak{d}}, T_{\mathfrak{d}})$

The maximal invariance probability for the MDP $\mathfrak{D}$ is defined as

$$p^*_{z_0}(A_{\mathfrak{d}}) = \max\left\{p^{\boldsymbol{\vartheta}}_{z_0}(A_{\mathfrak{d}}),\, \boldsymbol{\vartheta} \in \mathcal{M}_{\mathfrak{d}}\right\},$$

for all $z_0 \in A_{\mathfrak{d}}$, where $\mathcal{M}_{\mathfrak{d}}$ is the set of all Markov policies over the pair $(\mathcal{S}_{\mathfrak{d}}, \mathcal{U}_{\mathfrak{d}})$. A Markov policy $\boldsymbol{\vartheta}^*$ is *maximally safe* if $p^{\boldsymbol{\vartheta}^*}_{z_0}(A_{\mathfrak{d}}) = p^*_{z_0}(A_{\mathfrak{d}})$ for all $z_0 \in A_{\mathfrak{d}}$.

The next theorem is the discrete version of Theorem 5.1, which formulates the maximal invariance probability for the MDP $\mathfrak{D}$ through value functions D_k, D^*_k.

Theorem 5.3 *Consider value functions $D_k : \mathcal{S}_{\mathfrak{d}} \times \mathcal{U}_{\mathfrak{d}} \to [0,1]$, $D^*_k : \mathcal{S}_{\mathfrak{d}} \to [0,1]$, $k \in \mathbb{Z}_{N+1}$, computed by the backward recursion:*

$$D_k(z, v) = \boldsymbol{1}_{A_{\mathfrak{d}}}(z) \sum_{\bar{z} \in \mathcal{S}_{\mathfrak{d}}} D^*_{k+1}(\bar{z}) T_{\mathfrak{d}}(z, v, \bar{z}) \quad \forall z \in \mathcal{S}_{\mathfrak{d}}, v \in \mathcal{U}_{\mathfrak{d}},$$
$$D^*_k(z) = \max_{v \in \mathcal{U}_{\mathfrak{d}}} D_k(z, v), \quad \forall z \in \mathcal{S}_{\mathfrak{d}},$$

and initialized with:

$$D^*_N(z) = \boldsymbol{1}_{A_{\mathfrak{d}}}(z) = \begin{cases} 1, & \text{if } z \in A_{\mathfrak{d}}, \\ 0, & \text{if } z = \triangledown. \end{cases}$$

Then $p^*_{z_0}(A_{\mathfrak{d}}) = D^*_0(z_0)$.

Since the state and input spaces of MDP $\mathfrak{D}$ are finite, the maximally safe Markov policy always exists. It is of interest to provide a quantitative comparison between

the discrete outcome obtained by Theorem 5.3 and the continuous solution that results from Theorem 5.1. For this purpose we relate the discrete value functions D_k, D_k^* to the continuous value functions W_k, W_k^* of $\mathfrak{P}_c$. Lemma 5.2 accomplishes this goal which is subsequently used in Theorem 5.4 to compute the error of abstraction, presented in Algorithm 8, for maximal invariance probability.

Lemma 5.2 *The value functions associated to $\mathfrak{P}_c$ and $\mathfrak{D}$ are related together with*

$$W_k(s,u) = \sum_{i=1}^{m_{\mathfrak{s}}} \sum_{j=1}^{m_{\mathfrak{u}}} D_k(z_i, v_j) \mathbf{1}_{U_j}(u) \mathbf{1}_{A_i}(s) \quad s \in \mathcal{S}, u \in \mathcal{U},$$

$$W_k^*(s) = \sum_{i=1}^{m_{\mathfrak{s}}} D_k^*(z_i) \mathbf{1}_{A_i}(s) \quad s \in \mathcal{S}.$$

In other words, the value functions W_k,W_k^* of $\mathfrak{P}_c$ are piecewise constant functions and the constants in each domain of continuity is specified by the related values of D_k and D_k^* for the MDP $\mathfrak{D}$.

Theorem 5.4 *Under Assumption 5.1 the maximal invariance probability $p_{s_0}^*(\mathcal{A})$ for the model $\mathfrak{S}_c$ initialized at $s_0 \in \mathcal{A}$ satisfies:*

$$|p_{s_0}^*(\mathcal{A}) - p_{z_0}^*(A_{\mathfrak{d}})| \leq N(h_{\mathfrak{s}}\delta_{\mathfrak{s}} + h_{\mathfrak{u}}\delta_{\mathfrak{u}})\mathscr{L}(\mathcal{A}), \tag{5.2}$$

where $p_{z_0}^*(A_{\mathfrak{d}})$ is the maximal invariance probability for the MDP $\mathfrak{D}$ obtained by Algorithm 8, and initialized at the discrete state $z_0 = \xi(s_0) \in A_{\mathfrak{d}}$. The constants $h_{\mathfrak{s}}, h_{\mathfrak{u}}$ are defined in Assumption 5.1, where $\delta_{\mathfrak{s}}$ and $\delta_{\mathfrak{u}}$ are the largest diameters of the partition sets $\{A_i,\ i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$ and $\{U_j,\ i \in \mathbb{N}_{m_{\mathfrak{u}}}\}$, respectively.

Theorem 5.4 enables us to approximate the maximal invariance probability with any desired accuracy provided that the density function is Lipschitz continuous with respect to the conditioned state and input variables. Beyond the approximate computation of $p_{s_0}^*(\mathcal{A})$, we seek to find a Markov policy μ to be applied to the model $\mathfrak{S}_c$ and to guarantee such a (sub-optimal) invariance probability. The next section characterizes the set of policies that lead to the approximate maximal invariance probability value.

5.5 Construction of the ε–Maximally Safe Policies

In the previous section we have discussed how a controlled dtMP can be abstracted as an MDP, and have further provided an upper bound on the approximation error of the quantity $p_{s_0}^*(\mathcal{A})$. In this section we focus on the construction of a policy for the original dtMP $\mathfrak{S}_c$ that is ε–maximally safe. For this purpose we first construct the set of maximally safe policies for model $\mathfrak{P}_c$ based on the maximally safe policy of the MDP $\mathfrak{D}$.

Lemma 5.3 *The MDP $\mathfrak{D}$ has at least one maximally safe policy, indicated by $\boldsymbol{\vartheta}^* = (\vartheta_0^*, \vartheta_1^*, \ldots)$, where $\vartheta_k^* : \mathcal{S}_\mathfrak{d} \to \mathcal{U}_\mathfrak{d}$, such that $\vartheta_k^*(z) \in \arg\max_v \{D_k(z, v),\ v \in \mathcal{U}_\mathfrak{d}\}$. The dtMP $\mathfrak{P}_c$ has infinitely many maximally safe policies, represented by $\boldsymbol{\varrho}^* = (\varrho_0^*, \varrho_1^*, \ldots)$, where $\varrho_k^* : \mathcal{S} \to \mathcal{U}$, such that $\varrho_k^*(s) \in \arg\sup_u \{W_k(s, u),\ u \in \mathcal{U}\}$, and which are characterized as*

$$\varrho_k^*(s) = \sum_{i=1}^{m_\mathfrak{s}} f_{ki}(s) \mathbf{1}_{A_i}(s), \quad f_{ki} : A_i \to \Xi_\mathfrak{u}(\vartheta_k^*(z_i)), \tag{5.3}$$

where f_{ki} is any arbitrary function with domain A_i and range $\Xi_\mathfrak{u}(\vartheta_k^*(z_i))$ for $i \in \mathbb{N}_{m_\mathfrak{s}}$, $k \in \mathbb{Z}_N$. One of these maximally safe policies has the simple piecewise constant form

$$\varrho_k^*(s) = \sum_{i=1}^{m_\mathfrak{s}} \vartheta_k^*(z_i) \mathbf{1}_{A_i}(s).$$

Once the policy $\boldsymbol{\vartheta}^*$ is obtained for the MDP $\mathfrak{D}$, we use (5.3) to construct the policy $\boldsymbol{\varrho}^*$ for the model $\mathfrak{P}_c$ and apply this policy to $\mathfrak{S}_c$. The next theorem compares $\boldsymbol{\varrho}^*$ with the maximally safe policy of the model $\mathfrak{S}_c$.

Theorem 5.5 *Suppose $\boldsymbol{\varrho}^*$ is one of the maximally safe policies of $\mathfrak{P}_c$, characterized in Lemma 5.3. Under Assumption 5.1, the following inequality holds when we apply this policy to the original Model $\mathfrak{S}_c$:*

$$|p_{s_0}^*(\mathcal{A}) - p_{s_0}^{\boldsymbol{\varrho}^*}(\mathcal{A})| \leq 2N\mathscr{L}(\mathcal{A})(h_\mathfrak{s}\delta_\mathfrak{s} + h_\mathfrak{u}\delta_\mathfrak{u}) \quad \forall s_0 \in \mathcal{S}. \tag{5.4}$$

Proof: The solution of the invariance problem over the models $\mathfrak{S}_c, \mathfrak{P}_c$, for the fixed policy $\boldsymbol{\varrho}^* = (\varrho_0^*, \varrho_1^*, \varrho_2^*, \ldots)$, can be obtained by the recursions

$$V_k^{\boldsymbol{\varrho}^*}(s) = \mathbf{1}_\mathcal{A}(s) \int_\mathcal{S} V_{k+1}^{\boldsymbol{\varrho}^*}(\bar{s}) t_\mathfrak{s}(\bar{s}|s, \varrho_k^*(s)) d\bar{s} \quad \forall s \in \mathcal{S},$$
$$W_k^{\boldsymbol{\varrho}^*}(s) = \mathbf{1}_\mathcal{A}(s) \int_\mathcal{S} W_{k+1}^{\boldsymbol{\varrho}^*}(\bar{s}) t_\mathfrak{p}(\bar{s}|s, \varrho_k^*(s)) d\bar{s} \quad \forall s \in \mathcal{S},$$

initialized with $V_N^{\boldsymbol{\varrho}^*}(s) = W_N^{\boldsymbol{\varrho}^*}(s) = \mathbf{1}_\mathcal{A}(s), s \in \mathcal{S}$. Since the policy is given, the models $\mathfrak{S}_c, \mathfrak{P}_c$ can be thought of a time-inhomogeneous autonomous dtMPs with density functions $t_\mathfrak{s}(s_{k+1}|s_k, \varrho_k^*(s))$, $t_\mathfrak{p}(s_{k+1}|s_k, \varrho_k^*(s))$ respectively. It can be shown that under Assumption 5.1 we have, for all $s \in \mathcal{S}$, that

$$|V_k^{\boldsymbol{\varrho}^*}(s) - W_k^{\boldsymbol{\varrho}^*}(s)| \leq (N - k)\mathscr{L}(\mathcal{A})(h_\mathfrak{s}\delta_\mathfrak{s} + h_\mathfrak{u}\delta_\mathfrak{u}). \tag{5.5}$$

Using the triangular inequality we can write

$$|p_{s_0}^*(\mathcal{A}) - p_{s_0}^{\boldsymbol{\varrho}^*}(\mathcal{A})| \leq |p_{s_0}^*(\mathcal{A}) - p_{z_0}^*(\mathcal{A})| + |p_{z_0}^*(\mathcal{A}) - W_0^{\boldsymbol{\varrho}^*}(s_0)| + |W_0^{\boldsymbol{\varrho}^*}(s_0) - V_0^{\boldsymbol{\varrho}^*}(s_0)|.$$

The first term is upper bounded by $N\mathscr{L}(\mathcal{A})(h_\mathfrak{s}\delta_\mathfrak{s} + h_\mathfrak{u}\delta_\mathfrak{u})$ in the inequality (5.2). The third term is also upper bounded by the same quantity (replace $k = 0$ in the inequality (5.5)). The second term is equal to zero due to the specific construction

of policy $\boldsymbol{\varrho}^*$ (5.3) in Lemma 5.3. Summing up the three upper bounds results in the inequality (5.4). □

The error computed in (5.4) can be used to provide a uniform gridding approach for abstraction of the dtMP $\mathfrak{S}_c$ to an MDP $\mathfrak{D}$ which is presented in Algorithm 9. Moreover, the set of all ε–maximally safe policies can be computed using Algorithm 10.

Algorithm 9 Generation of the uniform grid for the abstraction

Require: model $\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$ under Assumption 5.1; threshold ε for the abstraction error

1: pick partition diameters $\delta_{\mathfrak{s}}$ and $\delta_{\mathfrak{u}}$ based on bound (5.4) such that

$$2N\mathscr{L}(\mathcal{A})(h_{\mathfrak{s}}\delta_{\mathfrak{s}} + h_{\mathfrak{u}}\delta_{\mathfrak{u}}) \leq \varepsilon$$

2: perform partitioning of $\mathcal{S}$ and $\mathcal{U}$ using uniformly-packed hypercubes with diameters $\delta_{\mathfrak{s}}$ and $\delta_{\mathfrak{u}}$

Ensure: $\mathfrak{D}$, error ε

Algorithm 10 Computation of ε–maximally safe policies

Require: model $\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$ under Assumption 5.1; threshold ε

1: Use Algorithm 9 to generate uniform grids for state and input spaces
2: Use Algorithm 8 to abstract $\mathfrak{S}_c$ to the MDP $\mathfrak{D}$
3: Use Theorem 5.3 to compute maximally safe policies of $\mathfrak{D}$, $\boldsymbol{\vartheta}^*$
4: Use Lemma 5.3 to construct all ε–maximally safe policies $\boldsymbol{\varrho}^*$ for $\mathfrak{S}_c$

Ensure: The set of all ε–maximally safe policies $\boldsymbol{\varrho}^*$

The characterization of all ε–maximally safe policies enables us to optimize a performance criterion, while maintaining a desired level of safety. This characterization is through the functions $f_{ki}(\cdot)$ in (5.3). For instance one could maximize at each time step, over the ε–maximally safe policies, the probability of reaching a target set $B \subset \mathcal{A}$ in the next time step:

$$\sup_{u_k} \left\{ \int_B t_{\mathfrak{s}}(s_{k+1}|s_k, u_k) ds_{k+1}, \;\; u_k \in \Xi_{\mathfrak{u}}\left(\vartheta_k^*(\xi(s_k))\right) \right\},$$

which means keeping a desired level of safety on the system trajectories while forcing the system to reach the target set B.

To complete our discussion on the class of controlled Markov processes we generalize our results to dtMPs with state-dependent input sets in the next section.

5.6 Extension to Models with State-Dependent Input Spaces

In the previous sections we assumed that the set of feasible inputs at each state is independent of the current state. In this section we generalize the previous results to the case $u \in \mathcal{U}(s) \subset \mathcal{U}$ when the process is at state $s \in \mathcal{S}$. Consider the controlled dtMP $\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, \{\mathcal{U}(s) | s \in \mathcal{S}\}, T_{\mathfrak{s}})$. Let us define the product of state and input spaces as

$$\mathcal{K} := \{(s, u) \in \mathcal{S} \times \mathcal{U} | u \in \mathcal{U}(s)\}$$

and assume that it is a measurable subset of $\mathcal{S} \times \mathcal{U}$. $T_{\mathfrak{s}}$ is a stochastic kernel on $\mathcal{S}$ given $\mathcal{K}$ and admits the conditional density function $t_{\mathfrak{s}}(\bar{s}|s, u)$.

Algorithm 11 is adapted from Algorithm 8 to include state-dependent input sets. In order to construct the MDP $\mathfrak{D}$ we again partition the safe set $\mathcal{A} = \cup_{i=1}^{m_{\mathfrak{a}}} A_i$ and select representative points $\{z_i \in A_i, i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$. The main difference is the selection of arbitrary finite partitions for input sets $\mathcal{U}(z_i)$ and representative points $v_{ij} \in \mathcal{U}(z_i)$. Note that the cardinality of the selected partition and representative points of partition sets is now state-dependent.

Algorithm 11 Abstraction of model $\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, \{\mathcal{U}(s) | s \in \mathcal{S}\}, T_{\mathfrak{s}})$ by MDP $\mathfrak{D}$

Require: input model $\mathfrak{S}_c$ with a state-dependent input space, set $\mathcal{A}$

1: Select a finite partition of set $\mathcal{A}$ as $\mathcal{A} = \cup_{i=1}^{m_{\mathfrak{a}}} A_i$, where $m_{\mathfrak{a}}$ represents the cardinality of the partition
2: For each A_i, select single representative point $z_i \in A_i$
3: For each $i \in \mathbb{N}_{m_{\mathfrak{a}}}$, select a finite partition of the input set $\mathcal{U}(z_i)$ as $\mathcal{U}(z_i) = \cup_{j=1}^{m_{\mathfrak{u}_i}} U_{ij}$ where $m_{\mathfrak{u}_i}$ represents the cardinality of the partition of $\mathcal{U}(z_i)$
4: For each U_{ij}, select single representative point $v_{ij} \in U_{ij}$
5: Define $A_{\mathfrak{d}} = \{z_i, i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$ and take $\mathcal{S}_{\mathfrak{d}} = A_{\mathfrak{d}} \cup \{\nabla_{\mathfrak{s}}\}$ as the finite state space of the MDP $\mathfrak{D}$ ($\nabla_{\mathfrak{s}}$ being the dummy variable of the state space)
6: Define $\mathcal{U}_{\mathfrak{d}} = \{v_{ij}, j \in \mathbb{N}_{m_{\mathfrak{u}_i}}, i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$ as the finite input space of the MDP $\mathfrak{D}$, $\mathcal{U}_{\mathfrak{d}}(z_i) = \{v_{ij}, j \in \mathbb{N}_{m_{\mathfrak{u}_i}}\}$ as the set of feasible inputs when $\mathfrak{D}$ is at state z_i, and $\mathcal{U}_{\mathfrak{d}}(\nabla_{\mathfrak{s}}) = \{\nabla_{\mathfrak{u}}\}$ ($\nabla_{\mathfrak{u}}$ being the dummy variable of the input space)
7: Compute the state transition matrix $T_{\mathfrak{d}}$ for $\mathfrak{D}$ as:

$$T_{\mathfrak{d}}(z, v, z') = \begin{cases} T_{\mathfrak{s}}(\Xi_{\mathfrak{s}}(z')|z, v), & z' \in A_{\mathfrak{d}}, z \in A_{\mathfrak{d}}, v \in \mathcal{U}_{\mathfrak{d}}(z) \\ 1 - \sum_{\bar{z} \in A_{\mathfrak{d}}} T_{\mathfrak{s}}(\Xi_{\mathfrak{s}}(\bar{z})|z, v), & z' = \nabla_{\mathfrak{s}}, z \in A_{\mathfrak{d}}, v \in \mathcal{U}_{\mathfrak{d}}(z) \\ 1, & z' = z = \nabla_{\mathfrak{s}}, v = \nabla_{\mathfrak{u}} \\ 0, & z' \in A_{\mathfrak{d}}, z = \nabla_{\mathfrak{s}}, v = \nabla_{\mathfrak{u}}, \end{cases}$$

Ensure: output MDP $\mathfrak{D} = (\mathcal{S}_{\mathfrak{d}}, \mathcal{U}_{\mathfrak{d}}, \{\mathcal{U}_{\mathfrak{d}}(z) | z \in \mathcal{S}_{\mathfrak{d}}\}, T_{\mathfrak{d}})$

The error quantification of the MDP abstraction approach, presented in Algorithm 11, requires to study the dependence of the input sets from the states. For this purpose, we assign the Hausdorff distance to the family of non-empty subsets of

$\mathcal{U}$, which is defined as

$$d_H(X,Y) = \max\left\{\sup_{x\in X}\inf_{y\in Y}\|x-y\|, \sup_{y\in Y}\inf_{x\in X}\|x-y\|\right\} \quad \forall X, Y \subset \mathcal{U}.$$

Assumption 5.2 poses a regularity condition on state-dependent input sets.

Assumption 5.2 *There exists a constant $\lambda_{\mathfrak{u}} \in \mathbb{R}$ such that the family of state-dependent input sets $\{\mathcal{U}(s)|s \in \mathcal{S}\}$ satisfies the Lipschitz inequality*

$$d_H(\mathcal{U}(s),\mathcal{U}(s')) \leq \lambda_{\mathfrak{u}}\|s-s'\| \quad \forall s, s' \in \mathcal{S}.$$

The maximal invariance probability $p^*_{s_0}(\mathcal{A})$ can be computed using the value functions $V^*_k : \mathcal{S} \to [0,1]$,

$$V^*_k(s) = \sup_{u\in\mathcal{U}(s)} \mathbf{1}_{\mathcal{A}}(s) \int_{\mathcal{S}} V^*_{k+1}(\bar{s}) T_{\mathfrak{s}}(d\bar{s}|s,u) \quad \forall s \in \mathcal{S}, \tag{5.6}$$

initialized with $V^*_N(s) = \mathbf{1}_{\mathcal{A}}(s)$ (notice the differences with the formulation of Theorem 5.1). The next lemma establish continuity of these value functions on Assumptions 5.1 and 5.2, which is essential for the error quantification.

Lemma 5.4 *Under Assumptions 5.1 and 5.2, the value functions $V^*_k(\cdot)$ of the recursion (5.6) are Lipschitz continuous, namely $|V^*_k(s) - V^*_k(s')| \leq \lambda\|s-s'\|$, for all $s, s' \in \mathcal{A}$, where $\lambda = (h_{\mathfrak{s}} + h_{\mathfrak{u}}\lambda_{\mathfrak{u}})\mathscr{L}(\mathcal{A})$. The constants $h_{\mathfrak{s}}, h_{\mathfrak{u}}$ were defined in Assumption 5.1 and $\lambda_{\mathfrak{u}}$ is according to Assumption 5.2.*

Lemma 5.4 enables us to quantify the error of MDP abstraction of Algorithm 11 on the maximal invariance probability $p^*_{s_0}(\mathcal{A})$ for the model $\mathfrak{S}_c$ as follows

$$|p^*_{s_0}(\mathcal{A}) - p^*_{z_0}(A_{\mathfrak{d}})| \leq N(h_{\mathfrak{s}}\delta_{\mathfrak{s}} + h_{\mathfrak{u}}\lambda_{\mathfrak{u}}\delta_{\mathfrak{s}} + h_{\mathfrak{u}}\delta_{\mathfrak{u}})\mathscr{L}(\mathcal{A}), \tag{5.7}$$

where $p^*_{z_0}(A_{\mathfrak{d}})$ is the maximal invariance probability for the MDP $\mathfrak{D}$ obtained by Algorithm 11, and initialized at the discrete state $z_0 = \xi(s_0) \in A_{\mathfrak{d}}$. The constants $\delta_{\mathfrak{s}}$ and $\delta_{\mathfrak{u}}$ are the largest diameters of the selected partitions $\{A_i,\, i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$ and $\{U_{ij},\, j \in \mathbb{N}_{m_{\mathfrak{u}_i}}, i \in \mathbb{N}_{m_{\mathfrak{a}}}\}$, respectively. Construction of the ε–maximally safe policy follows the same lines of Section 5.5 and is omitted for brevity.

Remark 5.2 *The inequality (5.7) generalized (5.2) by adding the term $h_{\mathfrak{u}}\lambda_{\mathfrak{u}}\delta_{\mathfrak{s}}$ to the error to account for approximation of the state-dependent input sets: for the case of state-independent input sets, Assumption 5.2 is automatically satisfied with $\lambda_{\mathfrak{u}} = 0$.*

For the case of finite input space, i.e. when $\mathcal{U}$ is finite, all the discussions of this chapter remain valid by putting $\delta_{\mathfrak{u}} = 0$: the input space does not need any discretization. This fact is used in the optimal temperature control problem of the next section.

5.7 Case Study: Optimal Temperature Control Problem

In this section we study the problem of optimal temperature control for thermal loads. The aggregation and control of a population of such loads, based on the formal techniques discussed in this thesis, are extensively discussed in the next chapter.

Evolution of the temperature in a cooling thermal load can be characterized by the following stochastic difference equation [20, 58]:

$$\theta(k+1) = a\,\theta(k) + (1-a)(\theta_a - m(k)RP_{rate}) + w(k), \tag{5.8}$$

where θ_a is the ambient temperature, P_{rate} is the rate of energy transfer to the load, C and R indicate the thermal capacitance and resistance respectively, and the parameter $a = e^{-h/RC}$ with a discretization time step h. The process noise $w(k), k \in \mathbb{N}_0$, is made up by i.i.d. random variables characterized by a density function $t_{\mathfrak{w}}(\cdot)$. We denote with $m(k) = 0$ and $m(k) = 1$ a thermal load in the OFF and ON mode at time step k.

The optimal temperature control for a thermal load is defined as follows: find the state feedback laws $m(k+1) = \varrho_k(m(k), \theta(k)), k \in \mathbb{Z}_N$ that maximize the probability that the temperature trajectory remains within a given interval $[\theta_{min}, \theta_{max}]$ over the time horizon $\mathbb{Z}_{N+1}$. The optimal feedback laws can be non-linear and time variant in general.

To solve this problem we construct a controlled SHS model with the hybrid dynamical equations

$$\begin{aligned}\theta(k+1) &= a\,\theta(k) + (1-a)(\theta_a - m(k)RP_{rate}) + w(k),\\ m(k+1) &= u(k).\end{aligned}$$

The hybrid state space is characterized by a variable $s = (m, \theta) \in \mathcal{S} = \{0,1\} \times \mathbb{R}$ with two components, a discrete (m) and a continuous (θ) one. The input space of the process is $\mathcal{U} = \{0,1\}$. The one-step transition density function of this stochastic process, $t_{\mathfrak{s}}(\cdot|s,u)$, can be computed as

$$t_{\mathfrak{s}}\left((\bar{m},\bar{\theta})|(m,\theta),u\right) = \boldsymbol{\delta}_{\mathfrak{d}}[\bar{m}-u]\,t_{\mathfrak{w}}(\bar{\theta} - a\,\theta - (1-a)(\theta_a - mRP_{rate})), \tag{5.9}$$

where $\boldsymbol{\delta}_{\mathfrak{d}}[\cdot]$ denotes the discrete unit impulse function.

Then the optimal temperature control problem is equivalent to finding a maximally safe policy $\boldsymbol{\varrho}^* = (\varrho_0^*, \varrho_1^*, \ldots, \varrho_{N-1}^*)$ over this SHS model with the safe set $\mathcal{A} = \{0,1\} \times [\theta_{min}, \theta_{max}]$ and the time horizon N. We employ the abstraction approach of this chapter to solve the problem. The process noise is assumed to be Gaussian $w(\cdot) \sim \mathcal{N}(0, \sigma^2)$, thus the density function (5.9) satisfies Assumption 5.1 with the constants $h_{\mathfrak{s}} = \frac{ae^{-1/2}}{\sigma^2\sqrt{2\pi}}, h_{\mathfrak{u}} = 0$. Numerical values of Table 5.1 are used for the dynamical equation (5.8). The temperature interval $[\theta_{min}, \theta_{max}] = [19.75, 20.25]$ is considered to reflect the user preference on the desired tempera-

Parameter	Interpretation	Value
θ_a	ambient temperature	$32\,[^\circ C]$
R	thermal resistance	$2\,[^\circ C/kW]$
C	thermal capacitance	$10\,[kWh/^\circ C]$
P_{rate}	power	$14\,[kW]$
h	time step	$10\,[sec]$

Table 5.1: Nominal values of parameters for a residential air conditioner [20].

ture. The time horizon $N = 20$ and the standard deviation $\sigma = 0.01\sqrt{h}$ are also selected.

Algorithm 9 is used to generate a uniform grid for partitioning the state space $\mathcal{S}$. The input space $\mathcal{U}$ is finite and does not need partitioning ($\delta_{\mathfrak{u}} = 0$). The error of $\varepsilon = 0.1$ on the maximally safe policies requires the partition diameter $\delta_{\mathfrak{s}} = 10^{-4}$. Given the partition, the SHS model is abstracted to an MDP $\mathfrak{D}$ using Algorithm 8. Theorem 5.3 is employed to compute the maximal invariance probability $p^*_{z_0}(A_{\mathfrak{d}})$ over $\mathfrak{D}$ which is an approximation of $p^*_{s_0}(\mathcal{A})$ with the guaranteed error 0.05. Figure 5.1 presents the maximal invariance probability $p^*_{s_0}(\mathcal{A})$ over the SHS model as a function of the initial state $s_0 = (m_0, \theta_0)$.

The set of maximally safe policies of $\mathfrak{D}$ is used according to Algorithm 10 to find an ε–maximally safe policy for the SHS model. Numerical implementation indicates that the policy $\boldsymbol{\varrho}^* = (\varrho^*_0, \varrho^*_1, \ldots)$ comprises the switching strategies $\varrho^*_k(m, \theta)$ of the form

$$\varrho^*_k(m(k), \theta(k)) = \begin{cases} 0 & \theta(k) < \theta_-(k) \\ 1 & \theta(k) > \theta_+(k) \\ 1 - m(k) & \text{else,} \end{cases} \tag{5.10}$$

is ε–maximally safe for the SHS model. The quantities $\theta_-(k), \theta_+(k)$ in (5.10) are constant which depend on time and parameters of the SHS model. The function ϱ^*_0 is presented in Figure 5.2.

5.8 Conclusions

In this chapter we have presented an abstraction procedure based on a partitioning of the state as well as of the input spaces. The focus of this chapter has been on the study of the maximal invariance probability over a finite horizon for controlled discrete-time Markov processes. The presented approach is a generalization of the results of Chapter 2 to controlled models. While only the uniform gridding procedure is discussed in this chapter, the study enables the adaptive gridding procedure using local continuity properties of the model, to mitigate the curse of dimensionality unavoidably related to the partitioning procedure.

We have applied the approach to the problem of optimal temperature control of a thermal load. The obtained optimal policy is quite similar to the switching control strategy that is widely used empirically for temperature regulation of thermal

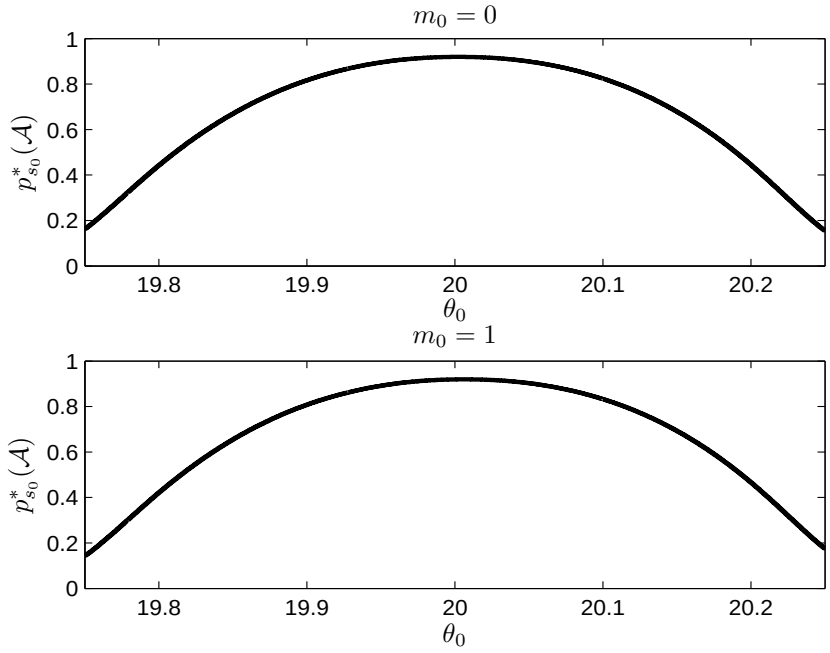


Figure 5.1: Maximal invariance probability $p^*_{s_0}(\mathcal{A})$, when the process is initialized at $s(0) = s_0 = (m_0, \theta_0)$, approximated with $p^*_{z_0}(A_\mathfrak{d})$ such that $z_0 = \xi(s_0)$. The initial mode is OFF in the top panel and is ON in the bottom panel.

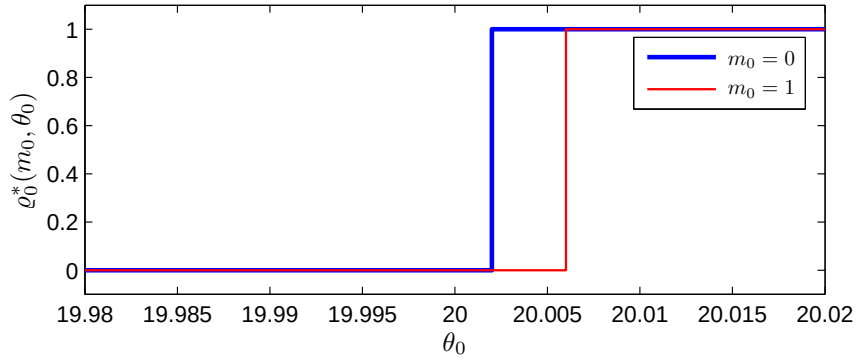


Figure 5.2: The ε–maximally safe policy function $\varrho^*_0(m_0, \theta_0)$ which is a switching strategy.

loads, which are also known as *Thermostatically Controlled Loads (TCLs)*. Chapter 6 discusses modeling and control of a population of TCLs using our formal abstraction methods. The abstraction approach of this chapter has been implemented as a part of the software tool FAUST^2, which is presented in Chapter 7.

6 CHAPTER

Aggregation of Thermostatically Controlled Loads by Formal Abstractions

This chapter discusses a two-step procedure, based on the use of formal abstractions, to generate a finite-space stochastic dynamical model as an aggregation of the continuous temperature dynamics of a homogeneous population of Thermostatically Controlled Loads (TCLs). The temperature of a TCL is described by a stochastic difference equation and the TCL status (ON, OFF) by a deterministic switching mechanism. The procedure is deemed to be formal, as it allows quantification of the error introduced by the abstraction. As such, it builds and improves on a known, earlier approximation technique used in the literature. Further, this chapter investigates the problem of global (population-level) power reference tracking and load balancing for TCLs that are explicitly dependent on a control input. The procedure is tested on a case study and benchmarked against the mentioned existing approach in the literature.

6.1 Introduction

Models for Thermostatically Controlled Loads (TCLs) have shown potential to be engaged in power system services such as load shifting, peak shaving, and demand response programs. The regulation of the total power consumption of large populations of TCLs, with the goal of smoothing the uncertain demand over the grid or tracking the uncertain power production, while abiding by strict requirements on users' comfort, can lead to economically relevant repercussions for an energy provider. The modeling of TCLs in view of their application to load control has a rich history, which can be traced back to the work [23], where the model of a TCL is used to describe the evolution of the thermostat state. A diffusion approximation framework is notably introduced in [58], and a discrete-time

simulation model is developed in [64]. Building on these foundations, recent studies have focused on the development of practically usable models for aggregated populations of TCLs. In particular, [20] provides an approximate analytical solution to the coupled Fokker–Planck equations originally developed in [58] for a population of homogeneous TCLs (meaning that TCLs are assumed to have the same dynamics and parameters), and puts forward a Linear Time-Invariant (LTI) population model, where the coefficients of its transfer function are estimated by means of system identification techniques. The contribution in [9] develops a bilinear PDE model and designs a Lyapunov stable controller. The work in [53] proposes a new technique, based on the partitioning of the TCL temperature range, to obtain an aggregate state-space model for a population of TCLs that is now heterogeneous over the thermal capacitances of the single TCLs. The full information of the state variables of the model is used to synthesize a control strategy over the model output (namely, the total power consumption), in order to attain tracking via a (deterministic) Model Predictive Control (MPC) scheme. The contributions in [60, 62] extend the results in [53] by considering a population of TCLs that are heterogeneous over all their parameters: the general setup requires the use of the extended Kalman filter to estimate the states of the model and to identify its characteristic transition matrix. The control of the population is performed by switching ON/OFF a portion of the TCL population. Additional recent contributions have targeted the application of the approaches in [53, 60, 62], towards higher-order dynamics [81, 82] and energy arbitrage [61].

Matrices and parameters of the state-space aggregate model can be computed analytically or via on system identification techniques [20, 9, 53, 60, 62]. The only available analytical derivation of the state-space aggregate model [53] works in discrete time and is based, under two rather restrictive assumptions, on the underlying model of the TCL: first, the TCL temperature evolution is assumed to be deterministic, thus leading to a deterministic state-space model; second, after partitioning the temperature range in separate intervals, the temperatures of the TCLs within each bin are assumed to be uniformly distributed in the population. Moreover, from a practical standpoint there seems to be no clear connection between the precision of the aggregation and the performance of the aggregated model: more specifically, an increase in the number of state bins (that is, a decrease in the width of the introduced temperature intervals) does not necessarily improve the performance of the aggregated model. The approach based on system identification on the other hand estimates the parameters of an LTI aggregate model based on data.

This chapter proposes a new, formal two-step abstraction procedure to generate a finite stochastic dynamical model as the aggregation of the dynamics of a population of TCLs. The approach relaxes the limiting assumptions employed in [53] by providing a model based on the natural probabilistic evolution of the single TCL temperature. The abstraction comprises two separate parts: (1) employing the results of Chapter 2 to translate a continuous-space model for a single TCL to a finite state-space model, which generates a Markov chain; and over a population of TCLs (2) taking the cross product of the single Markov chains and lumping the obtained model, by finding its coarsest probabilistically bisimilar Markov chain

[7]. The approach makes it possible to quantify the abstraction error of the first step, and furthermore in the homogeneous case the error of the overall abstraction procedure can be quantified – this is unlike the approach based on approximations in [53] and the approach based on system identification in [20, 60, 62].

This chapter also describes a dynamical model for the time evolution of the abstraction, and shows asymptotic results as the population size grows: increasing number of state bins always improves the accuracy, leading to a convergence of the introduced abstraction error to zero. This result is aligned with the work in [9, 10] on the aggregation of continuous-time deterministic thermostatic loads. The explicit relationship between aggregate model and population parameters enables the development of a set-point control strategy aimed at reference tracking over the population total power consumption (cf. Figure 6.2): a conditional Kalman filter [22] is employed to estimate the state of the model and the power consumption of the population is regulated via a simple one-step prediction approach. As such, the control architecture does not require knowledge of the states of all TCLs, but leverages directly the measurement of the total power consumption. Alternatively, a stochastic model predictive control scheme is proposed. Both procedures are tested on a case study and the abstraction technique is benchmarked against the analytical approach proposed in [53].

This chapter is organized as follows. Section 6.2, after introducing the model of the single TCL dynamics, describes its abstraction as a Markov Chain, and further discusses the aggregation of a homogeneous population of TCLs – the errors introduced by both steps are quantified. Section 6.3 discusses TCL models endowed with a control input, and the synthesis of global (acting at the population level – cf. Figure 6.2) controllers to achieve regulation of the total consumed power – this is achieved by two alternative schemes. Finally, all the discussed techniques are tested on a case study described in Section 6.4. Throughout this chapter we denote vectors with bold typeset and use the corresponding indexed letter with normal typeset for its elements.

6.2 Formal Abstraction of a Homogeneous Population of TCLs

6.2.1 Continuous Model of the Temperature of a TCL

The evolution of the temperature in a TCL can be characterized by the following stochastic difference equation [20, 58]:

$$\theta(k+1) = a\,\theta(k) + (1-a)(\theta_a \pm m(k)RP_{rate}) + w(k), \tag{6.1}$$

where θ_a is the ambient temperature, P_{rate} is the rate of energy transfer, C and R indicate the thermal capacitance and resistance respectively, and $a = e^{-h/RC}$ with a discretization step h. The process noise $w(k), k \in \mathbb{N}_0$, is made up by i.i.d. random variables characterized by a density function $t_{\mathfrak{w}}(\cdot)$. We denote with $m(k) = 0$ a

TCL in the OFF mode at time k, and with $m(k) = 1$ a TCL in the ON mode. In equation (6.1) the symbol $\pm$ signifies the following: a plus sign is used for a heating TCL, whereas a minus sign for a cooling TCL. We focus on a population of cooling TCLs, with the understanding that the case of heating TCLs can be treated similarly. The distributions of the initial temperature and mode are denoted by $\pi_0(m, \theta)$, respectively. The temperature dynamics for the cooling TCL is regulated by a control signal $m(k+1) = f(m(k), \theta(k))$ based on discrete switching as

$$f(m,\theta) = \begin{cases} 0, & \theta < \theta_s - \delta/2 \doteq \theta_- \\ 1, & \theta > \theta_s + \delta/2 \doteq \theta_+ \\ m, & \text{else,} \end{cases} \tag{6.2}$$

where θ_s denotes a temperature set-point and δ a dead-band, and together characterize an operating temperature range. The power consumption of the TCL at time k is equal to $\frac{1}{\eta} m(k) P_{rate}$, which is equal to zero in the OFF mode and positive in the ON mode, and where the parameter η is the coefficient of performance. The constant $\frac{1}{\eta} P_{rate}$, namely the power consumption of a TCL in the ON mode, will be shortened as P_{ON} in the rest of this chapter.

6.2.2 Finite Abstraction of a TCL by State-Space Partitioning

The composition of the dynamical equation in (6.1) with the algebraic relation in (6.2) allows one to consider a TCL as a SHS model (cf. Chapter 2 Section 2.4), namely as a dtMP evolving over a hybrid state space, which is characterized by a variable $s = (m, \theta) \in \{0, 1\} \times \mathbb{R}$ with two components, a discrete (m) and a continuous (θ) one. The one-step transition density function of the stochastic process, $t_{\mathfrak{s}}(\cdot|s)$, made up of the dynamical equations in (6.1), (6.2), and conditional on point s, can be computed as

$$t_{\mathfrak{s}}\left((m', \theta')|(m, \theta)\right) = \boldsymbol{\delta}_{\mathfrak{d}}[m' - f(m, \theta)] t_{\mathfrak{w}}(\theta' - a\,\theta - (1-a)(\theta_a - mRP_{rate})),$$

where $\boldsymbol{\delta}_{\mathfrak{d}}[\cdot]$ denotes the discrete unit impulse function. This interpretation allows leveraging the abstraction technique of Chapter 2 to reduce the model into a finite-state Markov chain. Consider an arbitrary, finite partition of the continuous domain $\mathbb{R} = \cup_{i=1}^{n} \Theta_i$, and arbitrary representative points within the partitioning regions denoted by $\{\bar{\theta}_i \in \Theta_i, i \in \mathbb{N}_n\}$. Introduce a finite-state Markov chain $\mathcal{M}$, characterized by $2n$ states $s_{im} = (m, \bar{\theta}_i), m \in \{0, 1\}, i \in \mathbb{N}_n$. Algorithm 1 indicates that the transition probability matrix related to $\mathcal{M}$ is made up of the following elements

$$\mathbb{P}(s_{im}, s_{i'm'}) = \int_{\Theta_{i'}} t_{\mathfrak{s}}\left((m', \theta')|m, \bar{\theta}_i\right) d\theta', \quad \forall m' \in \{0, 1\}, i' \in \mathbb{N}_n. \tag{6.3}$$

The initial probability mass for $\mathcal{M}$ is obtained as $p_0(s_{im}) = \int_{\Theta_i} \pi_0(m, \theta) d\theta$. For simplicity of notation we rename the states of $\mathcal{M}$ by the bijective map $\ell(s_{im}) =$

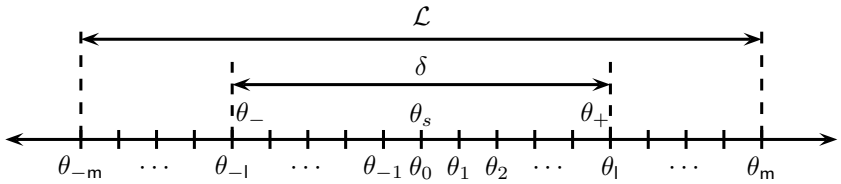


Figure 6.1: Partitioning of the temperature axis for the abstraction of the dynamics of a TCL.

$mn + i, m \in \{0, 1\}, i \in \mathbb{N}_n$, and accordingly we introduce the new notation

$$P_{ij} = \mathbb{P}(\ell^{-1}(i), \ell^{-1}(j)), \quad p_{0i} = p_0(\ell^{-1}(i)), \quad \forall i, j \in \mathbb{N}_{2n}.$$

Notice that the conditional density function of the stochastic system capturing the dynamics of a TCL is discontinuous, due to the presence of equation (6.2). This can be emphasized by the following alternative representation of the discontinuity in the discrete conditional distribution, for all $m, m' \in \{0, 1\}, \theta \in \mathbb{R}$:

$$\boldsymbol{\delta}_{\mathfrak{d}}\left[m' - f(m, \theta)\right] = m'\mathbf{1}_{(\theta_+, \infty)}(\theta) + (1 - m')\mathbf{1}_{(-\infty, \theta_-)}(\theta) + (1 - |m - m'|)\mathbf{1}_{[\theta_-, \theta_+]}(\theta),$$

where $\mathbf{1}_A(\cdot)$ denotes the indicator function of a general set A. The selection of the partitioning sets then requires special attention: a convenient way to obtain that is to select a partition for the dead-band $[\theta_-, \theta_+]$, thereafter extending it to a partition covering the whole real line $\mathbb{R}$ (cf. Figure 6.1). Let us select two constants $\mathsf{l}, \mathsf{m} \in \mathbb{N}$, $\mathsf{l} < \mathsf{m}$, compute the partition size $\upsilon = \delta/2\mathsf{l}$ and quantity $\mathcal{L} = 2\mathsf{m}\upsilon$. Now construct the boundary points of the partition sets $\{\theta_i\}_{i=-\mathsf{m}}^{i=\mathsf{m}}$ for the temperature axis as follows:

$$\begin{array}{l} \theta_{\pm\mathsf{l}} = \theta_s \pm \delta/2, \quad \theta_{\pm\mathsf{m}} = \theta_s \pm \mathcal{L}/2, \quad \theta_{i+1} = \theta_i + \upsilon, \quad i \in \mathbb{N}_{n-2}, \\ \Theta_1 = (-\infty, \theta_{-\mathsf{m}}), \quad \Theta_n = [\theta_{\mathsf{m}}, \infty), \quad \Theta_{i+1} = [\theta_{-\mathsf{m}+i-1}, \theta_{-\mathsf{m}+i}), \\ \mathbb{R} = \cup_{i=1}^{n} \Theta_i, \quad n = 2\mathsf{m} + 2, \end{array} \tag{6.4}$$

and let us render the Markov states of the infinite-length intervals Θ_1, Θ_n reflecting. Let us emphasize that the discontinuity in the discrete transition kernel $\boldsymbol{\delta}_{\mathfrak{d}}\left[m' - f(m, \theta)\right]$ and the above partition induce the following structure on the transition probability matrix of the chain $\mathcal{M}$:

$$P = [P_{ij}]_{i,j} = \begin{bmatrix} Q_{11} & 0 & Q_{31} & 0 \\ 0 & Q_{22} & 0 & Q_{42} \end{bmatrix}^T, \tag{6.5}$$

where $Q_{11}, Q_{42} \in \mathbb{R}^{n \times (\mathsf{m}+\mathsf{l}+1)}$, whereas $Q_{22}, Q_{31} \in \mathbb{R}^{n \times (\mathsf{m}-\mathsf{l}+1)}$, which leads to $P \in \mathbb{R}^{2n \times 2n}$.

Clearly, the abstraction of the dynamics in (6.1)-(6.2) over this partition of the state space leads to a discretization error: in Section 6.2.5 we formally derive bounds on this error as a function of the partition size υ and of the quantity $\mathcal{L}$, which guarantees the convergence (in expected value) of the power consumption of the abstracted model to that of the entire population.

6.2.3 Aggregation of a Population of TCLs by Bisimulation Relation

Consider now a population of n_p homogeneous TCLs, that is a population of TCLs which, after possible rescaling of (6.1)-(6.2), share the same set of parameters[1] $\theta_s, \delta, \theta_a, C, R, P_{rate}, P_{ON}$ (and thus η), h, and noise terms $t_{\mathfrak{w}}(\cdot)$. Each TCL can then be abstracted as a Markov chain $\mathcal{M}$ with the same transition probability matrix $P = [P_{ij}]_{i,j}$, where $i, j \in \mathbb{N}_{2n}$, which leads to a population of n_p homogeneous Markov chains. The initial probability mass vector $p_0 = [p_{0i}]_i$ might still vary over the TCL population.

The homogeneous population of TCLs can be represented by a single Markov chain Ξ, built as the cross product of the n_p homogeneous Markov chains, and endowed with the state

$$\mathbf{z} = [z_1, z_2, \ldots, z_{n_p}]^T \in \mathcal{Z} = \mathbb{N}_{2n}^{n_p},$$

where $z_j \in \mathbb{N}_{2n}$ represents the state of the j^{th} Markov chain. We denote by P_Ξ the transition probability matrix of Ξ.

It is understood that Ξ, having exactly $(2n)^{n_p}$ states, can in general be quite large, and thus cumbersome to manipulate computationally. As the second step of the abstraction procedure, we are interested in aggregating this model and we employ the notion of probabilistic bisimulation [7] to achieve this. Let us introduce a finite set of atomic propositions as a constrained vector with a dimension corresponding to the number of bins of the single TCL $\mathcal{M}$:

$$AP = \left\{ \mathbf{x} = [x_1, x_2 \ldots, x_{2n}]^T \in \mathbb{Z}_{n_p+1}^{2n} \middle| \sum_{r=1}^{2n} x_r = n_p \right\}.$$

The labeling function $L : \mathcal{Z} \to AP$ associates to a configuration $\mathbf{z}$ of Ξ a vector $\mathbf{x} = L(\mathbf{z})$, which elements $x_i \in \mathbb{Z}_{n_p+1}$ count the number of thermostats in bin $i, i \in \mathbb{N}_{2n}$. Notice that the set AP is finite with cardinality $|AP| = (n_p+2n-1)!/(n_p!(2n-1)!)$, which for $n_p \geq 2$ is (much) less than the cardinality $(2n)^{n_p}$ of Ξ.

Let us define an equivalence relation $\mathcal{R}$ [7] on the state space of $\mathcal{Z}$, such that

$$(\mathbf{z}, \mathbf{z}') \in \mathcal{R} \Leftrightarrow L(\mathbf{z}) = L(\mathbf{z}').$$

A pair of elements of $\mathcal{Z}$ (each of them a vector representing a state of Ξ) is in the relation whenever the corresponding number of TCLs in any of the introduced bins is the same (recall that the TCL population is assumed to be homogeneous). Such an equivalence relation provides a partition of the state space of $\mathcal{Z}$ into equivalence classes belonging to the quotient set $\mathcal{Z}/\mathcal{R}$, where each class is uniquely specified by the label associated to its elements. We plan to show that $\mathcal{R}$ is an exact probabilistic bisimulation relation on Ξ [7], which requires proving that, for

[1] This homogeneity assumption might seem to be restrictive which induces limitation on the practical application of the results of this chapter. The reader may refer to [33] in which we have extended the results to the heterogeneous populations of TCLs.

any set $\mathscr{T} \in \mathcal{Z}/\mathcal{R}$ and any pair $(\mathbf{z}, \mathbf{z}') \in \mathcal{R}$

$$\mathbb{P}_{\Xi}(\mathbf{z}, \mathscr{T}) = \mathbb{P}_{\Xi}(\mathbf{z}', \mathscr{T}), \tag{6.6}$$

This is achieved by Corollary 6.1 in the next Section. We now focus on the stochastic properties of Ξ, which we study via its quotient Markov chain obtained with $\mathcal{R}$.

6.2.4 Properties of the Aggregated Quotient Markov Chain

We study the one-step probability mass function associated to the codomain of the labeling function (that is, to the labels set), conditional on the state of the chain Ξ.

Lemma 6.1 *The conditional random variable $(x_i(k+1)|\mathbf{z}(k))$, $i \in \mathbb{N}_{2n}$, has a Poisson-binomial distribution over the sample space $\mathbb{Z}_{n_p+1}$, with the following mean and variance:*

$$\mathbb{E}[x_i(k+1)|\mathbf{z}(k)] = \sum_{r=1}^{n_p} P_{z_r(k)i}, \quad var(x_i(k+1)|\mathbf{z}(k)) = \sum_{r=1}^{n_p} P_{z_r(k)i}(1-P_{z_r(k)i}). \tag{6.7}$$

Conditional on an observation $\mathbf{x} = [x_1, x_2, \ldots, x_{2n}]^T$ at time k over the Markov chain Ξ, it is of interest to compute the probability mass function of the conditional random variable $(x_i(k+1)|\mathbf{x}(k))$ as $\mathbb{P}(x_i(k+1) = j|\mathbf{x}(k))$, for any $j \in \mathbb{Z}_{n_p+1}$ — notice the difference with the quantity discussed in (6.7) where the conditioning is over variable $\mathbf{z}(k)$. For any label $\mathbf{x} = [x_1, x_2, \ldots, x_{2n}]^T$ there are exactly $n_p!/(x_1!x_2!\ldots x_{2n}!)$ states of Ξ such that $L(\mathbf{z}) = \mathbf{x}$. We use the notation $\mathbf{z} \hookrightarrow \mathbf{x}$ to indicate the states in Ξ associated to label $\mathbf{x}$, that is $\mathbf{z} : L(\mathbf{z}) = \mathbf{x}$. Based on the law of total probability for conditional probabilities, we can write

$$\begin{aligned}
\mathbb{P}(x_i(k+1) = j|\mathbf{x}(k)) &= \frac{\sum_{\mathbf{z}(k)\hookrightarrow\mathbf{x}(k)} \mathbb{P}(x_i(k+1) = j|\mathbf{z}(k))\mathbb{P}(\mathbf{z}(k))}{\mathbb{P}(\mathbf{x}(k))} \\
&= \mathbb{P}(x_i(k+1) = j|\mathbf{z}(k)) \frac{\sum_{\mathbf{z}(k)\hookrightarrow\mathbf{x}(k)} \mathbb{P}(\mathbf{z}(k))}{\mathbb{P}(\mathbf{x}(k))} \\
&= \mathbb{P}(x_i(k+1) = j|\mathbf{z}(k)),
\end{aligned} \tag{6.8}$$

where the sum is over all states $\mathbf{z}(k)$ of Ξ such that $L(\mathbf{z}(k)) = \mathbf{x}(k)$: in these states we have $x_1(k)$ Markov chains in state 1 with probability P_{1i}, $x_2(k)$ Markov chains in state 2 with probability P_{2i}, and so on. The simplification above is legitimate since the probability of having a label $\mathbf{x} = [x_1, x_2, \ldots, x_{2n}]^T$ is exactly the sum of the probabilities associated to the states $\mathbf{z}$ generating such a label. This further allows expressing the quantities in (6.7) as

$$\mathbb{E}[x_i(k+1)|\mathbf{z}(k)] = \sum_{r=1}^{n_p} P_{z_r(k)i} = \sum_{r=1}^{2n} x_r(k) P_{ri}.$$

The generalization of the previous results to vector labels leads to the following statement.

Theorem 6.1 *The conditional random variables $(x_i(k+1)|\mathbf{x}(k))$ are characterized by Poisson-binomial distributions, whereas the conditional random vector $(\mathbf{x}(k+1)|\mathbf{x}(k))$ by a generalized multinomial distribution. Mean, variance, and covariance are described, $\forall i, j \in \mathbb{N}_{2n}, i \neq j$, by*

$$\mathbb{E}[x_i(k+1)|\mathbf{x}(k)] = \sum_{r=1}^{2n} x_r(k) P_{ri},$$

$$var(x_i(k+1)|\mathbf{x}(k)) = \sum_{r=1}^{2n} x_r(k) P_{ri}(1 - P_{ri}),$$

$$cov(x_i(k+1), x_j(k+1)|\mathbf{x}(k)) = -\sum_{r=1}^{2n} x_r(k) P_{ri} P_{rj}.$$

Theorem 6.1 indicates that the distribution of the conditional random variable $(\mathbf{x}(k+1)|\mathbf{x}(k))$ is independent of the underlying state $\mathbf{z}(k) \hookrightarrow \mathbf{x}(k)$ of Ξ. With focus on equation (6.6), this result allows us to claim the following [7].

Corollary 6.1 *The equivalence relation $\mathcal{R}$ is an exact probabilistic bisimulation over the Markov chain Ξ. The resulting quotient Markov chain is the coarsest probabilistic bisimulation of Ξ.*

Without loss of generality, let us normalize the values of the labels $\mathbf{x}$ by the total population size n_p, thus obtaining a new variable $\mathbf{X}$. The conditional variable $(\mathbf{X}(k+1)|\mathbf{X}(k))$ is characterized by the following parameters:

$$\begin{aligned}
&\mathbb{E}[X_i(k+1)|\mathbf{X}(k)] = \sum_{r=1}^{2n} X_r(k) P_{ri}, \\
&var(X_i(k+1)|\mathbf{X}(k)) = \frac{1}{n_p} \sum_{r=1}^{2n} X_r(k) P_{ri}(1 - P_{ri}), \\
&cov(X_i(k+1), X_j(k+1)|\mathbf{X}(k)) = -\frac{1}{n_p} \sum_{r=1}^{2n} X_r(k) P_{ri} P_{rj},
\end{aligned} \tag{6.9}$$

for all $i, j \in \mathbb{N}_{2n}, i \neq j$. Based on the expression of the first two moments of $(\mathbf{X}(k+1)|\mathbf{X}(k))$, we apply a translation (shift) on this conditional random vector as $\mathbf{W}(k) = \mathbf{X}(k+1) - \mathbb{E}[\mathbf{X}(k+1)|\mathbf{X}(k)]$, where $\mathbf{W}(k) = [\omega_1(k), \ldots, \omega_{2n}(k)]^T$ and $\omega_i(k)$ are guaranteed to be (dependent) random variables with zero mean and covariance described by (6.9) and dependent on the state $\mathbf{X}(k)$. Such a translation allows expressing the following dynamical model for the variable $\mathbf{X}$:

$$\mathbf{X}(k+1) = P^T \mathbf{X}(k) + \mathbf{W}(k). \tag{6.10}$$

Remark 6.1 *We have modeled the evolution of the TCL population with the abstract aggregated model (6.10), characterized by a stochastic difference equation. The dynamics in (6.10) represent a direct generalization of the model abstraction provided in [53]. A closer look on the covariance terms in (6.9) is that they are bounded by the quantity $1/n_p$, and thus decrease to zero as n_p grows, independent of the number of bins n.*

We employ the Lyapunov central limit theorem [13] to prove the following asymptotic result.

Theorem 6.2 *The random variable $(X_i(k+1)|\mathbf{X}(k))$ can be explicitly expressed as*

$$X_i(k+1) = \sum_{r=1}^{2n} X_r(k) P_{ri} + \omega_i(k), \tag{6.11}$$

where the random vector $\mathbf{W}(k) = [\omega_1(k), \ldots, \omega_{2n}(k)]^T$ has a covariance matrix $\Sigma(\mathbf{X}(k))$ as in (6.9), and converges (in distribution) to a multivariate Gaussian random vector $\mathcal{N}(0, \Sigma(\mathbf{X}(k)))$, as $n_p \uparrow \infty$.

Theorem 6.2 practically states that the conditional distribution of the random vector $\mathbf{W}(k)$ for a relatively large population size can be effectively replaced by a multivariate Gaussian distribution with known moments. We shall exploit this result in the state estimation step using Kalman filter, as in Section 6.3. Notice that the above conclusion can be applied to any population of homogeneous TCLs, which are characterized by Markov chains that have the same transition probability matrix. The initial distributions of the single Markov chain can instead vary.

In the previous theorem we have developed a model for the evolution of $X_i(k)$, which in the limit includes a Gaussian noise $\omega_i(k)$. As discussed in (6.9), these Gaussian random variables are dependent. The covariance matrix in (6.9) is guaranteed to be positive semi-definite for all $X_r \in \{0, \frac{1}{n_p}, \frac{2}{n_p}, \ldots, \frac{n_p-1}{n_p}, 1\}$, provided that $\sum_{r=1}^{2n} X_r = 1$. In order to enable a more general use in (6.10), we next show that the covariance matrix remains positive semi-definite when the model is extended over the variables $X_r \in [0, 1]$.

Proposition 6.1 *The covariance matrix $\Sigma(\mathbf{X})$ is positive semi-definite for all $X_r \geq 0$. The entries of the random vector $\mathbf{W}$ are dependent on each other, since $\sum_{r=1}^{2n} \omega_r = 0$ whenever $\sum_{r=1}^{2n} X_r = 1$. Condition $\sum_{r=1}^{2n} X_r(0) = 1$ implies that $\sum_{r=1}^{2n} X_r(k) = 1$, for all $k \in \mathbb{N}$.*

6.2.5 Explicit Quantification of the Errors of the Abstraction and of the Aggregation Procedures

Let us now quantify the power consumption of the aggregate model, as an extension of the quantity discussed after equation (6.2). The total power consumption

obtained from the aggregation of the original models in (6.1)-(6.2), with variables $(m_j, \theta_j)(k), j \in \mathbb{N}_{n_p}$, is

$$y(k) = \sum_{j=1}^{n_p} m_j(k) P_{ON}. \tag{6.12}$$

With focus on the abstract model (with the normalized variable $\mathbf{X}$), the power consumption is

$$y_{abs}(k) = H\mathbf{X}(k), \quad H = n_p P_{ON}[0_n, \mathbb{1}_n], \tag{6.13}$$

where 0_n and $\mathbb{1}_n$ are row vectors with all the entries equal to zero and one, respectively. For the quantification of the error we consider a homogeneous population of TCLs with dynamics affected by Gaussian process noise $w(\cdot) \sim \mathcal{N}(0, \sigma^2)$, and the abstracted model constructed based on the partition introduced in (6.4). The main result of this section hinges on two features of the Gaussian distribution, its continuity and its decay at infinity. In order to keep the discussion focused we proceed considering Gaussian distributions, however the results can be extended to any distribution with these two features. Since the covariance matrix in (6.9) is small for large population sizes, the first moment of the random variable $y(k)$ provides sufficient information on its behavior over a finite time horizon. The total power consumption in (6.12) is the sum of n_p independent Bernoulli trials over the sample space $\{0, P_{ON}\}$, each with different success probability. Then for the quantification of the modeling error we study the error produced by the abstraction over the expected value of the binary TCL mode (ON, OFF).

Consider a single TCL, with initial state $s_0 = (m_0, \theta_0)$. Also select a desired final time T_d and discrete time horizon $N = T_d/h$, where h is the discretization step. Based on the evolution equation of the discrete mode (6.2), the TCL is in the ON mode at time step N, namely $m(N) = 1$, if and only if $m(N-1) \in \mathcal{A}$, where $\mathcal{A} = \{1\} \times [\theta_-, +\infty) \cup \{0\} \times [\theta_+, +\infty)$. Then the expected value of its mode at time N, $m(N)$, can be computed as

$$\mathbb{E}[m(N)|m_0, \theta_0] = \mathbb{P}\left(m(N) = 1 | m_0, \theta_0\right) = \mathbb{P}\left(s(N-1) \in \mathcal{A} | m_0, \theta_0\right). \tag{6.14}$$

This quantity can be characterized via value functions $\mathcal{V}_k : \mathcal{S} \to [0, 1]$, $k \in \mathbb{N}_N$, which are computed recursively as follows:

$$\mathcal{V}_k(s_k) = \int_{\mathcal{S}} \mathcal{V}_{k+1}(s_{k+1}) t_{\mathfrak{s}}(s_{k+1}|s_k) ds_{k+1}, \quad \forall k \in \mathbb{N}_{N-1}, \quad \mathcal{V}_N(s) = \mathbf{1}_{\mathcal{A}}(s). \tag{6.15}$$

Knowing these value functions, we have that $\mathbb{E}[m(N)|m_0, \theta_0] = \mathcal{V}_1(m_0, \theta_0)$. Computationally, the calculation of these quantities can leverage the results of Chapter 2, which however require extensions 1) to conditional density functions of the process that are discontinuous, and 2) to an unbounded state space. The first issue is addressed by the following lemma.

Lemma 6.2 *The density function $t_{\mathfrak{s}}(s'|\cdot)$ is piecewise-continuous within the continuity regions*

$$\{0\} \times (-\infty, \theta_+], \quad \{0\} \times (\theta_+, +\infty), \quad \{1\} \times (-\infty, \theta_-), \quad \{1\} \times [\theta_-, +\infty).$$

The value functions $\mathcal{V}_k(s)$ are piecewise-Lipschitz continuous, namely:

$$|\mathcal{V}_k(m,\theta) - \mathcal{V}_k(m,\theta')| \leq \frac{2a}{\sigma\sqrt{2\pi}}|\theta - \theta'|,$$

where a, σ represent respectively the TCL parameter and the variance of the process noise, and where $(m,\theta),(m,\theta')$ is any pair of points belonging to one of the four continuity regions of the density $t_{\mathfrak{s}}$.

In order to cope with the second issue, we study the limiting behavior of the value functions at infinity, apply a truncation over the state space, as proposed in Section 6.2.2, and properly select the value of the functions outside of this region. Lemma 6.3 shows that $\lim_{\theta\to-\infty}\mathcal{V}_k(m,\theta) = 0$, $\lim_{\theta\to+\infty}\mathcal{V}_k(m,\theta) = 1$, and provides an upper bound on the distance between $\mathcal{V}_k(m,\theta)$ and its limiting values, which hinges on the parameter $\mathcal{L}$. This parameter represents the length of the truncated part of the temperature range $[\theta_{-\mathsf{m}},\theta_{\mathsf{m}}]$, which is further partitioned to construct the abstract Markov chain. An upper bound on the error of the value functions produced by state-space truncation and partitioning is further quantified in Theorem 6.3.

Lemma 6.3 *For the partitioning procedure in (6.4) we have that*

$$\begin{aligned}&\mathcal{V}_k(m,\theta) \leq (N-k)\epsilon \quad \forall\theta \leq \theta_{-\mathsf{m}} = \theta_s - \mathcal{L}/2 \quad m\in\{0,1\},\\ &\mathcal{V}_k(m,\theta) \geq 1-(N-k)\epsilon \quad \forall\theta \geq \theta_{\mathsf{m}} = \theta_s + \mathcal{L}/2 \quad m\in\{0,1\},\end{aligned} \tag{6.16}$$

where $\epsilon = \dfrac{e^{-\gamma^2/2}}{\gamma\sqrt{2\pi}}$, *and where*

$$\gamma = \frac{1-a}{2\sigma}\left[\frac{a^N\mathcal{L}+\delta}{1-a^N} - \lambda\right], \quad \lambda = RP_{rate} + |2(\theta_s-\theta_a) + RP_{rate}|. \tag{6.17}$$

Notice in particular the linear dependence of γ on $\mathcal{L}$, the temperature interval of interest.

Theorem 6.3 *If we abstract a TCL as a Markov chain based on the procedure of Section 6.2.2, compute the solution of problem (6.15) over the Markov chain, and construct a piecewise-constant approximation function $\mathcal{W}_1(m,\theta)$ using the solution of (6.15) over the Markov chain, then the approximation error can be upper bounded as follows:*

$$|\mathcal{V}_1(m,\theta) - \mathcal{W}_1(m,\theta)| \leq (N-1)\left[\frac{N-2}{2}\epsilon + \frac{2a}{\sigma\sqrt{2\pi}}\upsilon\right] \quad \forall(m,\theta)\in\{0,1\}\times[\theta_{-\mathsf{m}},\theta_{\mathsf{m}}].$$

Notice that the error has two terms: one term accounts for the error of the approximation over infinite-length intervals ϵ, whereas the second is related to the choice of the partition size υ.

Collecting the results above, the following theorem quantifies an upper bound on the abstraction error over the total power consumption.

Corollary 6.2 *The difference in the expected value of the total power consumption of the population $y(N)$, and that of the abstracted model $y_{abs}(N)$, both conditional on the corresponding initial conditions, is upper bounded by*

$$\left|\mathbb{E}[y(N)|\mathbf{s}_0] - \mathbb{E}[y_{abs}(N)|\mathbf{X}_0]\right| \leq n_p P_{ON}(N-1)\left[\frac{(N-2)}{2}\epsilon + \frac{2a}{\sigma\sqrt{2\pi}}\upsilon\right], \quad (6.18)$$

for all $\mathbf{s}_0 \in (\{0,1\} \times [\theta_{-\mathsf{m}}, \theta_{\mathsf{m}}])^{n_p}$. The initial state $\mathbf{X}_0$ is obtained as a function of the initial states in the TCL population $\mathbf{s}_0$, as can be evinced from the definition of the state vector $\mathbf{X}$.

The result in Corollary 6.2 allows tuning the error over the total power consumption of the population made with the abstraction procedure. In practice, it can be reduced to a desired level by increasing the abstraction precision: this can be achieved by increasing γ and the number of state bins (by decreasing their size). This results in a larger-dimensional model in (6.10). To address this issue, model-order reduction techniques like *balanced realization and truncation* or *Hankel singular values* [5] can be employed to obtain a low dimensional model describing the dynamics of the population power consumption. These known techniques follows the observation that the dynamics of the linear model are mostly determined by the largest eigenvalues of the transition probability matrix.

6.3 Abstraction and Control of a Population of Non-Autonomous TCLs

One can imagine a number of different strategies for controlling the total power consumption of a population of TCLs. With focus on the dynamics of a single TCL, one strategy could be to vary the rate of the energy transfer P_{rate}, for instance by circulating cold/hot water through the load with higher or lower speed. Another approach could be to act on the thermal resistance R, for instance opening or closing doors and windows at the load end. Yet another strategy could be to apply changes to the set-point θ_s, as suggested in [20].

Let us observe that the first two actions would modify the dynamics of (6.1), whereas the third control action would affect the relation in (6.2). While abstracting the TCL model as a finite-state Markov chain, a control action results in a modification of the elements of the transition probability matrix. With reference to (6.5), the entries of the matrices $Q_{11}, Q_{22}, Q_{31}, Q_{42}$ are computed based on (6.1), while the size of these matrices is determined based on (6.2). Since the set-point θ_s affects only equation (6.2), a set-point alteration affects the *structure* of the probability matrix in (6.5), whereas other approaches affect the *value* of its non-zero elements. It follows that in view of the abstraction procedure the control by set-point variation has the advantage of requiring a single computation of marginals, while the other discussed methods would require this computation to be a function of the allowed control inputs.

Based on the discussion above, we consider case where the control input is taken to be the set-point θ_s of the TCL. We intend to apply the control input to all TCLs uniformly (cf. Figure 6.2), which does not require differentiating among the states of different TCLs. Moreover, in order to retain validity of the definition of state bins $\mathbf{X}(\cdot)$ regardless of the applied input signal, we discretize the domain of allowable set-points by the same parameter ν used for the partition size.

Considering closed-loop control schemes in the literature, [53] assumes full knowledge of the state vector $\mathbf{X}(k)$ and employs a Model Predictive Control architecture to design the control signal. Moving forward, [60, 62] consider different scenarios for the configuration of the control architecture: states are measured completely, or known partially and a Kalman filter is used for state estimation, or states and transition matrix are estimated by use of an Extended Kalman filter. The minimum required infrastructure for the practical implementation of the strategies in [60, 62] ranges from a TCL temperature sensor and a two-way data connection for transmitting the state information and control signal, to a one-way data connection for sending the specific control signal to the single TCLs. The presence of a local decision maker is essential in all the scenarios: each TCL receives a control signal at each time step, determines its current state, and generates a local control action. In contrast, the set-point control strategy in this chapter does not require single TCL to know its individual state, which makes the approach applicable regardless of the thermometer precision [20].

In the following we attempt to mitigate the above limitations by showing that the knowledge of the actual values of the TCL states or of vector $\mathbf{X}(k)$ in the aggregated model are not necessary. Given the model parameters, all that is needed is an online measurement of the total power consumption of the TCL population, which allows estimating the states in $\mathbf{X}(k)$ and using the set-point θ_s to track a given reference signal. The control action comprises a simple signal for the set-point, which is applied to all TCLs uniformly: no local decision maker is then required.

6.3.1 State Estimation and One-Step Regulation

Suppose we consider a homogeneous population of TCLs with known parameters. As discussed earlier, we assume that the control input is discrete and takes values over a finite set, $\theta_s(k) \in \{\theta_{-\mathsf{l}}, \theta_{-\mathsf{l}+1}, \ldots, \theta_{\mathsf{l}-1}, \theta_{\mathsf{l}}\}, \forall k \in \mathbb{N}_0$: the parameter l is arbitrary and has been chosen to match the abstraction parameter in Figure 6.1 and the scheme in (6.4). Based on (6.10), we set up the following discrete-time switched stochastic system:

$$\mathbf{X}(k+1) = F_{\sigma(k)}\mathbf{X}(k) + \mathbf{W}(k),$$

where by switched model we mean that the state matrix $F_{\sigma(k)}$ takes values in

$$\{P^T(\theta_{-\mathsf{l}}), P^T(\theta_{-\mathsf{l}+1}), \ldots, P^T(\theta_{\mathsf{l}-1}), P^T(\theta_{\mathsf{l}})\},$$

for all $k \in \mathbb{N}_0$, (cf. (6.10)), and the switching signal $\sigma(\cdot) : \mathbb{N}_0 \to \mathbb{Z}_{2l+1}$ is a map specifying the set-point θ_s, and hence the TCL dynamics, as a function of time. The process noise $\mathbf{W}(k)$ is normal with zero mean and the state-dependent covariance matrix $\Sigma(\mathbf{X}(k))$ in (6.9). The total power consumption of the TCL population is measured as $y_{meas}(k) = H\mathbf{X}(k) + v(k)$, where $v(k) \sim \mathcal{N}(0, R_v)$ is a measurement noise characterized by $\sqrt{R_v}$, the standard deviation of the real-time measurement in the power meter instrument.

Since the process noise $\mathbf{W}$ is state-dependent, the state of the system can be estimated by the conditional Kalman filter [22] with the following time update:

$$\begin{aligned}\hat{\mathbf{X}}^-(k+1) &= F_{\sigma(k)}\hat{\mathbf{X}}(k),\\ P^-(k+1) &= F_{\sigma(k)}\mathbb{P}(k)F_{\sigma(k)}^T + \Sigma(\hat{\mathbf{X}}(k)),\end{aligned}$$

and the following measurement update:

$$\begin{aligned}K_{k+1} &= P^-(k+1)H^T\left[HP^-(k+1)H^T + R_v\right]^{-1},\\ \mathbb{P}(k+1) &= [I - K_{k+1}H]P^-(k+1),\\ \hat{\mathbf{X}}(k+1) &= \hat{\mathbf{X}}^-(k+1) + K_{k+1}[y_{meas}(k+1) - H\hat{\mathbf{X}}^-(k+1)].\end{aligned}$$

When the state estimates $\hat{\mathbf{X}}$ are available, we formulate the following optimization problem based on a one-step output prediction, in order to synthesize the control input at the next step:

$$\begin{aligned}&\min\{|y_{est}(k+2) - y_{des}(k+2)|,\ \sigma(k+1) \in \mathbb{Z}_{2l+1}\}\\ &\text{such that } \begin{cases}\hat{\mathbf{X}}(k+2) = F_{\sigma(k+1)}\hat{\mathbf{X}}(k+1)\\ y_{est}(k+2) = H\hat{\mathbf{X}}(k+2),\end{cases}\end{aligned}$$

where $y_{des}(\cdot)$ is a desired reference signal and $\hat{\mathbf{X}}(k+1)$ is provided by the conditional Kalman filter above. The obtained optimal value for $\sigma(k+1)$ provides the set-point $\theta_s(k+1)$, which is applied to the entire TCL population at the following $(k+1)^{\text{th}}$ iteration. Figure 6.2 illustrates the closed-loop configuration for state estimation and one-step regulation of the power consumption.

6.3.2 Regulation via Stochastic Model Predictive Control (SMPC)

We can perform power tracking by formulating and solving the following SMPC problem [46]:

$$\begin{aligned}&\min_{\sigma(\tau)} J_k = \mathbb{E}\left[\sum_{\tau=k+1}^{T}[y_{abs}(\tau) - y_{des}(\tau)]^2 + \kappa^T\mathbf{X}(T)\middle|\mathbf{X}(k)\right] \qquad (6.19)\\ &\text{such that } \begin{cases}\mathbf{X}(\tau+1) = F_{\sigma(\tau)}\mathbf{X}(\tau) + \mathbf{W}(\tau), \quad y_{abs}(\tau) = H\mathbf{X}(\tau),\\ \sigma(\tau) \in \mathbb{Z}_{2l+1}, \quad \forall\tau \in \{k, k+1, \ldots, T-1\}.\end{cases}\end{aligned}$$

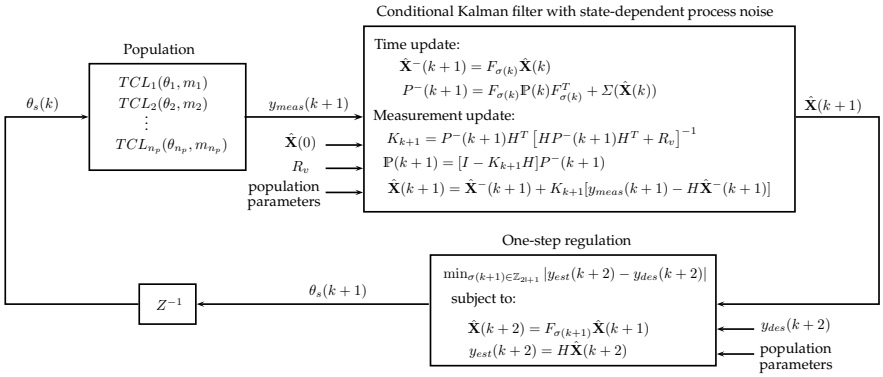


Figure 6.2: State estimation and one-step regulation for the closed-loop control of the power consumption.

The function J_k is called the conditional cost-to-go [72] which comprises a running cost for tracking and a terminal cost. The terminal cost is assumed to be a linear combination (with weighting vector κ) of the model states at final time T, and practically accounts for possible penalty weights over the number of TCLs within the temperature intervals. The expectation is taken over the underlying probability space for the trajectories of the process over the time interval $[k+1, T]$. The dynamics are nonlinear due to the switching nature of the control signal. The average evolution of the states and output of the system can be expressed by the following deterministic difference equation:

$$\mathbb{E}[\mathbf{X}(\tau+1)] = F_{\sigma(\tau)}\mathbb{E}[\mathbf{X}(\tau)], \quad \mathbb{E}[y_{abs}(\tau)] = H\mathbb{E}[\mathbf{X}(\tau)].$$

The associated state transition matrix $\Phi_\sigma(T,k) = F_{\sigma(T-1)}F_{\sigma(T-2)}\dots F_{\sigma(k)}$ provides a closed form for the average evolution over the interval $[k, T]$:

$$\mathbb{E}[\mathbf{X}(T)] = \Phi_\sigma(T,k)\mathbb{E}[\mathbf{X}(k)], \quad \mathbb{E}[y_{abs}(T)] = H\Phi_\sigma(T,k)\mathbb{E}[\mathbf{X}(k)].$$

Thanks to the linearly state-dependent covariance matrix, we can establish the following result.

Theorem 6.4 *The cost function of the SMPC problem can be computed explicitly as*

$$J_k = \sum_{\tau=k+1}^{T} \left[H\Phi_\sigma(\tau,k)\mathbf{X}(k) - y_{des}(\tau)\right]^2 + \Psi_\sigma(T,k)\mathbf{X}(k), \tag{6.20}$$

where the matrix

$$\Psi_\sigma(T,k) = \kappa^T\Phi_\sigma(T,k) + \frac{1}{n_p}\sum_{\tau_1=k}^{T}\sum_{\tau_2=\tau_1+1}^{T} \mathscr{R}(H\Phi_\sigma(\tau_2,\tau_1+1), F_{\sigma(\tau_1)})\Phi_\sigma(\tau_1,k),$$

and where $\mathscr{R} : \mathbb{R}^{1\times 2n} \times \mathbb{R}^{2n\times 2n} \to \mathbb{R}^{1\times 2n}$ *is a matrix-valued map with* $\mathscr{R}(C,D) =$

$C^{\circ 2}D - (CD)^{\circ 2}$, *where the operator* $\circ 2$ *is the* Hadamard *square of the matrix (element-wise square).*

Proof: The proof is derived by computing the summation of J_k in (6.19) backwards, conditioning the expected value to the intermediate states and utilizing the equality $\nu\Sigma(\mathbf{X})\nu^T = \frac{1}{n_p}\mathscr{R}(\nu, P^T)\mathbf{X}$, for any $\nu \in R^{1\times 2n}$. □

The obtained explicit cost function is the sum of a quadratic cost for the deterministic average evolution of the system state and of a linear cost related to the covariance of the process noise.

Remark 6.2 *For both formulations of the power tracking problem, the reference signal* $y_{des}(\cdot)$ *is assumed to be given. This is in practice the case when the TCL population is controlled to provide ancillary services. Moreover, this holds when a power utility company (or aggregator) participating in an energy market: it can observe the profile of the energy price, solve an optimization problem at a higher level minimizing the total energy cost based on an energy storage model, and thus obtain the power reference signal [61].*

Example 6.1 *The SMPC formulation can accommodate problems where the population participates in the energy market to minimize the energy costs. In the real-time energy market the Locational Marginal Pricing algorithms result in the profile of energy price for time intervals of 5-minutes [79]. Given that profile, the population can save money by minimizing the total cost of its energy usage within the given time frame, i.e. consuming less energy when the price is high and more energy when the price is low, under some constraints, in the next 24 hours. Suppose the final time* T *is selected such that* $T = 24/h$*, where* h *is the length of the sampling time (5 minutes), and let the sequence* $\{\lambda_\tau, \tau = k+1, k+2, \dots, T\}$ *be the profile of the energy price provided by the energy market. The total energy consumption of the population is then* $\sum_{\tau=k+1}^{T} \lambda_\tau y_{abs}(\tau)h$*. The following optimization problem can be solved, given the model dynamics, in order to minimize the expected value of the energy consumption:*

$$\min_{\sigma(\tau)} \mathbb{E}\left[\sum_{\tau=k+1}^{T} \lambda_\tau y_{abs}(\tau)h \middle| \mathbf{X}(k)\right] = \min_{\sigma(\tau)} \sum_{\tau=k+1}^{T} \lambda_\tau h H \Phi_\sigma(\tau, k)\mathbf{X}(k).$$

6.4 Numerical Case Study and Benchmarks

In this section we compare the performance of the aggregation procedure with that developed in [53], which as discussed obtains an aggregated model with dynamics that are deterministic, and has in fact been shown to be a special (limiting) case of the model in this chapter (cf. Remark 6.1). We further synthesize global controls over the temperature set-point to perform tracking of the total power consumption of the population.

Parameter	Interpretation	Value
θ_s	temperature set-point	$20\,[°C]$
δ	dead-band width	$0.5\,[°C]$
θ_a	ambient temperature	$32\,[°C]$
R	thermal resistance	$2\,[°C/kW]$
C	thermal capacitance	$10\,[kWh/°C]$
P_{rate}	power	$14\,[kW]$
η	coefficient of performance	2.5
h	time step	$10\,[sec]$

Table 6.1: Nominal values of parameters for the case study as from [20].

For all simulations we consider a population size of $n_p = 500$, however recall that our abstraction is proved to work as desired for any value n_p of the population size. As a benchmark, we have run 50 Monte Carlo simulations for the TCL population based on the dynamics in (6.1)-(6.2) and aggregated explicitly, and computed the empirical average total power consumption.

6.4.1 Aggregation of a Homogeneous Population of TCLs

Each TCL is characterized by parameters that take value in Table 6.1. All TCLs are initialized with a temperature at the set-point ($\theta(0) = \theta_s$), half of them in the OFF mode ($m(0) = 0$), and the other half in the ON model ($m(0) = 1$). Unlike the deterministic dynamics considered in [53], the model in (6.1) includes a process noise: we select initially a small value for its standard deviation as $\sigma = 0.001\sqrt{h} = 0.0032$.

The abstraction in [53] is obtained by partitioning exclusively the dead-band and by "moving the probability mass" outside of this interval to the nearest bin in the opposite mode. Recall that in the new approach put forward in this chapter we need to provide a partition not only for the dead-band but for the larger range of temperatures (cf. Fig. 6.1). Sample trajectories of the TCL population are presented in Figure 6.3: the second set of trajectories, obtained for a larger value of noise level, confirms that we need to partition the whole temperature range, rather than exclusively the dead-band. The abstraction in [53] depends on a parameter n_d, denoting the number of bins: we select $n_d = 70$, which leads to a total of 140 states. The selection of n_d has been steered by empirical tuning, targeted toward optimal performance; however, in general there seems to be no clear correspondence between the choice of n_d and the overall precision of the abstraction procedure in [53].

For the formal abstraction proposed in this chapter, we construct the partition as in (6.4) with parameters $\mathsf{l} = 70, \mathsf{m} = 350$, which leads to $2n = 1404$ abstract states. Here notice that the presence of a small standard deviation σ for the process noise (not included in the dynamics of [53]) requires a smaller partition size to finely resolve the jumping probability between adjacent bins. Let us emphasize again

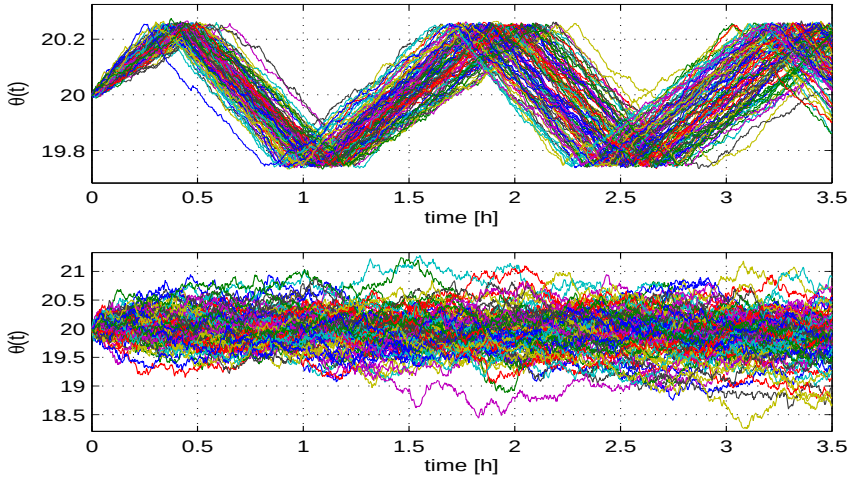


Figure 6.3: Sample trajectories of the TCL population for two different values of the standard deviation of the process noise, $\sigma = 0.0032$ and $\sigma = 0.032$.

that an increase in n_d for the method in [53] does not lead to an improvement of the outcomes.

The results obtained for a small noise level $\sigma = 0.0032$ are presented in Figure 6.4 (top). The aggregate power consumption has an oscillatory decay, since all thermostats are started in a single state bin (they share the same initial condition) and are thus "synchronized" at the outset. This outcome matches that presented in [53]: the deterministic abstraction[2] in [53] produces precise results for the first few (2-3) oscillations, after which the disagreement increases.

Let us now select a larger standard deviation for the process, to take the value $\sigma = 0.01\sqrt{h} = 0.032$, all other parameters being the same as before. We now employ $n_d = 5$ (by empirical optimal tuning), and $\mathsf{l} = 7$, and $\mathsf{m} = 35$, which leads to 10 and 144 abstract states, respectively. Figure 6.4 (bottom) presents the results of the experiment. It is clear that the model abstraction in [53] is not capable to generate a valid trajectory for the aggregate power, whereas the output of the formal abstraction proposed in this chapter nicely matches that of the average aggregated power consumption. Let us again remark that increasing number of bins n_d does not seem to improve the performance of the deterministic abstraction in [53], but rather renders the oscillations more evident. On the contrary, our approach allows the quantification of an explicit bound on the error made: for instance, the error on the normalized power consumption with parameters $N = 2$ and $\mathsf{l} = 70$ is equal to 0.226 (absolute quantity). As a final remark, let us emphasize that the outputs of both the abstract models converge to steady-state values that

[2]Let us again remark that by "deterministic abstraction" we mean that the aggregate model $\mathbf{X}(k+1) = P^T\mathbf{X}(k)$ obtained in [53] is a deterministic equation. However, the process noise is included in the temperature evolution of the TCLs.

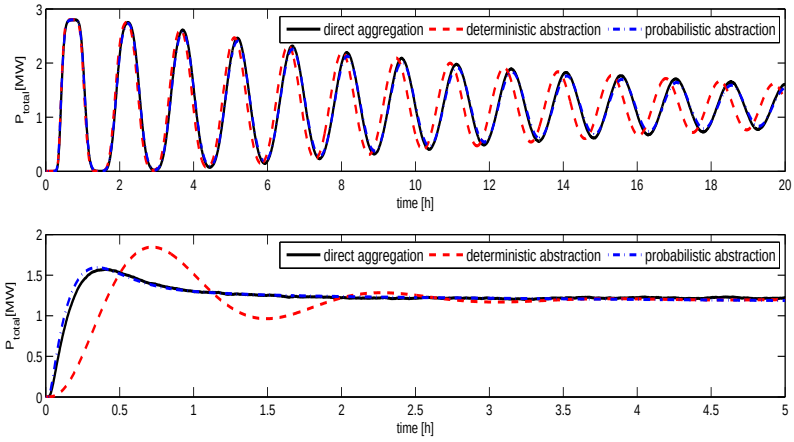


Figure 6.4: Comparison of the deterministic abstraction from [53] with the formal stochastic abstraction, for a small process noise $\sigma = 0.0032$ (top panel) and a larger value $\sigma = 0.032$ (bottom panel).

may be slightly different from those obtained as the average of the Monte Carlo simulations for the model aggregated directly. This discrepancy is due to the intrinsic errors introduced by both the abstraction procedures, which approximate a concrete continuous-space model (discontinuous stochastic difference equation) with discrete-space abstractions (finite-state Markov chains). However, whereas the abstraction in [53] does not offer an explicit quantification of the error, the formal abstraction proposed in this chapter does, and further allows the tuning (decrease) of such an error bound, by choice of a larger cardinality for the partitions set. However as a tradeoff, recall that increasing the number of partitions demands managing a Markov chain abstraction with a larger size.

6.4.2 Abstraction and Control of a Population of TCLs

With focus on the abstraction proposed in this chapter for a homogeneous population (again of $n_p = 500$ TCLs), the one-step output prediction and regulation scheme of Section 6.3.1 is applied with the objective of tracking a randomly generated piecewise-constant reference signal. We have used discretization parameters $\mathsf{l} = 8$, $\mathsf{m} = 40$, and the standard deviation of the measurement ($\sqrt{R_v}$) has been chosen to be 0.5% of the total initial power consumption. Figure 6.5 displays the tracking outcome (top), as well as the required set-point signal synthesized by the above optimization problem (bottom). Notice that the set-point variation is bounded to within a small interval, which practically means that the users in the TCLs are unaffected by that.

Finally, we have employed the SMPC scheme described in Section 6.3.2, combined with the conditional Kalman state estimator of Section 6.3.1, to track a piecewise-constant reference signal over a homogeneous population of TCLs. A prediction

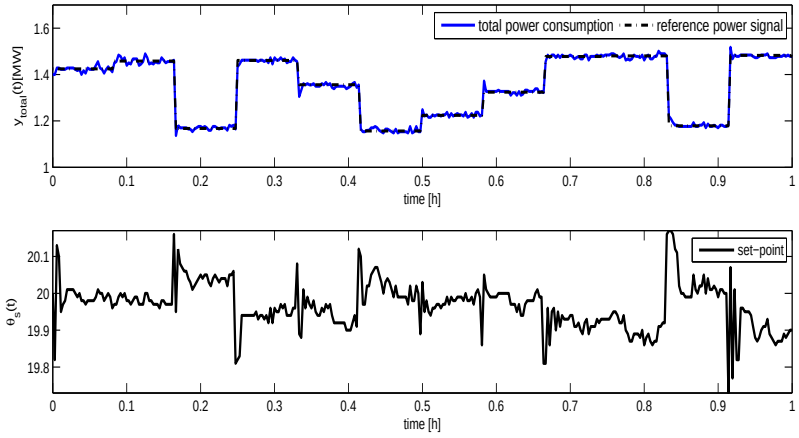


Figure 6.5: Tracking of a piecewise-constant reference signal (top panel) by set-point control (bottom panel) in a homogeneous population of TCLs abstracted by the formal probabilistic approach.

horizon of $T - k = 5$ steps has been selected. The discrete nature of the optimization variable in (6.19) requires us, at each time step, to compute the cost function J_k for each sequence of $\sigma(\cdot)$ and find the optimal one. In order to reduce computational burden of the optimization we introduce the following constraint on the variation of the set-point has been considered: $\left|\frac{d\theta_s}{dt}\right| \simeq \left|\frac{\theta_s(k+1)-\theta_s(k)}{h}\right| \leq \upsilon = 0.025$. Figure 6.6 presents the power consumption of the population (top) and the required set-point variation (bottom). The displayed response consists of a transient and of a steady-state phases. It takes 3 minutes to reach the steady-state phase because of the limitations on the max rate of set-point changes. This can be seen from the plot of the set-point control signal, which first decreases and then increases within the transient phase with a constant rate. In order to obtain a faster transient phase, the upper bound for the set-point changes may be increased.

6.5 Conclusions

This chapter has put forward a formal approach for the abstraction of the dynamics of TCLs and the aggregation of a population model. The approach starts by applying the Markov chain abstraction algorithm of Chapter 2 to the dynamics of the single TCL. Given the transition probability matrices of the obtained Markov chains, it is possible to write down the explicit state-space model of the population and further aggregate it. In this chapter we have also discussed approaches to perform controller synthesis over the aggregated model. It is worth mentioning that the error bound derived for the autonomous populations can be extended to the case of controlled populations of TCL.

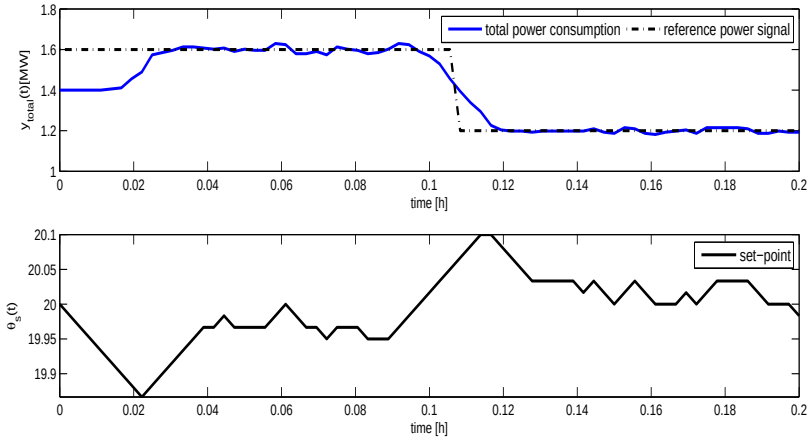


Figure 6.6: Tracking of a piecewise-constant reference signal (top panel) by set-point control (bottom panel) for a homogeneous population of TCLs using the SMPC scheme.

Looking forward, developing approaches for the heterogeneous case, synthesizing improved control schemes, and ameliorating the error bounds are directions that are research-worthy in order to render the approach further applicable in practice.

7 CHAPTER

FAUST^2: Formal Abstractions of Uncountable-STate STochastic processes

FAUST^2 is a software tool that generates formal abstractions of (possibly non-deterministic) discrete-time Markov processes (dtMP) defined over uncountable (continuous) state spaces. A dtMP model is specified in MATLAB and abstracted as a finite-state Markov chain or a Markov decision process. The abstraction procedure runs in MATLAB and employs parallel computations and fast manipulations based on vector calculus. The abstract model is formally put in relationship with the concrete dtMP via a user-defined maximum threshold on the approximation error introduced by the abstraction procedure. FAUST^2 allows exporting the abstract model to well-known probabilistic model checkers, such as PRISM or MRMC. Alternatively, it can handle internally the computation of PCTL properties (e.g. safety or reach-avoid) over the abstract model, and refine the outcomes over the concrete dtMP via a quantified error that depends on the abstraction procedure and the given formula. The toolbox is available at

http://sourceforge.net/projects/faust2/.

7.1 Models: Discrete-Time Markov Processes

The focus of this thesis and the applicability of FAUST^2 is given to the class of discrete-time Markov processes. A dtMP $s(k), k \in \mathbb{N}_0$ is defined over a general state space, such as a finite-dimensional Euclidean domain [63] or a hybrid state space [4]. As we discussed in Chapter 2, the model is denoted by the pair $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$. $\mathcal{S}$ is a continuous (uncountable) but bounded state space, e.g. $\mathcal{S} \subset \mathbb{R}^n, n < \infty$. We denote by $\mathcal{B}(\mathcal{S})$ the associated sigma algebra. The conditional stochastic

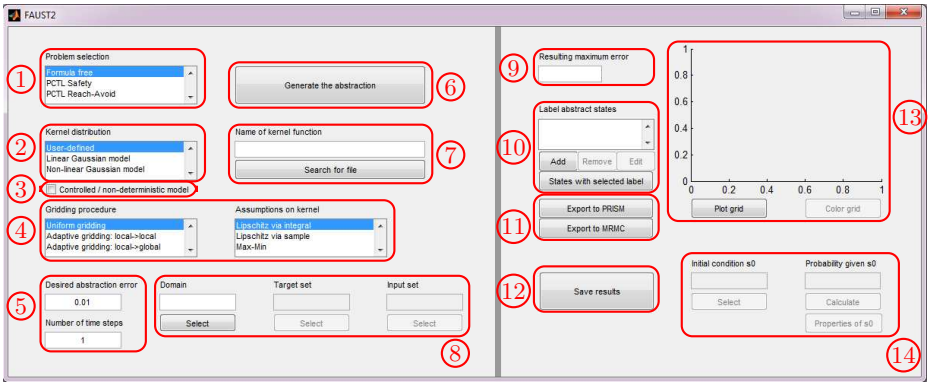


Figure 7.1: Graphical User Interface of FAUST^2, overlaid with numbered boxes

kernel $T_{\mathfrak{s}} : \mathcal{B}(\mathcal{S}) \times \mathcal{S} \to [0,1]$ assigns to each point $s \in \mathcal{S}$ a probability measure $T_{\mathfrak{s}}(\cdot|s)$, so that for any set $A \in \mathcal{B}(\mathcal{S})$, $k \in \mathbb{N}_0$, $\mathbb{P}(s(k+1) \in A | s(k) = s) = \int_A T_{\mathfrak{s}}(d\bar{s}|s)$. (Please refer to Chapter 2 for more details on the definition and observe the code or the case study of Section 7.5 for a modeling example.)

Implementation: The user interaction with FAUST^2 is enhanced by a Graphical User Interface. A dtMP model is fed into FAUST^2 as follows. Select the Formula free option in the box Problem selection ① in Figure 7.1, and enter the bounds on the state space $\mathcal{S}$ as a $n \times 2$ matrix in the prompt Domain in box ⑧. Alternatively if the user presses the button Select ⑧, a pop-up window prompts the user to enter the lower and upper values of the box-shaped bounds of the state space. The transition kernel $T_{\mathfrak{s}}$ can be specified by the user (select User-defined ②) in an m-file, entered in the text-box Name of kernel function, or loaded by pressing the button Search for file ⑦. Please open the files `./Templates/SymbolicKernel.m` for a template and `ExampleKernel.m` for an instance of kernel $T_{\mathfrak{s}}$. As a special case, the class of affine dynamical systems with additive Gaussian noise is described by the difference equation $s(k+1) = \mathtt{A}s(k) + \mathtt{B} + \eta(k)$, where $\eta(\cdot) \sim \mathcal{N}(0, \mathtt{Sigma})$. (Refer to the case study of Section 7.5 on how to express the difference equation as a stochastic kernel.) For this common instance, the user can select the option Linear Gaussian model in the box Kernel distribution ②, and input properly-sized matrices `A`, `B`, `Sigma` in the MATLAB workspace. FAUST^2 also handles Gaussian dynamical models $s(k+1) = f(s(k)) + g(s(k))\eta(k)$ with nonlinear drift and variance: select the bottom option in box ② and enter the symbolic function `[f g]` via box ⑦. □

The software also handles models with non-determinism [77]: a controlled dtMP is a tuple $\mathfrak{S} = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$, where $\mathcal{S}$ is as before, $\mathcal{U}$ is a continuous control space (e.g. a bounded set in $\mathbb{R}^m$), and $T_{\mathfrak{s}}$ is a Borel-measurable stochastic kernel $T_{\mathfrak{s}} : \mathcal{B}(\mathcal{S}) \times \mathcal{S} \times \mathcal{U} \to [0,1]$, which assigns to any state $s \in \mathcal{S}$ and input $u \in \mathcal{U}$ a probability measure $T_{\mathfrak{s}}(\cdot|s,u)$. Please refer to Chapter 5 for a more detailed definition.

Implementation: In order to specify a non-deterministic model in FAUST^2, tick the relevant check Controlled/non-deterministic model ③, and enter the bounds on the space $\mathcal{U}$ as a $m \times 2$ matrix in the window Input set ⑧. □

7.2 Formal Finite-State Abstractions of dtMP Models

This section discusses the basic procedure to approximate a dtMP $\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}})$ as a finite-state Markov chain (MC) $\mathfrak{P} = (\mathcal{P}, T_p)$, as implemented in FAUST^2. The software uses Algorithm 1 to abstract a general model $\mathfrak{S}$ as a finite-state MC $\mathfrak{P}$, regardless of the specifics of the probabilistic invariance problem studied in Chapter 2 by assuming that $\mathcal{A} = \mathcal{S}$. The quantification of the abstraction error requires that the state space $\mathcal{S}$ is bounded.

Consider the representation of the kernel $T_{\mathfrak{s}}$ by its density function $t_{\mathfrak{s}} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^{\geq 0}$, namely $T_{\mathfrak{s}}(ds'|s) = t_{\mathfrak{s}}(s'|s)ds'$ for any $s, s' \in \mathcal{S}$. The abstraction error over the next-step probability distribution introduced by Algorithm 1 hinges on the Lipschitz continuity of the function $t_{\mathfrak{s}}$: the next-step abstraction error is $E = h_{\mathfrak{s}}\delta_{\mathfrak{s}}\mathscr{L}(\mathcal{S})$, where $h_{\mathfrak{s}}$ is the Lipschitz constant of the density function (cf. Assumption 2.1), $\delta_{\mathfrak{s}}$ is the max diameter of the state-space partition sets and $\mathscr{L}(\mathcal{S})$ is the volume of the state space. When interested in working over a finite, N-step time horizon, the error results in the quantity EN. Notice that the error can be reduced via $\delta_{\mathfrak{s}}$ by considering a finer partition, which on the other hand results in a MC $\mathfrak{P}$ with a larger state space (cf. Theorem 2.2).

Implementation: FAUST^2 enables the user to enter the time horizon N of interest (box Number of time steps ⑤), and a threshold on the maximum allowed error (box Desired abstraction error ⑤). The software generates a Markov chain with the desired accuracy by pressing the button Generate the abstraction ⑥. Among other messages, the user is prompted with an estimated running time, which is based on an over-approximation of the Lipschitz constant of the kernel, on a uniform partitioning of the space $\mathcal{S}$ [1], and on the availability of parallelization procedures in MATLAB, and is asked whether to proceed. □

In the case of a non-deterministic dtMP, FAUST^2 uses Algorithm 8 to abstract the model $\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}})$ as an MDP $\mathfrak{D} = (\mathcal{S}_{\mathfrak{d}}, \mathcal{U}_{\mathfrak{d}}, T_{\mathfrak{d}})$, regardless of the specifics of the maximal probabilistic invariance problem studied in Chapter 5 by assuming that $\mathcal{A} = \mathcal{S}$. The next-step abstraction error can be formally quantified as $E = 2(h_{\mathfrak{s}}\delta_{\mathfrak{s}} + h_{\mathfrak{u}}\delta_{\mathfrak{u}})\mathscr{L}(\mathcal{S})$, where $\delta_{\mathfrak{u}}$ is the max diameter of the input-space partitions and $h_{\mathfrak{u}}$ is the Lipschitz constant of the density function with respect to the inputs (cf. Assumption 5.1 and Theorem 5.4).

Implementation: The user may tick the check in ③ to indicate that the dtMP is controlled (non-deterministic), specify a box-shaped domain for the input in box Input set ⑧, enter a time horizon in box Number of time steps ⑤, and require an error threshold in box Desired abstraction error ⑤. FAUST^2 automatically generates an MDP according to the relevant formula on the error.

Notice that the quantification of the abstraction error requires state and input spaces to be bounded. In case of an unbounded state space, the user should truncate it to a bounded, box-shaped domain: selecting the Formula free option in the box Problem selection ①, the domain is prompted in box Domain ⑧. Algorithms

[1] At the moment we assume to have selected options Uniform gridding and Lipschitz via integral among the lists in box ④. Comments on further options are in Section 7.3.

1,8 used for the abstraction automatically assign an absorbing abstract state to the truncated part of the state space. □

The states of the abstract model $\mathfrak{P}$ may be labeled. The state labeling map $\mathsf{L} : \mathcal{P} \to \Sigma$, where Σ is a finite alphabet, is defined by a set of linear inequalities: for any $\alpha \in \Sigma$ the user characterises the set of states $\mathsf{L}^{-1}(\alpha)$ as the intersection of half-planes (say, as a box or a simplex): the software automatically determines all points $z \in \mathcal{P}$ belonging to set $\mathsf{L}^{-1}(\alpha)$. The obtained labeled finite-state model can be automatically exported to well-known model checkers, such as PRISM and MRMC [45, 50], for further analysis. In view of the discussed error bounds, the outcomes of the model checking procedures over the abstract model $\mathfrak{P}$ may be refined over the concrete dtMP $\mathfrak{S}$ – more details can be found in Chapter 2 on adaptive abstraction algorithms.

Implementation: Labels are introduced in FAUST^2 as follows: suppose that the intersection of half-planes $A_\alpha z \leq B_\alpha$ (where A_α, B_α are properly-sized matrices) tags states z by label $\alpha \in \Sigma$. The user may add such a label by pressing button Add (10) and subsequently entering symbol α and matrices A_α, B_α in the pop-up window. The user can also edit or remove any previously defined label using buttons Edit, Remove in (10), respectively. The button States with selected label (10) shows the sets associated with the active label over the plot in (13).

The user may click the buttons in (11) to export the abstracted model to PRISM or to MRMC. Alternatively, FAUST^2 is designed to automatically check or optimize over (quantitative, non-nested) PCTL properties, without relying on external model checkers: Section 7.3 elaborates on this capability. □

7.3 Formula-Dependent Abstractions for Verification

Probabilistic Computation Tree Logic (PCTL) [43] are used to describe properties over stochastic systems. PCTL is an extension of Computation Tree Logic (CTL) which allows for probabilistic quantification of described properties. It includes a probabilistic operator to write specifications of the system. PCTL is a useful logic for stating soft properties, e.g. "eventually reach A and then B with probability greater than 0.9 within 10 time steps". The syntax of PCTL is composed of state and path formulas. The state formulas include the property that a path formula has a probability in a given interval. The path formulas include next (X), bounded until ($\mathsf{U}^{\leq N}$), and until (U) operators.

Algorithms 1,8, presented in Chapters 2,5, can be employed to abstract a dtMP as a finite-state MC/MDP, and to directly check it against properties such as probabilistic invariance or reach-avoid, that is over (quantitative, non-nested) bounded-until specifications in PCTL. Next, we detail this procedure for the finite-horizon probabilistic invariance (a.k.a. safety) problem, which was formalized in Section 2.2.2. The invariance probability $p_{s_0}(\mathcal{A})$ can be employed to characterize the satisfiability set of a corresponding bounded-until PCTL formula, namely

$$s_0 \models \mathbb{P}_{\sim\epsilon}\{\text{true } \mathsf{U}^{\leq N}(\mathcal{S}\backslash\mathcal{A})\} \quad \Leftrightarrow \quad p_{s_0}(\mathcal{A}) \backsim 1-\epsilon,$$

where $\mathcal{S}\backslash\mathcal{A}$ is the complement of $\mathcal{A}$ over $\mathcal{S}$, true is a state formula valid everywhere on $\mathcal{S}$, the inequality operator $\sim \in \{>, \geq, <, \leq\}$, and $\backsim$ represents its complement.

FAUST^2 formally approximates the computation of $p_{s_0}(\mathcal{A}), \forall s_0 \in \mathcal{S}$, as follows. $\mathfrak{S}$ is abstracted as an MC $\mathfrak{P}$ via Algorithm 1. Given the obtained MC $\mathfrak{P} = (\mathcal{P}, T_p)$ and considering the finite safe set $A_p \subset \mathcal{P}$, FAUST^2 internally computes the safety probability over $\mathfrak{P}$ via Bellman recursion of Theorem 2.1 along with the associated abstraction error which is now tailored to the PCTL formula of interest.

Implementation: The user may select option PCTL Safety in the list within box ①, enter the boundaries of the Safe set within box ⑧, and press button ⑥ to proceed obtaining the abstraction and computing the probability of the selected formula. The computed value of $p_{s_0}(\mathcal{A})$ is displayed in box Probability given s0 ⑭, for any user-selected initial state s_0 that is input in box Initial condition s0 ⑭. The user can optionally press button Properties of s0 ⑭ to get more information about the concrete state s_0, including the related discrete state $z = \xi(\Xi(s))$ of the MC, as well as the associated labels. Furthermore, the quantity $p_{s_0}(\mathcal{A})$ can be plotted, as a function of the initial state s_0, by pressing buttons Plot grid and Color grid in ⑬. Clearly these outputs are exclusively available for models of dimensions $n = 1, 2, 3$. □

It is of interest to obtain tight bounds on the error associated with the abstraction procedure since, given a user-defined error threshold, tighter bounds would generate abstract models $\mathfrak{P}$ with fewer states. The abstraction error bound in FAUST^2, tailored around the discussed safety problem, can be efficiently decreased under different types of regularity assumptions on the conditional density function of the dtMP $\mathfrak{S}$. FAUST^2 uses the adaptive Algorithms 3,4 to generate the abstracted MC based on local computation of the error.

Implementation: FAUST^2 allows the user to select three different gridding procedures in box Gridding procedure ④: the reader is referred to Chapter 2 for the details of these three options. The Uniform gridding option leads to a one-shot (non sequential) procedure, as already discussed in Section 7.2, whereas the two Adaptive gridding options result in sequential and adaptive procedures leading to better errors and to smaller abstractions, but in general requiring more computation time. The error bound quantification hinges on the constants introduced in Theorem 2.7, which are made available in box Assumptions on kernel ④: tighter errors lead to longer computations. In order to provide with full control on the chosen inputs, for any possible selection of gridding procedure, desired abstraction error, and error bound computation, the user is prompted in a pop-up window with an estimated running time, and asked whether to proceed.

This range of algorithms and procedures are also implemented for probabilistic reach-avoid (constrained reachability) problems, which are encompassed by general bounded-until PCTL formulas $\mathbb{P}_{\sim\epsilon}\{\Phi\ \mathsf{U}^{\leq N}\Psi\}$. The user can select this option in box Problem selection ①, and is asked to input sets Φ, Ψ as safe and target sets in the boxes of ⑧.

Let us remark that the described abstraction algorithms and procedures are as well available for the formula-free abstraction discussed in Section 7.2. □

Computation of the maximal safety probability and *maximally safe* policies for a controlled dtMP were discussed in Chapter 5. Similarly we can compute the *minimally safe* policy, or an optimal policy for the reach-avoid problem.

Implementation: FAUST^2 computes a suboptimal policy for a given problem over an MDP, with a given threshold on the distance to the optimal safety probability, and quantifies the corresponding approximate quantity $p^{\boldsymbol{\mu}^*}_{s_0}(\mathcal{A})$. The approximate optimal policy can be stored by pushing button Save results (12), which provides the user with two options: either storing it on the disk as a .mat file, or loading it to the workspace. □

7.4 Accessing and Testing FAUST^2

The toolbox is available at `http://sourceforge.net/projects/faust2/`. This toolbox has been successfully tested with MATLAB R2012a, R2012b, R2013a, R2013b, on machines running Windows 7, Apple OSX 10.9, and OpenSUSE Linux. FAUST^2 exploits the command integral of MATLAB (introduced in version R2012a) for numerical integrations. (The previous versions of MATLAB contain instruction quad and its variations, which will be removed in the future versions of MATLAB – we have thus opted for the most up-to-date version.) Optimization and symbolic computation toolboxes of MATLAB are necessary. FAUST^2 automatically checks the presence of these packages and displays an error to the user in their absence. The software also takes the advantage of the MATLAB parallel computation toolbox if present. The use of parallel computation toolbox is currently disabled for Apple operating systems due to a conflict.

The user can download FAUST^2 from Sourceforge. The files are organized in the main folder as follows: the sub-folder `Autonomous Models` contains the codes for deterministic systems (without input); the sub-folder `Controlled Models` includes the codes for non-deterministic systems (input dependent); the sub-folder `Templates` contains templates and examples for the definition of symbolic conditional density functions; the sub-folder `Case Study` contains the files used in the next Section to test the software on a practical study. The file `README` can be opened with your preferred text editor and contains instructions on how to set up and run the software. As a starting point, FAUST^2 can be tested on a case study as elaborated in the next Section. To perform this test, set the current directory of MATLAB to the folder where the software is stored and run `FAUST2.m` from the MATLAB command line.

7.5 Case Study

FAUST^2 can be used to obtain all the results of Chapters 2,5. To illustrate the use of our software we apply FAUST^2 to compute optimal control strategies for the known room temperature regulation benchmark [37]. Probabilistic models for

the underlying dynamics are based on [58] and on [4]. We consider the temperature regulation in multiple rooms via cooling water circulation. The amount of extracted heat is changed via a flow-control valve. Then the input signal is the percentage of the valve in the open position. The dynamics of the room temperature evolve in discrete time according to the equations

$$\begin{aligned} s_1(k+1) &= s_1(k) + \frac{\Delta}{C_{ra}}((s_2(k) - s_1(k))k_{cw}u(k) + (T_a - s_1(k))k_{out}) + \eta_{ra}(k), \\ s_2(k+1) &= s_2(k) + \frac{\Delta}{C_{cw}}((s_1(k) - s_2(k))k_{cw}u(k) + Q) + \eta_{cw}(k), \end{aligned} \tag{7.1}$$

where s_1 is the air temperature inside the room, s_2 is the cooling water temperature, T_a is the ambient temperature, Δ is the discrete sampling time [min], and $\eta_{ra}(\cdot), \eta_{cw}(\cdot)$ are stationary, independent random processes with normal distributions $\mathcal{N}(0, \sigma_{ra}^2\Delta)$ and $\mathcal{N}(0, \sigma_{cw}^2\Delta)$, respectively. Equations (7.1) can be encompassed in the condensed two-dimensional model

$$s(k+1) = f(s(k), u(k)) + \eta(k), \quad \eta(\cdot) \sim \mathcal{N}(0, \Sigma_\eta),$$

which results in a stochastic kernel that is a Gaussian conditional distribution $\mathcal{N}(f(s,u), \Sigma_\eta)$, where $\Sigma_\eta = diag(\Delta[\sigma_{ra}^2, \sigma_{cw}^2])$. The file `Chiller_Kernel_2d.m` appearing with the first release of the software provides numerical values and physical interpretations of the parameters in equations (7.1), as well as the symbolic structure of the conditional density function. The dynamical model in (7.1) can be as well extended to a two-room temperature control (which results in a three-dimensional model), and its conditional density function can be found in file `Chiller_Kernel_3d.m`. We will run FAUST^2 on both 2D and 3D setups.

We are interested in keeping the temperature of the room(s) within a given temperature interval over a fixed time horizon: this can be easily stated as a (probabilistic) safety problem, where we maximise over the feasible inputs to the model. We instantiate and compute this problem over the model above as described in the main text, while providing a step-by-step guide to the user.

In order to select the problem and import the model in FAUST^2, please follow these steps: select PCTL Safety in box ①, choose User-defined in box ②, tick the check-box ③ to indicate a controlled model, and write the name Chiller_Kernel_2d.m in the text of box ⑦ to load the density function of the two-dimensional model (7.1).

In the next stage we perform the abstraction and compute the quantity of interest (maximal safety probability). Select the most straightforward (but coarsest) abstraction algorithm, by choosing options Uniform gridding and Lipschitz via integral in ④. Proceed entering the problem parameters as follows: input the number of time steps as 3 and select a desired abstraction error equal to 0.5 in box ⑤; enter the safe temperature interval A as `[19.7,20.3; 4.7,5.3]`, as well as the input space $\mathcal{U}$ as `[0,1]` in the text within box ⑧.

At this point the software can proceed with the main computations. Please press the button in box ⑥, in order to generate the abstract MDP, to compute the opti-

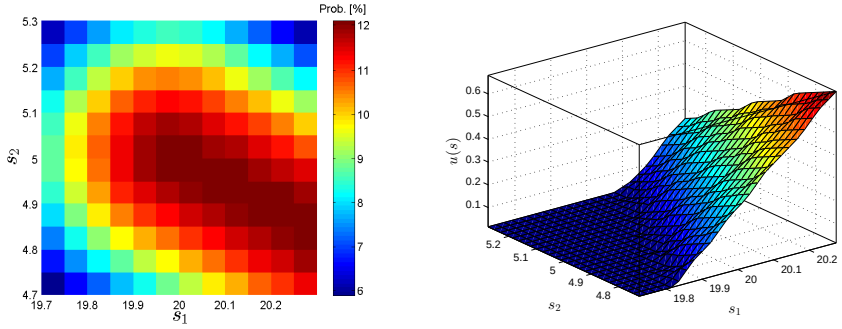


Figure 7.2: Room temperature control problem. Left: obtained uniform partition of the safe set, along with optimal safety probability for each partition set (colour bar on the right). The safety probability is equal to zero over the complement of the safe set. Right: optimal Markov policy at step $N-1$, as a function of the state.

mal policy and the related maximal safety probability. When the computation is complete, let us proceed with some post-processing: press the buttons Plot grid and Color grid in box ⑬, to generate Figure 7.2 (left) representing the maximal safety probability. The result of the computation can be stored for further analysis by pressing button ⑫: for instance Figure 7.2 (right) is generated by retrieving the optimal state-dependent Markov policy at step $N-1$.

A similar procedure can be followed to study the same probabilistic safety problem over a two-room temperature control, instantiated via the density function `Chiller_Kernel_3d.m`. Figure 7.3 presents the outcomes obtained using the Adaptive gridding and Lipschitz via integral options, selected in box ④. The abstraction parameters used in this problem is as follows: number of time steps `3`, safe temperature interval `[19.5,20.5; 19.5,20.5; 4.5,5.5]`, input space `[0,1; 0,1]`. We have selected a large abstraction error equal to 12 in box ⑤ to be able to visualize the adaptive grid generated by the software. The user can choose a smaller error at the cost of a larger computation time.

7.6 Summary of the Commands in the Graphical User Interface

We provide a summary of the commands of the GUI in FAUST^2, as they appear in the boxes highlighted in Figure 7.1.

① The box Problem selection provides a list with three options: select Formula free to obtain an abstraction of the model which can be exported to PRISM or to MRMC for further analysis; choose PCTL Safety in order to abstract

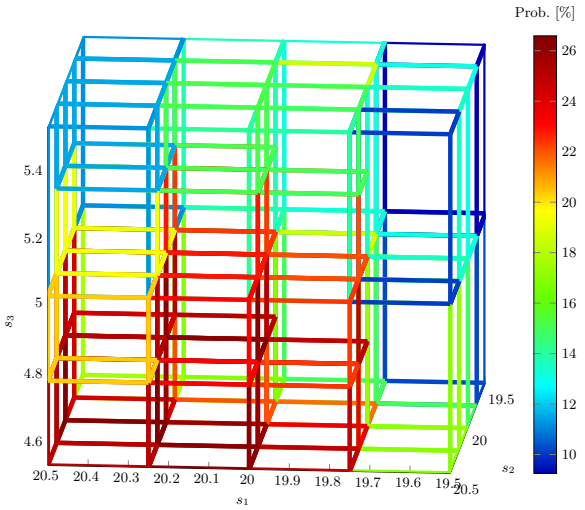


Figure 7.3: Two-room temperature control problem. Obtained partition of the safe set, together (bar) with optimal safety probability.

the model and compute a safety probability; or opt for PCTL Reach-Avoid to get the abstraction tailored around the computation of the reach-avoid probability.

② The box Kernel distribution gives three options in a list: select Linear Gaussian model if the model belongs to the class of Linear Gaussian difference equations (cf. Section 7.1) and define matrices `A`, `B`, `Sigma` in the MATLAB workspace; choose Non-linear Gaussian model if the process noise is Gaussian and the drift and variance are non-linear (cf. Section 7.1), enter the drift and variance as a single symbolic function with two outputs via box ⑦; otherwise choose User-defined and enter your kernel as a symbolic function using ⑦.

③ Check this box if the model is non-deterministic (controlled).

④ Box Gridding procedure provides three options: select Uniform gridding to generate a grid based on global Lipschitz constant h (cf. Section 7.2), where the state space is partitioned uniformly along each dimension; choose Adaptive gridding: local->local to generate the grid adaptively based on local Lipschitz constants $h(i, j)$ (cf. Assumption 2.2), where the size of partition sets is smaller where the local error is higher; select Adaptive gridding: local->global to generate the grid adaptively based on local Lipschitz constants $h(i)$ (cf. Assumption 2.3). The first option is likely to generate the largest number of partition sets and to be the fastest in the generation of the grid. The second option is likely to generate the smallest number of partition sets but to be the slowest in the grid generation. For the detailed comparison of these gridding procedures, please see Chapter 2.

The box Assumptions on kernel provides three choices: option Lipschitz via integral requires the density function $t_{\mathfrak{s}}(\bar{s}|s)$ to be Lipschitz continuous with respect to the current state s, and the quantity $T_p(z, z') = T_{\mathfrak{s}}(\Xi(z')|z)$ is used in the marginalization (integration) step; option Lipschitz via sample requires the density function $t_{\mathfrak{s}}(\bar{s}|s)$ to be Lipschitz continuous with respect to both current and the next states $s, \bar{s}$, and the quantity $T_p(z, z') = T_{\mathfrak{s}}(z'|z)\mathscr{L}(\Xi(z'))$ is used in the marginalization step; option Max-Min (cf. equation (2.13)) does not require any continuity assumption, but takes longer time in the computation of the error.

⑤ The time horizon of the desired PCTL formula or of the problem of interest, and the required upper bound on the abstraction error should be input in these two boxes. For the case of formula-free abstraction you may enter `1` as the number of time steps.

⑥ Press this button after entering the necessary data to generate the abstraction: this runs the main code. First, various checks are done to ensure the correctness of the inputed data. Then the partition sets are generated via gridding, the transition matrix is calculated, and the probability and the optimal policy are computed if applicable.

⑦ This box is activated for options User-defined and Non-linear Gaussian model in ②. For the first option, the conditional density function must be an m-file that generates $t_{\mathfrak{s}}(\bar{s}|s, u)$ symbolically. Please refer to `SymbolicKernel.m` for a template and `ExampleKernel.m` for an example. The name of kernel function should be entered in the text-box or the function should be loaded by pressing the button Search for file. For the option Non-linear Gaussian model, the non-linear drift and variance must be specified as a single symbolic function with two outputs. Please refer to `NonLinKernel.m` for a template and `NonLinKernelExample.m` for an example.

⑧ If the Formula-free option is selected in ①, the user can enter the bounds of the state space in the first of the boxes, named Domain. In case any of the additional two options in ① are selected, the boundaries of the safe set should be entered in the first text-box named Safe set. If the PCTL Reach-Avoid option in ① is selected, the second box is activated and the boundaries of the target set should be entered in the text-box named Target set. If the model is non-deterministic and the check in box ③ is ticked, the third box is also activated and the boundaries of the Input space may be entered in the box named Input set. In all cases the boundaries are to be given as a matrix with two columns, where the first and second columns contain lower and upper bounds, respectively. Alternatively, the user can press the Select button and separately enter the lower and upper bounds in the pop-up window.

⑨ The resulting error of the abstraction procedure, which is less than or equal to the desired abstraction error introduced in ⑤. This box shows the error associated with the abstracted model.

⑩ The user can add, remove, or edit labels associated with the abstract states. The set of states with any label $\alpha \in \Sigma$ can be represented by the intersection

of half-planes $A_\alpha z \leq B_\alpha$. In order to tag these states with the associated label, the user presses button Add and subsequently enters symbol α and matrices A_α, B_α in a pop-up window. The user can also edit or remove any previously defined label by activating its symbol in the static-box and using buttons Edit, Remove. The button States with selected label will show the set of states associated with the active label in (13). Adding labels is essential in particular for exporting the result to PRISM or to MRMC.

(11) The abstracted Markov chain or MDP can be exported to PRISM or to MRMC using these buttons. FAUST^2 enables two ways of exporting the result to PRISM: as a `.prism` format that is suitable for its GUI, or as the combination of `.tra` and `.sta` files, which are appropriate for the command line.

(12) Use this button to store the results. A pop-up window appears after pushing the button and the user can opt for storing the date over the workspace, or on disk as an `.mat` file.

(13) The user can plot the generated grid for the state space using the first button. Pressing this button opens a new window showing the partitioned input space for the controlled model. The solution of the safety and of the reach-avoid probability can also be visualized by pressing the second button. This option obviously works exclusively for dimensions $n = 1, 2, 3$.

(14) The user can enter any initial state s_0 in the first box and calculate the safety or the reach-avoid probability of the model starting from that initial state, by pressing the button Calculate. The button Properties of s0 gives the abstracted state associated to s_0, namely $z = \xi(\Xi(s_0))$ (cf. Algorithm 1), and all the labels assigned to this state.

7.7 Extensions and Outlook

FAUST^2 is presently implemented in MATLAB, which is the modelling software of choice in a number of engineering areas. We plan to improve part of its functionalities on a lower-level programming language. We further plan to extend the applicability of FAUST^2 to models with discontinuous and degenerate kernels (Chapter 3), to implement higher-order approximations (Chapter 4), and to embed formal techniques to obtain spatial truncations of the model dynamics [34]. Finally, we plan to look into developing bounds for infinite horizon properties.

8 CHAPTER

Conclusions and Future Research

This chapter summarizes the thesis and shortly discusses the main contributions presented in the manuscript. We also provide some directions of research are presently pursued by the author.

8.1 Conclusions

In this thesis we discussed finite abstractions of discrete-time Markov processes. With focusing on probabilistic safety problem we showed how the abstraction can be done formally based on regularity assumptions on the model. We relaxed the proposed assumptions and also developed further efficient methods under stronger assumptions. We used the abstraction technique to formally construct an aggregate model for a population of TCL. In the following we summarize our main contributions.

- **Adaptive and sequential gridding procedures.** We addressed the issue of curse of dimensionality in Markov chain abstraction of discrete-time Markov processes over general (continuous or hybrid) state space. We showed that the abstraction can be done adaptively respect to the local error made by the abstraction.
- **Abstraction of partially-degenerate stochastic processes.** We observed that this class of systems does not satisfy the Lipschitz continuity assumption. We developed new techniques for abstraction and error computation of these systems which hinges upon the regularity of the vector field characterizing the dynamical equation of the system.
- **Higher-order approximations.** We showed that if the conditional density function of the system satisfies stronger regularity assumptions, i.e. having higher-order partial derivatives, we could increase efficiency of the proposed numerical algorithms by using piece-wise polynomial interpolants.

- **Controlled discrete-time Markov processes.** We proved that such processes can be abstracted to a finite-state MDP. We showed that the abstraction error is composed of three terms: state-space discretization, input-space discretization, and state-dependent input spaces.
- **Application to aggregation of TCL.** We applied our techniques to the model of a single TCL and proposed a two step abstraction approach to describe the behavior of a population of TCL with a finite state system. Furthermore, we constructed a stochastic difference equation for the population dynamics and used stochastic MPC to regulate the total power consumption of the population.
- **Implementation.** We have implemented the abstraction methods as a software tool, which is available for download online.

8.2 Recommendations for Future Research

In this section we discuss some interesting topics that ought to considered as future research lines.

- **Partially-observed Markov processes.** In the results presented in this thesis we assume that the system states are fully accessible, while only the output information is available in real applications. This fact requires developing new techniques for abstraction of partially observed Markov processes and in particular *partially observed SHS*. The candidate for the abstract model would be hidden Markov chains or partially observable Markov decision processes. Such techniques can be integrated into FAUST^2.
- **Structured systems.** The adaptive gridding approach of Chapter 2 increases the scalability of the abstraction technique. Real systems like models of multi-robot environments, are of large dimensions but featuring a certain structure (hierarchy or loops) for the dependency between the evolution of states. We could exploit such a structure to overcome the curse of dimensionality utilizing the directed graph associated with the system equations. This idea is aligned with the decomposition and lumping techniques developed for Markov chains. In order to further scale up the algorithms used for verification, compositional model checking in the known *assume-guarantee reasoning* style [15] is proposed in the literature: this approach uses proof techniques to decomposed a property to easier, localized ones, and verifies assumptions made by the local statements to infer the the desired global property.
- **Pure-jump Markov processes.** New abstraction methods are required for continuous-time stochastic systems. The literature provides approximation methods which are based on time and space discretization [55]. However, these asymptotic results are not formal in the sense that they do not provide a priori bounds for the quality of approximation. One could develop

formal abstraction methods for this class of systems and extends them to continuous-time SHS where the jumps are triggered by a discrete-space process.

- **Approximation of the moments of a Markov process.** The described abstraction techniques can be applied to the density and moment approximation of a Markov process in time. Mean-field game problems and aggregate model of hybrid systems can benefit from these methods for the computation of optimal strategies.

- **Tighter connection of FAUST^2 to probabilistic model checking software.** The tool FAUST^2 presented in this thesis, generates formal abstractions of stochastic processes defined over continuous state spaces and allows exporting the abstract model to PRISM and MRMC. These well-known probabilistic model checkers can then be employed to check safety and other rich properties of the process. Future versions of FAUST^2 ought to tighten the connection with available probabilistic model checkers.

- **Heterogeneous population of TCLs.** While all the discussions of Chapter 6 are focused on homogeneous population of TCLs, real thermal loads feature heterogeneity in their set of parameters. An extensive scientific study needs to be done on aggregation of heterogeneous population of TCLs.

- **Aggregation of dynamical systems.** The two-step abstraction approach of Chapter 6 can be generalized to develop an aggregate model for any population of dynamical systems characterized by stochastic difference equations.

- **Numerical stability of the proposed algorithms.** All the results of this thesis are founded on the assumption of performing arithmetic operations with exact precision. Moreover, it is assumed that the marginalization step for creation of the transition probability matrix of the constructed Markov chain is performed without any error. These two assumptions are obviously violated in the simulation results of this thesis. Future research is needed to provide upper bound for these type of error introduced in the final computation of the quantities of interest.

Bibliography

[1] Hybrid system tools. `wiki.grasp.upenn.edu/hst/index.php`.

[2] A. Abate, J.-P. Katoen, J. Lygeros, and M. Prandini. Approximate model checking of stochastic hybrid systems. *European Journal of Control*, 6:624–641, 2010.

[3] A. Abate, J.-P. Katoen, and A. Mereacre. Quantitative automata model checking of autonomous stochastic hybrid systems. In *ACM Proceedings of the 14th International Conference on Hybrid Systems: Computation and Control*, pages 83–92, Chicago, IL, April 2011.

[4] A. Abate, M. Prandini, J. Lygeros, and S. Sastry. Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. *Automatica*, 44(11):2724–2734, November 2008.

[5] A.C. Antoulas. *Approximation of large-scale dynamical systems*. Society for Industrial Mathematics, 2005.

[6] L. Bachelier, M. Davis, A. Etheridge, and P.A. Samuelson. *Louis Bachelier's Theory of Speculation*. Princeton University Press, 2006.

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

[8] D. Barnes and D. Chu. *Introduction to Modelling for Biosciences*. Springer Verlag, 2010.

[9] S. Bashash and H.K. Fathy. Modeling and control insights into demand-side energy management through setpoint control of thermostatic loads. In *Proceedings of the 2011 American Control Conference*, pages 4546–4553, June 2011.

[10] S. Bashash and H.K. Fathy. Modeling and control of aggregate air conditioning loads for robust renewable power management. *IEEE Transactions on Control Systems Technology*, 21(4):1318–1327, July 2013.

[11] D.P. Bertsekas. Convergence of discretization procedures in dynamic programming. *IEEE Transactions on Automatic Control*, 20(3):415–419, 1975.

[12] D.P. Bertsekas and S.E. Shreve. *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific, 1996.

[13] P. Billingsley. *Probability and Measure - Third Edition*. Wiley Series in Probability and Mathematical Statistics, 1995.

[14] H.A.P. Blom and J. Lygeros (Eds.). *Stochastic Hybrid Systems: Theory and Safety Critical Applications*. Number 337 in Lecture Notes in Control and Information Sciences. Springer Verlag, Berlin Heidelberg, 2006.

[15] Mihaela Gheorghiu Bobaru, Corina S. Păsăreanu, and Dimitra Giannakopoulou. Automated assume-guarantee reasoning by abstraction refinement. In *Proceedings of the 20th international conference on Computer Aided Verification*, pages 135–148. Springer Verlag, Berlin Heidelberg, 2008.

[16] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[17] R. Brown. A brief account of microscopical observations made in the months of june, july and august, 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies. *Phil. Mag.*, 4:161–173, 1828.

[18] M.L. Bujorianu and J. Lygeros. Reachability questions in piecewise deterministic Markov processes. In O. Maler and A. Pnueli, editors, *Hybrid Systems: Computation and Control*, volume 2623 of *Lecture Notes in Computer Science*, pages 126–140. Springer Verlag, Berlin Heidelberg, 2003.

[19] R. Bundschuh, F. Hayot, and C. Jayaprakash. The role of dimerization in noise reduction of simple genetic networks. *Journal of Theoretical Biology*, 220(2):261–269, 2003.

[20] D.S. Callaway. Tapping the energy storage potential in electric loads to deliver load following and regulation, with application to wind energy. *Energy Conversion and Management*, 50(5):1389–1400, 2009.

[21] C.G. Cassandras and J. Lygeros (Eds.). *Stochastic Hybrid Systems*. Number 24 in Control Engineering. CRC Press, Boca Raton, 2006.

[22] Han-Fu Chen, P.R. Kumar, and J.H. van Schuppen. On kalman filtering for conditionally gaussian systems with random matrices. *Systems & Control Letters*, 13(5):397–404, 1989.

[23] C. Y. Chong and A.S. Debs. Statistical synthesis of power system functional load models. In *18th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, volume 18, pages 264–269, 1979.

[24] M. Dardo. *Nobel laureates and twentieth-century physics*. Cambridge University Press, Cambridge, 2004.

[25] M.H.A. Davis. *Markov Models and Optimization*. Chapman & Hall/CRC Press, London, 1993.

[26] F. Dufour and T. Prieto-Rumeau. Approximation of markov decision processes with general state space. *Journal of Mathematical Analysis and Applications*, 388(2):1254 – 1267, 2012.

[27] A. Einstein. *Concerning an Heuristic Point of View Toward the Emission and Transformation of Light*. American Journal of Physics. 1905.

[28] S. Esmaeil Zadeh Soudjani and A. Abate. Adaptive gridding for abstraction and verification of stochastic hybrid systems. In *Proceedings of the 8th International Conference on Quantitative Evaluation of Systems*, pages 59–69, Sept. 2011.

[29] S. Esmaeil Zadeh Soudjani and A. Abate. Higher-Order Approximations for Verification of Stochastic Hybrid Systems. In S. Chakraborty and M. Mukund, editors, *Automated Technology for Verification and Analysis*, volume 7561 of *Lecture Notes in Computer Science*, pages 416–434. Springer Verlag, Berlin Heidelberg, 2012.

[30] S. Esmaeil Zadeh Soudjani and A. Abate. Probabilistic invariance of mixed deterministic-stochastic dynamical systems. In *ACM Proceedings of the 15th International Conference on Hybrid Systems: Computation and Control*, pages 207–216, Beijing, PRC, April 2012.

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

[32] S. Esmaeil Zadeh Soudjani and A. Abate. Aggregation of thermostatically controlled loads by formal abstractions. In *European Control Conference*, pages 4232–4237, Zurich, Switzerland, July 2013.

[33] S. Esmaeil Zadeh Soudjani and A. Abate. Aggregation and control of populations of thermostatically controlled loads by formal abstractions. *IEEE Transactions on Control Systems Technology*, 2014. accepted.

[34] S. Esmaeil Zadeh Soudjani and A. Abate. Precise approximations of the probability distribution of a Markov process in time: an application to probabilistic invariance. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, Lecture Notes in Computer Science. Springer Verlag, Berlin Heidelberg, 2014. to appear.

[35] S. Esmaeil Zadeh Soudjani and A. Abate. Probabilistic reach-avoid computation for partially-degenerate stochastic processes. *IEEE Transactions on Automatic Control*, 59(2):528–534, 2014.

[36] S. Esmaeil Zadeh Soudjani, C. Gevaerts, and A. Abate. FAUST^2: Formal abstractions of uncountable-state stochastic processes. *arXiv preprint*, 2014. arXiv:1403.3286.

[37] A. Fehnker and F. Ivančić. Benchmarks for hybrid systems verifications. In R. Alur and G.J. Pappas, editors, *Hybrid Systems: Computation and Control*, volume 2993 of *Lecture Notes in Computer Science*, pages 326–341. Springer Verlag, Berlin Heidelberg, 2004.

[38] M. Fränzle, H. Hermanns, and T. Teige. Stochastic satisfiability modulo theory: A novel technique for the analysis of probabilistic hybrid systems. In M. Egerstedt and B. Misra, editors, *Hybrid Systems: Computation and Control*, volume 4981 of *Lecture Notes in Computer Science*, pages 172–186. Springer Verlag, Berlin Heidelberg, 2008.

[39] D.T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22:403–434, 1976.

[40] D.T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[41] D.T. Gillespie. The chemical Langevin equation. *Journal of Chemical Physics*, 113(1):297–306, 2000.

[42] M. Greenstreet. Verifying safety properties of differential equations. In *In the Proceedings of the 1996 Conference on Computer Aided Verification (CAV 96)*, pages 277–287. Springer, 1996.

[43] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[44] O. Hernández-Lerma and J. B. Lasserre. *Discrete-time Markov control processes*, volume 30 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, 1996.

[45] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In H. Hermanns and J. Palsberg, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, volume 3920 of *Lecture Notes in Computer Science*, pages 441–444. Springer Verlag, Berlin Heidelberg, 2006.

[46] P. Hokayem, D. Chatterjee, and J. Lygeros. On stochastic receding horizon control with bounded control inputs. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 6359–6364, Shanghai, PRC, December 2009.

[47] C.N. Jones, E.C. Kerrigan, and J.M. Maciejowski. On polyhedral projection and parametric programming. *Journal of Optimization Theory and Applications*, 3(137), 2008.

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

[49] M. Kamgarpour, C. Ellen, S. Esmaeil Zadeh Soudjani, S. Gerwinn, J.L. Mathieu, N. Mullner, A. Abate, D.S. Callaway, M. Fränzle, and J. Lygeros. Modeling options for demand side participation of thermostatically controlled loads. In *International Conference on Bulk Power System Dynamics and Control (IREP)*, pages 1–15, August 2013.

[50] J.-P. Katoen, M. Khattri, and I. S. Zapreev. A Markov reward model checker. In *IEEE Proceedings of the International Conference on Quantitative Evaluation of Systems*, pages 243–244, Los Alamos, CA, USA, 2005.

[51] R. Khanin and D. Higham. Chemical Master Equation and Langevin regimes for a gene transcription model. In M. Calder and S. Gilmore, editors, *Computational Methods in Systems Biology*, volume 4695 of *Lecture Notes in Computer Science*, pages 1–14. Springer Verlag, Berlin Heidelberg, 2007.

[52] P.E. Kloeden and E. Platen. *Numerical Solution of Stochastic Differential Equations*. Springer Verlag, 1992.

[53] S. Koch, J.L. Mathieu, and D.S. Callaway. Modeling and control of aggregated heterogeneous thermostatically controlled loads for ancillary services. In *17th Power Systems Computation Conference*, Stockholm, Sweden, August 2011.

[54] K. Koutsoukos and D. Riley. Computational methods for reachability analysis of stochastic hybrid systems. In J. Hespanha and A. Tiwari, editors, *Hybrid Systems: Computation and Control*, volume 3927 of *Lecture Notes in Computer Science*, pages 377–391. Springer Verlag, Berlin Heidelberg, 2006.

[55] H. Kushner and G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.

[56] H. J. Kushner and P.G. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*. Springer-Verlag, New York, 2001.

[57] M. Kvasnica, P. Grieder, and M. Baotić. Multi-parametric toolbox (MPT), 2004.

[58] R. Malhamé and C.-Y. Chong. Electric load model synthesis by diffusion approximation of a high-order hybrid-state stochastic system. *IEEE Transactions on Automatic Control*, 30(9):854–860, 1985.

[59] G. Mastroianni and G.V. Milovanovic. *Interpolation Processes: Basic Theory and Applications*. Springer Verlag, 2008.

[60] J.L. Mathieu and D.S. Callaway. State estimation and control of heterogeneous thermostatically controlled loads for load following. In *Hawaii International Conference on System Sciences*, pages 2002–2011, Hawaii, USA, 2012.

[61] J.L. Mathieu, M. Kamgarpour, J. Lygeros, and D.S. Callaway. Energy arbitrage with thermostatically controlled loads. In *Control Conference (ECC), 2013 European*, pages 2519–2526, 2013.

[62] J.L. Mathieu, S. Koch, and D.S. Callaway. State estimation and control of electric loads to manage real-time energy imbalance. *IEEE Transactions on Power Systems*, 28(1):430–440, 2013.

[63] S.P. Meyn and R.L. Tweedie. *Markov chains and stochastic stability*. Springer Verlag, 1993.

[64] R.E. Mortensen and K. P. Haggerty. A stochastic computer model for heating and cooling loads. *IEEE Transactions on Power Systems*, 3(3):1213–1219, 1988.

[65] R. Munos and A. Moore. Variable resolution discretization in optimal control. *Machine Learning*, 49:291–323, 2002.

[66] 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*, 52(8):1415–1428, 2007.

[67] M. Prandini and J. Hu. Stochastic reachability: Theory and numerical approximation. In C.G. Cassandras and J. Lygeros, editors, *Stochastic hybrid systems*, Automation and Control Engineering Series 24, pages 107–138. Taylor & Francis Group/CRC Press, 2006.

[68] F. Ramponi, D. Chatterjee, S. Summers, and J. Lygeros. On the connections between PCTL and dynamic programming. In *ACM Proceedings of the 13th International Conference on Hybrid Systems: Computation and Control*, pages 253–262, April 2010.

[69] M. Rathinam, L. R. Petzold, Y. Cao, and D.T. Gillespie. Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method. *Journal of Chemical Physics*, 119(24):12784–12794, 2003.

[70] M. Smoluchowski. Zur kinetischen theorie der brownschen molekularbewegung und der suspensionen. *Ann.˜Phys.*, 326(14):756–780, 1906.

[71] P. Somerville. Numerical computation of multivariate normal and multivariate-t over convex regions. *Journal of Computational and Graphical Statistics*, 7(4):529–544, 1998.

[72] C. Striebel. *Optimal Control of Discrete Time Stochastic Systems*. Lecture Notes in Economics and Mathematical Systems. Springer Berlin Heidelberg, 1975.

[73] T.N. Thiele. *Theory of Observations*. Charles and Edwin Layton, 1903.

[74] I. Tkachev and A. Abate. On infinite-horizon probabilistic properties and stochastic bisimulation functions. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, pages 526–531, Orlando, FL, December 2011.

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

[76] I. Tkachev and A. Abate. Characterization and computation of infinite-horizon specifications over markov processes. *Theoretical Computer Science*, 515(0):1–18, 2014.

[77] I. Tkachev, A. Mereacre, J.-P. Katoen, and A. Abate. Quantitative automata-based controller synthesis for non-autonomous stochastic hybrid systems. In *Proceedings of the 16th international conference on Hybrid Systems: Computation and Control*, HSCC '13, pages 293–302, 2013.

[78] C. Traxler. An algorithm for adaptive mesh refinement in n dimensions. *Computing*, 59:115–137, 1997.

[79] S. Widergren, C. Marinovici, T. Berliner, and A. Graves. Real-time pricing demand response in operations. In *Power and Energy Society General Meeting, 2012 IEEE*, pages 1–5, 2012.

[80] W.M. Wonham. *Linear multivariable control: A geometric approach*. Springer-Verlag, Berlin, 1979.

[81] W. Zhang, K. Kalsi, J. Fuller, M. Elizondo, and D. Chassin. Aggregate model for heterogeneous thermostatically controlled loads with demand response. In *IEEE PES General Meeting*, San Diego, CA, July 2012.

[82] W. Zhang, J. Lian, C.-Y. Chang, and K. Kalsi. Aggregated modeling and control of air conditioning loads for demand response. *IEEE Transactions on Power Systems*, 28(4):4655–4664, Nov 2013.

List of symbols

$\mathbb{N} = \{1, 2, \ldots\}$	positive natural numbers
$\mathbb{N}_0 = \mathbb{N} \cup \{0\}$	non-negative natural numbers
$\mathbb{N}_n = \{1, 2, \ldots, n\}$	finite set of positive natural numbers for $n \in \mathbb{N}$
$\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$	set of integers modulo n
$\to$	going to
$\mapsto$	maps to
k	time instant belonging to $\mathbb{N}_0$
$s(k)$	state of a Markov process at time k
$\mathcal{S}$	state space of a Markov process
$\mathcal{B}(\mathcal{S})$	Borel sigma-algebra on the set $\mathcal{S}$
$(\mathcal{S}, \mathcal{B}(\mathcal{S}))$	measurable space
$\mathbb{P}$	probability measure
Ω	sample space
$T_{\mathfrak{s}}$	stochastic kernel of a Markov process
$\mathfrak{S} = (\mathcal{S}, T_{\mathfrak{s}}(\cdot\|s))$	an autonomous Markov process
$t_{\mathfrak{s}}$	conditional density function of a Markov process
$\pi : \mathcal{B}(\mathcal{S}) \to [0, 1]$	probability measure of initial state $s(0)$
$\Omega = (\mathcal{S})^{N+1}$	product space over a finite time horizon
$\mathbb{R}^n$	n-dimensional Euclidean space
$\mathcal{A}$	safe set
N	time horizon of the safety problem
$p_{s_0}(\mathcal{A})$	solution of the safety problem for initial state s_0
$p_\pi(\mathcal{A})$	solution of the safety problem for initial measure π
$\mathbf{1}_A$	indicator function of a set A
$[a, b]$	closed interval for any $a, b \in \mathbb{R}, a \leq b$
$V_k : \mathcal{S} \to [0, 1]$	value functions for the solution of safety
2^A	power set of any set A
$\mathscr{L}(A)$	Lebesgue measure of any set A
$\doteq$	is defined as
$\mathcal{N}(\mu, \Sigma)$	multivariate Gaussian random vector with mean μ and covariance matrix Σ
$R \succ 0$	a positive definite matrix R
$R \succeq 0$	a positive semi-definite matrix R
$\mathbb{I}_n$	n-dimensional identity matrix for $n \in \mathbb{N}$
$\lambda(A)$	eigenvalue of a square matrix A

Δ	sample time of discretization
$\mathbb{E}$	expectation operator
$\triangledown$	absorbing state of a Markov chain
$\phi_\sigma(\alpha)$	Gaussian density function with zero mean and standard deviation σ
$\delta(\cdot)$	continuous Dirac delta function
$supp(f)$	support set of a function f
$A \setminus B$	the set difference of set B from A
$\triangle$	symmetric difference of two sets $A \triangle B = (A \setminus B) \cup (B \setminus A)$
Π_1, Π_2	projection operators
$\emptyset$	empty set
$\frac{\partial f}{\partial x}$	partial derivative of the function f respect to x
$\mathrm{erf}(x)$	Gauss error function
$f^{(n)}(x)$	the n^{th} derivative of the function f
$\mathfrak{S}_c = (\mathcal{S}, \mathcal{U}, T_{\mathfrak{s}}(\cdot\|s,u))$	a controlled Markov process
$\mathcal{U}$	input space of the controlled Markov process
$\boldsymbol{\mu} = (\mu_0, \mu_1, \mu_2, \ldots)$	Markov policy
$\mathcal{M}_m$	the set of all Markov policies

List of Abbreviations

i.i.d.	independent identically distributed
SHS	Stochastic Hybrid Systems
MC	Markov Chain
dtMP	discrete-time Markov Process

Summary

Formal Abstractions for Automated Verification and Synthesis of Stochastic Systems

Sadegh Esmaeil Zadeh Soudjani

Stochastic hybrid systems involve the coupling of discrete, continuous, and probabilistic phenomena, in which the composition of continuous and discrete variables captures the behavior of physical systems interacting with digital, computational devices. Because of their versatility and generality, methods for modeling, analysis, and verification of stochastic hybrid systems (SHS) have proved invaluable in a wide range of applications, including biology, smart grids, air traffic control, finance, and automotive systems. The problems of verification and of controller synthesis over SHS can be algorithmically studied using methodologies and tools developed in computer science, utilizing proper symbolic models describing the overall behaviors of the SHS.

A promising direction to address formal verification and synthesis against complex logic specifications, such as PCTL and BLTL, is the use of abstraction with finitely many states. This thesis is devoted to formal abstractions for verification and synthesis of SHS by bridging the gap between stochastic analysis, computer science, and control engineering. A SHS is first considered as a discrete-time Markov process over a general state space, then is abstracted as a finite-state Markov chain to be formally verified against the desired specification.

We generate finite abstractions of general state-space Markov processes based on the partitioning of the state space, which provide a Markov chain as an approximation of the original process. We put forward a novel adaptive and sequential gridding algorithm based on non-uniform quantization of the state space that is expected to conform to the underlying dynamics of the model and thus to mitigate the curse of dimensionality unavoidably related to the partitioning procedure.

PCTL and BLTL properties are defined over trajectories of a system. Examples of such properties are probabilistic safety and reach-avoid specifications. While the developed techniques are applicable to a wide arena of probabilistic properties, the thesis focuses on the study of the particular specification probabilistic safety or invariance, over a finite horizon.

Abstraction of *controlled* discrete-time Markov processes to Markov decision processes over finite sets of states is also studied in the thesis. The proposed abstraction scheme enables us to solve the problem of obtaining a maximally safe Markov policy for the Markov decision process and synthesize a control policy for the original model. The total error is quantified which is due to the abstraction procedure and caused by exporting the result back to the original process. The abstraction error hinges on the regularity of the stochastic kernel of the process, i.e. its Lipschitz continuity. Furthermore, this thesis extends the results in the following directions:

- Partially degenerate stochastic processes suffer from non-smooth probabilistic evolution of states. The stochastic kernel of such processes does not satisfy Lipschitz continuity assumptions which requires us to develop new techniques specialized for this class of processes. We have shown that the probabilistic invariance problem over such processes can be separated into two parts: a deterministic reachability analysis, and a probabilistic invariance problem that depends on the outcome of the first. This decomposition approach leads to computational improvements.
- The abstraction approach have leveraged piece-wise constant interpolations of the stochastic kernel of the process. We extend this approach for systems with higher degrees of smoothness in their probabilistic evolution and provide approximation methods via higher-order interpolations that are aimed at requiring less computational effort. Using higher-order interpolations (versus piece-wise constant ones) can be beneficial in terms of obtaining tighter bounds on the approximation error. Furthermore, since the approximation procedures depend on the partitioning of the state space, higher-order schemes display an interesting tradeoff between more parsimonious representations versus more complex local computation.

From the application point of view, an example of SHS is the model of thermostatically controlled loads (TCLs), which captures the evolution of temperature inside a building. This thesis proposes a new, formal two-step abstraction procedure to generate a finite stochastic dynamical model as the aggregation of the dynamics of a population of TCLs. The approach relaxes the limiting assumptions employed in the literature by providing a model based on the natural probabilistic evolution of the single TCL temperature. We also describe a dynamical model for the time evolution of the abstraction, and develop a set-point control strategy aimed at reference tracking over the total power consumption of the TCL population.

The abstraction algorithms discussed in this thesis have been implemented as a MATLAB tool FAUST^2 (abbreviation for "Formal Abstractions of Uncountable-STate STochastic processes"). The software is freely available for download at `http://sourceforge.net/projects/faust2/`.

Samenvatting

Formele Abstracties voor Automatische Verificatie en Synthese van Stochastische Systemen

Sadegh Esmaeil Zadeh Soudjani

Stochastische hybride systemen beschrijven de koppeling van discrete, continue en probabilistische fenomenen, waarbij de combinatie van continue en discrete variabelen bijvoorbeeld het gedrag beschrijft van fysische systemen die interageren met digitale rekenapparatuur. Vanwege hun veelzijdigheid en algemene toepasbaarheid hebben methoden voor het modelleren, analyseren en verifiëren van stochastische hybride systemen (SHS) bewezen van onschatbare waarde te zijn in een breed scala van wetenschapsgebieden waaronder de biologie, smart grids, regeling van vliegverkeer, financiën en automobielsystemen. De problemen van verificatie voor SHS en van het samenstellen van regelwetten voor SHS kunnen vanuit een algorithmisch oogpunt bestudeerd worden met methoden en met hulpmiddelen die ontwikkeld zijn binnen de informatica, gebruikmakend van passende symbolische modellen voor het gedrag van SHS.

Een veelbelovende onderzoeksrichting is om het gedrag van SHS te vergelijken met complexe specificaties door gebruik te maken van logica's, zoals PCTL en BLTL. Dit kan door middel van een abstractie van een SHS naar een automaat met een eindig aantal toestanden. Dit proefschrift is gewijd aan formele abstracties voor verificatie en synthese van SHS waarmee het gat wordt gedicht tussen stochastische analyse, informatica en regeltechniek. Allereerst wordt een SHS benaderd als een discrete-tijd Markov-proces met een algemene toestandsruimte. Daarna wordt het geabstraheerd tot een Markov-keten met een eindig aantal toestanden om formeel geverifiëerd te worden ten opzichte van de gewenste specificatie.

We genereren eindige abstracties van Markov-processen met een algemene toestandruimte op basis van een partitie van de toestandsruimte. Dit geeft een Markov-keten als benadering van het originele proces. We laten een innovatief adaptief en recursief rooster-algoritme zien dat is gebaseerd op een niet-uniforme verdeling van de toestandsruimte en dat naar verwachting voldoet aan de onderliggende dynamica van het model. Deze aanpak verzwakt de problemen met de complexiteit die onlosmakelijk verbonden zijn met de partitieprocedure.

PCTL- en BLTL-eigenschappen worden gedefinieerd op de ruimte van systeemtrajectoriën. Voorbeelden van zulke eigenschappen zijn probabilistische veiligheidsspecificaties en specificaties om een bepaalde deelverzameling van de toestandsruimte juist wel of juist nooit te bereiken. Alhoewel de ontwikkelde technieken toepasbaar zijn op een breed scala aan probabilistische eigenschappen, is dit proefschrift in het bijzonder gericht op de probabilistische veiligheids- of invariantiespecificatie over een eindige horizon.

Hiernaast wordt in dit proefschrift de abstractie van een discrete-tijd stochastisch regelsysteem naar een eindige-toestands Markov-beslissingsproces behandeld. Het voorgestelde abstractieschema maakt het mogelijk om een maximaal veilige Markov regelwet voor het Markov-beslissingsproces te verkrijgen en om een regelwet te bepalen voor het oorspronkelijke systeem. De benaderingsfout in de kosten als gevolg van de abstractieprocedure en als gevolg van het converteren van de regelwet naar een regelwet voor het oorspronkelijke systeem wordt in een formule samengevat. De benaderingsfout van de abstractie is afhankelijk van de regulariteit van de stochastische kern (de kansdichtheid van de transitiemaat) van het proces, namelijk diens graad van Lipschitz-continuïteit. Daarnaast breidt dit proefschrift resultaten uit in de volgende richtingen:

- Stochastische processen die gedeeltelijk gedegenereerd zijn, lijden aan niet-gladde (*non-smooth*) toestandstransitiefuncties. De stochastische kern van zulke processen voldoet niet aan de Lipschitz-continuiteitsvoorwaarden waardoor het nodig is nieuwe technieken te ontwikkelen die specifiek zijn voor deze klasse van processen. We laten zien dat het probabilistische invariantieprobleem voor zulke processen opgesplitst kan worden in twee delen: (1) een deterministische bereikbaarheidsanalyse en (2) een probabilistisch invariantieprobleem dat afhankelijk is van de uitkomst van de bereikbaarheidsanalyse. Deze decompositie-aanpak leidt tot rekenkundige verbeteringen.

- De abstractiemethode maakt kundig gebruik van stuksgewijs-constante interpolaties van de stochastische kern van het proces. We breiden deze methode uit naar systemen met hoge graden van gladheid in hun probabilistische evolutie en stellen benaderingsmethoden voor via hoge-orde interpolaties die minder rekenkracht vereisen. Het gebruik van hoge-orde interpolaties in plaats van stuksgewijs-constante interpolaties kan nuttig zijn voor het behalen van nauwe grenzen op de benaderingsfout. Omdat de benaderingsmethoden afhankelijk zijn van de partitie van de toestandsruimte, laten rekenschema's van hoge orde daarnaast een interessante wisselwerking zien tussen een meer beperkte voorstelling en een meer gecompliceerde lokale berekening.

Een voorbeeld van een SHS is het modelleren van thermostatisch-geregelde eenheden (termostaticaly-controlled loads (TCLs)), die de veranderingen van de temperatuur binnen een gebouw beschrijven. Dit proefschrift stelt een nieuwe formele abstractieprocedure met twee stappen voor om een eindig stochastisch systeem te genereren door de samenvoeging van de dynamica van een populatie van TCLs. De methode verzwakt de beperkende voorwaarden die in de literatuur

worden gebruikt door een model voor te stellen dat gebaseerd is op de natuurlijke probabilistische veranderingen van een enkele TCL-temperatuur. We beschrijven tevens een dynamisch model voor de tijdsevolutie van de abstractie en ontwikkelen een regelwet met een streefpunt gericht op het volgen van een referentiewaarde voor het totale energieverbruik van de TCL-populatie.

De abstractie-algoritmes die in dit proefschrift behandeld worden, zijn geïmplementeerd als MATLAB-pakket genaamd FAUST^2 (afkorting voor "Formal Abstractions of Uncountable-STate STochastic processes"). Deze software kan men gratis downloaden van het adres `http://sourceforge.net/projects/faust2/`.

Curriculum Vitae

Sadegh Esmaeil Zadeh Soudjani was born on 16 September 1984 in Fasa and grew up in Shiraz, Iran. After graduating from secondary school in 2002, he started the bachelor's program in Electrical Engineering at the University of Tehran. Due to his interest in mathematics and being an exceptionally talented student, he got permission to simultaneously study a second bachelor's program in Pure Mathematics at the same school. He graduated in both programs with honors in 2007.

Becoming fascinated with the structural and systematic analysis of dynamical models for physical systems in Control Engineering community, he pursued this major for his master's study at the University of Tehran and graduated with honors in 2009. His M.Sc. research work, advised by Professor F.R. Salmasi, was on the analysis and development of nonlinear observability problem in electrical drive systems. To increase his knowledge and experience in real industrial large-scale applications of control theory, he worked on the project "Revamping and Renovation of Instrumentation and Control Systems of Sarcheshmeh Coper Complex" (the second largest copper deposit worldwide) during the years 2006-2009. Having excellent teaching skills, he was active in lecturing mathematics and control theory courses to undergraduate students in the next year.

In September 2010, Sadegh moved to the Netherlands to start his doctoral research at Delft Center for Systems and Control under the supervision of Professor A. Abate. In his research, he was involved in the European project "MoVeS" on modeling, verification, and control of complex systems. He has also worked on the project "AMBI" on advanced methods for building diagnostics and maintenance. During his PhD, he received the DISC certificate for following the graduate course program at the Dutch Institute of Systems and Control. His research is focused on developing efficient formal abstraction methods for verification and synthesis of Markov processes over general state spaces, applied to smart grids and biological systems. The high-level goal of his research is to bridge the gap between Computer Science and Control Engineering by exporting the concepts used in Computer Science to the analysis and synthesis of dynamical models.

List of Publications

Journal papers

S. Esmaeil Zadeh Soudjani and A. Abate, "Aggregation and Control of Populations of Thermostatically Controlled Loads by Formal Abstractions," *Transactions on Control Systems Technology*, June 2014, accepted.

S. Esmaeil Zadeh Soudjani and A. Abate, "Probabilistic Reach-Avoid Computation for Partially Degenerate Stochastic Processes," *IEEE Transactions on Automatic Control*, pp. 528–534, February 2014.

S. Esmaeil Zadeh Soudjani and A. Abate, "Adaptive and Sequential Gridding for the Abstraction and Verification of Stochastic Processes," *SIAM Journal on Applied Dynamical Systems*, pp. 921–956, June 2013.

Book chapters, springer's LNCS

S. Esmaeil Zadeh Soudjani, S. Gerwinn, C. Ellen, M. Fränzle, and A. Abate, "Formal Synthesis and Validation of Inhomogeneous Thermostatically Controlled Loads," *Quantitative Evaluation of SysTems (QEST)*, pp. 57–73, September 2014.

[1]D. Adzkiya, **S. Esmaeil Zadeh Soudjani**, and A. Abate, "Finite Abstractions of Stochastic Max–Plus–Linear Systems," *Quantitative Evaluation of SysTems (QEST)*, pp. 74–89, September 2014.

S. Esmaeil Zadeh Soudjani and A. Abate, "Precise Approximations of the Probability Distribution of a Markov Process in Time: an Application to Probabilistic Invariance," *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pp. 547–561, April 2014.

S. Esmaeil Zadeh Soudjani and A. Abate, "Higher-Order Approximations for Verification of Stochastic Hybrid Systems," *Automated Technology for Verification and Analysis (ATVA)*, pp. 416–434, October 2012.

[1]The first two authors have equally contributed to this work.

Tool papers

S. Esmaeil Zadeh Soudjani and C. Gevaerts and A. Abate, "FAUST^2: Formal Abstractions of Uncountable–STate STochastic processes," arXiv:1403.3286, March 2014. Freely available at http://sourceforge.net/projects/faust2/.

Conference proceedings

M. Kamgarpour, C. Ellen, **S. Esmaeil Zadeh Soudjani**, S. Gerwinn, J.L. Mathieu, N. Mullner, A. Abate, D.S. Callaway, M. Fränzle, and J. Lygeros, "Modeling Options for Demand Side Participation of Thermostatically Controlled Loads," *Bulk Power System Dynamics and Control (IREP)*, Rethymnon, Greece, pp. 1–15, August 2013.

S. Esmaeil Zadeh Soudjani and A. Abate, "Aggregation of Thermostatically Controlled Loads by Formal Abstractions," *European Control Conference (ECC)*, Zurich, Switzerland, pp. 4232–4237, July 2013.

S. Esmaeil Zadeh Soudjani and A. Abate, "Probabilistic Invariance of Mixed Deterministic-Stochastic Dynamical Systems," 15^{th} *International Conference on Hybrid Systems: Computation and Control (HSCC)*, Beijing, China, pp. 207–216, April 2012.

S. Esmaeil Zadeh Soudjani and A. Abate, "Adaptive Gridding for Abstraction and Verifications of Stochastic Hybrid Systems," 8^{th} *International Conference on Quantitative Evaluation of SysTems (QEST)*, Aachen, Germany, pp. 59–68, September 2011.