Formal Controller Synthesis for Dynamical Systems: Decidability & Scalability

PhD Thesis

Thesis approved by
the Department of Computer Science
University of Kaiserslautern-Landau
for the award of the Doctoral Degree
Doctor of Natural Sciences (Dr. rer. nat.)

from

Mahmoud Salamati

Date of Defense:	June 20th, 2024
Dean:	Prof. Dr. Christoph Garth
Reviewer:	Prof. Dr. Rupak Majumdar
Reviewer:	Prof. Dr. Necmiye Ozay
Reviewer:	Dr. Sadegh Soudjani
PhD Advisor:	Prof. Dr. Rupak Majumdar

DE-386

Abstract

In cyber-physical systems research an important challenge is the synthesis of reliable controllers with respect to a general temporal specification. The synthesized controller must provide formal guarantees against different sources of disturbance, such as measurement noise and mismatch between the dynamics of the physical system and its model. By synthesizing correct-by-construction controllers for complex dynamical systems, we can enable a large number of exciting applications in various domains, including autonomous vehicle industry, energy systems and healthcare.

In this thesis, we plan to study controller synthesis for several different classes of dynamical systems. For some specific classes of systems, we provide sound and complete decision procedures. For general nonlinear dynamical systems for which undecidability of basic synthesis problems is proven, we propose sound but scalable technique that can be applied to the real-world dynamical systems.

First, we consider continuous dynamical systems with bounded disturbances. The underlying dynamics for every continuous dynamical system can—in the bounded adversarial setting—be modeled by a (non-linear) differential inclusion system, provided that a bound over the range of disturbances is known. A promising approach to tackle the continuous nature of the state space is to use abstraction-based controller design (ABCD) schemes. The controller designed by the ABCD scheme is described as being formal due to the guarantees on satisfaction of the specification by the original system in closed loop with the designed controller. In the first part of the thesis, we present methods to improve applicability of ABCD by proposing (1) a data-driven scheme for relaxing the requirement of having analytical model, (2) a neural abstraction method to reduce memory requirements of both synthesis and deployment, and (3) a scalable method for solving reach-avoid problems for multi-agent systems.

Second, we study continuous-time Markov decision processes (CTMDPs), which are a widely used model for continuous-time stochastic systems. A fundamental problem in the analysis of CTMDPs is time-bounded reachability, which asks whether we can synthesize a control policy with which the probability of reaching a target set of states within a finite horizon is greater than a given threshold. Time-bounded reachability is the core technical problem for model checking stochastic temporal logics such as Continuous Stochastic Logic, and having efficient implementations of time-bounded reachability is crucial for scaling up formal analysis of CTMDPs. Existing work either considers time-abstract policies or focuses on numerical approximation. Despite the importance of this problem, its decidability is yet open. Moreover, the existing discretization-based approximation methods are not scalable for CTMDPs with a large number of states. In the second part of the thesis, we study the time-bounded reachability problem for CTMDPs, and propose (1) a conditional decidability result, and (2) a systematic method for improving scalability of numerical approximation methods.

Finally, we study the class of linear dynamical systems, which are fundamental models in many different domains of science and engineering. In the third part of this thesis, we consider several reachability-related problems for linear dynamical systems, and propose (1) a hardness result for point-to-point reachability of linear dynamical systems with hyper-rectangular control sets, and (2) decidability of pseudo-reachability for hyperplane target sets.

Contents

Acknowledgments

This thesis is the result of numerous fruitful collaborations, and, perhaps even more significantly, the unwavering support I have received from my loved ones. I will make every effort to express my sincere gratitude to all these individuals, although it may be challenging to do so comprehensively. Therefore, I extend my heartfelt apologies in advance if I inadvertently omit anyone in this brief note.

Foremost, I wish to convey my profound gratitude to my supervisor, Rupak Majumdar. Throughout my Ph.D. journey, he consistently extended his invaluable support, wisdom, boundless enthusiasm, and patience. One of the most rewarding aspects of working under his guidance was his unwavering commitment to the pursuit of genuine scientific inquiry over a mere quest for additional publications. Beyond the realm of scientific mentorship, he also provided invaluable assistance in navigating the path to securing future employment.

The other person to whom I owe a tremendous debt of gratitude is Sadegh Soudjani. If I had not met him, it is quite probable that I would have not found my way to MPI. Furthermore, Sadegh served as my close and dedicated supervisor from the outset, and it is with him that I invested the greatest number of hours in collaborative brainstorming throughout my Ph.D. journey.

I was fortunate to have Eva Darulova, James Worrell, and Joël Ouaknine as mentors and collaborators. Their contributions were instrumental in guiding the direction of my PhD.

I had the privilege to have excellent student collaborators during the course of my PhD with whom I learned a lot. I would like to thank Ben Wooding, Ghazal Ebrahimi, Julian D'Costa, Kaushik Mallik, Mehrdad Zareian, Milad Kazemi, Rocco Salvia and Toghrol Karimov.

Huge thanks to the amazing administrative staffs of MPI in Kaiserslautern, including Corinna, Geraldine, Mouna, Roslyn, Susanne, and Vera, and the super-efficient IT staffs, including Pascal, and Tobias, who simplified non-scientific aspects effortlessly, ensuring they never became sources of stress. Many thanks to Rose, who taught us crucial skills for writing beautiful scientific texts and delivering fluent talks. I am grateful to Mary-Lou, who not only helped wading through the administrative requirements of the university, but also showered encouragement on every occasion we met.

The six years that I spent in Kaiserslautern have been the most joyful six years of my life so far. The credit is owed to the wonderful friends I've met. Alongside them, I've invested my leisure time, crafting cherished memories through adventures in travel, sports, dining, and delightful experiences. In alphabetical order, they include: Abir, Alexandra, Aman, Amir, Andrea, Andreea, Ana, Anne, Anthony, Arabinda, Arash, Aristotelis, Arka, Ava, Arpan, Ashwani, Azalea, Bala, Burcu, Cedric, Chris, Clothilde, Damien, Daniel, Dmitry, Ehsan, Eirene, Eleni, Ezgi, Felix, Filip, Hasan, Germano, Giovanni, Iason, Irmak, Isa, Ivan F., Ivan G., Ivi, Javad, James, Johannes, Kata, Kaushik, Klara, Kyle, Laura, Leo, Lia, Lovro, Mahdi, Manohar, Manuel, Mareike, Marco M., Marco P., Marcus, Marko D., Marko H., Maryam, Mateusz, Maziar, Mehrdad, Mia, Michalis, Milad, Mitra, Moses, Mohammad Hossein, Munko, Murat, Nastaran, Natsuki, Nazerke, Nina, Numair, Ori,

Pascal, Pasha, Pavel, Rajarshi, Ram, Rayna, Rosa, Sadegh, Sathiya, Satya, Seungeon, Simin, Srinidhi, Soham, Stanly, Stratis, Suhas, Utkarsh, Vinayak, Xuan, and Yunjun. I would like to particularly mention Aman, Ehsan, Javad, Maryam, Milad, Kaushik, Mehrdad, Parinaz with whom I spent most of my time.

I feel extremely lucky that I met Zohreh during the course of my PhD to whom I married later. She has offered unconditional help and support throughout, and with her I had the chance to create the best memories during my time in Germany.

Finally, I am eternally grateful to my parents for making me who I am, and to my brother for his love and support all along. They are my inner strengths.

Mahmoud Salamati,
July 4, 2024

1 Introduction

Recent technological advancements in the field of cyber-physical systems have been nothing short of revolutionary. Autonomous vehicles are becoming increasingly sophisticated, with self-driving cars and trucks undergoing extensive testing and deployment in some regions. Smart cities are utilizing cyber-physical systems to optimize traffic management, reduce energy consumption, and enhance overall urban living. In healthcare, wearable devices and remote monitoring systems are revolutionizing patient care, allowing for more personalized and timely interventions. Additionally, advancements in robotics and automation are transforming industries, from manufacturing and logistics to agriculture and healthcare, with robots working alongside humans in collaborative environments. These innovations are not only improving efficiency and productivity but also raising important questions about safety, ethics, and cybersecurity in the rapidly evolving landscape of cyber-physical systems. As technology continues to advance, the potential for further breakthroughs and their profound impact on our daily lives remains both exciting and challenging.

The importance of safety design in cyber-physical systems in the industry is underscored by a series of catastrophic incidents, including Tesla Autopilot crashes, the Uber self-driving car accident, the Volkswagen factory incident where a robot killed a worker, and the Boeing 737 Max crashes, each serving as stark reminders of the catastrophic consequences when safety specifications are not rigorously met in the design of controllers for cyber-physical systems. These events have prompted a reevaluation of how cyber-physical systems are developed and deployed, underlining the need for rigorous formal design, robust control systems, and comprehensive safety specifications. Therefore, in the age of automation and interconnected systems, safety must be a foundational pillar, safeguarding both lives and the integrity of our industries.

Design of reliable controllers for cyber-physical systems is a very challenging due to their complex dynamics—defined over a *continuous* state space—and also specifications that can only be expressed using *natural-like* specifications—which are beyond the scope of classical control. As a result of these challenges, majority of the existing controller design methods for cyber-physical systems are non-systematic and hence do not provide any useful formal guarantees.

The main objective of this thesis is to propose new methods that broaden the scope of formal controller design. To that end, we consider different classes of dynamics. For the more general classes of dynamical systems, we propose *sound* methods to enhance the *scalability* of the controller design. We notice that a sound controller design method may miss some solutions; therefore, we also consider less general classes of dynamical systems

for which we provide *sound and complete* design methods.

In the sequel, we provide a brief description for the list of main challenges we addressed for each of the considered classes of control system in this thesis.

1.1. Continuous-Space Systems with Adversarial Disturbances

Majority of cyber-physical systems can be modeled as *continuous control systems*, whose state can evolve continuously in a *compact* Cartesian space over continuous time horizons. To design controllers for safety critical systems, we always need to take the effect of model uncertainties into account. In many circumstances, we have no information about the exact probability distribution over the range of model uncertainties ,but a worst-case estimation of the compact domain of uncertainties is known. This setting, which shall be referred to as *adversarial disturbances* setting, is largely studied. *Continuous control systems with adversarial disturbances* are the first class of control systems which will be studied in this thesis. To synthesize *correct-by-construction* controllers for continuous control systems with bounded adversarial disturbances, one popular technique is to use methods from the field of *reactive synthesis*. To that end, one needs to first discretize time, using an appropriate *sampling time*, and state and input spaces, using uniformly-sized rectangular partition elements, to compute a finite *transition system* whose behavior is connected to the original system via some appropriate behavioral relation. The derived system with the finite set of states and inputs is referred to as *abstraction* and the corresponding method to synthesize correct-by-construction controllers using the system's abstraction is called *abstraction-based controller design* (ABCD) [172, 22, 118, 161]. ABCD works very well for low-dimensional systems. However, increasing the dimension of state space results in an exponential increase in the size of the abstraction and leads into very long run-times and memory blow-ups. In the first part of this proposed thesis, we consider the general class of continuous control systems, in the adversarial disturbance setting and propose methods to improve the scalability of ABCD from both memory and time complexity perspectives, as this is the main challenge for the application of ABCD. Below, we mention a list of the challenges we addressed for this class of systems.

Extending ABCD to systems with unknown model. ABCD schemes generally rely on a precise mathematical model of the system. Such exact mathematical description is not available for most of real-world control systems. A promising approach to tackle this issue is to develop data-driven controller synthesis schemes with appropriate formal (probabilistic) guarantees. We provide a data-driven method to synthesize controllers formally, for general temporal specifications over unknown control systems. We compute the growth bound of the system by using a finite number of sample trajectories, and construct the abstraction based on the computed growth bound. We also provide a lower-bound over the sample complexity of our method. By several experiments, we show that our method can be used to synthesize formally guaranteed controllers for unknown control systems.

Reducing memory requirements for ABCD. The computed abstractions of high-dimensional dynamical systems are frequently characterized by a substantial size, posing a significant bottleneck when it comes to storage in memory through conventional abstraction-based synthesis methods. We propose an on-the-fly memory-efficient synthesis method, which relies on expressive power of *neural networks* for representing the system's abstraction and also the synthesized controller. We train neural networks, which take a state-input pair and generate the characterization of tight over-approximations for the corresponding sets of successors and predecessors. Once these neural networks are trained, we plug them into our on-the-fly synthesis algorithm to solve the given instance of reachability problem. Therefore, we can significantly lower the space complexity by avoiding the need for storage of the full transition system into RAM. Further, we propose a similar method that computes compressed neural representations for the controller and hence reduces the memory requirements of the deployment phase significantly.

Extending ABCD to multi-agent systems. We consider the decentralized controller synthesis problem for multi-agent systems with global reach-avoid specifications. Each agent is modeled as a nonlinear dynamical system with disturbance. The objective is to synthesize *local* feedback controllers that guarantee that the overall multi-agent system meets the global specification under the influence of disturbances. Existing techniques based on planning or trajectory optimization usually ignore the effects of disturbance and produce open-loop *nominal* trajectories which may not suffice in the presence of disturbances. Techniques based on formal synthesis that guarantee satisfaction of temporal specifications do not scale as the number of agents increase. We address these limitations by proposing a two-level solution approach that combines fast global nominal trajectory generation and local application of formal synthesis. At the top level, we ignore the effect of disturbances and obtain a joint open-loop plan for the system using a fast trajectory optimizer. At the lower level, we use abstraction-based controller design to synthesize a set of decentralized feedback controllers that track the high level plan against worst-case disturbances, thus ensuring satisfaction of the global specification.

1.2. Continuous-Time Markov Decision Processes

In cases wherein a distribution over the model uncertainty is attainable, one can use the existing stochastic models for describing the underlying dynamics. Markov decision processes (MDPs) are a very well-known framework for modeling discrete-time stochastic control systems. Despite their popularity, there are *continuous-time* systems, whose behavior cannot be captured by discrete-time MDPs and we need to use continuous-time Markov decision processes (CTMDPs) instead. In case that there is no control, CTMDPs reduce to continuous-time Markov chains (CTMCs). Examples of such systems include queuing systems and virus spread networks among others. In general, CTMDPs can be derived from a continuous control stochastic system through appropriate abstraction of the state space into finite number of cells and computing the probabilities for jumping from one cell into the others as a continuous function of time. In this way, we can connect the first two classes of systems studied in this thesis, i.e., continuous control systems

with adversarial disturbances and CTMDPs. For both CTMCs and CTMDPs, the core problem for model checking stochastic temporal logics such as Continuous Stochastic Logic [9, 14] is time-bounded reachability. In fact, even decidability of this problem is unknown for CTMDPs. Furthermore, most of the existing methods, which are aimed at *approximating* the value of time-bounded reachability for CTMDPs, do not scale well for reasonably large CTMDPs. In the second part of the thesis, we consider the time-bounded reachability problem in CTMCs and CTMDPs and address it from both decidability and scalability perspectives. Below, we list a number of challenges, that we have addressed in the thesis, concerning time-bounded reachability of CTMDPs.

Decidability. We consider the time-bounded reachability problem for continuous-time Markov decision processes and show that this problem is decidable subject to *Schanuel's conjecture*. Our decision procedure relies on the structure of optimal policies and the conditional decidability (under Schanuel's conjecture) of the theory of reals extended with exponential and trigonometric functions over bounded domains. We further show that any unconditional decidability result would imply unconditional decidability of the bounded continuous Skolem problem, or equivalently, the problem of checking if an *exponential polynomial* has a *non-tangential zero* in a bounded interval. We note that the latter problems are also decidable subject to Schanuel's conjecture but finding unconditional decision procedures remain longstanding open problems.

Scalability. As already mentioned, time-bounded reachability is a fundamental problem in model checking continuous-time Markov chains (CTMCs) and Markov decision processes (CTMDPs) for specifications in continuous stochastic logics. It can be computed by numerically solving a characteristic linear dynamical system but the procedure is computationally expensive. We take a control-theoretic approach and propose a reduction technique that finds another dynamical system of lower dimension, such that numerically solving the reduced dynamical system provides an approximation to the solution of the original system with guaranteed error bounds. Our technique generalizes lumpability (or probabilistic bisimulation) to a quantitative setting. Our main result is a Lyapunov function characterization of the difference in the trajectories of the two dynamics that depends on the initial mismatch and exponentially decreases over time. In particular, the Lyapunov function enables us to compute an error bound between the two dynamics as well as a convergence rate. Finally, we show that the search for the reduced dynamics can be computed in polynomial time using a Schur decomposition of the transition matrix. This enables us to efficiently solve the reduced dynamical system by computing the exponential of an upper-triangular matrix characterizing the reduced dynamics. For CTMDPs, we generalize our approach using piecewise quadratic Lyapunov functions for switched affine dynamical systems. We synthesize a policy for the CTMDP via its reduced-order switched system that guarantees the time-bounded reachability probability lies above a threshold. We provide error bounds that depend on the minimum dwell time of the policy. We demonstrate the technique on examples from queuing networks, for which lumpability does not produce any state space reduction but our technique synthesizes policies using reduced version of the model.

1.3. Linear Dynamical Systems

Linear dynamical systems are fundamental models in many different domains of science and engineering, and the computability and complexity of decision problems for linear dynamical systems are of both theoretical and practical interest. Therefore, we take a closer look into the class of linear dynamical systems for which certain problems, including point-to-point reachability, are known to be decidable in the absence of control. Also, there are many seemingly simple problems for linear dynamical systems whose decidability are yet open after decades of continuous effort. Skolem (point-to-hyperplane reachability) and Positivity (point-to half-space-reachability) are two such well-known problems.

In the fourth part of this thesis, we first show that point-to-point reachability problem for linear dynamical system with hyper-rectangular control set is at least as hard as the positivity problem for linear dynamical systems. Therefore, we consider an important related problem, known as pseudo-reachability for linear dynamical systems. Intuitively, pseudo-reachability problem asks whether a target is reachable under every hyper-cubic control set with non-zero volume. We show that for both point-to-point and point-to-hyperplane cases, the pseudo-reachability problem is decidable for linear dynamical systems.

Hardness of the reachability problem for linear control systems. A very natural question would be to ask about the hardness of (point-to-point) reachability problem for a linear dynamical system with hyper-cubic control set. We show that this (restricted) version of reachability problem is indeed hard. Motivated by this hardness result, we turn our focus into investigation of other reachability-related problems for linear dynamical systems.

Decidability of pseudo-reachability for hyperplane target sets. Pseudo-orbits are generalizations of orbits in the topological theory of dynamical systems. We study the *pseudo-orbit problem*, whether a state belongs to the pseudo-orbit of another state, and the *pseudo-Skolem problem*, whether a *hyperplane* is reachable by an ϵ-pseudo-orbit for *every* ϵ. These problems are analogous to the well-studied orbit problem and Skolem problem on unperturbed dynamical systems. Our main results show that the pseudo-orbit problem is decidable in polynomial time and, surprisingly, the Skolem problem on pseudo-orbits is also decidable. The former extends the seminal result of Kannan and Lipton from orbits to pseudo-orbits. The latter is in contrast to the Skolem problem for linear dynamical systems, which remains open for proper orbits.

1.4. Outline of the Thesis

This document is organized as follows. In the subsequent chapters, we describe our contributions over the three classes of control systems considered in this dissertation. Chapter 2 is dedicated to overcoming three main bottlenecks of ABCD, with the goal of enhancing the scalability of controller synthesis for continuous control systems with adversarial disturbances and infinite-horizon temporal specifications. First, in Section 2.2 we propose a method that extends applicability of ABCD to systems with unknown

dynamics. Section 2.3 discusses a technique that significantly lowers the memory requirements of ABCD. In Section 2.4, we explore the extension of ABCD into multi-agent settings. Chapter 3 focuses on the time-bounded reachability problem for CTMDPs. In Section 3.2, we show a conditional decidability result, and in Section 3.3, we propose a method for enhancing scalability of discretization-based approximation methods. Finally, In Chapter 4, we consider the class of linear dynamical systems and study the decidability of reachability-related specifications for them from a decidability perspective.

1.5. List of Publications

The material in the thesis has been published in the following papers:

1. "Approximate Time Bounded Reachability for CTMCs and CTMDPs: A Lyapunov Approach" with Sadegh Soudjani and Rupak Majumdar, 15th International Conference on Quantitative Evaluation of SysTems (QEST'2018).

2. "A Lyapunov Approach for Time-Bounded Reachability of CTMCs and CTMDPs" with Sadegh Soudjani and Rupak Majumdar, ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS).

3. "On Decidability of Time-Bounded Reachability in CTMDPs" with Rupak Majumdar and Sadegh Soudjani, 47th International Colloquium on Automata, Languages, and Programming (ICALP'2020).

4. "Symbolic reach-avoid control of multi-agent systems" with Rupak Majumdar, Kaushik Mallik, Sadegh Soudjani and Mehrdad Zareian, ACM/IEEE 12th International Conference on Cyber-Physical Systems (ICCPS'2021).

5. "The Pseudo-Skolem Problem is Decidable" with Julian D'Costa, Toghrul Karimov, Rupak Majumdar, Joël Ouaknine, Sadegh Soudjani and James Worrell, 46th International Symposium on Mathematical Foundations of Computer Science (MFCS'2022).

6. "The Pseudo-Reachability Problem for Diagonalisable Linear Dynamical Systems" with Julian D'Costa, Toghrul Karimov, Rupak Majumdar, Joël Ouaknine and James Worrell, 47th International Symposium on Mathematical Foundations of Computer Science (MFCS'2023).

7. "Neural Abstraction-Based Controller Synthesis and Deployment" with Rupak Majumdar and Sadegh Soudjani, ACM Transactions on Embedded Computing Systems (TECS).

8. "Data-Driven Abstraction-Based Control Synthesis" with Milad Kazemi, Rupak Majumdar, Sadegh Soudjani and Ben Wooding, **submitted to** the Elsevier Journal on Nonlinear Analysis: Hybrid Systems (NAHS).

2

Abstraction-Based Controller Design

One of the major objectives in the design of safety-critical systems is to ensure their safe operation while satisfying high-level requirements. Examples of safety-critical cyber-physical systems include power grids, autonomous vehicles, traffic control, and battery-powered medical devices. Automatic design of controllers for such systems that can fulfill the given requirements has received significant attention recently. These systems can be represented as control systems with continuous state spaces and may be affected by adversarial bounded disturbances. Within these continuous spaces, it is challenging to leverage automated control synthesis methods that provide satisfaction guarantees for high-level specifications, such as those expressed in Linear Temporal Logic [13, 22, 172, 67].

A common approach to tackle the continuous nature of the state space is to use abstraction-based controller design (ABCD) schemes [172, 22, 118, 161]. The first step in the ABCD scheme is to compute a finite abstraction by discretizing the state and action spaces. Finite abstractions are connected to the original system via an appropriate behavioral relation such as feedback refinement relations or alternating bisimulation relations [146, 172]. Under such behavioral relations, trajectories of the abstraction are related to the ones of the original system. Therefore, a controller designed for the simpler finite abstract system can be refined to a controller for the original system. The controller designed by the ABCD scheme is described as being formal due to the guarantees on satisfaction of the specification by the original system in closed loop with the designed controller.

In the rest of this chapter, we take ABCD as a sound, but not complete method for synthesizing formally guaranteed controllers for continuous control systems with bounded adversarial disturbance, and address its intrinsic shortcomings. First, in Section 2.1, we describe the notations used in this chapter. In Section 2.2, our proposed data-driven method for learning abstractions is depicted. In Section 2.3, we describe how neural networks can be used in order to mitigate huge memory requirements of data-driven ABCD. Finally, in Section 2.4, we describe a method to extend ABCD to multi-agent systems with joint reach-avoid specifications.

2.1. Preliminaries

We denote the set of natural, integer, real, positive real, and non-negative real numbers by $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$, $\mathbb{R}_{>0}$, and $\mathbb{R}_{\geq 0}$, respectively. The set of natural numbers including zero is denoted by $\mathbb{N}_{\geq 0}$. We use superscript $n > 0$ with these sets to denote the Cartesian product of n copies of these sets. The power set of a set A is denoted by 2^A and includes all the subsets of A. For any $x, y \in \mathbb{R}^n$ with $x = (x(1), \ldots, x(n))$ and $y = (y(1), \ldots, y(n))$, and a relational symbol $\triangleright \in \{\leq, <, =, >, \geq\}$, we write $x \triangleright y$ if $x(i) \triangleright y(i)$ for every $i \in \{1, 2, \ldots, n\}$. A matrix $M \in \mathbb{R}^{n \times n}$ is said to be non-negative if all of its entries are non-negative. The operator $|\cdot|$ is used to denote both the absolute value of a vector and cardinality of a set, depending on the type of the operand. We use the operators $\|\cdot\|_2$ and $\|\cdot\|_\infty$ to denote the two norm, and the infinity norm, respectively. We use the notations $\Omega_\varepsilon^{(2)}(c) \coloneqq \{x \in \mathbb{R}^n \mid \|x - c\|_2 \leq \varepsilon\}$ and $\Omega_\varepsilon^{(\infty)}(c) \coloneqq \{x \in \mathbb{R}^n \mid \|x - c\|_\infty \leq \varepsilon\}$ to denote the ball with respect to, respectively, the two norm and the infinity norm centered at $c \in \mathbb{R}^n$ with radius $\varepsilon \in \mathbb{R}^n_{>0}$. We consider a probability space $(\Omega, \mathcal{F}_\Omega, \mathbb{P}_\Omega)$, where Ω is the sample space, $\mathcal{F}_\Omega$ is a sigma-algebra on Ω comprising its subsets as events, and $\mathbb{P}_\Omega$ is a probability measure that assigns probabilities to events.

For a vector $a \in \mathbb{R}^n$, we denote its i^{th} component, element-wise absolute value, ℓ_2 norm and ℓ_∞ norm by $a(i)$, $|a|$, $\|a\|_2$ and $\|a\|_\infty$, respectively. For a pair of vectors $a, b \in \mathbb{R}^n$, $[\![a, b]\!]$ denotes the hyper-rectangular set $[a(1), b(1)] \times \cdots \times [a(n), b(n)]$. Further, given $c \in \mathbb{R}^n$, $c + [\![a, b]\!]$ is another hyper-rectangular set which is shifted compared to $[\![a, b]\!]$ to the extent determined by c. Similarly, for a vector $\eta \in \mathbb{R}^n$ and a pair of vectors $a, b \in \mathbb{R}^n$, for which $a = \alpha\eta$, $\alpha \in \mathbb{Z}$ and $b = \beta\eta$, $\beta \in \mathbb{Z}$, we define $[\![a, b]\!]_\eta = \prod_{i=1}^n A_i$, where $A_i = \{\gamma\eta(i) \mid \gamma \in \mathbb{Z},\ \alpha \leq \gamma \leq \beta\}$. For two integers $a, b \in \mathbb{Z}$, we define $[a; b] = \{c \in \mathbb{Z} \mid a \leq c \leq b\}$.

Let A be a finite set of size $|A|$. The empty set is denoted by $\emptyset$. When A inherits a coordinate structure, i.e., when its members are vectors on the Euclidean space, $A(i)$ denotes the projection of set A onto its i^{th} dimension.

2.1.1. Control Systems

A continuous-time control system is a tuple $\Sigma = (X, x_{\mathsf{in}}, U, W, f)$, where $X \subset \mathbb{R}^n$ is the state space, $x_{\mathsf{in}} \in X$ is the initial state, $U \subset \mathbb{R}^m$ is the input space, and $W \subset \mathbb{R}^n$ is the disturbance space which is assumed to be a compact set containing the origin. The vector field $f : X \times U \to X$ is such that $f(\cdot, u)$ is locally Lipschitz for all $u \in U$. The evolution of the state of Σ is characterized by the differential equation

$$\dot{x}_t = f(x_t, u_t) + w_t, \tag{2.1}$$

where $w_t \in W$ represents the additive disturbance.

Trajectories of Control Systems

We first define *continuous-time trajectories* of control systems. Given a sampling time $\tau > 0$, an initial state $x_0 \in X$, a constant input $u \in U$, and a constant disturbance

$w \in W$, define the continuous-time trajectory $\zeta_{x_0,u,w}$ of the system on the time interval $[0,\tau]$ as an absolutely continuous function $\zeta_{x_0,u,w} : [0,\tau] \to X$ such that $\zeta_{x_0,u,w}(0) = x_0$, and $\zeta_{x_0,u,w}$ satisfies the differential equation $\dot{\zeta}_t(x_0,u,w) = f(\zeta_t(x_0,u,w),u)+w$ for almost all $t \in [0,\tau]$. The solution of (2.1) from x_0 for the constant control input u with $w_t = 0$ for all $t \geq 0$ is called the *nominal trajectory* of the system. For a fixed τ, we define the operators

$$\varphi(x,u,w) := \zeta_{x,u,w}(\tau) \text{ and}$$
$$\Phi(x,u) := \{\varphi(x,u,w) \mid w \in W\}$$

respectively for the trajectory at time τ and the set of such trajectories starting from x. A sequence $x_0, x_1, x_2, \ldots$ is a time-sampled trajectory of Σ if for each $i \geq 0$, we have $x_{i+1} \in \Phi(x_i, u_i)$ for some $u_i \in U$.

2.1.2. Linear Temporal Logic

Our control tasks are defined using Linear Temporal Logic (LTL). Here, we give a brief introduction to LTL. For detailed syntax and semantics of LTL, we refer to the book by Baier et al. [13] and references therein. We consider linear temporal logic (LTL) specifications with syntax [13]

$$\psi := \text{true} \mid p \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \bigcirc\psi \mid \psi_1 \,\mathsf{U}\, \psi_2,$$

where $p \subset \mathbb{R}^n$ is an element of the set of atomic propositions AP. Let $\rho = x_0, x_1, \ldots$ be an infinite sequence of elements from $\mathbb{R}^n$, and $\rho_i = x_i, x_{i+1}, \ldots$ denote a prefix of ρ for $i \in \mathbb{N}$. Then the satisfaction relation between ρ and a property ψ, expressed in LTL, is denoted by $\rho \models \psi$. Furthermore, $\rho \models \neg\psi$ if $\rho \not\models \psi$ and we say that $\rho \models \psi_1 \wedge \psi_2$ if $\rho \models \psi_1$ and $\rho \models \psi_2$. The next operator $\rho \models \bigcirc\psi$ holds if the property holds at the next time instance. The temporal until operator $\rho \models \psi_1 \,\mathsf{U}\, \psi_2$ holds if $\exists i \in \mathbb{N} : \rho_i \models \psi_2, \text{and } \forall j \in \mathbb{N} : 0 \leq j < i, \rho_j \models \psi_1$. Disjunction ($\vee$) can be defined by $\rho \models \psi_1 \vee \psi_2 \;\Leftrightarrow\; \rho \models \neg(\neg\psi_1 \wedge \neg\psi_2)$. The operator $\rho \models \Diamond\psi$ is used to denote that the property will eventually happen at some point in the future. The operator $\rho \models \Box\psi$ signifies that ψ must always be true at all times in the future.

2.1.3. Finite Abstractions

In order to satisfy a temporal specification on the trajectories of the system, it is generally needed to over-approximate the dynamics of the system with a finite discrete-time model. Let $\bar{X} \subset X$ and $\bar{U} \subset U$ be the finite sets of states and inputs, computed by (uniformly) quantizing the compact state and input spaces X and U using the rectangular discretization partitions of size $\eta_x \in \mathbb{R}^n_{>0}$ and $\eta_u \in \mathbb{R}^m_{>0}$, respectively. A *finite abstraction* associated with the dynamics in Eq. (2.1) is characterized by the tuple $\bar{\Sigma}$: $(\bar{X}, \bar{U}, T_F)$, where $T_F \subseteq \bar{X} \times \bar{U} \times \bar{X}$ denotes the system's *forward-in-time transition system.* The transition system T_F is defined such that

$$(\bar{x}, \bar{u}, \bar{x}') \in T_F \Leftrightarrow \exists (x,u,x') \in \Omega_{\frac{\eta_x}{2}}(\bar{x}) \times \Omega_{\frac{\eta_u}{2}}(\bar{u}) \times \Omega_{\frac{\eta_x}{2}}(\bar{x}') \text{ s.t. } x' \in \varphi(x,u,\tau).$$

When the dynamics in Eq. (2.1) are known and satisfy the required Lipschitz continuity condition, the finite abstraction can be constructed using the method proposed in [146]. For systems with unknown dynamics, data-driven schemes for learning finite abstractions can be employed [93, 52, 123]. By abusing the notation, we denote the *reachable set* for a state-input pair $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$ by $T_F(\bar{x}, \bar{u}) = \{\bar{x}' \in \bar{X} \mid \bar{x}' \in \varphi(\bar{x}, \bar{u}, \tau)\}$. We assume that the reachable sets take hyper-rectangular form, meaning that for every $\bar{x} \in \bar{X}$, $\bar{u} \in \bar{U}$ the corresponding reachable set $H = T_F(\bar{x}, \bar{u})$ can be rewritten as $H = \prod_{i=1}^{n} H(i)$, where $H(i)$ corresponds to the projection of the set H onto its i^{th} coordinate. Otherwise, in case that H is not hyper-rectangular, it is over-approximated by $\prod_{i=1}^{n} H(i)$. Note that $\bar{\Sigma}$ can in general correspond to a *non-deterministic* control system, i.e., $|T_F(\bar{x}, \bar{u}))| > 1$ for some $\bar{x} \in \bar{X}, \bar{u} \in \bar{U}$. Given $\bar{\Sigma}$, one can easily compute the characterization of the *backward-in-time* dynamics as

$$\bar{\Sigma}_B = (\bar{X}, \bar{U}, T_B),\ T_B = \{(\bar{x}, \bar{u}, \bar{x}') \in \bar{X} \times \bar{U} \times \bar{X} \mid (\bar{x}', \bar{u}, \bar{x}) \in T_F\}. \tag{2.2}$$

A *trajectory* of $\bar{\Sigma}$ is a finite or infinite sequence $x_0, x_1, x_2, \ldots \in \bar{X}^{\infty}$, such that for each $i \geq 0$, there is a control input $\bar{u}_i \in \bar{U}$ such that $(x_i, \bar{u}_i, x_{i+1}) \in T_F$. The operator $\mathrm{Pre}(\cdot)$ acting on sets $P \subseteq \bar{X}$ is defined as

$$\mathrm{Pre}(P) = \{\bar{x} \in \bar{X} \mid \exists \bar{u} \in \bar{U}\ s.t.\ T_F(\bar{x}, \bar{u}) \subseteq P\}.$$

Finally, to compute an over-approximating set of the discrete states that have overlap with a hyper rectangular set $[\![x_{lb}, x_{ub}]\!]$, we define the (over-approximating) quantization mapping as

$$\bar{K}(x_{lb}, x_{ub}) = \{\bar{x}' \in \bar{X} \mid [\![\bar{x}' - \eta_x/2, \bar{x}' + \eta_x/2]\!] \cap [\![x_{lb}, x_{ub}]\!] \neq \emptyset\}.$$

Similarly, the under-approximating quantization mapping is defined as

$$\underline{K}(x_{lb}, x_{ub}) = \{\bar{x}' \in \bar{X} \mid [\![\bar{x}' - \eta_x/2, \bar{x}' + \eta_x/2]\!] \subseteq [\![x_{lb}, x_{ub}]\!]\}.$$

2.1.4. Controllers

A feedback controller for Σ is denoted by $C\colon X \mapsto U$. We denote the feedback composition of Σ with C as $C \parallel \Sigma$. A feedback controller for Σ over a time interval $[0; T]$, $T \in \mathbb{N}$, is a function $C\colon X \times [0; T] \to U$. The set of trajectories of $C \parallel \Sigma$ having length $T \in \mathbb{N}$ is the set of sequences $x_0, x_1, x_2, \ldots, x_{T-1}$, s.t. $x_0 = x_{\mathsf{in}}$, $x_{i+1} \in \Phi(x_i, u_i)$ and $u_i = C(x_i)$ for $i \in [0; T-2]$.

An *open-loop controller* for $\Sigma = (X, x_{\mathsf{in}}, U, W, f)$ over a time interval $[0; T]$ with $T \in \mathbb{N}$ is a function $C\colon [0; T] \to U$. The open loop is obtained when we connect C with Σ serially, denoted by $C \triangleright \Sigma$. The set of trajectories of the open-loop system $C \triangleright \Sigma$ consists of all finite trajectories $x_0, x_1, \ldots, x_T$ such that $x_0 = x_{\mathsf{in}}$ and $x_{i+1} = f(x_i, C(i)) + w_i$ for some $w_i \in W$ for all $i \in [0; T-1]$.

Now let $\{\Sigma^i\}$ be a set of control systems. We can define *global* open-loop and feedback controllers by defining the respective controllers on the product system $\Sigma^{\times}$ over a time

interval $[0;T]$. We can also define *local* open-loop and feedback controllers C^i for each Σ^i. In this latter case, the set of trajectories of the system $\{C^i\} \triangleright \{\Sigma^i\}$ (respectively, $\{C^i\} \parallel \{\Sigma^i\}$) are finite sequences $x_0^\times, x_1^\times, \ldots, x_T^\times$ such that $x_0^\times = x_{\mathsf{in}}^\times$ and for each $j \in [0;T-1]$, we have $\mathbf{proj}^i(x_{j+1}^\times) = f^i(\mathbf{proj}^i(x_j^\times), C^i(j)) + w_{ji}$ (respectively, $\mathbf{proj}^i(x_{j+1}^\times) = f^i(\mathbf{proj}^i(x_j^\times), C^i(\mathbf{proj}^i(x_j^\times), j)) + w_{ji}$) for some $w_{ji} \in W^i$, for each $i \in [1;N]$.

2.1.5. Feedback Refinement Relation

Let Σ be a control system and $\bar{\Sigma}$ be its finite-state abstraction. A *feedback refinement relation* (FRR) from Σ to $\bar{\Sigma}$ is a relation $Q \subseteq X \times \bar{X}$ s.t. for all $x \in X$ there is some $\bar{x} \in \bar{X}$ such that $Q(x,\bar{x})$ and for all $(x,\bar{x}) \in Q$, we have (i) $\bar{U} \subseteq U$, and (ii) $u \in \bar{U} \Rightarrow Q(f(x,u)) \subseteq \bar{f}(\bar{x},u)$. We write $\Sigma \preceq_Q \bar{\Sigma}$ if Q is an FRR from Σ to $\bar{\Sigma}$.

2.1.6. Abstraction-based Controller Synthesis

Our synthesis objective is expressed as Linear Temporal Logic (LTL) specifications. The abstraction-based controller design (ABCD) [146] is a 3-step method to find a robust controller for the control system Σ: First, we compute a finite state abstraction $\bar{\Sigma}$ s.t. $\Sigma \preceq_Q \bar{\Sigma}$. Second, we synthesize an abstract controller of the form $\bar{C} : \bar{X} \to \bar{U}$ for $\bar{\Sigma}$ using methods from the reactive synthesis literature. Finally, we obtain the desired controller C as $C := \bar{C} \circ Q$. It is known that this three step process produces a controller C such that $C \parallel \Sigma$ satisfies the specification [146]. If a controller cannot be found, we reduce the discretization parameters η_x and η_u and try again. For the details of the tool implementation using abstract models we refer to [152].

2.1.7. Neural Networks

A *neural network* $\mathcal{N}(\theta,\cdot)\colon \mathbb{R}^d \to \mathbb{R}^q$ of depth $v \in \mathbb{N}$ is a parameterized function which transforms an input vector $a \in \mathbb{R}^d$ into an output vector $b \in \mathbb{R}^q$, and is constructed by the forward combination of v functions as follows:

$$\mathcal{N}(\theta,a) = G_v(\theta_v, G_{v-1}(\theta_{v-1}, \ldots, G_2(\theta_2, G_1(\theta_1,a)))),$$

where $\theta = (\theta_1,\ldots,\theta_v)$ and $G_i(\theta_i,\cdot)\colon \mathbb{R}^{p_{i-1}} \to \mathbb{R}^{p_i}$ denotes the i^{th} *layer* of $\mathcal{N}$ parameterized by θ_i with $p_0 = d$, $p_i \in \mathbb{N}$ for $i \in [1;v]$ and $p_v = q$. The i^{th} layer of the network, $i \in [1;v]$, takes an input vector in $\mathbb{R}^{p_{i-1}}$ and transforms it into an output representation in $\mathbb{R}^{p_i}$ depending on the value of parameter vector θ_i and type of the used *activation function* in G_i. During the *training phase* of the network, the set of parameters θ is learned over the *training set* which consists of a number of input-output pairs $\{(a_k,b_k) \mid k = 1,2,\ldots,N\}$, in order to achieve the highest performance with respect to an appropriate metric such as mean squared error. For a trained neural network, we drop its dependence on the parameters θ. In this section, we characterize a neural network of depth v using its corresponding list of layer sizes, i.e., $(p_1,p_2,\ldots,p_v)$, and the type of the activation function used, e.g., hyperbolic tangent, Rectified Linear Unit (ReLU), etc.

Neural networks can be used for both *regression* and *classification* tasks. In a regression task, the goal is to predict a numerical value given an input, whereas, a classification task requires predicting the correct class label for a given input. In order to measure performance of the trained neural network, we consider *prediction error*. Note that prediction error is different from the metrics such as *mean squared error* (MSE) which are used during the training phase for defining the objective function for the training. The prediction error for regression and classification tasks is defined differently. For *our regression tasks*, we define the prediction error for a trained neural network $\mathcal{N}$ over a training set $\{(a_k, b_k) \mid k = 1, 2, \ldots, N\}$ as

$$e = \max_{k \in [1;N]} |\mathcal{N}(a_k) - b_k|.$$

We consider the classification tasks wherein there may exist more than one valid class label for each input. Therefore, the training set would be of the form $\{(a_k, b_k) \mid k = 1, 2, \ldots, N\}$, where $b_k \in \{0, 1\}^q$ and $b_k(i) = 1$ iff $i \in [1; q]$ corresponds to a valid label at a_k. Since the number of valid labels for each input can be different, we define the prediction error of a trained classifier $\mathcal{N}$ in the following way:

$$err = \frac{|\{k \in [1; N] \mid b_k(i) = 0 \text{ with } i = argmax(\mathcal{N}(a_k))\}|}{N}.$$

For a given neural network $\mathcal{N}$ with the training set $\{(a_k, b_k) \mid k = 1, 2, \ldots, N\}$, we define the *continuity index* as

$$\alpha_{\mathcal{N}} = \max_{1 \leq i,j \leq N,\, i \neq j} \frac{\|\mathcal{N}(a_i) - \mathcal{N}(a_j)\|}{\|a_i - a_j\|}. \tag{2.3}$$

2.2. Data-Driven Abstraction Based Controller Design

ABCD schemes generally rely on a precise mathematical model of the system. This stems from the fact that establishing a behavioral relation between the original system and its finite abstraction uses reachability analysis over the dynamics of the original system that require knowledge of the dynamical equations. Although such equations can in principle by derived for instance by using physics laws, the real-world control systems are a mixture of differential equations, block diagrams, and lookup tables. Therefore, extracting a clean analytical model for systems of practical interest could be infeasible. A promising approach to tackle this issue is to develop data-driven control synthesis schemes with appropriate formal (probabilistic) guarantees.

The main objective of this section is to provide a data-driven approach for formal synthesis of controllers to satisfy temporal specifications. A brief overview of the approach can be seen in Figure 2.1. We focus on continuous-time nonlinear dynamical systems whose dynamics are unknown but sampled trajectories are available. Our approach constructs a *finite* abstract model of the system using only a finite number of sampled trajectories and the growth bound of the system. We formulate the computation of the growth bound as a robust convex program (RCP) that has infinite uncountable number of

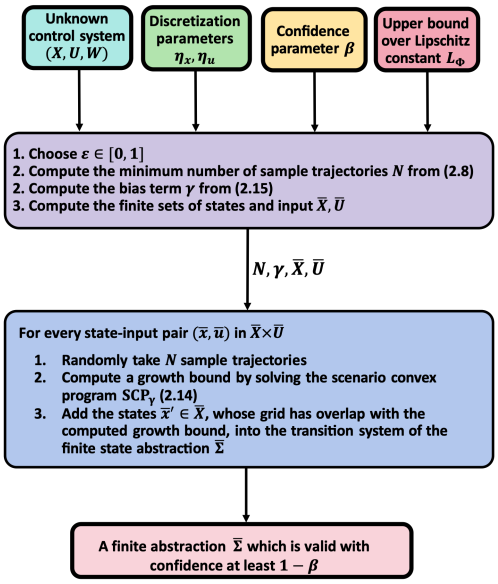

Figure 2.1.: A flow diagram illustrating the proposed data-driven method for constructing finite abstractions.

constraints. We then approximate the solution of the RCP with a scenario convex program (SCP) that has a finite number of constraints and can be solved using only a finite set of sampled trajectories. We establish a sample complexity result that gives a lower bound for the required number of trajectories to guarantee the correctness of the growth bound *over the whole state space* with a given confidence. We also provide a sample complexity result for the satisfaction of the specification on the system in closed loop with the designed controller for a given confidence. Our result requires estimating a bound on the Lipschitz constant of the system with respect to the initial state, that we obtain using extreme value theory. As our last contribution, we show that our approach can be extended to a model-free abstraction refinement scheme by modifying the formulation of the growth bound and providing similar sample complexity results. We demonstrate the performance of our approach on three case studies. The content of this section is based on our paper [93].

Related Work. There is an extensive body of literature on *model-based* formal synthesis for both deterministic and probabilistic systems. We refer the reader to the books [13, 22, 172], seminal papers [67, 1], and the survey paper [106]. *Data-driven* approaches for analysis, verification, and synthesis of systems have received significant attention recently to improve efficiency and scalability of model-based approaches, and to study problems in which a model of the system is either not available or costly and time-consuming to construct. Given a prior inaccurate knowledge about the model of the system, a research line is to use data for refining the model and then synthesize a controller. Such approaches assume a class of models and improve the estimation of the uncertainty within the model class. These approaches range from using Gaussian processes [128, 16],

differential inclusions [53], rapidly-exploring random graphs [70], piecewise affine models [154], and model-based reinforcement learning algorithms [42]. A data-driven framework is proposed by Fan et al.[57] for verifying properties of hybrid systems when the continuous dynamics are unknown but the discrete transitions are known.

Data-driven approaches for solutions of scenario convex programs are developed for switching systems by Wang and Jungers to establish stability [182] and by Berger et al. for invariant subspace identification [23]. Ahmad et al. [3] have developed an adaptive sampling-based approach for motion planning using deterministic nonlinear control systems and robust control barrier functions. Zhong et al. [194] have studied linear dynamical systems with bounded disturbances by proposing a data-driven method to compute state feedback controllers that enforce staying in safety invariant sets by using finite number of state-input data points. Cohen et al. [43] have developed a model-based reinforcement learning approach to satisfy linear temporal logic specifications on continuous-time nonlinear systems.

Data-driven model-free approaches compute the solution of the synthesis problem directly from data without constructing a model. Hsu et al. [80] provide a reach-avoid Q-learning algorithm with convergence guarantees for an arbitrarily tight conservative approximation of the reach-avoid set. Wang et al. [181] propose a falsification-based adversarial reinforcement learning algorithm for metric temporal logic specifications. Satisfying signal temporal logic specifications is studied by Verdier et al. [178] using counterexample-guided inductive synthesis on nonlinear systems, and using model-free reinforcement learning by Kalagarla et al. [90] to satisfy signal temporal logic specifications. A learning framework for synthesis of control-affine systems is provided by Sun et al. [169]. Watanabe et al. [183] study learning from demonstration while preventing the violation of safety under the learned policy. The recent papers [105, 157] propose a data-driven approach to compute barrier certificates with correctness guarantees on satisfaction of safety specifications.

The research on data-driven constructions of abstract models is very limited. Legat et al. [110] provide an abstraction-based controller synthesis approach for hybrid systems by computing Lyapunov functions and Bellman-like Q-functions, and using a branch and bound algorithm to solve the optimal control problem. This differs from our approach where we want to satisfy temporal specifications instead of solving optimal control problems. Makdesi et al. [122] studied unknown monotone dynamical systems and sampled a set of trajectories generated by the system to find a minimal map overapproximating the dynamics of any system that produces these transitions. Consequently, they calculate an abstraction of the system related to this map and prove that an alternating bisimulation relation exists between them. In contrast to this work, our approach is not restricted to monotone systems and is applicable to any nonlinear dynamical system. Abstract models are also constructed for stochastic systems using sampled data.

Data-driven construction of abstract models for stochastic systems has also been studied recently. Badings et al. [12, 11] consider constructing abstract models in the form of interval Markov decision processes (IMDPs) by computing probably approximately correct (PAC) bounds on the transition probabilities of the system. This makes the approach

applicable for satisfying infinite-horizon specifications and providing confidence bounds on the (probabilistic) satisfaction of the specification. The work by Lavaei et al. [107] constructs finite MDPs using data for general nonlinear stochastic systems utilizing the concept of stochastic bisimulation functions. The focus of these works is on stochastic systems, but our work develops the results for non-probabilistic systems.

Since our results rely on knowing a possibly conservative bound on the system's Lipschitz constant, we review here the corresponding literature for finding such a bound. Lipschitz learning algorithms have been proposed to estimate a Lipschitz constant of a function under the assumption of knowing the function, e.g., for neural networks [180, 60, 87]. When the function is not known, available approaches [188, 162, 168] use and improve a traditional estimator by Strongin [167]. All these approaches provide proof of convergence to the true Lipschitz constant when the number of samples goes to infinity and can only provide an underapproximation of the system's Lipschitz constant for finite sample sizes. A recent result by Huang et al. [83] extends the Strongin's estimation method to handle bounded observational noise. It assumes having prior knowledge of an upper bound on the second-order partial derivatives of the function. The authors use least squares regression and provide a guarantee on the sample complexity of the approach for both noiseless and noisy samples. Additionally, the authors obtain a theoretical minimum for the sample complexity, showing that their algorithm performs optimally without noise and near-optimally with noise. Under the assumption of knowing an upper bound over the second partial derivatives, the proposed method provides both asymptotic and finite sample guarantees that give closeness to the true Lipschitz constant with a certain confidence. In our approach, the underlying analytical model of the system is assumed to be unknown. The provided sample complexity result is based on the assumption of knowing the system's Lipschitz constant, which can be estimated using e.g., the approach of [188]. Our result still requires assuming that the estimation is gives correct bound. Alternatively, the recent approach of Huang et al. [83] can be used to estimate the Lipschitz constant with finite number of samples with a certain confidence under the assumption of knowing a bound on the second-order partial derivatives of the system trajectories.

The closest works to our problem formulation is the work by Devonport et al. [52] and the work of Xue et al. [191], where data-driven abstraction techniques are provided for satisfying finite-horizon specifications. Our results are more general and provide stronger guarantees in two main aspects. First, our constructed abstraction can be used for synthesizing a controller against any linear temporal logic (LTL) specification and is not restricted to a fragment of LTL specifications. Our sample complexity result is independent of the horizon of the specification and does not limit using the approach on finite-horizon specifications. Second, the guarantee provided by Devonport et al. and by Xue et al. are based on PAC bounds, which means the constructed abstraction is always wrong on a small subset of the state space whose size can be made smaller at the cost of high computational efforts, and the approach will require infinite number of samples if the size of this subset is set to zero. Our formulated guarantee ensures that the abstraction is valid on the *entire* state space with high confidence (i.e., confidence

close to 1). The confidence is specified by $(1-\beta)$ in this our work and is interpreted from the frequentist view of probability: if we run our algorithm multiple times, we always get a correct abstraction except for a small number of times reflected in the confidence value. Having such a confidence value is essential in our approach since it relies on data gathered from the system. Smaller values of β gives higher confidence on getting a correct abstraction. This in turn increases the computational complexity of our approach since β appears directly in our sample complexity results.

In our approach, we formulate the synthesis problem as a robust convex program and approximate it with a scenario program. Calafiore and Campi [36] provide an approximately feasible solution for the associated chance-constrained program by solving a scenario program, and give a sample complexity result. Relaxing the convexity assumption is studied by Soudjani and Majumdar [165] by assuming additional properties of the underlying probability distributions. We will use the results by Esfahani et al. [56], where the optimality of the robust program is connected directly to the scenario program for performing data-driven verification and synthesis. Inspired by the works of Wood and Zhang [188], and Weng et al. [184], we will use extreme value theory to estimate the Lipschitz constant needed for the sample complexity results. Our results still require assuming that the estimation gives a correct bound.

2.2.1. Problem Statement

We study abstraction-based control design (ABCD) for systems with *unknown* dynamics using available data from the system such that a given specification is satisfied with high confidence on the closed-loop system.

Assumption 1 *The vector field f of the control system $\Sigma = (X, x_{\mathsf{in}}, U, W, f)$ in unknown, but sampled trajectories of the system can be obtained in the form of $\mathcal{S}_N := \{(x_k, u_k, x'_k) \,|\, x'_k \in \Phi(x_k, u_k),\ k = 1, 2, \ldots, N\}$.*

Problem 2.1 (Data-driven ABCD) ***Inputs:*** *Control system $\Sigma = (X, x_{\mathsf{in}}, U, W, f)$ with unknown vector field f, specification Ψ, sampled trajectories $\mathcal{S}_N$, and confidence parameter $\beta \in (0, 1)$.*

Outputs: *Abstract model $\bar{\Sigma}$, abstract controller $\bar{C}$, and refined controller C for Σ, such that $C \parallel \Sigma$ satisfies Ψ with confidence $(1-\beta)$.*

The first step of the ABCD is to compute a finite abstraction $\bar{\Sigma}$ for Σ. Once such an abstraction is computed, synthesis of the controller $\bar{C}$ and refining it to C follow the model-based ABCD scheme. Therefore, the main challenge is to provide a data-driven computation of the abstraction $\bar{\Sigma}$ that is a true overapproximation of Σ with confidence $(1-\beta)$.

Problem 2.2 (Data-driven Abstraction) ***Inputs:*** *Control system $\Sigma = (X, x_{\mathsf{in}}, U, W, f)$ with unknown vector field f, sampled trajectories $\mathcal{S}_N$, discretization parameters η_x and η_u, and confidence parameter $\beta \in (0, 1)$.*

Outputs: *Finite model $\bar{\Sigma}$ that is an abstraction of Σ with confidence $(1-\beta)$.*

In this section, we tackle Problem 2.2 by showing how to construct $\bar{\Sigma}$ from sampled trajectories $\mathcal{S}_N$, and provide a lower bound on the data size N in order to ensure correctness of the abstraction with confidence $(1-\beta)$. The required theoretical tools are presented in the following.

2.2.2. Robust Convex Programs

In this subsection, we describe robust convex programs (RCPs) and data-driven approximation of their solution. In Subsections 2.2.3 and 2.2.4, we show how such an approximation can be used for solving the data-driven abstraction in Problem 2.2.

Let $T \subset \mathbb{R}^q$ be a compact convex set for some $q \in \mathbb{N}$ and $c \in \mathbb{R}^q$ be a constant vector. Let $(\mathcal{D}, \mathcal{B}, \mathbb{P})$ be the probability space of the *uncertainty* and $g\colon T \times \mathcal{D} \to \mathbb{R}$ be a measurable function, which is convex in the first argument for each $d \in \mathcal{D}$, and bounded in the second argument for each $\theta \in T$. The robust convex program (RCP) is defined as

$$\text{RCP: } \begin{cases} \min_\theta c^\top \theta \\ s.t.\ \theta \in T \text{ and } \ g(\theta, d) \leq 0 \quad \forall d \in \mathcal{D}. \end{cases} \tag{2.4}$$

Computationally tractable approximations of the optimal solution of the RCP (2.4) can be obtained using *scenario convex programs* (SCPs) that only require gathering finitely many samples from the uncertainty space [129]. Let $(d_i)_{i=1}^N$ be N independent and identically distributed (i.i.d.) samples drawn according to the probability measure $\mathbb{P}$. The SCP corresponding to the RCP (2.4) strengthened with $\gamma \geq 0$ is defined as

$$SCP_\gamma : \begin{cases} \min_\theta c^\top \theta \\ s.t.\ \theta \in T, \text{ and } \ g(\theta, d_i) + \gamma \leq 0 \ \forall i \in \{1, 2, \ldots, N\}. \end{cases} \tag{2.5}$$

We denote the optimal solution of RCP (2.4) as θ^*_{RCP} and the optimal solution of SCP_γ (2.5) as θ^*_{SCP}. Note that θ^*_{RCP} is a single deterministic quantity but θ^*_{SCP} is a random quantity that depends on the i.i.d. samples $(d_i)_{i=1}^N$ drawn according to $\mathbb{P}$. The RCP (2.4) is a challenging optimization problem since the cardinality of $\mathcal{D}$ is infinite and the optimisation has infinite number of constraints. In contrast, the SCP (2.5) is a convex optimization with finite number of constraints for which efficient optimization techniques are available [27]. The following theorem provides a sample complexity result for connecting the optimal solution of the SCP_γ to that of the RCP.

Theorem 2.2.1 ([129]) *Assume that the mapping $d \mapsto g(\theta, d)$ in (2.4) is Lipschitz continuous uniformly in $\theta \in T$ with Lipschitz constant L_d and let $h\colon [0,1] \to \mathbb{R}_{\geq 0}$ be a strictly increasing function such that*

$$\mathbb{P}(\Omega_\varepsilon(d)) \geq h(\varepsilon), \tag{2.6}$$

for every $d \in \mathcal{D}$ and $\varepsilon \in [0,1]$. Let θ^*_{RCP} be the optimal solution of the RCP (2.4) and θ^*_{SCP} the optimal solution of SCP_γ (2.5) with

$$\gamma = L_d h^{-1}(\varepsilon) \tag{2.7}$$

computed by taking N i.i.d. samples $(d_i)_{i=1}^N$ from $\mathbb{P}$. Then θ^*_{SCP} is a feasible solution for the RCP with confidence $(1-\beta)$ if the number of samples $N \geq N(\varepsilon, \beta)$, where

$$N(\varepsilon,\beta) := min\left\{N \in \mathbb{N} \,\Big|\, \sum_{i=0}^{q-1} \binom{N}{i} \varepsilon^i (1-\varepsilon)^{N-i} \leq \beta\right\}, \tag{2.8}$$

with q being the dimension of the decision vector $\theta \in T$.

2.2.3. Data-Driven Abstraction

In this section, we first discuss the steps required for model-based abstraction of control systems. We then show how this can be formulated as an RCP and present its associated SCP. Finally, we use the connection between the RCPs and SCPs in Theorem 2.2.1 to provide a lower bound for number of required samples to certify a desired confidence. The simplifying assumption used in this subsection is that samples from the *nominal trajectories* of the system Σ in also available in the form of $\{(x_k, u_k, x'_k) \,|\, x'_k = \varphi(x_k, u_k, 0),\, k = 1, 2, \ldots, N\}$. We discuss in the next subsection how this assumption can be relaxed by modifying the inequality of the growth bound.

Growth Bound for Reachable Sets

Consider a control system $\Sigma = (X, x_{\mathsf{in}}, U, W, f)$ with the disturbance set $W = [-\bar{w}, \bar{w}]$ for some vector $\bar{w} \in \mathbb{R}^n_{\geq 0}$. Let η_x and η_u be discretization parameters for the state and input spaces X and U used to construct $\bar{X}$ and $\bar{U}$ of sizes n_x and n_u, respectively. The first step of ABCD is to compute a finite abstraction $\widehat{\Sigma} = (\bar{X}, \bar{U}, \bar{f})$ using overapproximations of the reachable sets for every pair of abstract state and input. The reachable set for every pair $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$ is defined as

$$Reach(\bar{x}, \bar{u}) := \{x' \in \Phi(x, \bar{u}) \mid x \in \Omega_{\eta_x}(\bar{x})\}.$$

The set $Reach(\bar{x}, \bar{u})$ is usually overapproximated using the growth bound of the system dynamics [146].

Definition 2.2.1 *For a control system Σ with abstract state and input spaces $\bar{X}, \bar{U}$, a function $\kappa\colon \mathbb{R}^n_{\geq 0} \times \bar{X} \times \bar{U} \to \mathbb{R}^n_{\geq 0}$ is called a* growth bound function *for Σ if it satisfies*

$$\begin{aligned} &|\varphi(x, \bar{u}, w) - \varphi(\bar{x}, \bar{u}, 0)| \leq \kappa(|x - \bar{x}|, \bar{x}, \bar{u}) \\ &\quad \forall \bar{x} \in \bar{X},\, \forall \bar{u} \in \bar{U},\, \forall x \in \Omega_{\eta_x}(\bar{x}),\, \forall w \in W. \end{aligned} \tag{2.9}$$

Note that $\varphi(\bar{x}, \bar{u}, 0)$ is the nominal (disturbance-free) trajectory of the system. Using this definition, for every abstract state-input pair $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$, the reachable set $Reach(\bar{x}, \bar{u})$ is overapproximated with a ball centered at $z(\bar{x}, \bar{u}) := \varphi(\bar{x}, \bar{u}, 0)$ with radius $\lambda(\bar{x}, \bar{u}) := \kappa(\eta_x, \bar{x}, \bar{u})$.

When the system dynamics are known, it is shown in [146] that a growth bound for the system can be computed as

$$\kappa(r, \bar{x}, \bar{u}) = e^{L(\bar{u})\tau} r + \int_0^\tau e^{L(\bar{u})s} \bar{w} ds, \tag{2.10}$$

for all $r \in \mathbb{R}^n_{\geq 0}$, $\bar{x} \in \bar{X}$, and $\bar{u} \in \bar{U}$, where $\bar{w}$ is the upper bound of the disturbance and $L \colon \bar{U} \to \mathbb{R}^{n \times n}$ is a matrix such that the entries of $L(\bar{u})$ satisfy the following inequality for all $x \in X$:

$$L_{i,j}(\bar{u}) \geq \begin{cases} D_j f_i(x, \bar{u}) & i = j \\ |D_j f_i(x, \bar{u})| & i \neq j, \end{cases} \tag{2.11}$$

for all $i, j \in \{1, 2, \ldots, n\}$, where $f_i(x, u)$ is the i^{th} element of the vector field $f(x, u)$ and $D_j f_i$ is its partial derivative with respect to the j^{th} element of x.

SCP for the Computation of Growth Bound

When the model of the system is unknown, the growth bound in (2.10) is not available since the matrix $L(\bar{u})$ defined using (2.11) is not computable. To tackle this bottleneck, we aim at computing a growth bound for the system that has the following parameterized form

$$\kappa_\theta(r, \bar{x}, \bar{u}) := \theta_1(\bar{x}, \bar{u}) r + \theta_2(\bar{x}, \bar{u}), \forall r \in \mathbb{R}^n_{\geq 0}, \bar{x} \in \bar{X}, \bar{u} \in \bar{U}, \tag{2.12}$$

where $\theta_1 \in \mathbb{R}^{n \times n}$ and $\theta_2 \in \mathbb{R}^n$. We denote by $\theta \in \mathbb{R}^{n^2+n}$ the concatenation of columns of θ_1 and θ_2.

Remark 1 *The parameterized growth bound in* (2.12) *is linear with respect to r similar to* (2.10)*, but is more general and less conservative by allowing θ_1, θ_2 to depend on $\bar{x}$ (i.e., they are defined locally for each abstract state).*

Theorem 2.2.2 *The parameterised growth bound in Eq.* (2.12) *can be computed by solving the following robust convex program*

$$\begin{cases} \min_\theta c^\top \theta \\ s.t.\ 0 \leq \theta \leq \bar{\theta},\ \textit{and}\ \forall x \in \Omega_{\eta_x}(\bar{x}),\ \forall w \in W, \\ \quad |\varphi(x, \bar{u}, w) - \varphi(\bar{x}, \bar{u}, 0)| - \kappa_\theta(|x - \bar{x}|, \bar{x}, \bar{u}) \leq 0, \end{cases} \tag{2.13}$$

where $c = [1, 1, \ldots, 1] \in \mathbb{R}^{n^2+n}$ and $\bar{\theta}$ is a sufficiently large positive vector.

Proof *We first show that the optimization* (2.13) *is in fact a robust convex program. Let $\mathcal{D} = \Omega_{\eta_x}(\bar{x}) \times W$ be the uncertainty space and*

$$g(\theta, x, w) := |\varphi(x, \bar{u}, w) - \varphi(\bar{x}, \bar{u}, 0)| - \kappa_\theta(|x - \bar{x}|, \bar{x}, \bar{u})$$

for all $x \in \Omega_{\eta_x}(\bar{x})$ and $w \in W$ and fixed $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$. We need to show that g is convex in θ for each $(x, w) \in \mathcal{D}$ and bounded in (x, w) for every $\theta \in [0, \bar{\theta}]$. The convexity holds due to the parameterization of κ_θ in (2.12) *being linear with respect to the optimization*

variables in θ. The boundedness holds due to the set $\mathcal{D}$ being compact and trajectories of the system being continuous. We note that the goal of solving the optimization problem in (2.13) *is to find a parametrization which corresponds to the tightest possible growth bound. Therefore, with the current valuation of vector c and formulation of κ_θ in* (10)*, we need to restrict entries of θ to only take non-negative values ($0 \leq \theta$).*

To construct the SCP$_\gamma$ associated with the RCP (2.13), we fix $\bar{x} \in \bar{X}$ and $\bar{u} \in \bar{U}$, consider a uniform distribution on the space $\mathcal{D} = \Omega_{\eta_x}(\bar{x}) \times W$ and obtain N i.i.d. sample trajectories $\mathcal{S}_N = \{(x_i, \bar{u}, x_i') \,|\, x_i' \in \Phi(x_i, \bar{u}), i = 1, 2, \ldots, N\}$. Note that every x_i' corresponds to a random disturbance $w_i \in W$. The SCP$_\gamma$ is

$$\begin{cases} \min_\theta c^\top \theta \\ s.t.\ 0 \leq \theta \leq \bar{\theta} \text{ and } \forall i \in \{1, \ldots, N\}, \\ \quad |x_i' - x_{nom}'| - \theta_1(\bar{x}, \bar{u})|x_i - \bar{x}| + \theta_2(\bar{x}, \bar{u}) + \gamma \leq 0, \end{cases} \tag{2.14}$$

where $x_{nom}' := \varphi(\bar{x}, \bar{u}, 0)$ and $\gamma \in \mathbb{R}_{\geq 0}$.

Theorem 2.2.3 *Let $|\bar{X}| = n_x$ and $|\bar{U}| = n_u$. For any $\bar{x} \in \bar{X}$ constructed with discretization size η_x, any $\bar{u} \in \bar{U}$, and the disturbance set $W = [-\bar{w}, \bar{w}]$, the optimal solution of* (2.14) *gives a growth bound for the system Σ corresponding to $(\bar{x}, \bar{u})$ with confidence $(1 - \beta/(n_x n_u))$, when the number of samples $N \geq N(\varepsilon, \beta/(n_x n_u))$ and*

$$\gamma = 4L_\varphi(\bar{u}) \sqrt[2n]{\varepsilon \prod_{i=1}^{n} \eta_x(i) \prod_{i=1}^{n} \bar{w}(i)}, \tag{2.15}$$

where $\varepsilon \in [0, 1]$, n is the dimension of the state space and $L_\varphi(\bar{u})$ is an upper bound for the Lipschitz constant of the system trajectories $\varphi(x, \bar{u}, w)$ with respect to (x, w).

Proof *We apply Theorem 2.2.2 to the RCP* (2.13) *for fixed $\bar{x} \in \bar{X}$ and $\bar{u} \in \bar{U}$. Define*

$$\begin{aligned} g(\theta, x, w) := \max\{&|\varphi(x, \bar{u}, w) - \varphi(\bar{x}, \bar{u}, 0)| \\ &- \theta_1(\bar{x}, \bar{u})|x - \bar{x}| - \theta_2(\bar{x}, \bar{u})\}, \end{aligned} \tag{2.16}$$

where the $\max\{\cdot\}$ is applied to the elements of its argument that belongs to $\mathbb{R}^n$. Since the distribution on $\mathcal{D} = \Omega_{\eta_x}(\bar{x}) \times W$ is uniform, we choose

$$h(\varepsilon) = \mathbb{P}(\Omega_\varepsilon(d)) = \frac{(\varepsilon/2)^{2n}}{\prod_{i=1}^{n} \eta_x(i) \prod_{i=1}^{n} \bar{w}(i)}$$

to satisfy the inequality (2.6)*. Note that $h(\varepsilon)$ gives the probability of choosing a point within the $2n-$ball $\Omega_\varepsilon(d)$ uniformly at random. We use Equation* (2.7) *as $\gamma = L_d h^{-1}(\varepsilon)$ to get the value of γ in* (2.15)*. It only remains to show that $g(\theta, x, w)$ is Lipschitz continuous with constant $L_d = 2L_\varphi(\bar{u})$. Note that $L_\varphi(\bar{u})$ is the Lipschitz constant of $\varphi(x, \bar{u}, w)$ with respect to (x, w), and satisfies*

$$\|\varphi(x, \bar{u}, w) - \varphi(x', \bar{u}, w')\| \leq L_\varphi(\bar{u})\|(x, w) - (x', w')\| \tag{2.17}$$

for all $x, x' \in \Omega_{\eta_x}(\bar{x})$ *and* $w, w' \in W$. *Since* $\|\theta_1(\bar{x}, \bar{u})\|$ *can be bounded by* $L_\varphi(\bar{u})$, *we get that*

$$\begin{aligned}
&\|g(\theta, x, w) - g(\theta, x', w')\| \\
&\leq \|\varphi(x, \bar{u}, w) - \varphi(x', \bar{u}, w')\| + \|\theta_1(\bar{x}, \bar{u})\|\|x - x'\| \\
&\leq L_\varphi(\bar{u})\|(x, w) - (x', w')\| + L_\varphi(\bar{u})\|x - x'\| \\
&\leq 2L_\varphi(\bar{u})\|(x, w) - (x', w')\|,
\end{aligned}$$

Therefore, $g(\theta, x, w)$ *is Lipschitz continuous with constant* $2L_\varphi(\bar{u})$. *This completes the proof.*

Remark 2 *Note that the statement of Theorem 2.2.3 holds for any* $L_\varphi(\bar{u})$ *that is a (possibly conservative) upper bound on the Lipschitz constant of the system trajectories with respect to* (x, w). *To compensate for conservative values of* $L_\varphi(\bar{u})$, *smaller values of* ε *is chosen, which will require taking higher number of samples* N.

Remark 3 *We provide an algorithm in the next subsection for estimating* L_φ *using sampled trajectories of the system. Note that as the above proof shows, the estimated quantity* $\theta_1 = L_\varphi \mathbf{1}_{n\times n}$ *can be used to construct the abstraction, but this would give conservative results. We will demonstrate this observation on a case study in Subsection 2.2.5.*

corollary 2.2.1 *The abstract model constructed using the growth bounds as solutions of* SCP_γ *with confidence* $(1 - \beta/(n_x n_u))$ *for state-input pairs* $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$ *is a valid abstract model for* Σ *with confidence at least* $(1 - \beta)$.

Proof *Denote the optimal solution of* SCP_γ *in* (2.14) *by* θ^*. *The ball centered at* $z(\bar{x}, \bar{u}) := x'_{nom}$ *with radius* $\lambda(\bar{x}, \bar{u}) = \kappa_{\theta^*}(\eta_x, \bar{x}, \bar{u}) + \gamma$ *is a valid overapproximation of the reachable set from the state-input pair* $(\bar{x}, \bar{u})$ *with confidence at least* $1 - \beta/(n_x n_u)$. *Since the number of pairs* $(\bar{x}, \bar{u})$ *is* $n_x n_u$, *the chance of getting an invalid growth bound in at least one instance of* SCP_γ *is bounded by* β. *Therefore, we get a sound abstraction that truly overapproximates the behaviour of the system with confidence* $(1 - \beta)$.

Remark 4 *The parameter* $\varepsilon \in [0, 1]$ *gives a tradeoff between the required number of samples and the level of conservativeness applied to the SCP. Smaller* ε *results in a larger number of sample trajectories, but reduces the value of* γ *in* (2.15) *(less conservative constraints in the SCP and higher chance of finding a feasible solution). In contrast, larger* ε *results in a smaller number of sample trajectories but increases the value of* γ.

Remark 5 *The quantity* $2n$ *used in* (2.15) *is in fact the dimension of the sample space* $\mathcal{D} = \Omega_{\eta_x}(\bar{x}) \times W$. *If the system does not have any disturbance (i.e., the system can be modeled as an ODE having deterministic trajectories), the sample space will be* $\mathcal{D} = \Omega_{\eta_x}(\bar{x})$ *and its dimension* n *can be used in* (2.15)*:* $\gamma = 4L_\varphi(\bar{u}) \sqrt[n]{\varepsilon \prod_{i=1}^{n} \eta_x(i)}$. *This will substantially reduce the number of required sample trajectories. Similarly, if the disturbance does not affect some of the state equations,* $2n$ *can be replaced by* $(n + q)$ *where* q *is the dimension of the disturbance set considered as a non-zero measure set.*

Algorithm 1 uses the result of Corollary 2.2.1 to provide an algorithmic solution for Problem 2.2. This algorithm receives a confidence parameters β, divides it by the cardinality of $\bar{X} \times \bar{U}$ (i.e., $n_x n_u$), computes the growth bounds for each pair $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$ using the SCP$_\gamma$ in (2.14) with confidence $1 - \beta/(n_x n_u)$, and constructs the abstraction using these growth bounds.

Algorithm 1: Data-Driven Abstraction

Data: (X, U, W) of a control system Σ, confidence β, discretisation parameters η_x, η_u

Compute the finite state and input sets $\bar{X}$ and $\bar{U}$ using η_x, η_u;
Define n_x and n_u as cardinalities of $\bar{X}$ and $\bar{U}$;
Choose $\varepsilon \in [0, 1]$;
Set $N = N(\varepsilon, \frac{\beta}{n_x n_u})$ using Eq. (2.8);
Compute γ using Eq. (2.15);
for $\bar{x} \in \bar{X}$ **do**
  for $\bar{u} \in \bar{U}$ **do**
    $T_F(\bar{x}, \bar{u}) = \emptyset$;
    Consider the uncertainty space $\mathcal{D} = \Omega_{\eta_x}(\bar{x}) \times W$;
    Select N i.i.d sample trajectories using uniform distribution over $\mathcal{D}$;
    Simulate the nominal trajectory $(\bar{x}, \bar{u}, x'_{nom})$;
    Solve the SCP$_\gamma$ (2.14) to get the optimiser $\theta^*(\bar{x}, \bar{u})$;
    $z \leftarrow x'_{nom}$;
    $\lambda \leftarrow \kappa_{\theta^*}(\eta_x, \bar{x}, \bar{u}) + \gamma$;
    Find all states $\bar{x}' \in \bar{X}$ for which $\Omega_{\eta_x}(\bar{x}') \cap \Omega_\lambda(z) \neq \emptyset$ and add them to $T_F(\bar{x}, \bar{u})$;
  end
end

Result: $\bar{\Sigma} = (\bar{X}, \bar{U}, T_F)$ as a finite abstraction of Σ with confidence $(1 - \beta)$, $\theta^*(\bar{x}, \bar{u})$ as a growth bound for $\bar{x} \in \bar{X}, \bar{u} \in \bar{U}$

The finite abstraction $\bar{\Sigma}$ constructed by Algorithm 1 is a valid abstraction for Σ with confidence $(1-\beta)$. This means any controller $\bar{C}$ synthesized on $\bar{\Sigma}$ and refined to a controller C for Σ will satisfy the desired specification with confidence $(1 - \beta)$ on the closed loop system $\Sigma \parallel C$. In the next subsection, we extend our approach to make it suitable for abstraction refinement in case there is no controller $\bar{C}$ satisfying the specification due to the conservatism of the approach.

Lipschitz Constant Estimation

For estimating the Lipschitz constant L_φ in (2.17), we estimate an upper bound for the fraction

$$\Delta(\bar{u}) := \frac{\|\varphi(x, \bar{u}, w) - \varphi(x', \bar{u}, w')\|}{\|(x, w) - (x', w')\|}$$

that holds for all $x, x' \in X$ and $w, w' \in W$. We follow the line of reasoning in [188, 184] and use the extreme value theory for the estimation.

Let us fix a $\delta > 0$ and assign uniform distribution to the pairs (x, w) and (x', w') over the domain

$$\{x, x' \in X,\ w, w' \in W \text{ with } \|(x, w) - (x', w')\|_\infty \leq \delta\}. \tag{2.18}$$

Then $\Delta(\bar{u})$ is a random variable with an unknown cumulative distribution function (CDF). Based on the assumption of Lipschitz continuity of the system, the support of the distribution of $\Delta(\bar{u})$ is bounded from above, and we want to estimate an upper bound for its support. We take $\mathfrak{n}$ sample pairs (x, w) and (x', w'), and compute $\mathfrak{n}$ samples $\Delta_1, \Delta_2, \ldots, \Delta_\mathfrak{n}$ for $\Delta(\bar{u})$. The CDF of $\max\{\Delta_1, \Delta_2, \ldots, \Delta_\mathfrak{n}\}$ is called the limit distribution of $\Delta(\bar{u})$. Fisher-Tippett-Gnedenko theorem [71, 130] says that if the limit distribution exists, it can only be one of the three family of extreme value distributions – the Gumbel class, the Fréchet class, and the reverse Weibull class. These CDF's have the following forms:

$$\text{Gumbel class:} \quad G(s) = \exp\left[-\exp\left[\frac{s-a}{b}\right]\right],\ s \in \mathbb{R}$$

$$\text{Fréchet class:} \quad G(s) = \begin{cases} 0 & \text{if } s < a \\ \exp\left[-[\frac{s-a}{b}]^{-c}\right] & \text{if } s \leq a \end{cases}$$

$$\text{Reverse Weibull class:} \quad G(s) = \begin{cases} \exp\left[-[\frac{a-s}{b}]^{c}\right] & \text{if } s < a \\ 1 & \text{if } s \leq a \end{cases}$$

where $a \in \mathbb{R}, b > 0, c > 0$ are respectively the location, scale and shape parameters of the distributions.

Among the above three distributions, only the reverse Weibull class has a support bounded from above. Therefore, the limit distribution of $\Delta(\bar{u})$ will be from this class and the location parameter a is such an upper bound. As a result, we can estimate the location parameter of the limit distribution of $\Delta(\bar{u})$ to get an estimation of the Lipschitz constant.

The approach is summarized in Algorithm 2. The most inner loop computes samples of $\Delta(\bar{u})$. The middle loop computes samples of $\max\{\Delta_1, \ldots, \Delta_\mathfrak{n}\}$. The outer loop estimates the Lipschitz constant for each $\bar{u}$ by fitting a reverse Weibull distribution.

Remark 6 *The approach presented above can only be used for* estimating *the Lipschitz constant, which can then be enlarged by a factor greater than one to account for the effect of estimation using finite number of samples. Note that this factor can be selected depending on the system under study by fitting the Reverse Weibull distribution to datasets of varying size and observing its convergence behavior.*

2.2.4. Synthesis via Abstraction Refinement

The data-driven synthesis discussed in Subsection 2.2.3 inherits the soundness property from the ABCD approach: they both work with overapproximations of the dynamics and

Algorithm 2: Lipschitz Constant Estimation

Data: (X, U, W) of a control system Σ, abstract input space $\bar{U}$
Select number of samples $\mathfrak{n}$ and $\mathfrak{m}$ for the estimation
Select $\delta > 0$
for $\bar{u} \in \bar{U}$ **do**
  for $j = 1 : \mathfrak{m}$ **do**
    for $i = 1 : \mathfrak{n}$ **do**
      Sample pairs $(x, w), (x', w')$ uniformly from the domain in (2.18)
      Run Σ to get trajectories $\varphi(x, \bar{u}, w)$ and $\varphi(x', \bar{u}, w')$
      Compute $\Delta_i := \frac{\|\varphi(x,\bar{u},w) - \varphi(x',\bar{u},w')\|}{\|(x,w)-(x',w')\|}$
    end
    $\Gamma_j := \max\{\Delta_1, \ldots, \Delta_{\mathfrak{n}}\}$
  end
  Fit a reverse Weibull distribution to the sample set $\{\Gamma_1, \Gamma_2, \ldots, \Gamma_{\mathfrak{m}}\}$
  $L_\varphi(\bar{u})$ is the location parameter of the fitted distribution
end

Result: Estimated value of $L_\varphi(\bar{u})$ for all $\bar{u} \in \bar{U}$

may not return a controller despite one may exists. Therefore, there is a need for refining the abstraction in order to check for controllers using less conservative abstractions. While the method of Subsection 2.2.3 is good for a given fixed discretization parameter η_x, it is not suitable for reducing η_x, which requires re-computing all local parameters of the growth bounds $\theta_1(\bar{x}, \bar{u}), \theta_2(\bar{x}, \bar{u})$. Another shortcoming of the method is related to the data collection: the nominal trajectories of the system should be available and are used in the constraints of the SCP. In this subsection, we discuss an extension of the approach of Subsection 2.2.3, in order to

- enable reducing η_x without the need for re-computing the growth bound, and
- relax the assumption of having access to the nominal trajectories of the system.

Let us define a modified growth bound as a function $\kappa^e \colon \mathbb{R}^n_{\geq 0} \times \bar{X} \times \bar{U} \to \mathbb{R}^n_{\geq 0}$ that is strictly increasing in its first argument and satisfies

$$|\varphi(x_1, \bar{u}, w_1) - \varphi(x_2, \bar{u}, w_2)| \leq \kappa^e(|x_1 - x_2|, \bar{x}, \bar{u})$$
$$\forall \bar{x} \in \bar{X},\ \forall \bar{u} \in \bar{U},\ \forall x_1, x_2 \in \Omega_{\eta_x}(\bar{x}),\ \forall w_1, w_2 \in W. \tag{2.19}$$

This definition is more conservative than (2.9) in comparing trajectories under two arbitrary disturbances, and we always have that κ^e satisfies (2.9). Using this new definition, for every pair of abstract state and input $(\bar{x}, \bar{u})$, the corresponding overapproximation of the reach set can be computed as a ball centered at *any* $z(\bar{x}, \bar{u}) \in \Phi(\bar{x}, \bar{u})$ with radius $\lambda(\bar{x}, \bar{u}) = \kappa^e(\eta_x, \bar{x}, \bar{u})$.

we choose a parametrization for κ^e similar to (2.12), i.e.,

$$\kappa^e_\theta(r, \bar{x}, \bar{u}) = \theta_1(\bar{x}, \bar{u})r + \theta_2(\bar{x}, \bar{u}), \tag{2.20}$$

where $r \in \mathbb{R}_{\geq 0}$, $\theta_1 \in \mathbb{R}^{n\times n}$, $\theta_2 \in \mathbb{R}^n$, and $\theta \in \mathbb{R}^{n^2+n}$ is constructed by concatenating columns of θ_1 and θ_2. The SCP associated with this growth bound is constructed by considering a uniform distribution over $\Omega_{\eta_x}(\bar{x})\times W$ and obtain $2N$ i.i.d. sample trajectories $\mathcal{S}_{2N} = \{(x_i, \bar{u}_i, x'_i) \,|\, x'_i \in \Phi(x_i, \bar{u}), i = 1, 2, \ldots, 2N\}$ so that every x'_i corresponds to a random disturbance $w_i \in W$. The modified SCP$_\gamma$ is defined as

$$\begin{cases} \min c^\top \theta \\ s.t.\ 0 \leq \theta \leq \bar{\theta} \text{ and } \forall i \in \{1, \ldots, N\} \\ |x'_{2i-1} - x'_{2i}| - \theta_1(\bar{x}, \bar{u})|x_{2i-1} - x_{2i}| - \theta_2(\bar{x}, \bar{u}) + \gamma \leq 0 \end{cases}$$

where $c = [1, 1, \ldots, 1] \in \mathbb{R}^{n^2+n}$ is a constant vector, $\bar{\theta} \in \mathbb{R}^{n^2+n}_{>0}$ is sufficiently large, and $\gamma \geq 0$.

Theorem 2.2.4 *For any $\bar{x} \in \bar{X}$ constructed with the discretization size η_x, any $\bar{u} \in \bar{U}$, and the disturbance set $W = [-\bar{w}, \bar{w}]$, the optimal solution of (2.21) gives a growth bound for the system Σ corresponding to $(\bar{x}, \bar{u})$ that satisfies (2.19) with confidence $(1-\beta)$, when the number of samples $2N \geq N(\varepsilon, \beta)$ and*

$$\gamma = 8L_\varphi \sqrt[4n]{\varepsilon \left[\prod_{i=1}^{n} \eta_x(i) \prod_{i=1}^{n} \bar{w}(i)\right]^2}, \tag{2.21}$$

where $\varepsilon \in [0, 1]$, ,n is the dimension of the state space, and $L_\varphi(\bar{u})$ is the Lipschitz constant of the system trajectories $\varphi(x, \bar{u}, w)$ with respect to (x, w).

Proof *The proof of this theorem is similar to that of Theorem 2.2.3. Define*

$$\begin{aligned} g(\theta, x_1, w_1, x_2, w_2) := \max\{&|\varphi(x_1, \bar{u}, w_1) - \varphi(x_2, \bar{u}, w_2)| \\ &- \theta_1(\bar{x}, \bar{u})|x_1 - x_2| - \theta_2(\bar{x}, \bar{u})\}. \end{aligned}$$

To satisfy the inequality (2.6), we can choose

$$h(\varepsilon) = \mathbb{P}(\Omega_\varepsilon(d)) = \frac{(\varepsilon/2)^{4n}}{[\prod_{i=1}^{n} \eta_x(i) \prod_{i=1}^{n} \bar{w}(i)]^2},$$

since the distribution on $(\Omega_{\eta_x}(\bar{x}) \times W)^2$ is uniform. Using Equation (2.7), we have $\gamma = L_d h^{-1}(\varepsilon)$. In order to prove that γ takes the value in (2.21), we must show that g is Lipschitz continuous with constant $L_d = 4L_\varphi(\bar{u})$. Bounding $\|\theta_1(\bar{x}, \bar{u})\|$ by L_φ, for all (x_1, w_1, x_2, w_2) and (x'_1, w'_1, x'_2, w'_2) we have

$$\begin{aligned} &\|g(\theta, x_1, w_1, x_2, w_2) - g(\theta, x'_1, w'_1, x'_2, w'_2)\|_\infty \\ &\leq \|\varphi(x_1, \bar{u}, w_1) - \varphi(x'_1, \bar{u}, w'_1)\|_\infty \\ &+ \|\varphi(x_2, \bar{u}, w_2) - \varphi(x'_2, \bar{u}, w'_2)\|_\infty \\ &+ \|\theta_1(\bar{x}, \bar{u})\|_\infty(\|x_1 - x'_1\|_\infty + \|x_2 - x'_2\|_\infty) \\ &\leq 4L_\varphi(\bar{u})\|(x_1, w_1, x_2, w_2) - (x'_1, w'_1, x'_2, w'_2)\|_\infty. \end{aligned}$$

Table 2.1.: Results for the DC-DC boost converter.

Case-study	Dimension		Disturbance	Fixed Discretisation		
	X	U	W	N	time (min)	$\lvert\mathcal{V}\rvert$
DC-DC boost converter	2	1	$\{0\}$	$1,807$	22.2	$37,783$
			$[-0.01, 0.01]$	$2,285$	30.6	$37,414$

Therefore, g is Lipschitz continuous with constant $4L_\varphi(\bar{u})$. This completes the proof.

A statement similar to Corollary 2.2.1 holds for the growth bound computed using (2.21).

2.2.5. Experimental Evaluation

To demonstrate our approach, we apply it to a DC-DC boost converter and a path planning problem. These case studies are taken from [152, 68] and will be used as black-box models to generate sample trajectories. We also introduce a case study from power systems based on [112], that is implemented in the Power System Toolbox (PST) [40]. We will use trajectories from the black-box reduced model of the 30 state power system model. We apply our approach to construct finite abstractions of these systems and employ SCOTS [152] to design controllers. Our algorithms are implemented in C++ on a 64-bit Linux cluster machine with two Intel Xeon E5 v2 CPUs, 1866 MHz, and 50GB RAM.

DC-DC Boost Converter

The objective in the DC-DC boost converter problem is to design a controller to enforce a reach and stay specification. The DC-DC boost converter can be modeled as a two dimensional linear switching system with two functional modes. The state vector of the system at time $t \in \mathbb{R}_{\geq 0}$ is $x_t = (i_{l_t}, v_{c_t}(t))$, where i_l is the inductor current and v_c is the capacitor voltage. The system's evolution can be controlled by selecting the appropriate mode $u_t \in \{1, 2\}$ at every time $t \in \mathbb{R}_{\geq 0}$. The system's dynamics under the two modes can be represented as $\dot{x} = A_{u_t} x_t + b + c w_t$, $u \in \{1, 2\}$, with matrices A_1, A_2, b, c as reported in [68]. The state and input spaces are $X = [0.65, 1.65] \times [4.95, 5.95]$ and $U = [1, 2]$. The initial state is $(i_{l_0}, v_{c_0}) = (0.7, 5.4)$ and the target set is $[1.1, 1.6] \times [5.4, 5.9]$. The target set is shown in red color in Figure 2.2.

Our implementation results are reported in Table 2.1 for the system without disturbance ($\bar{w} = (0, 0)$) and with disturbance bound $\bar{w} = (0.01, 0)$. These results are obtained with discretization parameters $\eta_x = (0.005, 0.005)$ and $\eta_u = 1$, confidence parameter $\beta = 0.01$, $\varepsilon = 0.01$ and estimation for $L_\varphi = 0.9935$. The resulted finite abstraction has cadinalities $n_x = 40,000$ and $n_u = 2$. The required number of sample trajectories, N, for each $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$ is computed using equation (2.8). Runtimes and the resulting winning region sizes, $|\mathcal{V}|$, for the DC-DC boost converter are given in Table 2.1.

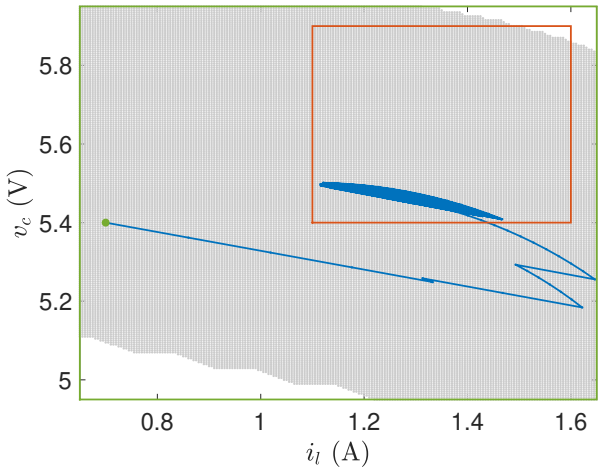

Figure 2.2.: The closed-loop trajectory of the DC-DC boost converter with $\bar{w} = (0, 0)$ under the controller designed by our data-driven abstraction approach. The rectangle in red color represents the target region and the area in gray shows the winning region of the controller.

We have used Algorithm 1 to compute the finite-state abstraction by collecting sample trajectories of the system. Subsequently, SCOTS is used for designing the controller. The performance of the controller is shown in Figures 2.2 and 2.3 for the system without and with the disturbance. These figures show one sample closed-loop trajectory of the system under the controllers designed by our data-driven ABCD approach. In both cases, without and with disturbance, it can be noticed from Figures 2.2 and 2.3 that our approach has been successful in finding controllers satisfying the given reach and stay specification, despite the the dynamics being unknown.

Path Planning Problem with Partition Refinement

We consider a path planning problem for a vehicle that is modeled as

$$\begin{aligned} \dot{x} &= v\cos(\alpha+\theta)/\cos(\alpha) + w \\ \dot{y} &= v\sin(\alpha+\theta)/\cos(\alpha) \\ \dot{\theta} &= v\tan(\omega), \end{aligned} \tag{2.22}$$

where the state variables x, y, θ represent the position of the vehicle in the 2-dimensional space and the orientation of the vehicle, respectively. Inputs are (v, ω), the disturbance is w, and $\alpha := \arctan(\tan(\omega)/2)$. The state and input spaces are $X = [0, 10] \times [0, 10] \times [-\pi - 0.4, \pi + 0.4]$ and $U = [-1, 1]^2$, respectively. The goal is to find a controller to steer the vehicle from the initial state $(x_0, y_0, \theta_0) = (0, 1.2, 0)$ to the target set $(x, y) \in [9, 9.51] \times [0, 0.51]$ while avoiding the obstacles. These obstacles are shown in blue color in Figures 2.4 and 2.5.

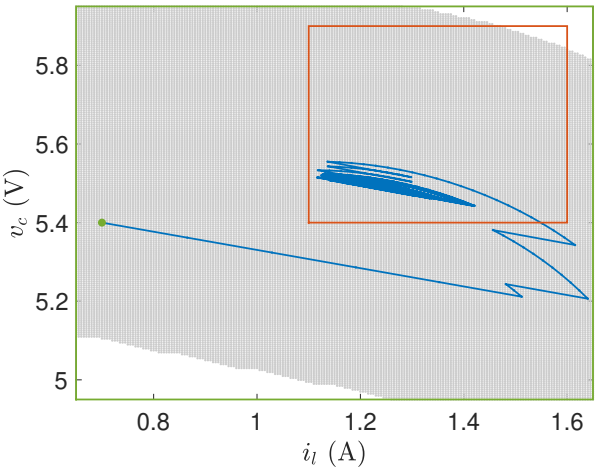

Figure 2.3.: The closed-loop trajectory of the DC-DC boost converter with $\bar{w} = (0.01, 0)$ under the controller designed by our data-driven abstraction approach. The rectangle in red color represents the target region and the area in gray shows the winning region of the controller.

Table 2.2.: Results for the path planning case study.

Case-study	Dimension		Disturbance	Abstraction Refinement		
	X	U	$\bar{w}$	N	time (min)	$\lvert\mathcal{V}\rvert$
Path planning	3	2	$(0, 0, 0)$	$3,127$	225	$405,493$
			$(0.01, 0, 0)$	$4,277$	513	$447,212$

We computed the growth bounds with a coarse discretization $\eta_x = (1.6, 1.6, 1.6)$ and reduced it iteratively with the factor of two. The algorithm successfully finds a controller for the system after five iterations. The implementation results are reported in Table 2.2. These results are obtained with $\eta_u = (0.3, 0.3)$, the confidence parameter $\beta = 0.01$, $\varepsilon = 0.01$ and estimated constant $L_\varphi = 1.46$. The resulted abstraction has cardinalities $n_x = 88,500$ and $n_u = 24$. For the case of disturbance-free model we set $\bar{w} = (0, 0, 0)$, and for the case of dynamics with disturbance, we set $\bar{w} = (0.01, 0, 0)$. The required number of sample trajectories for each $(\bar{x}, \bar{u})$ is computed using Equation (2.8) and marked with N in the table. Finally, runtimes and size of the winning regions $|\mathcal{V}|$ are reported.

We have used the synthesis method based on abstraction refinement presented in Subsection 2.2.4, to construct the finite-state abstraction by collecting sample trajectories of the system. We used SCOTS to design the controller fulfilling the given specification. The performance of the controller is shown in Figures 2.4 and 2.5 for the system without and with the disturbance, respectively. These figures compare the closed-loop trajectories

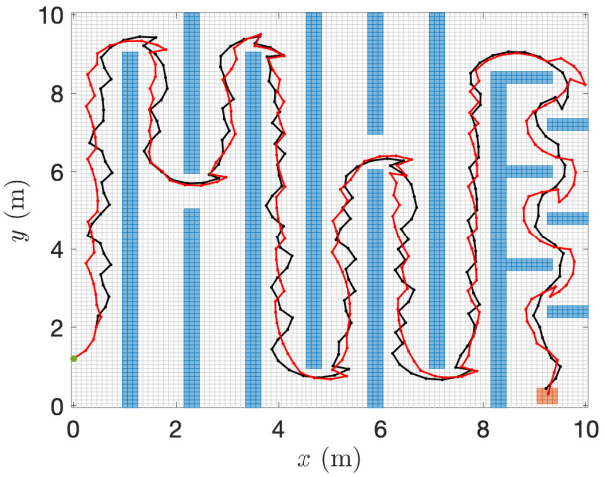

Figure 2.4.: Comparison between the closed-loop trajectories of the system (2.22)without disturbance under the controllers designed by our data-driven abstraction refinement approach (black) and by the model-based approach of SCOTS (red). Blue blocks represent the obstacles, the green dot represents the initial state, and the orange rectangle shows the target region.

of the system under the controllers designed by our data-driven abstraction refinement algorithm approach (black) and by the model-based approach of SCOTS (red). Our data-driven approach successfully finds a controller for the system that satisfies the specification without the need for knowing the dynamics of the system.

Three Area Three Machine Power System

We consider a three area three machine (3A3M) power system adapted from [112] and is shown in Figure 2.6. The system consists of three buses, which are each connected to a power source (generator) and a load. At bus 1 we consider a load which is bidirectional, meaning it can both draw power and inject power into the system. The loads at buses 2 and 3 can only draw power from the system; when these loads increase, more power will be drawn from the system, causing an imbalance between generation and consumption which may result in a reduction of the network frequency. The nominal frequency of the network is set to 60 Hz.

We consider a worst-case scenario when a sudden increase occurs in the loads at buses 2 and 3 by 0.2 and 0.3 per unit (pu), respectively. The control task is for the load at bus 1 to balance the load increase at buses 2 and 3 by either reducing its load or injecting power into the network. The simulation is run using PST on a 30 state model of this power system. Balanced realization of the system reduces its dynamics to three states. To compute the data-driven finite abstraction, sample trajectories are gathered using a black-box approach of the reduced system representation for the original model. The

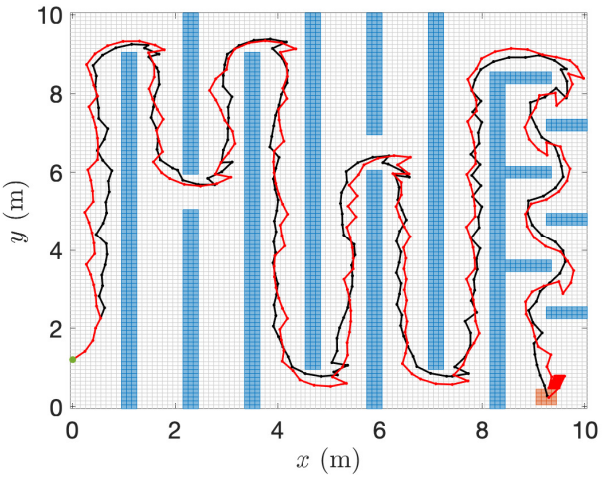

Figure 2.5.: Comparison between the closed-loop trajectories of the system (2.22) with disturbance bound $\bar{w} = (0.01, 0, 0)$ under the controllers designed by our data-driven abstraction refinement approach (black) and by the model-based approach of SCOTS (red).

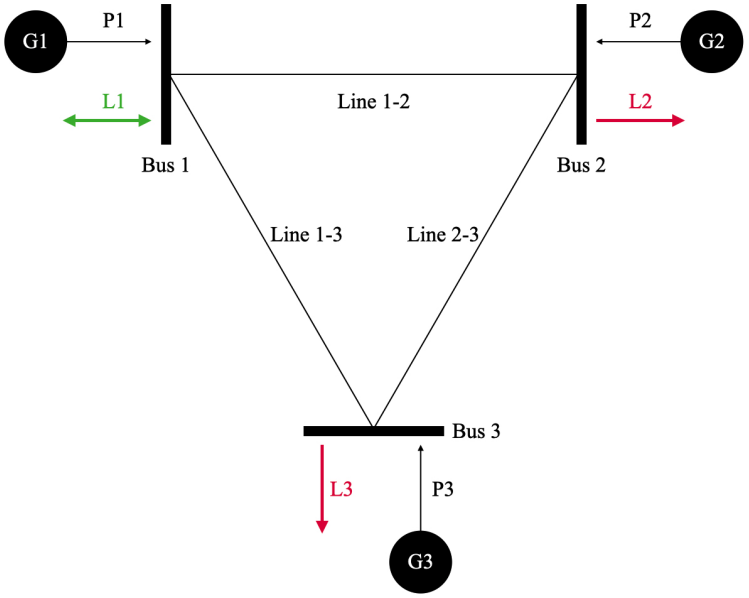

Figure 2.6.: 3A3M power system with generators (G) and loads (L). L1 represents a bidirectional load such as Electric Vehicles or Energy Storage Systems.

dynamics of the reduced system are given by

$$\begin{aligned} \dot{x} &= Ax + Bu + Ew \\ y &= Cx, \end{aligned} \tag{2.23}$$

where

$$A = \begin{bmatrix} 0.00027563 & 0 & 0 \\ 0 & -0.3951 & 0.687 \\ 0 & -0.6869 & -0.016 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.00031166 \\ 0.1359 \\ 0.0230 \end{bmatrix}$$

$$E = \begin{bmatrix} 0.00033103 & 0.00031244 \\ 0.1309 & 0.1308 \\ 0.0250 & 0.0233 \end{bmatrix}$$

$$C = \begin{bmatrix} -0.0115 & -0.2296 & 0.0412 \end{bmatrix}. \tag{2.24}$$

The state and input spaces are $X = [-0.02, 0.02] \times [-0.05, 0.05] \times [-0.12, 0.12]$ and $U = [0, 0.5]$. Further, we set $W = [-0.2, 0.2] \times [-0.3, 0.3]$, $\eta_u = 0.025$, $\tau = 0.4$, $\eta_x = (0.0015, 0.0015, 0.0015)$, $\beta = 0.01$ and $\varepsilon = 0.01$. The resulted abstraction has $n_x = 228,480$ and $n_u = 20$. The estimated Lipschitz constant is $L_\varphi = 1.5715$. The target set is given by $-0.008 < y < 0.008$ and the avoid set is given by $y < -0.015$. Multiplying by the nominal frequency to get the specification in Hertz, the target region is $[59.52, 60.48]$ and the avoid region is $(-\infty, 59.1)$. Figure 2.7 shows that the specification is violated when no control is applied.

We apply the data-driven approaches of Subsection 2.2.3 (fixed discretization) and Subsection 2.2.4 (abstraction refinement). Both controllers are synthesised with disturbance $W = [-0.2, 0.2] \times [-0.3, 0.3]$. A comparison of the two control approaches is shown in Table 2.3. The required number of sample trajectories for each $(\bar{x}, \bar{u})$ is computed using equation (2.8) and marked with N in the table. The abstraction refinement starts with $\eta_x = 0.012$ and refines the discretization iteratively with a factor of two. The algorithm successfully finds a controller after five iterations. The runtimes and the resulting winning region sizes $|\mathcal{V}|$ are also given in Table 2.3. The abstraction refinement synthesises the controller a factor of 100 times faster than the fixed discretization by iteratively decreasing the value of η_x.

The data-driven control approach with fixed discretization is simulated in PST and is reported in Figures 2.8 and 2.9. The controlled system successfully keeps the frequencies of the three areas outside of the avoid set (i.e., always above 59.1 Hz) and bring them back to the target set (i.e., above 59.52 Hz). Figure 2.9 shows the load changes in the system. Load at bus 1 is able to maintain the frequencies of the three areas above the avoid region and facilitate the system returning to the target set for the maximum disturbances applied at buses $2, 3$. Figures 2.10 and 2.11 show the results of simulating the system in PST with the control obtained from the abstraction refinement approach. The controlled system has the same performance in satisfying the specification.

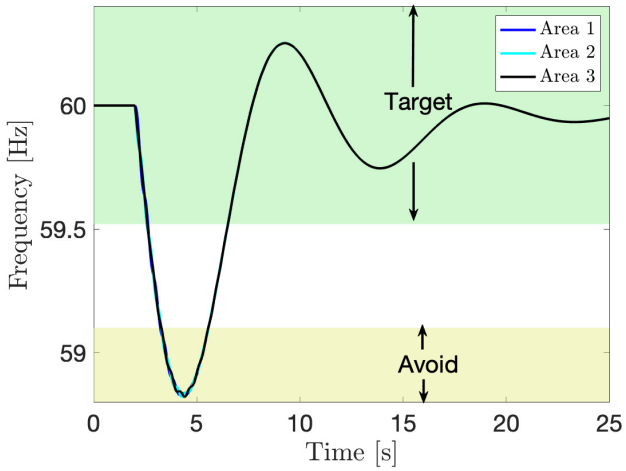

Figure 2.7.: 3A3M power system frequency without applying any control input. The frequency falls below 59.1 Hz thus violates the specification.

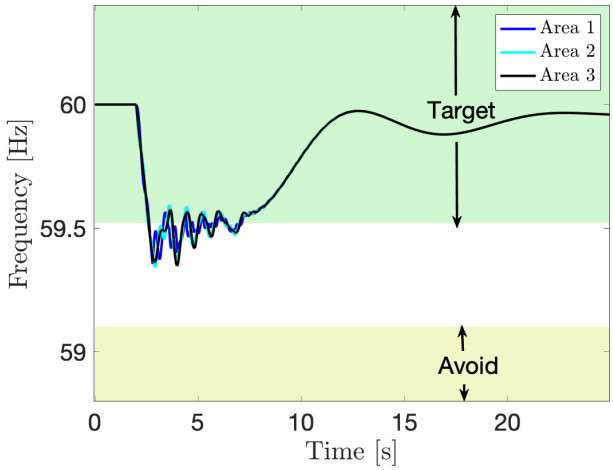

Figure 2.8.: 3A3M power system frequencies for the three areas, with the frequency of an area is measured at the corresponding bus in that area. The control synthesized by the fixed discretization approach successfully keeps the frequencies of the three areas outside of the avoid set. The frequencies leave the target set for around 4.4 seconds before staying in the target set.

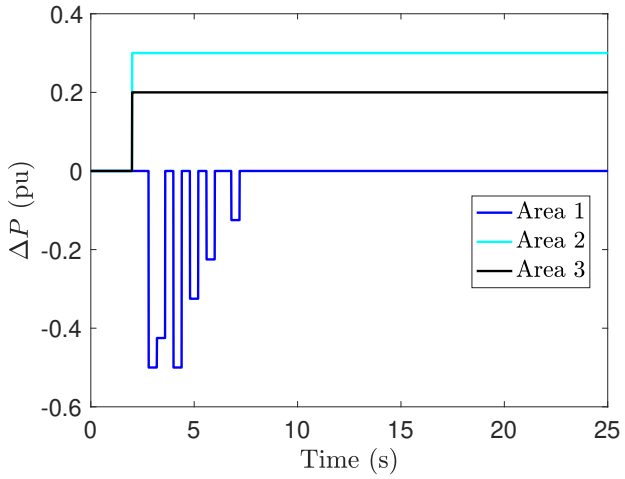

Figure 2.9.: 3A3M power system load changes for the three areas. Loads at buses 2 and 3 increase by 0.3 and 0.2 pu, respectively. Load at bus 1 is used to control the frequency using our data-driven approach with fixed discretization.

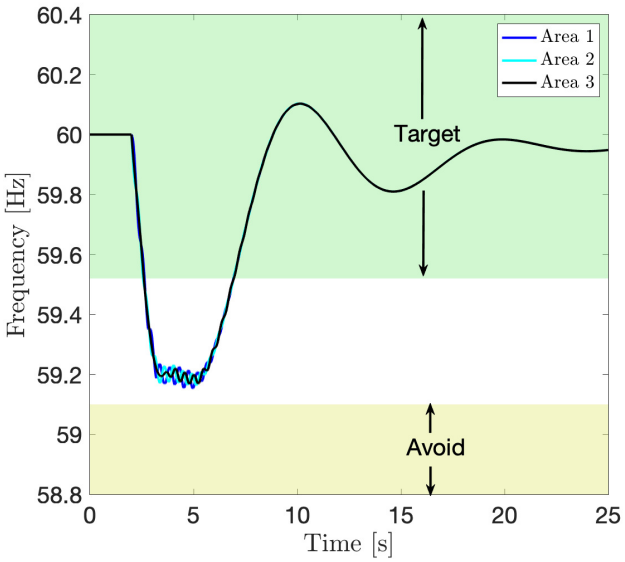

Figure 2.10.: 3A3M power system frequencies for the three areas, with the frequency of an area is measured at the corresponding bus in that area. The control synthesized by the abstraction refinement approach successfully satisfies the specification. The frequencies leave the target set for around 4.2 seconds before staying in the target set.

Table 2.3.: Results for the 3A3M power system.

Control Approach	Dimension X	Dimension U	Disturbance $\bar{w}$	N	time (min)	$\lvert\mathcal{V}\rvert$
Fixed Discretisation	3	1	$(0.2, 0.3)$	$3,290$	$5,253$	$230,760$
Adaptive Refinement			$(0.2, 0.3)$	$4,460$	50.25	$314,802$

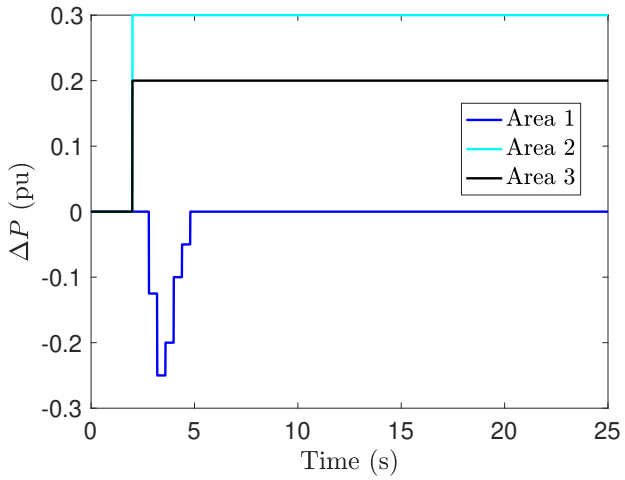

Figure 2.11.: 3A3M power system load changes for the three areas. Loads at buses 2 and 3 increase by 0.3 and 0.2 pu, respectively. Load at bus 1 is used to control the frequency using our data-driven approach with abstraction refinement.

Comparison with PAC Learning

We want to compare our approach with the results provided by Xue et al. [191] that is based on probably approximately correct (PAC) learning on the 3A3M power system case study. The PAC approach is designed for finite-horizon problems, and cannot deal with infinite-horizon problems. At each step of the PAC approach the error increases by a non-zero factor tending towards 1. Over an infinite horizon, the error associated with the PAC approach becomes too large. Additionally the formulation of the samples required in the PAC method is inversely proportional to its error level. If we translate their problem formulation to our problem formulation, the error level would need to be zero meaning the samples required would be infinity. Therefore the tradeoff in our approach is we require more samples than PAC, but can provide stronger guarantees. The abstraction approach of [191] has no bias term γ, but uses confidence parameter $\beta \in (0, 1)$, error level $\nu \in (0, 1)$, and cardinality of the parameter vector θ denoted by $q \in \mathbb{N}$. The required number of samples is

$$N \geq \frac{2}{\nu}(ln\frac{1}{\beta} + q), \tag{2.25}$$

Table 2.4.: Comparing the winning domain of controllers obtained from our RSA method, PAC method of [191], and the model-based approach of [146]. The pairwise comparison is made by computing the intersections ($\cap$) and set differences (row $\setminus$ column). The results are reported both in cardinalities and percentages.

Winning Domain	RSA		PAC		Model-based	
	$\cap$	$\setminus$	$\cap$	$\setminus$	$\cap$	$\setminus$
RSA	230,760	0	230,760	0	230,760	0
%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%
PAC	230,760	15,664	246,424	0	245,345	1,079
%	93.64%	6.36%	100.00%	0.00%	99.56%	0.44%
Model-based	230,760	22,216	245,345	7,631	252,976	0
%	91.22%	8.78%	96.98%	3.02%	100.00%	0.00%

which allows the constructed abstraction to hold for the entire state space except a subset measured by parameter ν. If we attempt to translate the PAC approach into our approach, the value ν would need to be set to zero. Note that as ν tends towards zero, the sample number required will reach infinity. Meaning it is impossible to completely remove the error with finite sampling. Therefore, the PAC approach is not usable for the situations we consider, and although it requires less sampling, it provides weaker guarantees.

We implement our data-driven robust scenario approach (RSA), the PAC approach in [191] with parameters $\beta = 0.01$ and $\nu = 0.01$, and the model-based approach of [146]. Table 2.4 compares the winning domain of the controllers by reporting the intersections ($\cap$) and set differences (row $\setminus$ column). It can be seen that the winning domain obtained by our RSA method is a subset of the ones computed by PAC and the model-based approaches. This shows that our approach is more conservative than the model-based approach but correctly finds a subset of the winning domain. In contrast, the PAC approach gives a winning domain that includes states not identified as winning by the model-based approach. It includes 1079 states outside of the winning domain obtained by the model-based approach. Due to the nature of the PAC learning, some of these states are incorrectly identified as winning. The main reason is that the PAC method may miss capturing some of the transitions and does not always generate an overapproximation of the system behaviors. Among these 1079 states, a counter-example can be found, demonstrating a lack of guarantee provided by the PAC method. At state $(0.0187, 0.0262, -0.1163)$ the PAC controller calculates $u = -0.075$ to be an input which will transition to a safe state under any disturbances. However, the system under disturbances $W_1 = 0.2$ and $W_2 = 0.3$ will lead to the state $(0.0188, 0.0131, -0.1167)$ that is outside of the winning domain of the controller. In comparison, the winning domain provided by our RSA method is a subset of the one from the model-based method and provides full guarantees of the satisfaction

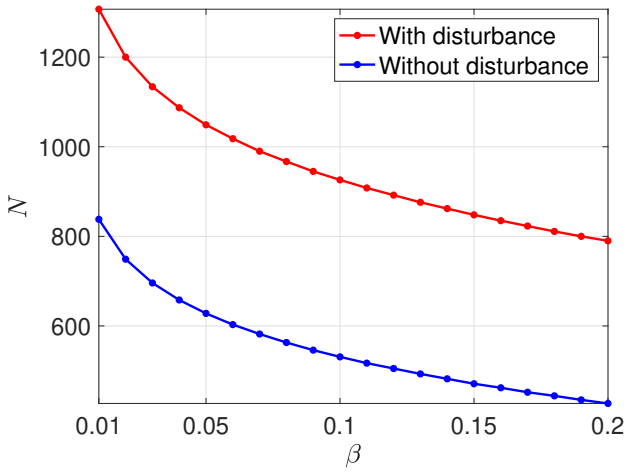

Figure 2.12.: Required number of samples for our approach as a function of β for a fixed $\varepsilon = 0.01$.

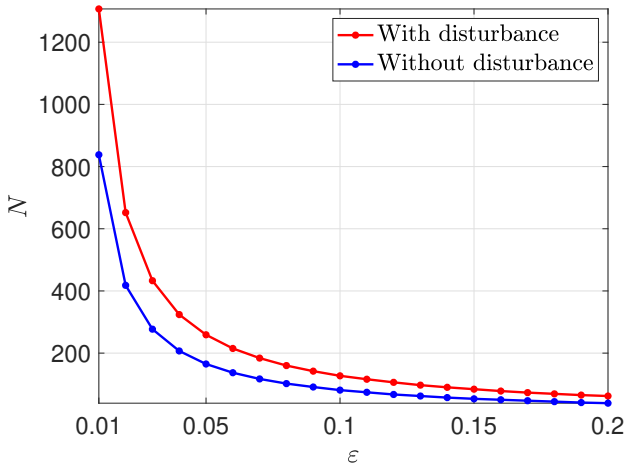

Figure 2.13.: Required number of samples for our approach as a function of ε for a fixed $\beta = 0.01$.

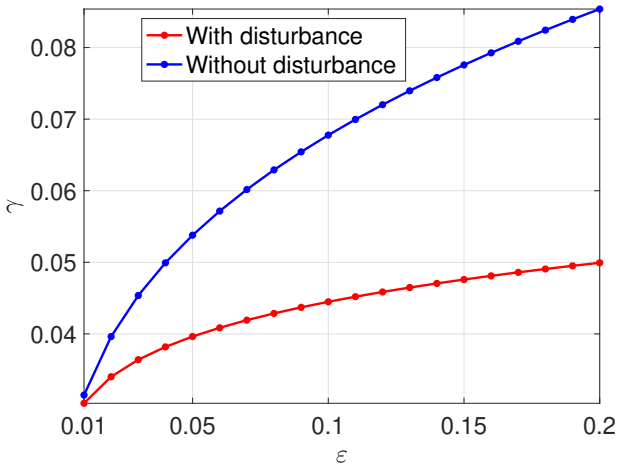

Figure 2.14.: The bias term γ as a function of ε.

of the specification and correctness of the controller. This guarantee is obtained at the cost of an increased number of samples and a bias term included in the growth bound calculations, which makes the controller more conservative.

As a final point on this case study, note that our sampling approach uses the Lipschitz constant estimated using sample trajectories. This Lipschitz constant can in turn be used to construct the abstraction. The direct use of the estimated Lipschitz constant does not provide a formal guarantee as it is an estimated value that converges to the true value only in the limit (i.e., the number of samples goes to infinity), and is likely to provide an overly conservative controller. To account for a finite sample size, the Lipschitz constant needs to be corrected by multiplying it with a factor greater than one after observing the convergence behavior of the distribution fitting for different sizes of the dataset. In this particular case study, the direct use of the Lipschitz constant (without correction) gives a controller that covers only 78.8% of the winning domain of the model-based approach.

Parameter Optimization

We now discuss how a selection of different parameters can affect the sample complexity and conservativeness of our method. We fix the path planning case study with the estimated Lipschitz constant of 1.46. Figures 2.12 and 2.13 illustrate the effect of changing parameters ε, β on the number of samples N required for each pair $(\bar{x}, \bar{u})$ in order to compute the growth bound with confidence $(1 - \beta)$. Figure 2.12 illustrates the effect of increasing the confidence parameter β on reducing the sample complexity, for a fixed $\varepsilon = 0.01$. Figure 2.13 shows that for a fixed $\beta = 0.01$, increasing ε leads to a rapid drop in N. In both Figures 2.12 and 2.13, the sample complexity increases in the presence of disturbance as the dimension of the sample space becomes larger.

Figure 2.14 demonstrate the effect of changing ε on the value of the bias term γ that makes the inequalities of the SCP more conservative. The bias term γ increases for larger

values of ε. Therefore, increasing ε can decrease the sample complexity while increasing γ. Finally, it can be observed that the value of γ is larger in the presence of disturbance.

2.3. Neural Abstraction-Based Controller Synthesis and Deployment

A main bottleneck of ABCD is the memory requirement, both in representing the finite abstract transition relation and in representing the controller. First, the state and input spaces of the abstraction grow exponentially with the system and input dimensions, respectively, and the size of the abstract transition relation grows quadratically with the abstract states and linearly with the input states. While symbolic encodings using BDDs can be used, in practice, the transition relation very quickly exceeds the available RAM. Memory-efficient methods sometimes exploit the analytic description of the system dynamics or growth bounds [116, 85, 151], but these techniques are not applicable when the finite abstractions are learned directly from sampled system trajectories, or when a compact analytical expression of the growth bound is not available. Second, the winning strategy in the graph game is extracted as a look-up table mapping winning states to one or more available inputs. Thus, the controller representation is also exponential in the system dynamics. Such controllers cannot be deployed on memory-constrained embedded systems.

In this section, we address the memory bottleneck using approximate, compressed, representations of the transition relation and the controller map using neural networks. We learn an approximate representation of the abstract transition relation as a neural network with a fixed architecture. In contrast to the predominant use of neural networks to learn a generalization of an unknown function through sampling, we train the network on the entire data set (the transition relation or the controller map) offline. We store the transitions on disk, and train our networks in batch mode by bringing blocks of data into the RAM as needed. The trained network is small and fits into RAM. Since the training of the network minimizes error but does not eliminate it, we apply a *correction* to the output to ensure that the representation is *sound* with respect to the original finite abstraction, i.e., every trajectory in the finite abstraction is preserved in the compressed representation. We propose an on-the-fly synthesis approach that works directly on the corrected representation of the forward and backward dynamics of the system. Although we present our results with respect to reach-avoid specifications, our approach can be generalized to other classes of properties and problems (e.g., linear temporal logic specifications [13]) in which the solution requires the computation of the set of predecessors and successors in the underlying transition system.

Similarly, we store the winning strategy as a look-up table mapping states to sets of valid inputs on disk and propose a novel training algorithm to find a neural network representation of the synthesized controller. The network is complemented with a look-up table that provides "exceptions" in which the network deviates from the winning strategy. We experimentally demonstrate that a controller can be correctly represented as a combination of a neural network and a look-up table that requires a substantially

smaller memory than the original representation.

An important aspect of our approach is that, instead of using neural networks for learning an unknown data distribution, we train them over the entire data domain. Therefore, in contrast to many other applications wherein neural networks provide function representation and generalization over the unseen data, we are able to provide *formal soundness guarantees* for the performance of the trained neural representations over the whole dataset.

Our compression scheme uses additional computation to learn a compressed representation and avoid the memory bottleneck. In our implementation, the original relations are stored on the hard drive and data batches are loaded sequentially into the RAM to perform the training. Hard drives generally have much higher memory sizes compared to the RAM, but reading data from the hard drive takes much longer. However, data access during training is predictable and we can perform prefetching to hide the latency. During the synthesis, the trained corrected neural representations fit into the RAM. In contrast, a disk-based synthesis algorithm does not have predictable disk access patterns and is unworkable. Similarly, the deployed controller only consists of the trained compact representation and (empirically) a small look-up table, which can be loaded into the RAM of the controlling chip for the real-time operation of the system.

We evaluate the performance of our approach on several examples of different difficulties and show that it is effective in reducing the memory requirements at both synthesis and deployment phases. For the selected benchmarks, our method reduces the space requirements of synthesis and deployment respectively by factors of 1.31×10^5 and 7.13×10^3 in average, and up to 7.54×10^5 and 3.18×10^4, compared to the abstraction-based method that requires storing the full transition system. Moreover, we empirically show that, unlike other encodings, the memory requirement of our method is not affected by the system dimension on the considered benchmarks. The content of this section is based on our paper [121].

In summary, our main contributions are:

- Proposing a novel and sound representation scheme for compressing finite transition systems using the expressive power of neural networks;
- Proposing a novel on-the-fly controller synthesis method using the corrected neural network representations of forward and backward dynamics;
- Proposing an efficient scheme for compressing the controller computed by abstraction-based synthesis methods;
- Demonstrating significant reduction in the memory requirements by orders of magnitude through a set of standard benchmarks. [1]

Related Work. Below, we give an overview of the existing literature in areas that are relevant to the subject of study in this section.

[1] Our implementations are available online at `https://github.com/msalamati/Neural-Representation`.

Synthesis via reinforcement learning. The idea of using neural networks as function approximators to represent tabular data for synthesis purposes has been used in different fields such as reinforcement learning (RL) literature and aircraft collision avoidance system design. RL algorithms try to find an optimal control policy by iteratively guiding the interaction between the agent and the environment modeled as a Markov decision process [171]. When the space of the underlying model is finite and small, q-tables are used to represent the required value functions and the policy. When the space is large and possibly uncountable, such finite q-tables are replaced with neural networks as function approximators. Convergence guarantees that hold with the q-table representation [24] are not valid for non-tabular setting [26, 29, 177]. A similar behavior is observed in our setting: we lose the correctness guarantees in our approach without correcting the output of the neural network representations of the transition systems and the tabular controller.

Neural-aided controller synthesis. Constructing neural network representations of the dynamics of the control system and using them for synthesis is studied in specific application domains including the design of unmanned airborne collision avoidance systems [81]. The central idea of [81] is to start from a large look-up table representing the dynamics, train a neural network on the look-up table, and use it in the dynamic programming for issuing horizontal and vertical advisories. Several techniques are used to reduce the storage requirement since the obtained score table—that is the table mapping every discrete state-input pair into the associated score—becomes huge in size (hundreds of gigabytes of floating numbers). Since simple techniques such as down sampling, block compression [99], and exploiting the natural symmetry of the score table [88] are unable to achieve the required storage reduction, Julian et al. have shown that deep neural networks can successfully approximate the score table [89]. However, as in the RL controller synthesis, there is no guarantee that the control input computed using the neural representation matches the one computed using the original score table. In contrast, our corrected neural representations are guaranteed to produce formally correct controllers.

Reactive synthesis. Binary decision diagrams (BDDs) are used extensively in the reactive synthesis literature to represent the underlying transition systems [149, 79]. While BDDs are compact enough for low-order dynamical systems, recent synthesis tools such as SCOTS v2.0 [152] have already migrated into the non-BDD setting in order to avoid the large runtime overheads. In fact, motivated by reducing the required memory foot print, the current trend is to synthesize controllers in a non-BDD on the fly to eliminate the need for storing the transition system [96, 97, 104, 116, 85, 151]. These memory-efficient methods exploit the analytic description of the system dynamics or growth bounds. In contrast, our technique is applicable also to the case where the finite abstractions are learned directly from the sampled system trajectories, i.e., when no compact analytical expression of the dynamics and growth bounds are available.

Verifying systems with neural controllers. An alternative approach developed for safety-critical systems is to use neural networks as a representation of the controller and learn the controller using techniques such as reinforcement learning and data-driven predictive control [55, 175]. In this approach, the controller synthesis stage does not provide any

safety guarantee on the closed loop system, i.e., on the feedback connection of the neural controller and the physical system. Instead, the safety of the closed-loop system is verified a posteriori for the designed controller. Ivanov et al. have considered dynamical systems with sigmoid-based neural network controllers, used the fact that sigmoid is the solution to a quadratic differential equation to transform the composition of the system and the neural controller into an equivalent hybrid system, and studied reachability properties of the closed-loop system by utilizing existing reachability computation tools for hybrid systems [84]. Huang et al. have considered dynamical systems with Lipschitz continuous neural controllers and used Bernstein polynomials for approximating the input-output model of the neural network [81]. Development of formal verification ideas for closed-loop systems with neural controllers has led into emergence of dedicated tools such as NNV [176] and POLAR [82]. While these methods provide guarantees on closed-loop control system with neural controllers, they can only consider *finite horizon* specifications for a given set of initial states. In contrast, we consider controllers that are synthesized for *infinite horizon* specifications.

Minimizing the memory foot print for symbolic controllers. Girard et al. have proposed a method to reduce the memory needed to store safety controllers by determinizing them, i.e., choosing one control input per state such that an algebraic decision diagram (ADD) representing the control law is minimized [66, 85]. Zapreev et al. have provided two methods based on greedy algorithms and symbolic regression to reduce the redundancy existing in the controllers computed by the abstraction-based methods [193]. Both of the ADD scheme in [66, 85] and the BDD-based scheme in [193] have the capability to determinize the symbolic controller and reduce its memory foot print. However, the computed controller still suffers from the additional runtime overhead of the ADD/BDD encoding. Further, as mentioned by the authors of [193], their regression-based method is not able to represent the original controller with high accuracy. In contrast, our tool produces real-valued representations for symbolic controllers and can (additionally) be computed on top of the simplified version found by either of the methods proposed in [85, 193].

Compressed representations for model predictive controllers (MPCs). Hertneck et al. have proposed a method to train an approximate neural controller representing the original robust (implicit) MPC satisfying the given specification [76]. While reducing the online computation time is the main motivation in implicit MPCs, minimizing the memory foot print is the main objective in explicit MPCs. Salamati et al. have proposed a method which is based on solving an optimization to compute a memory-optimized controller with mixed-precision coefficients used for specifying the required coefficients [159]. Our method considers a different class of controllers that can fulfill infinite horizon temporal specifications.

Overview of the Proposed Approach

We provide a high-level description of our approach for both synthesis and deployment.

Corrected neural representations. Figure 2.15 gives a pictorial description of the

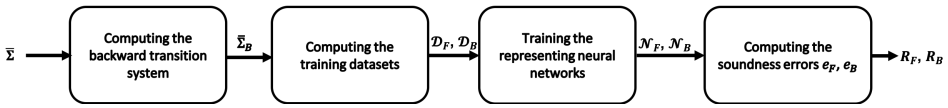

Figure 2.15.: Graphical description of the proposed scheme for compressing finite abstractions

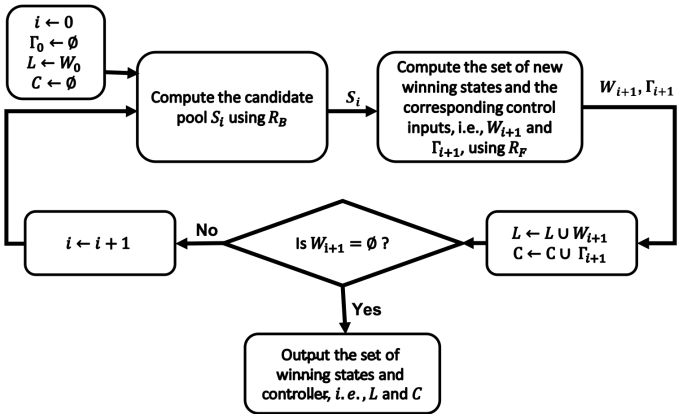

Figure 2.16.: Graphical description of the proposed synthesis scheme

steps for computing a corrected neural network representation. Given a finite abstraction $\bar{\Sigma}$ that corresponds to the forward dynamics of the system and stored on the hard drive, we first compute the transition system $\bar{\Sigma}_B$ corresponding to the backward dynamics. Next, we extract the input-output training datasets $\mathcal{D}_F$ and $\mathcal{D}_B$ respectively from the forward and backward systems, and store them on the hard drive. Each data point contains one state-input pair and the characterization of ℓ_∞ ball for the corresponding reachable set. We train two neural networks $\mathcal{N}_F$ and $\mathcal{N}_B$ such that they represent compressed input-output surrogates for the datasets $\mathcal{D}_F$ and $\mathcal{D}_B$, respectively. Finally, we compute the *soundness errors* e_F and e_B which correspond to the difference between the output of $\mathcal{N}_F$ and $\mathcal{N}_B$ and the respective values in $\mathcal{D}_F$ and $\mathcal{D}_B$, calculated over all of the state-input pairs. We use the computed errors e_F and e_B in order to construct the corrected neural representations R_F and R_B. We will get memory savings by using R_F and R_B instead of $\bar{\Sigma}$ and $\bar{\Sigma}_B$, respectively.

Synthesis. Figure 2.16 gives a pictorial description of our proposed synthesis algorithm for a reach-avoid specification with the target set *Goal* and obstacle set *Avoid* as subsets of the state space. Let $W_0 \subseteq \bar{X}$ represents a discrete under-approximation of the target set *Goal*. We initialize the winning set as $L = W_0$, the controller as $C = \emptyset$, and the set of state-input pairs that must be added to the controller as $\Gamma_0 = \emptyset$. In each iteration, we compute the set of new states that belong to the winning set and update the controller accordingly, until no new state is added to L. To this end, we first use R_B and its corresponding soundness error e_B to compute a set of candidates S_i out of which some belong to L and it is guaranteed that there will be no winning state outside of S_i in the i^{th} iteration. We use R_F and its corresponding soundness error e_F to compute the set of new winning states $W_{i+1} \subseteq S_i$. We also compute the set of control inputs for every

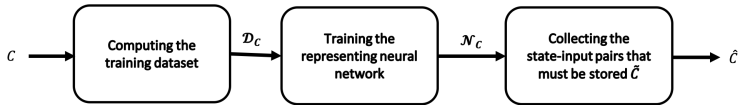

Figure 2.17.: Graphical description of the proposed scheme for compressing controllers

new winning state and compute the corresponding set of state-input pairs Γ_{i+1} that must be added to the controller. Finally, if $W_i = \emptyset$, we terminate the computations as we already have computed the winning set L and the controller C. Otherwise, we add the new winning set of states and state-input pairs, respectively, into the overall winning set ($L \leftarrow L \cup W_{i+1}$) and the controller ($C \leftarrow C \cup \Gamma_{i+1}$), and repeat the steps in the next iteration.

Deployment. Figure 2.17 shows our method for compressing controllers that are obtained from abstraction-based approaches. In the first step, we collect the training dataset $\mathcal{D}_C$ and reformat it to become appropriate for our specific formulation of a classification problem. Each data point contains one state and an encoding of the corresponding set of control inputs. We then train a neural network $\mathcal{N}_C$ on the data with the loss function designed for this specific classification problem. Finally, we find all the states at which the output label generated by $\mathcal{N}_C$ is invalid, and store the corresponding state-input pair in a look-up table, denoted by $\hat{C}$. We experimentally show that $\hat{C}$ only contains a very small portion of state-input pairs.

2.3.1. Problem Statement

We now consider the controller synthesis problem for finite abstractions w.r.t. a reach-avoid specification. Let $Goal, Avoid \subseteq X, Goal \cap Avoid = \emptyset$ be the set of states representing the target and unsafe spaces, respectively. The *winning domain* for the finite abstraction $\bar{\Sigma} = (\bar{X}, \bar{U}, T_F)$ is the set of states $\bar{x}^* \in \bar{X}$ such that there exists a feedback controller C such that all trajectories of $C \parallel \bar{\Sigma}$, which are started at $\bar{x}^*$, satisfy the given specification Φ. $\bar{x}_0 = \bar{x}^*, \bar{x}_1, \bar{x}_2, \ldots \models \Phi$. The aim is to find the set of the winning states L together with a feedback controller C such that $C \parallel \bar{\Sigma}$ satisfies the reach-avoid specification Φ. To compute the winning domain and the controller, one can use the methods from reactive synthesis. For many of interesting control systems, size of T_F in the finite abstraction becomes huge. This restricts the application of reactive-synthesis-based methods for computing the controller. Therefore, we are looking for a method which uses compressed surrogates of T_F to save memory. In particular, we want to train two *corrected neural* surrogates, i.e., neural network representations whose output is corrected to maintain the soundness property: R_F for the forward-in-time dynamics and R_B for the backward-in-time dynamics.

Problem 2.3 ***Inputs:*** *Finite abstraction $\bar{\Sigma} = (\bar{X}, \bar{U}, T_F)$, and specification $\Phi = \neg Avoid\, \mathcal{U}\, Goal$.*
Outputs: *Corrected neural representations R_F and R_B, winning domain L and a feedback controller C for $\bar{\Sigma}$ such that $C \parallel \bar{\Sigma}$ realizes Φ.*

It is important to notice that any solution for this problem is required to provide a *formal guarantee* on the satisfaction of Φ, i.e., the reach-avoid specification Φ *must* be satisfied under any disturbance affecting the control systems.

Let $C \in \bar{X} \times \bar{U}$ be the computed controller for the abstraction $\bar{\Sigma}$ such that $C \parallel \bar{\Sigma}$ realizes a given specification Φ. The size of this controller can be large due to the large number of discrete state and inputs. For deployment purposes, we would like to compute a corrected neural controller $\hat{C} \coloneqq \bar{X} \to \bar{U}$ s.t. $\hat{C} \parallel \bar{\Sigma}$ realizes Φ.

Problem 2.4 ***Inputs:*** *Controller C computed for the discrete control system $\bar{\Sigma}$, and specification Φ s.t. $C \parallel \bar{\Sigma}$ realizes Φ.*
Outputs: *A corrected neural controller $\hat{C}$ such that $\hat{C} \parallel \bar{\Sigma}$ realizes Φ.*

2.3.2. Synthesis

One approach to formally synthesize controllers for a given specification is to store the transition system corresponding to quantization of the state and input spaces, and to use the methods from reactive synthesis to design a controller. However, the memory required to store these transition systems increases exponentially with the number of state variables, which causes a memory blow-up for many real-world systems. In this subsection, we propose our memory-efficient algorithm for synthesizing controllers to satisfy reach-avoid specifications for finite abstractions and reach-avoid specifications. Our method requires computation of corrected neural representations for the finite abstraction. First, we provide two method for computing these representations. Later, we show how our synthesis method makes use of the computed representations.

Corrected Neural Representations for Finite Abstractions

Let $\bar{\Sigma} = (\bar{X}, \bar{U}, T_F)$ be a finite abstraction. Finite abstractions can be computed analytically when the system dynamics are known and certain Lipschitz continuity properties hold. Even when the system dynamics are unknown, one can use data-driven methods to learn finite abstractions that are correct with respect to a given confidence [93, 52, 123]. We show that T_F can be approximated by some *generator functions*. In particular, we show how to compute generator functions $R_F \colon \bar{X} \times \bar{U} \to \mathbb{R}^n \times \mathbb{R}^n_{\geq 0}$ and $R_B \colon \bar{X} \times \bar{U} \to \mathbb{R}^n \times \mathbb{R}^n_{\geq 0}$ which can produce characterization of an ℓ_∞ ball corresponding to the over-approximation of forward- and backward-in-time reachable sets, respectively, for every state-input pair picked from $\bar{X} \times \bar{U}$. Our aim is to use the expressive power of neural networks to represent the behavior of $\bar{\Sigma}$ such that the memory requirements significantly decrease.

Our compression scheme is summarized in Algorithm 3. We first compute the backward-in-time system $\bar{\Sigma}_B$ using Eq. (2.2). We then calculate the over-approximating ℓ_∞ ball for every state-input pair. Let $c_F(\bar{x}, \bar{u}) \in X$ and $r_F(\bar{x}, \bar{u}) \in \mathbb{R}^n_{\geq 0}$ characterize the tightest ℓ_∞ ball such that

$$(\bar{x}, \bar{u}, \bar{x}') \in T_F \Leftrightarrow \|\bar{x}' - c_F(\bar{x}, \bar{u})\|_\infty \leq r_F(\bar{x}, \bar{u}) - \eta_x/2.$$

This is illustrated in Figure 2.18 in two-dimensional space for a given state-input pair $(\bar{x}, \bar{u})$. The dotted red rectangle corresponds to the hyper-rectangular reachable set. The

Algorithm 3: Regression-based compression algorithm for finite abstractions

Data: Forward dynamics $\bar{\Sigma}$ and learning rate λ

1 Compute backward dynamics $\bar{\Sigma}_B$ and the datasets $\mathcal{D}_F$ and $\mathcal{D}_B$ using Eqs. (2.2), (2.26), and (2.27)

2 Train neural networks $\mathcal{N}_F$ on the dataset $\mathcal{D}_F$ and train $\mathcal{N}_B$ on $\mathcal{D}_B$ using the learning rate λ

3 Compute the soundness errors e_F and e_B using Eq. (2.28)

4 Compute the final corrected representations R_F and R_B using Eqs. (2.29) and (2.30)

Result: corrected neural representations R_F and R_B

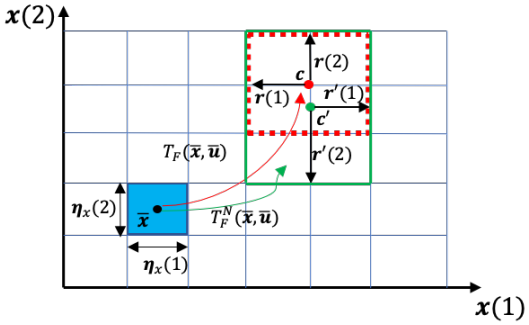

Figure 2.18.: Comparing the set of successor states in the transition system T_F and its representation T_F^N. We have $c = c_F(\bar{x},\bar{u})$, $r_F(\bar{x},\bar{u}) = (r(1), r(2))$, $c' = \mathcal{N}_F^c(\bar{x},\bar{u})$ and $\mathcal{N}_F^r(\bar{x},\bar{u}) = (r'(1), r'(2))$.

center $c_F(\bar{x},\bar{u})$ and radius $r_F(\bar{x},\bar{u})$ are computed using the lower-left and upper-right corners of the reachable set denoted, respectively, by $g_{FL}(\bar{x},\bar{u})$ and $g_{FU}(\bar{x},\bar{u})$. Then, we have $c_F(\bar{x},\bar{u}) = (g_{FU}(\bar{x},\bar{u}) + g_{FL}(\bar{x},\bar{u}))/2$ and $r_F(\bar{x},\bar{u}) = (g_{FU}(\bar{x},\bar{u}) - g_{FL}(\bar{x},\bar{u}))/2 + \eta_x/2$. At the end of the first step we have computed and stored the dataset

$$\mathcal{D}_F = \{((\bar{x},\bar{u}), (c_F(\bar{x},\bar{u}), r_F(\bar{x},\bar{u}))) \mid \bar{x} \in \bar{X}, \bar{u} \in \bar{U}\}. \tag{2.26}$$

Note that every data-point in $\mathcal{D}_F$ consists of two pairs: one specifies a state-input pair $(\bar{x},\bar{u})$ and the other one characterizes the center and radius corresponding to the over-approximating ℓ_∞ disc $(c_F(\bar{x},\bar{u}), r_F(\bar{x},\bar{u}))$. Similarly, we need to store another dataset corresponding to the backward dynamics. First, we define $c_B(\bar{x},\bar{u}) \in X$ and $r_B(\bar{x},\bar{u}) \in \mathbb{R}^n_{\geq 0}$ characterizing the tightest ℓ_∞ ball such that

$$(\bar{x},\bar{u},\bar{x}') \in T_B \Leftrightarrow \|\bar{x}' - c_B(\bar{x},\bar{u})\|_\infty \leq r_B(\bar{x},\bar{u}) - \eta_x/2.$$

The dataset corresponding to the backward dynamics is of the following form

$$\mathcal{D}_B = \{((\bar{x},\bar{u}), (c_B(\bar{x},\bar{u}), r_B(\bar{x},\bar{u}))) \mid \bar{x} \in \bar{X}, \bar{u} \in \bar{U}\}. \tag{2.27}$$

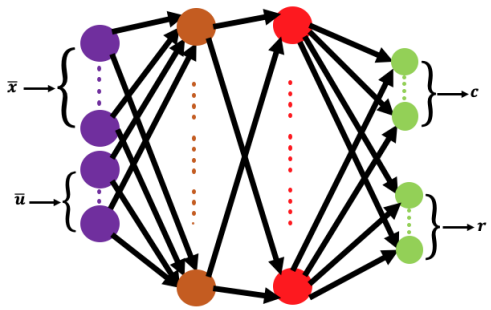

Figure 2.19.: The regression-based configuration used in compression of abstractions. The input to the neural network includes state-input pair $(\bar{x}, \bar{u})$, and the output includes the pair (c, r) corresponding to the center and radius of the rectangular reachable set, respectively. Right: The classification-based representation of finite abstractions. The representation receives a state-input pair $(\bar{x}, \bar{u})$. In the output, y_{lb} and y_{ub} correspond to the lower-left and upper-right corners for the rectangular reachable set.

The size of $\mathcal{D}_F$ and $\mathcal{D}_B$ grows exponentially with the dimension of state space. Hence, we store both the datasets $\mathcal{D}_F$ and $\mathcal{D}_B$ (potentially) into the hard drive. Next, we take the datasets $\mathcal{D}_F$ and $\mathcal{D}_B$, for which we train neural networks $\mathcal{N}_F$ and $\mathcal{N}_B$, taking the state-input pairs $(\bar{x}, \bar{u})$ as input and $(c_F(\bar{x}, \bar{u}), r_F(\bar{x}, \bar{u}))$ as output, and try to find an input-output mapping minimizing mean squared error (MSE). For systems with state and input spaces of dimensions n and m, the input and output layers of both neural networks are of sizes $n + m$ and $2n$, respectively. The configuration of the neural networks which we used is illustrated in Figure 2.19. During training, we load batches of data from $\mathcal{D}_F$ and $\mathcal{D}_B$, which are stored on the the hard drive, into the RAM. We use the stochastic gradient descent (SGD) method to minimize MSE.

As mentioned earlier, in contrast to the usual applications wherein neural networks are used to represent an unknown distribution, we have the full dataset and require computing representations which are *sound* with respect to the input dataset. A sound representation for the given finite abstractions produces reachable sets that include $T_F(\bar{x}, \bar{u})$ for every state-input pair $(\bar{x}, \bar{u})$. For instance, the solid green rectangle in Figure 2.18 contains the set of reachable states corresponding to $\mathcal{N}_F(\bar{x}, \bar{u})$ and contains the set of states included in the dotted red rectangle, i.e., $T_F(\bar{x}, \bar{u})$. Therefore, we can say that the representation $\mathcal{N}_F$ is sound for the pair $(\bar{x}, \bar{u})$. In order to guarantee *soundness*, we need to compute the maximum error induced during the training process among all the training data points. To that end, we go over all the state-input pairs (which are stored on the hard drive) and compute the maximum error in approximating the centers of the ℓ_∞ balls, denoted by e_F^c, e_B^c and radius e_F^r, e_B^r corresponding to the forward and backward representations:

$$e_F^c = \max_{\bar{x}\in\bar{X},\bar{u}\in\bar{U}} |c_F(\bar{x}, \bar{u}) - \mathcal{N}_F^c(\bar{x}, \bar{u})|, \quad e_F^r = \max_{\bar{x}\in\bar{X},\bar{u}\in\bar{U}} |r_F(\bar{x}, \bar{u}) - \mathcal{N}_F^r(\bar{x}, \bar{u})|.$$

Similarly, for the backward dynamics,

$$e_B^c = \max_{\bar{x}\in\bar{X},\bar{u}\in\bar{U}} |c_B(\bar{x},\bar{u}) - \mathcal{N}_B^c(\bar{x},\bar{u})|, \quad e_B^r = \max_{\bar{x}\in\bar{X},\bar{u}\in\bar{U}} |r_B(\bar{x},\bar{u}) - \mathcal{N}_B^r(\bar{x},\bar{u})|.$$

We define

$$e_F = e_F^c + e_F^r, \quad e_B = e_B^c + e_B^r, \tag{2.28}$$

and use the errors e_F and e_B to compute the *corrected representations* R_F and R_B, corresponding to $\mathcal{N}_F$ and $\mathcal{N}_B$, as described next. Let R_F^c and R_F^r correspond to the center and radius components of R_F. Similarly, R_B^c and R_B^r correspond to the center and radius components of R_B. For state-input pair $(\bar{x},\bar{u}) \in \bar{X}\times\bar{U}$, we define

$$R_F^c(\bar{x},\bar{u}) = \mathcal{N}_F^c(\bar{x},\bar{u}), \quad R_F^r(\bar{x},\bar{u}) = \mathcal{N}_F^r(\bar{x},\bar{u}) + e_F, \tag{2.29}$$

for the forward dynamics, and

$$R_B^c(\bar{x},\bar{u}) = \mathcal{N}_B^c(\bar{x},\bar{u}), \quad R_B^r(\bar{x},\bar{u}) = \mathcal{N}_B^r(\bar{x},\bar{u}) + e_B, \tag{2.30}$$

for the backward dynamics.

Let us define the forward transition system computed using the trained neural network as follows

$$T_F^N = \{(\bar{x},\bar{u},\bar{x}') \in \bar{X}\times\bar{U}\times\bar{X} \,|\, \bar{x}' \in \bar{K}(\mathcal{N}_F^c(\bar{x},\bar{u}) - \mathcal{N}_F^r(x,\bar{u}) - e_F, \mathcal{N}_F^c(\bar{x},\bar{u}) + \mathcal{N}_F^r(\bar{x},\bar{u}) + e_F)\}, \tag{2.31}$$

where $\mathcal{N}_F^c(\cdot,\cdot)$, $\mathcal{N}_F^r(\cdot,\cdot)$ denote the components of the output of $\mathcal{N}_F(\cdot,\cdot)$ corresponding to the center and radius of disc, respectively. Similarly, we can define the transition system T_B^N corresponding to the backward dynamics as follows

$$T_B^N = \{(\bar{x},\bar{u},\bar{x}') \in \bar{X}\times\bar{U}\times\bar{X} \,|\, \bar{x}' \in \bar{K}(\mathcal{N}_B^c(\bar{x},\bar{u}) - \mathcal{N}_B^r(\bar{x},\bar{u}) - e_B, \mathcal{N}_B^c(\bar{x},\bar{u}) + \mathcal{N}_B^r(\bar{x},\bar{u}) + e_B)\}. \tag{2.32}$$

The following lemma states that we can use the trained neural networks to compute sound transition systems for both forward and backward dynamics. However, our synthesis approach does not require the computation of T_F^N and T_B^N and only uses the compressed representations $\mathcal{N}_F$ and $\mathcal{N}_B$.

Lemma 2.3.1 *Transition systems T_F^N and T_B^N computed by* (2.31) *and* (2.32) *are sound for T_F and T_B, i.e., we have $T_F \subseteq T_F^N$ and $T_B \subseteq T_B^N$.*

To reduce the level of conservativeness, we require that T_F^N and T_B^N do not contain too many additional edges compared to T_F and T_B. The *mismatch rate* of the forward and backward dynamics are defined as

$$d_F \coloneqq \frac{|T_F^N \setminus T_F|}{|T_F|}, \quad d_B \coloneqq \frac{|T_B^N \setminus T_B|}{|T_B|}.$$

If the trained representations are accurate, the mismatch rate is low, which results in a *less restrictive* representation.

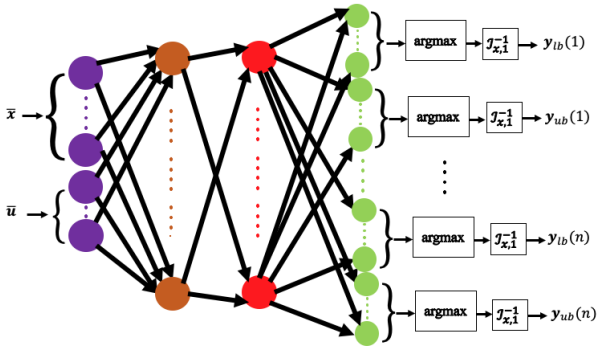

Figure 2.20.: The classification-based representation of finite abstractions. The representation receives a state-input pair $(\bar{x}, \bar{u})$. In the output, $\boldsymbol{y}_{lb}$ and $\boldsymbol{y}_{ub}$ correspond to the lower-left and upper-right corners for the rectangular reachable set.

Remark 7 *The method outlined in Algorithm 3 formulates the computation of the representations as a regression problem, wherein the representative neural networks are supposed to predict the center and radius corresponding to ℓ_∞ reachable sets. In the rest of this subsection, we describe a classification-based formulation for compressing finite abstractions, wherein the representative neural networks are supposed to predict the vectorized indices corresponding to the lower-left and upper-right corners of the reachable set. We experimentally show that this second formulation, while being more memory demanding, provides a less conservative representation compared to our regression-based formulation.*

Classification-Based Computation of Representations for Finite Abstractions

So far we have presented a formulation for training neural networks that can *guess* at any given state-input pair the center and radius of a hyper-rectangular over-approximation of the reachable states. This guess is then *corrected* using the computed soundness errors. A nice aspect of this formulation is that we only need to store the trained representations and their corresponding soundness errors. However, the result of using the soundness errors to correct the output values produced by the neural networks may give a very conservative over-approximation of the reachable sets, even when the trained representations have a very good performance on a large subset of the state-input pairs, since the soundness errors must be computed over *all* state-input pairs.

We provide an alternative formulation for computing a compressed representation of a given abstraction. Intuitively, our idea is to train neural network representations which can guess for any given state-input pair the *vectorized indices* corresponding to the lower-left and upper-right corner points of the hyper-rectangular reachable set. The architecture of the representation is shown in Figure 2.20. As illustrated, for every state-input pair $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$, the output of the representation gives the lower-left and upper-right corners of the rectangular set that is reachable by taking the control input $\bar{u}$ at the state $\bar{x}$. Algorithm 4 describes our classifier-based compression scheme for

Algorithm 4: Computing Classification-based representations of finite abstractions

Data: Forward dynamics $\bar{\Sigma}$ and learning rate λ

1 Compute backward dynamics $\bar{\Sigma}_B$ and the datasets $\mathcal{D}_F$ and $\mathcal{D}_B$ using Eqs. (2.2), (2.33), and (2.34)

2 Train neural networks $\mathcal{N}_F$ and $\mathcal{N}_B$ on the datasets $\mathcal{D}_F$ and $\mathcal{D}_B$ using the learning rate λ

3 Compute the set of misclassified state-input pairs E_F and E_B as in Eq. (2.35)

4 Compute the set of transitions $\tilde{\mathcal{N}}_F$ and $\tilde{\mathcal{N}}_B$ associated with E_F and E_B as in Eq. (2.37)

5 Compute the corrected neural representations R_F, R_B using Eqs. (2.38), (2.39)

Result: R_F, R_B

finite abstractions. We first compute the backward system $\bar{\Sigma}_B$ using Eq. (2.2). We then compute the training datasets for both the forward and backward systems $\bar{\Sigma}$ and $\bar{\Sigma}_B$. For $\bar{\Sigma}$, let $g_{FU}\colon \bar{X} \times \bar{U} \to \bar{X}$ and $g_{FL}\colon \bar{X} \times \bar{U} \to \bar{X}$ denote the mappings from the state-input pair $(\bar{x}, \bar{u}) \in \bar{X} \times \bar{U}$ into the corresponding upper-right and lower-left corners of the rectangular reachable set from $(\bar{x}, \bar{u})$. We define $z_F\colon \bar{X} \times \bar{U} \to \{0,1\}^{2\sum_{i=1}^{n} |\bar{X}(i)|}$ with $|\bar{X}(i)|$ being the cardinality of the projection of $\bar{X}$ along the i^{th} axis and $z_F(\bar{x}, \bar{u})(l) = 1$ if and only if

$$l = 2\sum_{k=1}^{i-1} |\bar{X}(k)| + \mathcal{I}_{x,i}(g_{FL}(\bar{x}, \bar{u})(i)) \text{ or } l = 2\sum_{k=1}^{i-1} |\bar{X}(k)| + |\bar{X}(i)| + \mathcal{I}_{x,i}(g_{FU}(\bar{x}, \bar{u})(i)),$$

for some $i \in \{1, 2, \ldots, n\}$. The indexing function $\mathcal{I}_{x,i}\colon \bar{X}(i) \to [1; |\bar{X}(i)|]$ maps every element of $\bar{X}(i)$ into a *unique* integer index in the interval $[1; |\bar{X}(i)|]$. The training dataset for $\bar{\Sigma}$ is defined as

$$\mathcal{D}_F \coloneqq \{(\bar{x}, \bar{u}, z_F(\bar{x}, \bar{u})) \,|\, \bar{x} \in \bar{X} \text{ and } \bar{u} \in \bar{U}\}. \tag{2.33}$$

Intuitively, each element of the dataset $\mathcal{D}_F$ contains a state-input pair $(\bar{x}, \bar{u})$ and a vector $\boldsymbol{h} \in \{0,1\}^{2\sum_{i=1}^{n} |\bar{X}(i)|}$ that has 1 only at the entries corresponding to $\mathcal{I}_{x,i}(g_{FL}(\bar{x}, \bar{u})(i))$ and $\mathcal{I}_{x,i}(g_{FU}(\bar{x}, \bar{u})(i))$ for $i \in \{1, 2, \ldots, n\}$. Similarly, we define $z_B\colon \bar{X} \times \bar{U} \to \{0,1\}^{2\sum_{i=1}^{n} |\bar{X}(i)|}$ for $\bar{\Sigma}_B$ such that $z_B(\bar{x}, \bar{u})(l) = 1$ if and only if

$$l = 2\sum_{k=1}^{i-1} |\bar{X}(k)| + \mathcal{I}_{x,i}(g_{BL}(\bar{x}, \bar{u})(i)) \text{ or } l = 2\sum_{k=1}^{i-1} |\bar{X}(k)| + |\bar{X}(i)| + \mathcal{I}_{x,i}(g_{BU}(\bar{x}, \bar{u})(i)),$$

for some $i \in \{1, 2, \ldots, n\}$. The training dataset for the backward dynamics is also defined similarly as

$$\mathcal{D}_B \coloneqq \{(\bar{x}, \bar{u}, z_B(\bar{x}, \bar{u})) \,|\, \bar{x} \in \bar{X} \text{ and } \bar{u} \in \bar{U}\}. \tag{2.34}$$

Once the training datasets are ready, we train the neural networks $\mathcal{N}_F$ and $\mathcal{N}_B$ respectively on the datasets $\mathcal{D}_F$ and $\mathcal{D}_B$. Note that the output layer of $\mathcal{N}_F$ and $\mathcal{N}_B$ will

be a vector of size $2\sum_{i=1}^{n}|\bar{X}(i)|$, while the final output of the representations are of size $2n$ (cf. Figure 2.20). These final outputs give an approximation of the coordinates of the lower-left and upper-right corners of the reachable set corresponding to the pair $(\bar{x},\bar{u})$. Note that, because $\bar{X}$ was computed by equally partitioning over X, both the indexing function $\mathcal{I}_{x,i}$ and its inverse can be implemented in a memory-efficient way using floor and ceil operators. We then evaluate the performance of the trained neural networks $\mathcal{N}_F$ and $\mathcal{N}_B$. Let $\rho_{FL}(\bar{x},\bar{u})$ and $\rho_{FU}(\bar{x},\bar{u})$ denote respectively the estimated lower-left and upper-right corners of the reachable set estimated by $\mathcal{N}_F$. Define $\rho_{BL}(\bar{x},\bar{u})$ and $\rho_{BU}(\bar{x},\bar{u})$ similarly for $\mathcal{N}_B$, and let the set of misclassified state-input pairs be

$$\begin{aligned} E_F &\coloneqq \{(\bar{x},\bar{u})\in\bar{X}\times\bar{U} \mid \ T_F(\bar{x},\bar{u})\setminus[\![\rho_{FL}(\bar{x},\bar{u}),\rho_{FU}(\bar{x},\bar{u})]\!]_{\eta_x}\neq\emptyset\}\\ E_B &\coloneqq \{(\bar{x},\bar{u})\in\bar{X}\times\bar{U} \mid \ T_B(\bar{x},\bar{u})\setminus[\![\rho_{BL}(\bar{x},\bar{u}),\rho_{BU}(\bar{x},\bar{u})]\!]_{\eta_x}\neq\emptyset\}. \end{aligned} \tag{2.35}$$

The *soundness error* of $\mathcal{N}_F$ and $\mathcal{N}_B$ can be considered as their misclassification rate:

$$err_F \coloneqq \frac{|E_F|}{|\bar{X}\times\bar{U}|} \quad \text{and} \quad err_B \coloneqq \frac{|E_B|}{|\bar{X}\times\bar{U}|}. \tag{2.36}$$

For the misclassified pairs in E_F and E_B, we extract the related transitions in the abstraction:

$$\tilde{\mathcal{N}}_F \coloneqq \{(\bar{x},\bar{u},\bar{x}')\,|\,(\bar{x},\bar{u})\in E_F,\bar{x}'\in T_F(\bar{x},\bar{u})\},\tilde{\mathcal{N}}_B \coloneqq \{(\bar{x},\bar{u},\bar{x}')\,|\,(\bar{x},\bar{u})\in E_B,\bar{x}'\in T_B(\bar{x},\bar{u})\}. \tag{2.37}$$

Finally, we *correct* the output of neural network representations to maintain soundness

$$R_F(\bar{x},\bar{u}) \coloneqq \begin{cases} [\![\rho_{FL}(\bar{x},\bar{u}),\rho_{FU}(\bar{x},\bar{u})]\!]_{\eta_x} & \text{if } (\bar{x},\bar{u})\notin E_F \\ \tilde{\mathcal{N}}_F(\bar{x},\bar{u}) & \text{if } (\bar{x},\bar{u})\in E_F, \end{cases} \tag{2.38}$$

$$R_B(\bar{x},\bar{u}) \coloneqq \begin{cases} [\![\rho_{BL}(\bar{x},\bar{u}),\rho_{BU}(\bar{x},\bar{u})]\!]_{\eta_x} & \text{if } (\bar{x},\bar{u})\notin E_B \\ \tilde{\mathcal{N}}_B(\bar{x},\bar{u}) & \text{if } (\bar{x},\bar{u})\in E_B. \end{cases} \tag{2.39}$$

Note that these corrected neural representations are memory efficient only if the misclassification rates are small, i.e., the size of E_F and E_B are small compared with $\bar{X}\times\bar{U}$.

On-the-Fly Synthesis

So far, we described the computation of the compressed representations corresponding to the forward and backward dynamics for finite abstractions. We use these representations in order to synthesize formally correct controllers.

Our synthesis procedure is provided in Algorithm 5. It takes the representations R_F and R_B to synthesize a controller which fulfills the given reach-avoid specification. Let

$$W_0 = \{\bar{x}\in\bar{X} \mid [\![\bar{x}-\boldsymbol{\eta_x}/2,\bar{x}+\boldsymbol{\eta_x}/2]\!]\subseteq \mathit{Goal}\}$$

be a discrete under-approximation of the target set *Goal*. We take W_0 as the input and perform a fixed-point computation to solve the given reach-avoid game. We initialize the

Algorithm 5: Controller synthesis algorithm

Data: Set $W_0 \subseteq \bar{X}$ and the corrected neural representations R_F and R_B

Initialize $C \leftarrow \emptyset$, $P_0 \leftarrow W_0$, $\Gamma_0 \leftarrow \emptyset$ and $i \leftarrow 0$

while $W_i \neq \emptyset$ **do**

  Compute the candidate pool S_i using Eq. (2.40)

  Compute the set of new winning states W_{i+1} using Eq. (2.41) and add them to the winning set ($P_{i+1} \leftarrow P_i \cup W_{i+1}$)

  Compute the set of new state-input pairs Γ_{i+1} using Eq. (2.42) and add them to the controller ($C \leftarrow C \cup \Gamma_{i+1}$)

  $i \leftarrow i + 1$

end

$L \leftarrow P_i$

Result: Controller C and its winning set L

winning set and controller with $P_0 = W_0$ and $C = \emptyset$, and in each iteration, we add the new winning set of states and state-input pairs, respectively, into the overall winning set and the controller, until no new state is found ($W_{i+1} = \emptyset$).

Let W_i be the set of *new* winning states in the beginning of the i^{th} iteration. Further, we denote the set of winning states in the beginning of the i^{th} iteration by $P_i = \bigcup_{k=0}^{i} W_k$. In every iteration, for every $\bar{x} \in W_i$ and $\bar{u} \in \bar{U}$, we compute the backward over-approximating ℓ_∞ ball and discretize it to get the *candidate pool* S_i defined as

$$S_i \coloneqq \bigcup_{\bar{u} \in \bar{U}} Y_i(\bar{u}), \tag{2.40}$$

with

$$Y_i(\bar{u}) \coloneqq \bigcup_{\bar{x} \in W_i} (\bar{X} \cap \bar{K}(R_B^c(\bar{x}, \bar{u}) - R_B^r(\bar{x}, \bar{u}), R_B^c(\bar{x}, \bar{u}) + R_B^r(\bar{x}, \bar{u}))),$$

where $R_B^c(\cdot, \cdot)$, $R_B^r(\cdot, \cdot)$ denote the components of the output of $R_B(\cdot, \cdot)$ corresponding to the center and radius of the ℓ_∞ ball, respectively. Note that we compute the candidate pool by running R_B over W_i *instead of* P_i. This is computationally beneficial, because $|W_i| \leq |P_i|$. Next lemma shows that S_i includes the *whole* set of new winning states W_{i+1}.

Lemma 2.3.2 *Let the set of candidates S_i be as defined in Eq.* (2.40)*. Then, we have $W_{i+1} \subseteq S_i$ for all $i \geq 0$.*

Proof *We prove this lemma by contradiction. Suppose that $W_{i+1} \not\subseteq S_i$. Then there exists at least one $\bar{x}^* \in W_{i+1} \setminus S_i$. Since $\bar{x}^* \in W_{i+1}$, we know that there exists at least one $\bar{u}^* \in \bar{U}$ such that $T_f(\bar{x}^*, \bar{u}^*) \subseteq P_i$ and $\bar{x}^* \notin P_i$. Moreover, since $\bar{x}^* \notin S_i$, by Eq.* (2.40) *we get $T_f(\bar{x}^*, \bar{u}^*) \cap W_i = \emptyset$. So, $T_f(\bar{x}^*, \bar{u}^*) \subseteq P_i \setminus W_i = P_{i-1}$. This gives $\bar{x}^* \in P_i$, which is a contradiction. This completes the proof.*

Now, we can use R_F, which represents the forward transition system, in order to choose the *legitimate* candidates out of S_i and add the new ones to W_{i+1}. Let

$$A = \{\bar{x} \in \bar{X} \mid [\![\bar{x} - \boldsymbol{\eta_x}/2, \bar{x} + \boldsymbol{\eta_x}/2]\!] \cap Avoid \neq \emptyset\}$$

be a discrete over-approximation over the set of obstacles. The next lemma states that we can use the representation R_F to compute W_{i+1}.

Lemma 2.3.3 *The set of states added to the winning set in the i^{th} step can be computed as*

$$W_{i+1} = \{\bar{x} \in S_i \mid \exists \bar{u} \in \bar{U} \; s.t. \bar{K}(R_F^c(\bar{x}, \bar{u}) - R_F^r(\bar{x}, \bar{u}), R_F^c(\bar{x}, \bar{u}) + R_F^r(\bar{x}, \bar{u})) \subseteq P_i\} \setminus (P_i \cup A). \tag{2.41}$$

Proof *To prove this lemma, we denote $G = \{\bar{x} \in S_i \mid \exists \bar{u} \in \bar{U} \; s.t. \; \bar{K}(R_F^c(\bar{x}, \bar{u}) - R_F^r(\bar{x}, \bar{u}), R_F^c(\bar{x}, \bar{u}) + R_F^r(\bar{x}, \bar{u})) \subseteq P_i\} \setminus (P_i \cup A)$, and show $W_{i+1} \subseteq G$ and $G \subseteq W_{i+1}$. The second direction ($G \subseteq W_{i+1}$) holds by definition. To prove the first direction ($W_{i+1} \subseteq G$), we note that $G \subseteq S_i$ and further, by the result of Lemma. 2.3.2, we have $W_{i+1} \subseteq S_i$. Assume $W_{i+1} \not\subseteq G$. Then there should exist at least one $\bar{x}^* \in W_{i+1} \setminus G$. Note that $\bar{x}^* \in S_i \setminus G$. Since $\bar{x}^* \in S_i$, we get that there exists at least one $\bar{u}^* \in \bar{U}$ for which $T_F(\bar{x}^*, \bar{u}^*) \subseteq W_i$. Also, because $\bar{x}^* \notin G$, we have $T_F(\bar{x}^*, \bar{u}^*) \not\subseteq W_i$, which is a contradiction. Therefore, $W_{i+1} \subseteq G$. Hence the proof ends.*

In each iteration, we calculate Γ_i, which is the set of new state-input pairs that must be added into the controller, and is defined as

$$\Gamma_{i+1} = \{(\bar{x}, \bar{u}) \mid \bar{x} \in W_{i+1}, \; \bar{K}(R_F^c(\bar{x}, \bar{u}) - R_F^r(\bar{x}, \bar{u}), R_F^c(\bar{x}, \bar{u}) + R_F^r(\bar{x}, \bar{u})) \subseteq P_i\}. \tag{2.42}$$

Finally, If $W_{i+1} = \emptyset$, we can terminate the computations as we already have computed the winning set and the controller. Otherwise, we add W_i and Γ_i into the overall winning set ($P_{i+1} \leftarrow P_i \cup W_{i+1}$) and controller ($C \leftarrow C \cup \Gamma_{i+1}$) and restart the depicted process.

2.3.3. Deployment

Once the controller C is computed such that $C \parallel \bar{\Sigma}$ realizes the given specification Φ, we need to deploy C onto an embedded controller platform, e.g., a microcontroller. Since such embedded controller platforms generally have a small on-board memory, we would like to minimize the size of the stored controller.

We define the *set of valid control inputs* corresponding to $\bar{x}$ as $C(\bar{x}) = \{\bar{u} \mid (\bar{x}, \bar{u}) \in C\}$. The approach we proposed for finding representations for the finite abstractions may not work, since we are not allowed to over-approximate $C(\bar{x})$, and thus the set of valid control inputs is not representable as a compact ℓ_∞ ball described by its center and radius. The following example illustrates a disconnected $C(\bar{x})$, which cannot be represented by an ℓ_∞ ball.

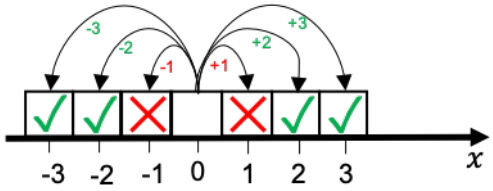

Figure 2.21.: Illustration of a disconnected set of valid control inputs.

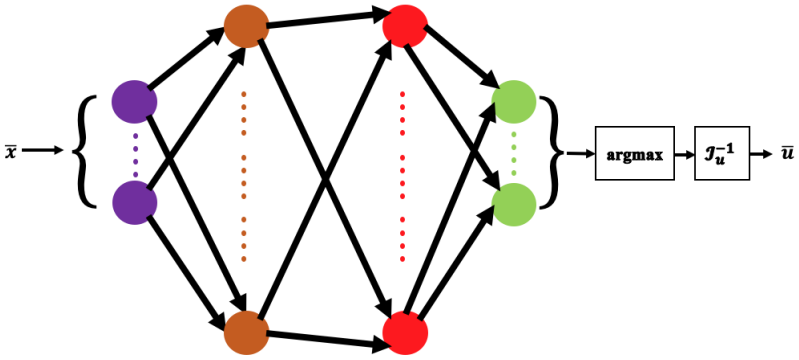

Figure 2.22.: The configuration used in compression of controllers. Given a state $\bar{x}$, the representation produces a corresponding control input $\bar{u}$.

Example 1 *Consider a system with one-dimensional state and input spaces ($n = m = 1$). Figure 2.21 illustrates the set of transitions starting from the white middle box ($\bar{x} = 0$). Let the boxes with green check mark and red cross mark correspond to the target and obstacle states and C be the controller for the corresponding reach-avoid specification. Then, we have $\{(0, 2), (0, 3), (0, -2), (0, -3)\} \subseteq C$ and $C(0) = \{-2, -3, 2, 3\}$. It is clear that $C(0)$ is a disconnected set, which is not characterizable by an ℓ_∞ ball.*

In contrast to the symbolic regression method proposed in [193], we formulate the controller compression problem as a classification task, that is, we train a neural network which assigns every state to a list of scores over the set of control inputs, and picks the control input with the highest score. The configuration of the neural network is illustrated in Figure 2.22. The justification for our formulation is that any representation for the controller can only perform well if it is trained over a dataset which respects the *continuity property*, i.e., neighboring states are not mapped into control input values which are very different from each other. A representation that respects the continuity property corresponds to a low continuity index (see Eq. (2.3)). During the training phase, we keep all the valid control inputs and let the training process to choose which value respects the continuity property more, by minimization of the cost function. Therefore, our formulation automatically takes care of the *redundancy* problem by mapping a neighborhood in the state space into close-in-value control inputs to respect the continuity requirement of the trained representation. The reason that our formulation does not correspond to a

Algorithm 6: Compression algorithm for the controller

Data: Controller C, learning rate λ

1 Compute the dataset $\mathcal{D}_C$ using Eq. (2.43)

2 Train the neural network $\mathcal{N}_C$ on the dataset $\mathcal{D}_C$ using the learning rate λ

3 Compute the set of state-input pairs $\tilde{C}$ using Eq. (2.45)

4 Compute $\hat{C}$ using Eq. (2.46)

Result: Corrected neural representation $\hat{C}$

standard classification setting is that during the training phase a *non-uniform* number of labels (corresponding to the control input values in the output stage of the neural network) per input (corresponding to the state values at the input layer of the neural network) are considered as valid, while we only will consider *one* label—corresponding to the highest score—as the trained representation's choice during the runtime.

Remark 8 *In order to formulate the problem of finding a neural-network-based representation for the controller as a regression problem, first the training data must be* pre-processed *such that the continuity property is respected, i.e., the set of valid control-inputs per each state is pruned so that neighboring states are mapped to close-in-value control inputs. However, this pre-processing is time consuming and does not work efficiently in practice (see, e.g., [193, 66]).*

Algorithm 6 summarizes the proposed procedure for computing a compressed representation for the original controller. In the first step, we need to store the training set

$$\mathcal{D}_C = \{(\bar{x}, \boldsymbol{h}(\bar{x})) \,|(\bar{x}, \bar{u}) \in C \Leftrightarrow \boldsymbol{h}(\bar{x})(\mathcal{I}_u(\bar{u})) = 1,\ (\bar{x}, \bar{u}) \notin C \Leftrightarrow \boldsymbol{h}(\bar{x})(\mathcal{I}_u(\bar{u})) = 0\}, \tag{2.43}$$

where $\mathcal{I}_u \colon U \to [1; |\bar{U}|]$ is an *indexing function* for the control set $\bar{U}$, which assigns every value in $\bar{U}$ into a *unique* integer in the interval $[1; |\bar{U}|]$. Intuitively, each point in the dataset $\mathcal{D}_C$ contains a state $\bar{x} \in L$ and a vector $\boldsymbol{h}(\bar{x})$ which is of length $|\bar{U}|$ and has ones at the entries corresponding to the valid control inputs and zeros elsewhere.

Once the training dataset is ready, we can train a neural network $\mathcal{N}_C$ which takes $\bar{x} \in \bar{X}$ as input and approximates ${\mathcal{I}_u}^{-1}(argmax(\boldsymbol{h}(\bar{x})))$ in the output, where ${\mathcal{I}_u}^{-1}(\cdot)$ denotes the inverse of the indexing function used in Eq. (2.43).

Remark 9 *Note that the output layer of $\mathcal{N}_C$ has to be of size $|\bar{U}|$ and for every $\bar{x} \in L$, we consider the value ${\mathcal{I}_u}^{-1}(argmax(\mathcal{N}_C(x)))$ as the final control input assigned by $\mathcal{N}_C$ to the state $\bar{x}$. Moreover, because $\bar{U}$ was computed by equally partitioning over U, both the indexing function $\mathcal{I}_u$ and its inverse can be implemented in a memory-efficient way using* floor *and* ceil *functions.*

Once the neural network $\mathcal{N}_C$ is trained, we evaluate its performance by finding all the states $\bar{x}$ at which using $\mathcal{N}_C$ produces an *invalid* control input, i.e.,

$$E = \{\bar{x} \in L \mid {\mathcal{I}_u}^{-1}(argmax(\mathcal{N}_C(\bar{x}))) \notin C(\bar{x})\}.$$

Table 2.5.: Catalog of models used to generate the finite abstractions in Subsection 2.3.4.

Case-study	Dynamical model	Configuration (1)				Configuration (2)			
		X	U	η_x	η_u	X	U	η_x	η_u
2D car $(x(1), x(2))$- position $(u(1), u(2))$- speed	$\begin{bmatrix}\dot{x}(1)\\ \dot{x}(2)\end{bmatrix} \in \begin{bmatrix}u(1)\\ u(2)\end{bmatrix} + W$ $\tau = 0.4,\ W = [-0.025, 0.025]^2$	$[0,5]^2$	$[-1,1]^2$	$\begin{bmatrix}0.05\\ 0.05\end{bmatrix}$	$\begin{bmatrix}0.23\\ 0.23\end{bmatrix}$	$[0,10]^2$	$[-2,2]^2$	$\begin{bmatrix}0.025\\ 0.025\end{bmatrix}$	$\begin{bmatrix}0.23\\ 0.23\end{bmatrix}$
3D car $(x(1), x(2))$- position $x(3)$- angle $u(1)$- speed $u(2)$- turn rate	$\begin{bmatrix}\dot{x}(1)\\ \dot{x}(2)\\ \dot{x}(3)\end{bmatrix} \in \begin{bmatrix}u(1)\cos(x(3))\\ u(1)\sin(x(3))\\ u(2)\end{bmatrix} + W$ $\tau = 0.3,\ W = \{\mathbf{0}\}$	$[0,5]^2 \times$ $[-1.6, 1.6]$	$[-1,1]^2$	$\begin{bmatrix}0.2\\ 0.2\\ 0.2\end{bmatrix}$	$\begin{bmatrix}0.3\\ 0.3\end{bmatrix}$	$[0,10]^2 \times$ $[-\pi, \pi]$	$[-1.5,1.5] \times$ $[-1,1]$	$\begin{bmatrix}0.1\\ 0.1\\ 0.1\end{bmatrix}$	$\begin{bmatrix}0.2\\ 0.2\end{bmatrix}$
4D car $(x(1), x(2))$- position $x(3)$- angle $x(4)$- speed $u(1)$- turn rate $u(2)$- acceleration control	$\begin{bmatrix}\dot{x}(1)\\ \dot{x}(2)\\ \dot{x}(3)\\ \dot{x}(4)\end{bmatrix} \in \begin{bmatrix}x(4)\cos(x(3))\\ x(4)\sin(x(3))\\ u(1)\\ u(2)\end{bmatrix} + W$ $\tau = 0.5,\ W = \{\mathbf{0}\}$	$[0,5]^2 \times$ $[-1.6, 1.6] \times$ $[-1,1]$	$[-1,1]^2$	$\begin{bmatrix}0.2\\ 0.2\\ 0.2\\ 0.2\end{bmatrix}$	$\begin{bmatrix}0.3\\ 0.3\end{bmatrix}$	$[0,10]^2 \times$ $[-\pi, \pi] \times$ $[-1,1]$	$[-2,2]^2$	$\begin{bmatrix}0.05\\ 0.05\\ 0.1\\ 0.1\end{bmatrix}$	$\begin{bmatrix}0.2\\ 0.2\end{bmatrix}$
5D car $(x(1), x(2))$- position $x(3)$- angle $x(4)$- speed speed $x(4)$- turn rate $u(1)$- acceleration $u(2)$- angular acceleration	$\begin{bmatrix}\dot{x}(1)\\ \dot{x}(2)\\ \dot{x}(3)\\ \dot{x}(4)\\ \dot{x}(5)\end{bmatrix} \in \begin{bmatrix}x(4)\cos(x(3))\\ x(4)\sin(x(3))\\ x(5)\\ u(1)\\ u(2)\end{bmatrix} + W$ $\tau = 0.5,\ W = \{\mathbf{0}\}$	$[0,5]^2 \times$ $[-1.6, 1.6] \times$ $[-1,1] \times$ $[-1,1]$	$[-1,1]^2$	$\begin{bmatrix}0.2\\ 0.2\\ 0.2\\ 0.2\\ 0.2\end{bmatrix}$	$\begin{bmatrix}0.3\\ 0.3\end{bmatrix}$	$[0,10]^2 \times$ $[-\pi, \pi] \times$ $[-1,1] \times$ $[-1,1]$	$[-2,2]^2$	$\begin{bmatrix}0.05\\ 0.05\\ 0.1\\ 0.1\\ 0.1\end{bmatrix}$	$\begin{bmatrix}0.2\\ 0.2\end{bmatrix}$

The misclassification rate of the trained classifier $\mathcal{N}_C$ is defined as:

$$err_C = \frac{|E|}{|L|}. \tag{2.44}$$

In order to maintain the guarantee provided by the original controller C, it is very important to *correct* the output of the trained representation, so that it outputs a valid control input at *every* state. In case the misclassification rate is small, we can store $\mathcal{N}_C$ together with $\tilde{C}$, where

$$\tilde{C} = \{(\bar{x}, \bar{u}) \mid \bar{x} \in E,\ \bar{u} \in C(\bar{x})\}. \tag{2.45}$$

The final deployable controller $\hat{C}$ consists of both $\mathcal{N}_C$ and $\tilde{C}$, and is defined as

$$\hat{C}(\bar{x}) :- \begin{cases} {\mathcal{I}_u}^{-1}(argmax(\mathcal{N}_C(\bar{x}))) & \text{if } \bar{x} \notin E \\ \tilde{C}(\bar{x}) & \text{if } \bar{x} \in E. \end{cases} \tag{2.46}$$

Lemma 2.3.4 *Let $\hat{C}$ be as defined in Eq. (2.46). The winning domain of both $\hat{C} \parallel \bar{\Sigma}$ and $C \parallel \bar{\Sigma}$ for satisfying a specification Φ is the same.*

2.3.4. Experimental Evaluation

We evaluate the performance of our proposed algorithms on several control systems. Dynamics of our control systems are listed in Table 2.5. We used configurations (1) and (2) in Table 2.5, respectively, for evaluating our methods for synthesis and deployment.

Table 2.6.: The results of regression-based controller synthesis for finite abstractions. $\bar{X} \times \bar{U}$ indicates the number of discrete state-input pairs, e_F, e_B denote the soundness errors, respectively, for the forward and backward representations, computed using Eq. (2.28), d_F and d_B give the graph mismatch rates for the forward and backward dynamics using using Eq. (4), $\mathcal{M}_T$ gives the memory needed to store the original transition system in kB, $\mathcal{M}_F + \mathcal{M}_B$ denotes the memory taken by the representing neural networks for the forward and backward dynamics in kB, $\mathcal{T}_c$ denotes the total execution time for computing the compressed representations in minutes. and $\mathcal{T}_s$ denotes the total execution time for synthesizing the controller in minutes.

Case study	$\lvert\bar{X}\rvert \times \lvert\bar{U}\rvert$	e_F	e_B	d_F	d_B	$\mathcal{M}_T$ (kB)	$\mathcal{M}_F + \mathcal{M}_B$ (kB)	$\mathcal{T}_c$ (min)	$\mathcal{T}_s$ (min)
2D car	810000	$\begin{bmatrix}1.02 \times 10^{-2}\\ 1.58 \times 10^{-2}\end{bmatrix}$	$\begin{bmatrix}2.81 \times 10^{-2}\\ 1.17 \times 10^{-2}\end{bmatrix}$	6.81×10^{-1}	9.64×10^{-1}	7.76×10^{4}	488	68.58	8.55
3D car	451584	$\begin{bmatrix}2.05 \times 10^{-2}\\ 2.19 \times 10^{-2}\\ 2.26 \times 10^{-2}\end{bmatrix}$	$\begin{bmatrix}2.48 \times 10^{-2}\\ 1.76 \times 10^{-2}\\ 2.32 \times 10^{-2}\end{bmatrix}$	7.11×10^{-1}	7.85×10^{-1}	1.35×10^{5}	488	65.46	14.50
4D car	4967424	$\begin{bmatrix}1.71 \times 10^{-2}\\ 2.40 \times 10^{-2}\\ 1.62 \times 10^{-2}\\ 1.96 \times 10^{-2}\end{bmatrix}$	$\begin{bmatrix}2.05 \times 10^{-2}\\ 1.54 \times 10^{-2}\\ 1.35 \times 10^{-2}\\ 1.25 \times 10^{-2}\end{bmatrix}$	4.24×10^{-1}	2.87×10^{-1}	5.58×10^{6}	488	446.23	20.55
5D car	30735936	$\begin{bmatrix}1.41 \times 10^{-2}\\ 1.18 \times 10^{-2}\\ 1.97 \times 10^{-2}\\ 2.22 \times 10^{-2}\\ 1.93 \times 10^{-2}\end{bmatrix}$	$\begin{bmatrix}2.11 \times 10^{-2}\\ 1.79 \times 10^{-2}\\ 1.13 \times 10^{-2}\\ 1.65 \times 10^{-2}\\ 2.45 \times 10^{-2}\end{bmatrix}$	5.34×10^{-1}	4.25×10^{-1}	$3.64 \times 10^{8}(OOM)$	488	3025.14	312.15
Inverted pendulum	17360	$\begin{bmatrix}2.53 \times 10^{-2}\\ 3.44 \times 10^{-2}\end{bmatrix}$	$\begin{bmatrix}2.31 \times 10^{-2}\\ 2.97 \times 10^{-2}\end{bmatrix}$	6.50×10^{-1}	5.61×10^{-1}	2.27×10^{4}	488	68.58	4.18
TORA	1433531	$\begin{bmatrix}2.53 \times 10^{-2}\\ 2.67 \times 10^{-2}\\ 2.39 \times 10^{-2}\\ 2.24 \times 10^{-2}\end{bmatrix}$	$\begin{bmatrix}2.21 \times 10^{-2}\\ 2.57 \times 10^{-2}\\ 1.88 \times 10^{-2}\\ 3.03 \times 10^{-2}\end{bmatrix}$	4.34×10^{-1}	4.15×10^{-1}	1.57×10^{7}	488	241.48	166.16

Table 2.7.: The results of classifier-based controller synthesis for finite abstractions. $\bar{X} \times \bar{U}$ indicates the number of discrete state-input pairs, err_F, err_B denote the soundness errors, respectively, for the forward and backward representations, computed using Eq. (2.36), d_F and d_B give the graph mismatch rates for the forward and backward dynamics, $\mathcal{M}_T$ gives the memory needed to store the original transition system in kB, $\mathcal{M}_F + \mathcal{M}_B$ denotes the memory taken by the representing neural networks for the forward and backward dynamics in kB, $\mathcal{T}_c$ denotes the total execution time for computing the compressed representations in minutes. and $\mathcal{T}_s$ denotes the total execution time for synthesizing the controller in minutes.

Case study	$\lvert\bar{X}\rvert \times \lvert\bar{U}\rvert$	err_F	err_B	d_F	d_B	$\mathcal{M}_T$ (kB)	$\mathcal{M}_F + \mathcal{M}_B$ (kB)	$\mathcal{T}_c$ (min)	$\mathcal{T}_s$ (min)
2D car	810000	2.75×10^{-2}	3.27×10^{-2}	2.65×10^{-2}	2.93×10^{-2}	7.76×10^{4}	1.33×10^{4}	68.58	10.71
3D car	451584	2.71×10^{-4}	2.21×10^{-6}	3.71×10^{-5}	9.47×10^{-7}	1.35×10^{5}	1.91×10^{4}	50.74	12.11
4D car	4967424	6.24×10^{-4}	0	2.84×10^{-4}	0	5.58×10^{6}	2.37×10^{4}	565.13	24.58
5D car	30735936	3.41×10^{-5}	5.33×10^{-8}	3.21×10^{-5}	2.19×10^{-8}	$3.64 \times 10^{8}(OOM)$	3.27×10^{4}	3421.21	215.88
Inverted pendulum	17360	6.03×10^{-2}	5.85×10^{-2}	0	0	2.27×10^{4}	2.08×10^{4}	8.21	8.33
TORA	1433531	1.27×10^{-1}	1.26×10^{-1}	1.55×10^{-1}	1.48×10^{-1}	1.57×10^{7}	2.38×10^{4}	234.87	159.75

We construct the transition system in all the case studies using the sampling approach in [93]. This approach generates T_F using sampled trajectories while providing confidence on the correctness of T_F. Our experiments were performed on a cluster with Intel Xeon E7-8857 v2 CPUs (32 cores in total) at 3GHz, with 100GB of RAM. For training neural networks, we did not use a distributed implementation as we found that distributing the process across GPUs actually decelerates the process. However, for the rest of our compression and synthesis algorithms, we used a distributed implementation.

Synthesis. We considered the ℓ_∞ ball centered at $(4, 4)$ with the radius 0.8 over the Euclidean plane as the target set for the multi-dimensional car examples, $[-0.5, 0.5] \times [-1, 1]$ for the inverted pendulum example, and $[-1, 1]^4$ for the TORA example. To evaluate our regression-based corrected neural method, we set the list of neuron numbers in different layers as $(n+m, 20, 40, 30, 2n)$, select the activation functions to be hyperbolic tangent, and set the learning rate to be $\lambda = 0.001$. As discussed in Subsection 2.3.2, the corrected neural representations for finite abstractions can also be constructed by solving a classification problem. To evaluate this method, we set the list of neuron numbers in different layers for both $\mathcal{N}_F$ and $\mathcal{N}_B$ as $(n + m, 40, 160, 160, 160, 160, 160, 160, 160, 160, 500, 800, 2\sum_{i=1}^{n} |\bar{X}(i)|)$, select the activation functions to be ReLU, and set the learning rate to be $\lambda = 0.0001$. We used stochastic gradient descent method with the corresponding learning rate for training the neural networks [150]. Tables 2.6 and 2.7 illustrate the synthesis results related to our experiments for finite abstractions, using the regression-based and classification-based methods, respectively. Although we used the same neural network structure for all the examples, soundness errors take small values that are bounded by 3.44×10^{-2} as the maximum of e_F and e_B in the regression-based method, and by 1.27×10^{-1} as the maximum of err_F and err_B in the classification-based method. Moreover, memory requirement of our proposed regression-based and classification-based methods at higher dimensions remains almost constant while the size of the transition system increases exponentially (see the illustration shown in Figure 2.23 (Left) for the multi-dimensional car case studies). Further, we notice that the regression-based method results in higher mismatch rates d_F and d_B compared to the classification-based method: on average, 5.87×10^{-1} versus 3.03×10^{-2} for d_F, and 6.15×10^{-1} versus 2.96×10^{-2} for d_B (see the illustration shown in Figure 2.23 (Right) for the multi-dimensional car case studies). Therefore, using the classification-based method, while being sound, produces a smaller graph, which is less restrictive for the synthesis purpose. Most importantly, memory requirement using both our approaches is way less than the memory needed to store the original (forward) transition system ($\mathcal{M}_F + \mathcal{M}_B << \mathcal{M}_T$). Regression-based method reduces the memory requirements by a factor of 1.31×10^5 and up to 7.54×10^5. However, the classification-based method reduces the memory requirements by a factor of 2.01×10^3 and up to 1.12×10^4. This shows that the regression-based method requires less memory compared to the classification-based method.

Deployment. Table 2.8 lists our experimental results for compressing the symbolic controllers. For $\mathcal{N}_C$, we set the list of neuron numbers in different layers for both $\mathcal{N}_F$ and $\mathcal{N}_B$ as $(n, 20, 80, 80, 80, 80, 80, 160, |\bar{U}|)$, select the activation functions to be rectified linear unit (ReLU), and set the learning rate to be $\lambda = 0.0001$. It can be noticed that

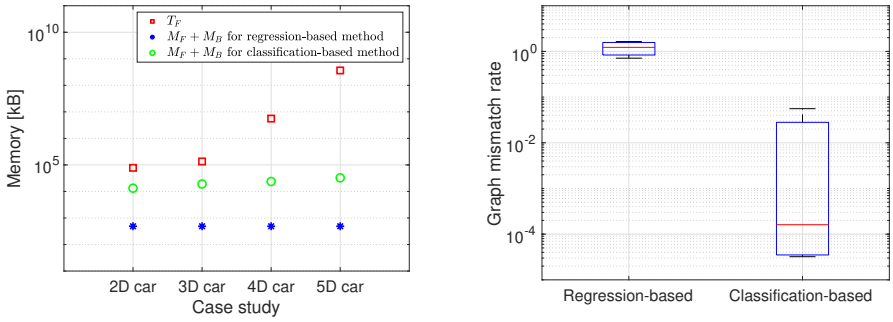

Figure 2.23.: Left: Memory requirement of different methods for storing transition systems of multi-dimensional cars (cf. Table 2.5) in logarithmic scale. Right: Distribution of total graph mismatch rate ($d_F + d_B$) for our proposed methods in logarithmic scale.

Table 2.8.: The results of controller compression. $|C|$ gives the number of state-input pairs in the original controller, err_C denotes the portion of the states at which the representing neural network produces non-valid control inputs computed using Eq. (2.44), $\mathcal{M}_C$ gives the memory needed to store the original controller in kB, $\mathcal{M}_{\hat{C}}$ denotes the memory taken by the representing neural network in kB, and $\mathcal{T}$ denotes the total execution time for our implementation in minutes.

Case study	$\lvert C\rvert$	err_C	$\mathcal{M}_C$ (kB)	$\mathcal{M}_{\hat{C}}(kB)$	$\mathcal{T}$ (min)
2D car	2.15×10^6	1.85×10^{-5}	2.75×10^5	1.21×10^3	6.31
3D car	2.87×10^6	2.16×10^{-3}	4.65×10^5	1.05×10^3	19.14
4D car	9.35×10^7	3.63×10^{-2}	2.24×10^6	1.35×10^3	39.48
5D car	1.69×10^9	4.51×10^{-3}	4.71×10^7	1.48×10^3	201.86
Inverted pendulum	8.16×10^5	1.08×10^{-3}	7.83×10^4	8.92×10^2	7.51
TORA	4.78×10^7	3.78×10^{-4}	7.65×10^6	8.92×10^2	113.97

err_C is very small for all the examples. Therefore, we only need to store a very small portion of C in addition to $\mathcal{N}_C$. As it can be observed in Table 2.8, our method has been successful in computing representations which are very accurate and compact-in-size ($\mathcal{M}_{\hat{C}} << \mathcal{M}_C$).

Parametrization. Our approach requires selecting the hyperparameters of the training process and choosing the structure of the neural networks. We have performed several experiments to select the hyperparameters of the training (e.g., the learning rate, epoch number, and batch size). Regarding the structure of the neural networks, we have explored different choices such as the type of the activation functions (hyperbolic tangent, ReLU, etc.), number of neurons per layer, and the depth. Increasing the complexity of the neural network, by increasing the number of neurons per layer or depth, leads to a better performance. Note that the neural networks employed in our setting are not supposed to make any generalization over unseen data. Therefore, our approach does not

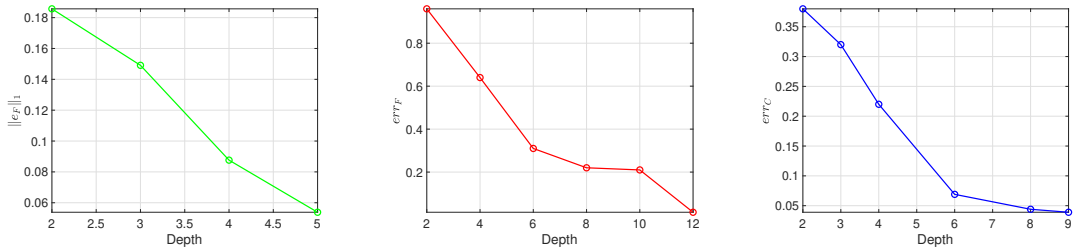

Figure 2.24.: Demonstrating the effect of increasing the depth of the neural representation on the norm of the soundness error e_F (cf. Eq. (2.28)) for regression-based controller synthesis (Left), the soundness error err_F (cf. Eq. (2.36)) for classification-based controller synthesis (Middle), and the misclassification rate (cf. Eq. (2.44)) for deployment (Right). The experiments are performed on the 3D car example.

suffer from over-parametrization of the neural networks. We have demonstrated this in Figure 2.24 by providing the error as a function of the depth of the neural representation for the 3D car example. The error always decreases by increasing the depth of the neural representation. Therefore, the structure of the neural representations can be selected for having an acceptable accuracy within a given time bound for the training process.

2.4. ABCD for Multi-Agent Systems with Reach-Avoid Specifications

We consider the *decentralized* feedback controller synthesis problem for *multi-agent*, nonlinear systems against temporal *reach-avoid* specifications. By multi-agent, we mean that the systems under study are composed of a number of concurrently executing components. Each component is modeled as a possibly nonlinear dynamical system that evolves under the influence of a control as well as an environmental disturbance. Our specifications require that the global state of the system eventually reaches a target while avoiding certain bad states along the way. While the dynamics of each component is independent of the others, the overall trajectories are coupled by the global specification. Decentralized means that we require a solution in which each component has a local feedback controller that sees only the local state, but the combination of all the closed loops satisfy the global specification. Above all, our goal is to ensure the resulting controllers are *provably correct* against the worst-case model of disturbances.

Such multi-agent control problems are ubiquitous in the domain of robotics, where a number of (possibly heterogeneous) mobile robots move concurrently in a shared workspace. A global specification can ask, for example, that a set of robots be able to reach certain locations while avoiding collisions among themselves or with obstacles in the environment, or that a set of drones fly in formation while reaching a target. Indeed, automatic generation of decentralized controllers is a classical problem in robotics, artificial intelligence, and control theory, and there is an enormous literature on the

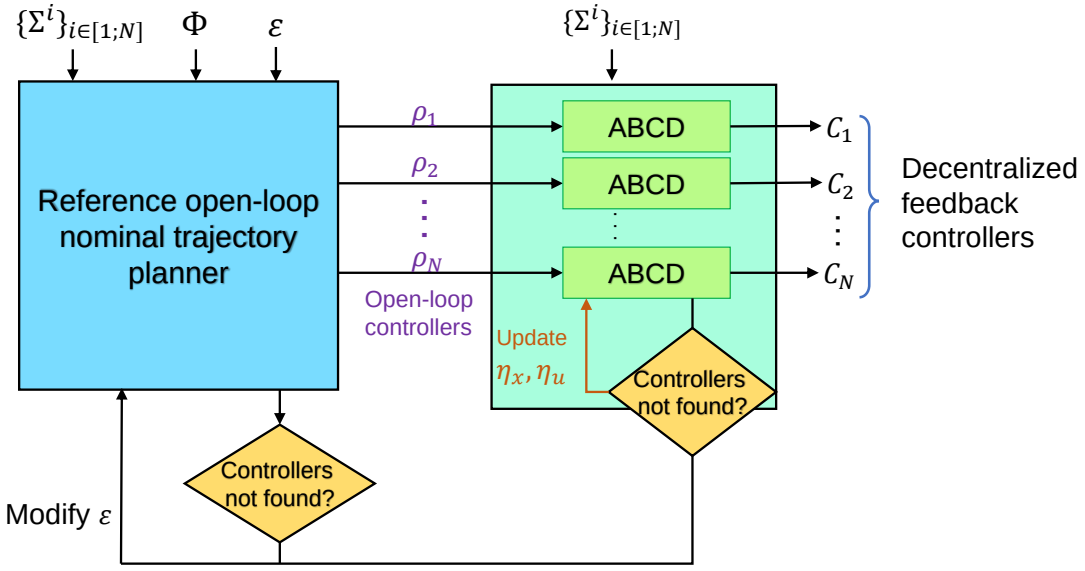

Figure 2.25.: Overall algorithm: The blue block on the left (centrally) computes a joint open-loop nominal trajectory for the overall system. The green block on the right computes decentralized controllers for tracking the nominal trajectory using Abstraction Based Controller Design (ABCD). $\{\Sigma^i\}_{i\in[1;N]}$ is a set of N agents, Φ is a global reach-avoid specification, ε is a robustness margin, $\rho_1, \ldots, \rho_N$ are the local projections of the nominal trajectory, η_x, η_u are parameters used in ABCD, and $C_1, \ldots, C_N$ are the sought local feedback controllers for the individual agents.

subject—too many to enumerate—across these disciplines.

Despite the large body of research, few techniques today can handle all our desiderata. Multi-agent planning algorithms, such as (hybrid variants of) A* search, scale to large systems but typically either disregard or simplify the underlying dynamics and work with geometric or discrete models, or disregard the effect of disturbances or nonlinear dynamics. Most planning and trajectory optimization techniques handle the *nominal* dynamics, i.e., the dynamics free of disturbances, and construct *open-loop* controllers. However, the open-loop behaviors do not guarantee satisfaction of the specifications in the presence of disturbances. On the other hand, correct-by-construction controller synthesis techniques from control theory, such as abstraction-based control design (ABCD) or Hamilton-Jacobi techniques, handle precise models of nonlinear dynamics and the effects of disturbance, but are difficult to scale beyond about 10 dimensions.

We provide a simple but effective *combined* approach. We use a global planning approach for nominal trajectory generation and a local correct-by-construction feedback controller synthesis approach for guaranteed adherence to specification for each component in the presence of disturbances. Figure 2.25 shows the overall algorithm. In the first step,

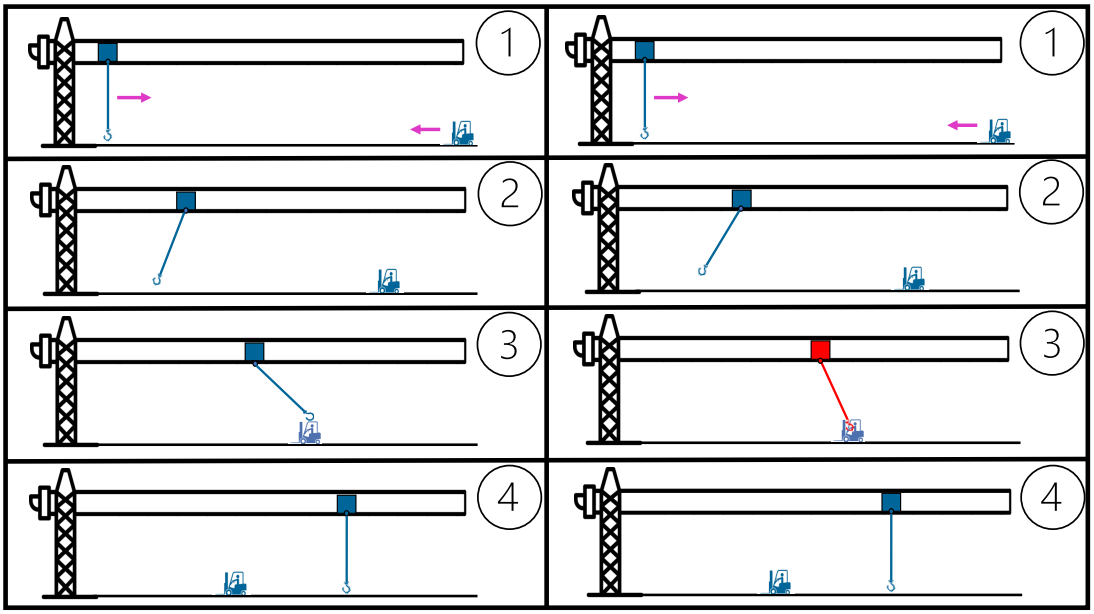

Figure 2.26.: Illustration of the trajectories generated by the open-loop controller for the crane and vehicle example under disturbance-free (**left**) and perturbed (**right**) situations.

given a set of control systems, one for each component, and a reach-avoid specification on the global state space, we use a trajectory planner to find a nominal open-loop controller for the global system. The trajectory planner ignores the effect of disturbances, but takes a *robustness parameter* ε. The role of the robustness parameter is to ensure that the specification is robustly satisfied: Every trajectory within an ε-tube of the open-loop trajectory also satisfies the specification.

Next, we project the unique open-loop trajectory produced by the open-loop controller on to a nominal trajectory for each individual component. The robustness of the trajectory means that there is a tube around each nominal trajectory. In a second step, we solve a number of local *guaranteed tracking control* problems, where we synthesize correct-by-construction controllers whose objective is to track the nominal trajectory while staying within the tube. The overall algorithm is more scalable and guarantees satisfaction of the global specification.

We show empirically that our algorithm is able to generate provably correct feedback controllers for many systems for which neither technique is individually effective. Of course, since we decompose the problem, it is possible that there is no controller for a particular choice of the robustness parameter, or indeed, for other parameters used by the individual tools. In that case, there is an outer loop that searches through the parameter space.

We have implemented our approach in an open-source tool called GAMARA (stands for GuAranteed Multi-Agent Reach-Avoid control) by combining the following two tools:

the ALTRO open-loop trajectory planner [78] and the SCOTS correct-by-construction controller synthesis tool [152]. ALTRO is a state-of-the-art trajectory planning tool based on optimal control. It handles nonlinear dynamics and scales to large dimensions, but ignores disturbances or modeling uncertainties. SCOTS implements a highly parallelizable ABCD algorithm that generates a feedback controller for satisfying temporal specification.

We empirically evaluate GAMARA on a number of multi-robot benchmarks, including a coordinated reach-avoid problem for ground robots, a formation control problem for drones, and a lane merging schenario for autonomous vehicles. In each case, we demonstrate that GAMARA can find decentralized and correct controllers within reasonable time and memory bounds.

Figure 2.26 shows a concrete multi-robot reach-avoid scenario with an overhead crane hanging from a trolley along a horizontal rail and a cart that drives underneath the crane in the same horizontal axis. The goal is to move the crane and the vehicle such that they do not collide. Figure 2.26 (**left**) shows an animation of a possible open-loop behavior from an initial configuration (Frame 1) to a final one (Frame 4), where the crane and the vehicle have crossed each other. The trajectory is generated by accelerating the trolley, causing the crane to swing up and thus creating enough space for the vehicle to pass. Unfortunately, in the presence of disturbances, such as wind or a slippery floor, a trajectory may not be free of collisions: the same open-loop behavior can cause a collision (Figure 2.26 **right**). Instead, GAMARA computes a global *robust* trajectory for the system; the robustness parameter ensures a "wider berth." The global trajectory is projected to the crane and the vehicle, and we compute guaranteed tracking controllers that ensure there is no collision despite the disturbances. In our experiments, planning with ALTRO took less than a second and feedback controller synthesis with SCOTS about 10 minutes. At the same time, a global approach to find a correct solution does not scale. The global state space is 10^{10} times larger and SCOTS timed out with 1.5 TB of memory. The content of this section is based on our paper [120].

Related Work. The field of multi-agent planning is too large for a comprehensive survey; we point to the text books [108, 109, 39, 153] for an introduction. We categorize closely related work into (1) those combining planning and tracking controller synthesis, (2) those addressing formal multi-agent controller synthesis, and (3) those combining (1) and (2). We provide a survey of these categories.

Combining planning and tracking. Techniques combining high-level planning and low-level tracking are a staple of classical planning and control. More recently, several techniques consider the problem of formal guarantees for such planners. Existing works differ in the dynamics that they can handle (e.g., linear or nonlinear), considered class of specifications, including disturbances, and scalability. A common approach is to perform the high-level planning over a lower dimensional model and then use Sum-of-Squares programming (SOS), Hamilton Jacobi (HJ), or satisfiability modulo convex programming (SMC) to obtain a low-level controller ensuring a bounded error between the two models [74, 125, 163, 137].

In contrast to [74, 163], our method does not require finding a linear mapping between the low and high dimensional models. Meyer et al. [125] considered reach-avoid problems for perturbed non-linear control affine systems. They create a lower dimensional model

and use SOS programming to compute a controller ensuring a bounded error between the two models. Then, they use ABCD to compute a controller for the low-order model while taking the error into account. While their method can provide guarantee against worst-case disturbances, it is not clear if SOS always scales to the higher dimensions.

Nilsson et al. [137] provide a method that decomposes the state space into a lower-order planning space, and a higher-order internal dynamics space, so that fast planning and accurate tracking can be achieved using a set of control barrier functions computed based on SOS. Despite providing guarantees for the worst-case bounded disturbances, their method is not capable of solving reach-avoid tasks which involve dynamic obstacles as in the multi-agent case. While we have chosen SCOTS since the underlying algorithm can be effectively parallelized [95], in principle, we could also use SOS, HJ, or SMC approaches.

Other works only consider special classes of models such as linear [58, 187, 148], disturbance-free [174, 59, 166], or finite transition systems [192]. In contrast, our method supports arbitrary nonlinear dynamics and provides a guarantee against worst-case bounded disturbances.

Formal multi-agent synthesis. Chen et al. [37] provide a method, using control barrier functions, that requires some form of inter-robot communication and does not consider external disturbances. Sahin et al. [156] propose a method that requires the group of robots to be homogeneous. There are methods which do not consider external disturbances and do not provide formal guarantees [86].

Combinations. Alonso-Mora et al. [5] provide a method for formation control of a group of communicating homogeneous robots. They first synthesize a nominal controller using a fast randomized geometric planning method, namely RRT, and then use optimal control to track the obtained nominal solution. Unlike us, they neither consider external disturbances nor provide formal guarantees. Pant et al. [141] have studied multi-quadrotor missions with signal temporal logic (STL) specifications. They find the reference trajectory by maximizing robustness of the STL specification, and then synthesize tracking controllers. Their method can only handle specifications with bounded horizon and does not provide any guarantee against disturbances.

Xiao et al. [190] propose a method for synthesis of distributed controllers for a set of autonomous vehicles in a lane merging situation. They consider only linear systems as vehicle models, use global optimal control to find a nominal controller, and employ local control barrier functions with safety constraints. Their designed controllers is not provably safe in the presence of disturbance and can occasionally violate the safety constraints. Nikou et al. [136] have studied the problem of robust navigation for multi-agent systems based on nominal reference trajectory and pre-computed feedback controllers. Their approach requires sensing capabilities of the agents to avoid collision. In contrast, our method does not requires any sensing capabilities of the agents. Sun et al. [170] have studied motion planning of multi-agent systems with linear temporal logic (LTL) specifications, under the presence of disturbances and denial of service attacks. Their approach uses SMC programming to compute a feasible nominal trajectory and employs feedback controllers to gain robustness. Despite being able to provide guarantees against disturbances, their implementation is centralized, thus the required time increases significantly

Table 2.9.: Features of the publicly available tools compared to GAMARA. Note that some of these tools can handle richer classes of specifications, compared to the reach-avoid problem handled by GAMARA.

Tool name	Non-linear Dynamics	Formal Guarantee	Multi-agent	Decentralized Controllers
GAMARA	✓	✓	✓	✓
SCOTS [152]	✓	✓		
ALTRO [78]	✓			
FastTrack [74]	✓	✓		
RealSyn [58]		✓		
Factest [59]	✓			
Model mismatch (SOS) [163]	✓			
RTD [100]	✓	✓		
Fly-by-Logic [141]	✓		✓	✓
Distributed team lift [86]	✓		✓	

for high-dimensional reach-avoid specifications.

There are other works that use a pre-defined motion primitive library to perform planning for multi-robot systems [155, 18, 64, 50]. In contrast, our method deals with the dynamical model directly.

Our construction can also be seen as an *assume-guarantee* technique that decomposes the global problem based on nominal trajectory tubes. Similar decompositions have been studied in the discrete case [6, 117]. The closest related work that matches our level of generality is the work by Bansal et al. [17]. However, they assume that each robot has its own reach-avoid specification while avoiding collision with the other robots. In contrast, we allow global reach-avoid specifications, which subsume their class of specifications. In fact, there are control problems that can be easily handled by our approach and cannot be encoded in their setting. An example is robots maintaining a formation while fulfilling their tasks [5].

A subset of approaches listed above have available implementations. In Table 2.9, we summarize the main features of the publicly available tools. We highlight that our tool GAMARA is the only one that fulfills all the criteria.

2.4.1. Problem Statement

We now consider the decentralized controller synthesis problem for a set of control systems $\{\Sigma^i\}$ w.r.t. a global reach-avoid specification $\Phi = \neg \mathit{Avoid}\, \mathcal{U}\, \mathit{Goal}$, where $\mathit{Avoid}, \mathit{Goal} \subseteq X^{\times}$ are subsets of the product state space.

First, we define a *robust version* of the control specification. Let $\varepsilon \in \mathbb{R}^n_{>0}$ be a *robustness margin*. We define the ε-robust version of Φ, denoted by $\Phi_\varepsilon := (\neg \mathit{Avoid}'\, \mathcal{U}\, \mathit{Goal}')$, where $\mathit{Avoid}' = \mathit{Avoid} \oplus \Omega_\varepsilon(0)$ and $\mathit{Goal}' = \mathit{Goal} \ominus \Omega_\varepsilon(0)$, and $\oplus$ and $\ominus$ are set operators denoting the Minkowski addition and difference, respectively. Intuitively, if a trajectory $x_0, x_1, \ldots$ satisfies Φ_ε, then any trajectory $y_0, y_1, \ldots$ such that $\|x_i - y_i\| \leq \varepsilon$ satisfies Φ.

Problem 2.5 (Decentralized Controller Synthesis) ***Inputs:*** *Control systems $\Sigma^i = (X^i, x^i_{\mathsf{in}}, U^i, W^i, f^i)$, $i \in [1; N]$, global specification $\Phi = \neg \mathit{Avoid}\, \mathcal{U}\, \mathit{Goal}$, and A robustness margin $\varepsilon \in \mathbb{R}^n_{>0}$.*
Outputs: *Local feedback controllers $\{C^i\}$ for $\{\Sigma^i\}$, $i \in [1; N]$, such that $\{C^i\} \parallel \{\Sigma^i\}$ realizes Φ.*

It is important to notice that any solution for this problem is required to provide a *formal guarantee* on the satisfaction of Φ, i.e., the reach-avoid specification Φ *must* be satisfied under every value of the disturbances affecting the control systems. Further, the solution must not require any information exchange between the different agents. Embedding this feature simplifies implementation by eliminating the need for regular synchronization between agents at run time.

2.4.2. Solution Outline

To solve the decentralized control problem, we first plan a high-level nominal trajectory for the product system by ignoring the disturbances, and then synthesize low-level formally verified controllers for robustly tracking the nominal trajectory under worst-case disturbances. We summarize our approach for solving Problem 2.5 in Algorithm 7. The approach is composed of three main steps: (1) Synthesize a global open-loop controller for the *nominal system* as a single planner task on the product system to satisfy Φ_ε; (2) Project the controller into local controllers and obtain a nominal trajectory for each system; and (3) Design local closed-loop controllers to track the nominal trajectory while always staying within the robustness margin. The soundness of the technique is summarized below.

Theorem 2.4.1 *Local feedback controllers $\{C^i\}$ synthesized by Algorithm 7 guarantee that the the product system $\{C^i\} \parallel \{\Sigma^i\}$ realizes the global specification Φ.*

Proof *Note that Φ_ε is a stronger version of Φ and is intentionally made conservative to allow for ε-deviation in the trajectory of the product system. Since $\{\rho^i\}$ is the unique solution of the nominal product system and satisfies Φ_ε, it is guaranteed that ε-perturbation of this nominal trajectory satisfies Φ. It can be observed that all solutions of Σ^i stay within distance ε^i of the nominal trajectory ρ^i regardless of the disturbance. This completes the proof.*

Algorithm 7: Multi-agent controller synthesis

1. For every i, let $\Sigma^i_{nom} = (X^i, x^i_{\mathsf{in}}, U^i, \{0\}, f^i)$ be the nominal control system of Σ^i that ignores the disturbance. Compute the product control system $\Sigma^\times_{nom}$ of $\{\Sigma^i_{nom}\}$. Use a scalable *planner* to compute a nominal open-loop controller $C^\times_{nom} : [0; T] \to U^\times$ such that the specification Φ_ε is satisfied by $C^\times_{nom} \triangleright \Sigma^\times_{nom}$. Note that T is the first time the set $Goal'$ is visited.

2. Decompose $C^\times_{nom}$ into *local* open-loop controllers $\{C^i_{nom}\}$ for the set of $\{\Sigma^i_{nom}\}$ by projecting the output of $C^\times_{nom}$ into local input spaces U^i. Further, for every i, find the unique nominal open-loop trajectory $\rho^i = (x^i_{0,nom}, \ldots, x^i_{T,nom})$ of $C^i_{nom} \triangleright \Sigma^i_{nom}$. These trajectories are unique since there is no disturbance.

3. Let $\varepsilon^i \in \mathbb{R}^{n_i}_{>0}$, $i \in [1; N]$, be the projections of ε compatible with the state dimensions of Σ^i. Each control system Σ^i uses a *guaranteed tracking* method to compute a closed-loop controller C^i such that $C^i \parallel \Sigma^i$ *tracks* the nominal trajectory ρ^i and stays within its ε^i-neighborhood, i.e., $C^i \parallel \Sigma^i$ satisfies the specification

$$\Phi^i_{track} \coloneqq \bigwedge_{k \in [0;T]} \bigcirc^k \Omega_{\varepsilon^i}(x^i_{k,nom}). \tag{2.47}$$

Next, we discuss some implementation details of Algorithm 7 for the global *open-loop planner* (Step 1) and the local *guaranteed trajectory tracking* (Step 3) in our tool GAMARA. Note that Step 2 is a simple projection from the product space into local spaces. While we instantiate particular techniques, our method can be used with other implementations as well.

2.4.3. Open-loop Planning

The planner used for generating nominal trajectories in Step 1 of our algorithm should be fast and scalable. In addition, it should be capable of handling non-linear dynamics and constraints. Our choice for the planner is ALTRO [78]. ALTRO is a fast and numerically robust solver for constrained trajectory optimization problems and and is capable of handling nonlinear state and input constraints. Given a product system $\Sigma^\times_{nom}$, reach-avoid specification Φ_ε and time horizon T, ALTRO computes an open-loop controller

$C^{\times}_{nom} : [0;T] \to U^{\times}$ by solving the optimization

$$\begin{aligned} \underset{u_0^{\times},u_1^{\times},\ldots,u_T^{\times}}{\text{minimize}} \quad & \ell_T(x_T^{\times}) + \sum_{k=0}^{T-1} \ell_k(x_k^{\times}, u_k^{\times}) \\ \text{subject to} \quad & x_{k+1}^{\times} = f^{\times}(x_k^{\times}, u_k^{\times}), \ \ \forall k \in [0; T-1] \\ & g(x_k^{\times}, u_k^{\times}) \leq 0, \qquad \forall k \in [0;T] \\ & h(x_k^{\times}, u_k^{\times}) = 0, \qquad \forall k \in [0;T], \end{aligned}$$

where $\ell_k(\cdot,\cdot)$ denotes a quadratic objective function assigning cost to each pair of state and input before the end of horizon, $\ell_T(\cdot)$ represents a quadratic objective function assigning penalty to the final state $x_T^{\times}$ being away from the goal set Goal'. The constraints $g(x_k^{\times}, u_k^{\times}) \leq 0$ and $h(x_k^{\times}, u_k^{\times}) = 0$ capture the requirement that at each time k the state should not be in Avoid', the state $x_T^{\times}$ should be in Goal', and the input $u_k^{\times}$ should always be in $U^{\times}$. In multi-robot scenarios, the inequality constraints can be used to define collision and obstacle avoidance specifications and the equality constraints can define fixed formation specification. Note that the reach-avoid specification is fulfilled if the corresponding equality and inequality constraints (i.e., $g(\cdot) \leq 0$, $h(\cdot) = 0$) are satisfied at every time-step and thus choice of the quadratic objective function (ℓ_k for $k \in [0;T]$) is not crucial.

Remark 10 *ALTRO only supports bounded horizon control problems. For this reason, we model the states in Goal' as a sink state and select a time horizon T for solving the planning task on the nominal product system. We increase the horizon T if ALTRO is not able to find a controller. We remark that this is an ALTRO-specific implementation detail, and our overall method does not rely on a fixed time horizon.*

2.4.4. Guaranteed Trajectory Tracking

Trajectories computed in the planning stage might not be followed in the presence of disturbance and therefore we need to use a formally guaranteed tracking controller to satisfy the given reach-avoid specification. We use abstraction-based controller design (ABCD) for Step 3. ABCD can handle nonlinear dynamics, (bounded) uncertainties, and ω-regular specifications. In particular, we use the implementation of ABCD in the tool called SCOTS [152].

Optmization: Local ABCD around the nominal trajectory. The abstraction process of ABCD usually requires computation of abstract transitions over the whole compact set X, which is computationally expensive. Luckily, for Step 3 of Algorithm 7, we only need to compute transitions in the ε-neighborhood of the given nominal trajectory. Given a control system Σ, together with a reference open-loop trajectory $\rho = (x_{0,nom}, \ldots, x_{T,nom})$ and a tube size $\varepsilon \in \mathbb{R}^n_{>0}$, we iteratively construct a tube as union of ε-balls around the reference trajectory (note that we have omitted the system index i for simpler notation). Next, we compute finite state abstraction for Σ for the chosen parameters η_x, η_u by

setting

$$X \coloneqq \bigcup_{k=0}^{T} \Omega_\varepsilon(x_{k,nom}).$$

In practice, this local computation of the abstraction is the key to scalability.

Remark 11 *Our decentralized controller synthesis approach features an interplay between the global open-loop planning and local formal synthesis via the robustness parameter ε and the discretization parameters η_x, η_u. The parameter ε should be large enough to allow deviation from the nominal trajectory caused by the disturbance. A small ε makes the local specification Φ^i_{track} difficult for ABCD synthesis thus requiring large computational complexity with smaller discretization parameters η_x, η_u. On the other hand, large ε makes the specification Φ_ε very conservative or infeasible for the global open-loop planning. Therefore, appropriate parameters should be selected iteratively for a successful controller synthesis.*

2.4.5. Hybrid vs Geometric Planning

Note that Step 3 of Algorithm 7 does not use the nominal controller obtained in Step 1 and requires only the nominal trajectories. Then one could argue that, instead of using ALTRO to generate nominal trajectories, a fast *geometric planner* [92] can be employed to generate geometric plans. However, fast geometric planners do not usually take into consideration the dynamics and control constraints. In our experience, the plans for nominal trajectories generated while ignoring the system dynamics are often untrackable unless the underlying system has special properties (e.g., differential flatness [131]). This is especially true for systems with restricted control capabilities or under-actuated systems. We demonstrate this phenomenon on a control system Σ that is a simple 2-dimensional pendulum with the following nominal dynamics:

$$\dot{x}(1) = x(2) \quad \dot{x}(2) = -\sin(x(1)) + u/5,$$

where x_1 represents the angle (in Radian) of the pendulum rod measured counter-clockwise from the vertical downward position, and x_2 represents the rate of change of x_1 or the angular velocity. Suppose the initial state of the pendulum is $(0, 0)$, i.e., when the pendulum is in the vertical downward position and is stationary. Suppose we want to find a controller for the goal $Goal = \{(\pi, 0)\}$, i.e., when the pendulum is in the vertical upward position and is stationary. The set of unsafe states *Avoid* is empty, i.e., no safety constraint is imposed. When we use ALTRO to compute an open-loop controller C, unsurprisingly, the controlled trajectory of $C \triangleright \Sigma$ looks like a spiral, as shown in blue Figure 2.27.

The synthesis of tracking controller using ABCD is indeed successful when we feed this nominal trajectory to our ABCD solver. However, if we use a geometric planner for this example that ignores the dynamics, the nominal trajectory would be a straight line path from $(0, 0)$ to $(\pi, 0)$ (shown using a dashed line in Figure 2.27), which gives an infeasible tracking problem for ABCD, due to the restrictions on possible trajectories of the pendulum coming from the its dynamics.

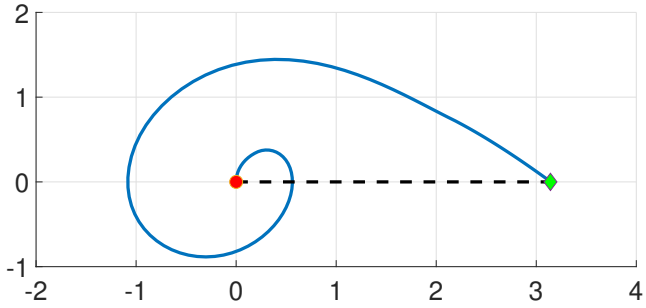

Figure 2.27.: The state trajectory of the inverted pendulum: the red point is the initial state, the green point is the final state, the blue spiral is the nominal trajectory obtained from ALTRO, and the dashed straight line is a geometric plan.

Table 2.10.: Runtimes for four case studies. Run times (in seconds) for computing open-loop controllers over the corresponding product spaces using ALTRO (T_g^{plan}), number of state-input pairs of the finite abstraction for the largest ABCD task (N_l), abstraction and synthesis times in SCOTS for that task (respectively T_l^{abs} and T_l^{syn}), number of state-input pairs of the finite state abstraction for global ABCD (N_g), abstraction and synthesis times for computing a global controller for the product system using SCOTS (respectively T_g^{abs} and T_g^{syn}). "OOM" denotes "out of memory" on a 1.5TB RAM machine.

Case-study	Global planning	Local ABCD			Global ABCD		
	T_g^{plan}	N_l	T_l^{abs}	T_l^{syn}	N_g	T_g^{abs}	T_g^{syn}
Multi-drone path planning	77.85	1.13×10^8	30.75	6.66	2.70×10^{110}	OOM	OOM
Crane and vehicle	0.65	8.56×10^8	511.24	91.43	2.16×10^{18}	OOM	OOM
Lane merging	89.02	1.07×10^8	22.79	5.29	1.69×10^{59}	OOM	OOM
Multi-drone formation control	114.34	1.55×10^8	39.46	7.83	3.65×10^{50}	OOM	OOM

2.4.6. Experimental Evaluation

We have implemented our approach in the open source tool GAMARA.[1] We evaluate the effectiveness of GAMARA on two distinct categories of problems: local reach-avoid problems with collision avoidance and global formation control problems. We consider four case studies: multi-drone path planning, crane and vehicle, lane merging, and multi-drone formation control. The design of nominal controller using ALTRO for all experiments was performed on a machine with core i5-4590 CPU at 3.30 GHz, with 16 GB of RAM. The formal controller synthesis using SCOTS for all systems except crane system was performed on the same machine. Controller synthesis for crane system is done on a cluster with 4 Intel Xeon E7-8857 v2 CPUs (48 cores in total) at 3 GHz, with 1.5 TB of RAM.

In all of our case studies, the robots are moving in a two-dimensional shared workspace (related but not exactly the same as the robots' state spaces) that possibly has obstacles.

[1] GAMARA is available online: https://github.com/MehrdadZareian/GAMARA.

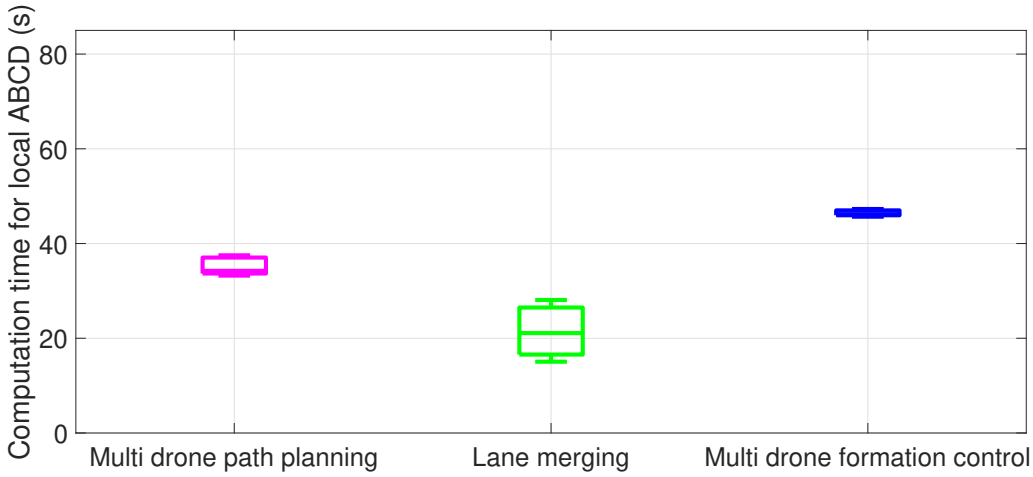

Figure 2.28.: Variations of run times of local ABCD among different agents for three case studies.

Table 2.10 shows run times for different stages of each experiment. For local ABCD, the reported numbers correspond to the maximum value among all of the agents. This choice is due to the fact that feedback controllers for different agents can be computed independently in parallel over different machines. To provide a more fine-grained comparison, Figure 2.28 shows the variations of run times of the different local ABCD tasks for every experiment. We have excluded the crane and vehicle case study in the figure due to an expected large variance originating from different dynamics. Notice that a higher number of state-input pairs does not necessarily result in a higher run time for local ABCD as the number of transitions and features of the parallel implementation can play a role. We compare GAMARA with ABCD applied to the product system to satisfy the global specification. As reflected in Table 2.10, memory requirement for global ABCD exceeds both system's (laptop and cluster) limit (1.5TB of RAM) in all of the experiments.

Local Reach-Avoid with Collision Avoidance

We first consider situations when each robot has an individual reachability specification, and they need to ensure a minimum safe distance from each other and the obstacles.

Suppose $\{\Sigma^i\}$ models a set of robots, $Goal^i \subseteq X^i$ are the individual goal sets, and $\delta \in \mathbb{R}_{>0}$ is a safety margin for collisions. We consider each robot as a point object with a bounding box for its physical dimensions. The parameter δ is chosen to be a constant greater than twice the radius of the bounding box around each robot. By keeping a distance at least δ from the other robots and the obstacles, the robots can avoid collision in their physical domain. The parameter δ can additionally take into account statutory minimum safe distances among the robots, such as in autonomous driving-like scenarios. The choice of δ is completely independent of the choice of ε. The latter is a robustness margin introduced to take into account the deviation of the system trajectories under

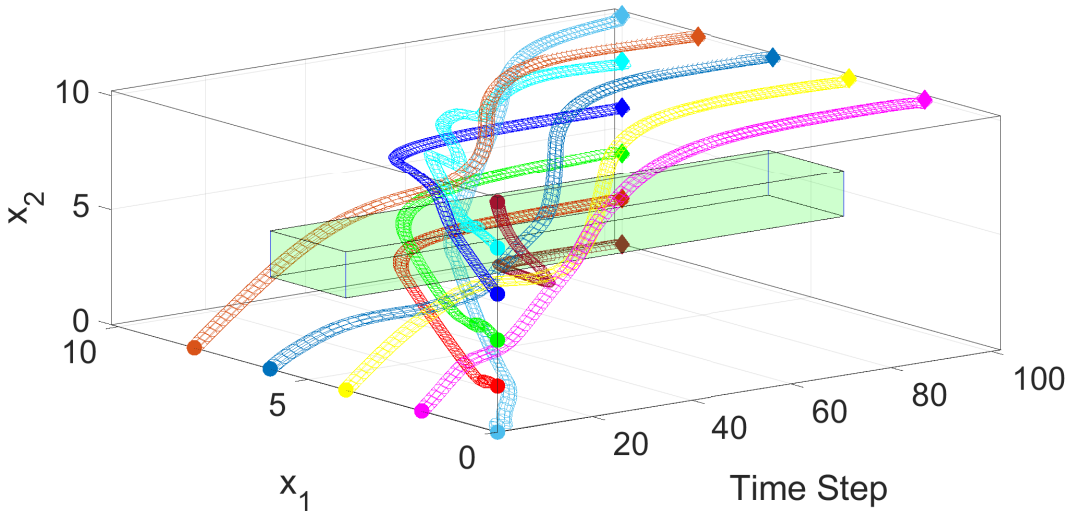

Figure 2.29.: Time-state space illustration of tubes enclosing nominal trajectories for the multi-drone path planning

external disturbances. Suppose the specification requires that each robot Σ^i eventually reaches $Goal^i$ while avoiding the obstacle $Obs \subseteq \mathbb{R}^2$ and collision with robots by the margin δ. The global specification on the product system $\Sigma^\times$ is as follows:

- The goal set $Goal \subseteq X^\times$ is defined as $Goal := Goal^1 \times \ldots \times Goal^N \subseteq X^\times$, and
- The avoid set $Avoid \subseteq X^\times$ is defined as

$$\{x^\times \in X^\times | \begin{array}{c} \exists i \in [1;N] \,.\, d_{Obs}(x^i, Obs) \leq \delta \\ \vee \\ \exists i,j \in [1;N] \,.\, i \neq j \,.\, d_{Col}\left(x^i, x^j\right) \leq \delta \end{array} \}, \tag{2.48}$$

  where x^i denotes the component of $x^\times$ corresponding to Σ^i, $d_{Col}(\cdot,\cdot)$ denotes a distance metric for measuring the geometric distance between positions of two systems located in two-dimensional space, and $d_{Obs}(\cdot,\cdot)$ denotes a distance metric for measuring the geometric distance between the position of one system and the obstacle $Obs \subseteq \mathbb{R}^2$.

In this category of problems, we apply our approach to three case studies as briefly discussed next. The detailed models for the systems and their different parameters have been presented in Appendix A.1, and the performance of GAMARA for these experiments have been summarized in the first three rows on Table 2.10.

Multi-drone path planning. We consider a planning scenario for ten identical drones ($N = 10$). The control objective is to synthesize a feedback controller for each drone so that in the presence of (bounded) disturbance, beginning from the specified initial state, the corresponding target state is reached within a finite horizon, while avoiding collision with other drones and the physical obstacle at every time point. Figure 2.29

gives a time-space illustration for the safe tubes around the nominal trajectories. The tracking feedback controllers are synthesized such that every drone remains within its safe tube until reaching its destination, even with worst-case disturbance. Additional analysis and detailed models for the systems and their different parameters are presented in Section A.1 in the appendix.

Crane and vehicle. The goal of this example is to study the performance of our method for controlling a number of robots with different dynamics. The goal is to move the overhead crane and the vehicle such that they do not collide. Formally guaranteed controllers are computed such that the generated open loop trajectory (see Figure 2.26 (**left**)) is tracked even under disturbance. More detailed discussion on dynamics of systems and further analysis are presented in Section A.1 in the appendix.

Lane merging. We study a lane merging problem wherein multiple controlled vehicles ($N = 6$) are driving over two merging lanes (Figure 2.30, **top** frame). A dangerous situation may occur at the merging point of the two lanes if vehicles are not controlled properly. Different variants of this problem have been studied in the literature (see, e.g., [190, 189]). Without seeking to optimize fuel consumption or travel time, we set the goal to control the vehicles to pass the merging zone safely. In particular, consider a situation where initially three cars are driving on each of the two lanes (Figure 2.30, (**top**)). The control objective for each vehicle is to pass the red dashed line within a finite horizon without hitting the road's sides or colliding with other vehicles. Figure 2.30 demonstrates snapshots of one sample trajectory when feedback controllers are employed under the presence of bounded disturbance. Additional analysis, systems' dynamics and parameters are reported in Section A.1 in the appendix.

Global Formation Control Problem

The second category of examples are about maintaining a global formation while satisfying a set of reach-avoid specifications. We show how the formation control problem can be expressed using a static obstacle *Avoid* on the product state space $X^{\times}$.

Let us first formalize the notion of *formation*. Let $\left\{ \Sigma^i = (X^i, x^i_{\mathsf{in}}, U^i, W^i, f^i) \right\}$ be a set of robots. A *formation constraint* is a set $\left\{ \lambda^{i,j} \in \mathbb{R} \right\}_{i,j \in [1;N]}$ where every $\lambda^{i,j}$ specifies the relative *Euclidean* distance between the projections of state of robot Σ^i and robot Σ^j.

Now suppose $Goal^i \subseteq X^i$ are the individual goal states, $Obs \subset \mathbb{R}^2$ is a common obstacle $\delta \in \mathbb{R}_{>0}$ is a safety margin, and $\mu \in \mathbb{R}_{>0}$ is a tolerance margin for the formation constraint. The formation control problem then asks to generate controllers $\{C_i\}$ such that every robot Σ^i eventually reaches $Goal^i$ while avoiding Obs by the margin δ, as well as while making sure that the *Euclidean* distance between robots Σ^i and Σ^j is in the range $\lambda^{i,j} \pm \mu$. Essentially the tolerance margin μ is to account for the possible slight deviations due to disturbances experienced by the robots. Notice that since the robots have their own goals but at the same time they need to "stay close" to their neighboring robots in the formation for the entire period, they might first need to accompany the other robots to their goals, before being accompanied by them to reach their own goal. We can express the formation control problem in the product state space as follows:

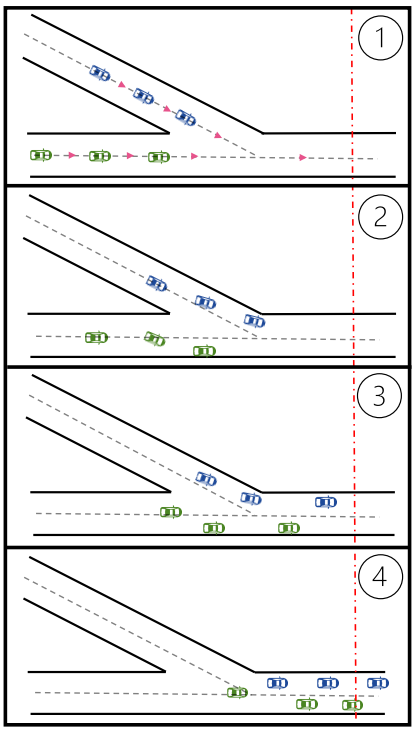

Figure 2.30.: Illustration of a sample trajectory generated by formally guaranteed controllers for the lane merging example

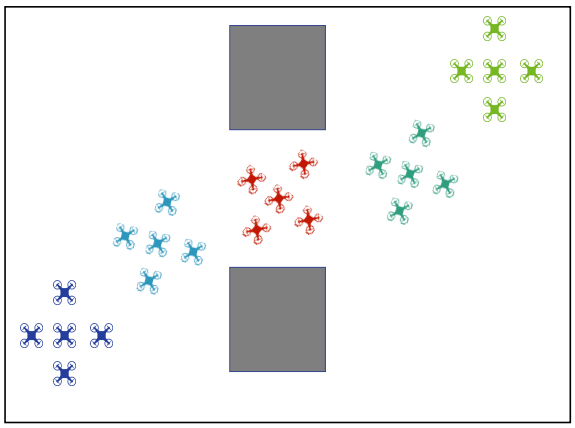

Figure 2.31.: Illustration of a sample trajectory generated by the feedback controllers for the formation control example

- The goal set $Goal \subseteq X^{\times}$ is defined as $Goal \coloneqq Goal^1 \times \ldots \times Goal^N$, and
- The avoid set $Avoid \subseteq X^{\times}$ is

$$\left\{ x^{\times} \in X^{\times} \;\middle|\; \begin{array}{c} \exists i \in [1;N] \,.\, D(x^i, Obs) \leq \delta \\ \vee \\ \exists i,j \in [1;N] \,.\, i \neq j \,.\, d\left(x^i, x^j\right) \notin \lambda^{i,j} \pm \mu \end{array} \right\}, \tag{2.49}$$

where the last disjunction in the definition of *Avoid* is the restriction required for maintaining the formation, and the rest are required in order to avoid hitting the obstacle.

Multi-drone formation control. Consider a formation control scenario where a set of five drones (identically modeled) need to go from a specified start point to a certain destination (both defined over the corresponding state spaces) within a finite horizon, while four of them forming a diamond around a fifth drone (positioned at the diamond's center) at every time point. There are two square obstacles from which the group needs to keep a certain minimum distance at all of the time points. Figure 2.31 illustrates four sequential frames of a sample perturbed trajectory generated by employing formally guaranteed feedback controllers. Notice that both relative position and orientation between drones are kept (almost) constant throughout the journey. Further analysis can be found in Section A.1 in the appendix.

2.5. Conclusion

In this chapter, we have explored the realm of non-linear dynamical systems with bounded disturbances, aiming to extend the scope of abstraction-based controller design for infinite-horizon temporal specifications. Within this context, we identified three primary limitations of the ABCD method: (1) its dependence on knowing the analytical dynamics, (2) its substantial memory requirements, and (3) its limited applicability in multi-agent scenarios.

In Section 2.2, we introduced a data-driven approach that learns an abstraction conforming to specified confidence levels, enabling the synthesis of controllers for infinite-horizon temporal specifications.

In Section 2.3, we proposed a memory-efficient method that employs neural representations for both the finite abstraction and the computed controller. Our extensive experiments demonstrated the successful reduction of memory requirements in comparison to the basic implementations of ABCD method.

Finally, in Section 2.4, we addressed scenarios involving a heterogeneous population of agents tasked with joint reach-avoid objectives, all while operating without inter-agent communication. Our method comprises a centralized planning phase for computing nominal temporal-spatial (open-loop) trajectories for all agents, followed by a decentralized tracking phase that employs the ABCD approach to synthesize feedback controllers ensuring the guaranteed tracking of the open-loop trajectories computed in the initial step. Empirical results underscored the effectiveness of our method in resolving diverse multi-agent challenges, including formation control, lane merging, and multi-agent reach-avoid tasks.

3

Continuous-Time MDPs with Reachability Specifications

Continuous-time Markov chains (CTMCs) and Markov decision processes (CTMDPs) play a central role in the modeling and analysis of performance and dependability properties of probabilistic systems evolving in real time. A CTMC combines probabilistic behavior with real time: it defines a transition system on a set of states, where the transition between two states is delayed according to an exponential distribution. Any state of the system may have multiple possible next states, each with an associated exponentially-distributed delay. The next state is chosen according to a race condition among these delays. A CTMDP extends a CTMC by introducing non-deterministic choice among a set of possible actions. It consists of a finite set of states, a finite set of actions, and for each action, a transition rate matrix that determines the rate (in an exponential distribution in continuous time) to go from one state to the next when the action is chosen. A *policy* for a CTMDP maps a timed execution path to state-dependent actions. Given a fixed policy, a CTMDP determines a stochastic process in continuous time, where the rate matrix determines the distribution of switches. Both CTMCs and CTMDPs have been used in a large variety of applications —from biology to finance.

The time-bounded reachability problem is at the core of model checking of CTMCs and CTMDPs with respect to stochastic temporal logics [15] and has been extensively studied [31, 133, 186, 134, 32]. The time-bounded reachability problem asks, given a CTMDP $\mathcal{M}$ with a designated "**good**" state, a time bound B, and a rational vector r, whether there exists a policy that controls the Markov decision process such that the probability of reaching the good state from state s within time bound B is at least $r(s)$. While the decidability of the time-bounded reachability problem remains an open question for CTMDPs, [9] demonstrated its decidability for CTMCs using tools from number theory. Furthermore, approximating the time-bounded reachability probability for both CTMCs and CTMDPs with large state spaces is computationally expensive. In this chapter, we focus on the time-bounded reachability of CTMCs and CTMDPs from both decidability and scalability perspectives.

3.1. Preliminaries

Definition 3.1.1 *A continuous-time Markov decision process (CTMDP) is a tuple* $\mathcal{M} = (S, \mathcal{D}, \mathbf{Q})$ *where*

- $S = \{1, 2, \ldots, \mathfrak{n}\}$ *is a finite set of states for some* $\mathfrak{n} > 0$*;*
- *a set* $\mathcal{D} = \prod_{s=1}^{\mathfrak{n}} \mathcal{D}_s$ *of* decision vectors*, where* $\mathcal{D}_s$ *is a finite set of* actions *that can be taken in state* $s \in S$*;*
- $\mathbf{Q}$ *is a* $\mathcal{D}$*-indexed family of* $\mathfrak{n} \times \mathfrak{n}$ generator matrices*; we write* $\mathbf{Q}^{\mathbf{d}}$ *for the generator matrix corresponding to the decision vector* $\mathbf{d} \in \mathcal{D}$*. The entry* $\mathbf{Q}^{\mathbf{d}}(s, s') \geq 0$ *for* $s' \neq s$ *gives the* rate of transition *from state* s *to state* s' *under action* $\mathbf{d}(s)$*, and* $\mathbf{Q}^{\mathbf{d}}(s, s')$ *is independent of elements of* $\mathbf{d}$ *except* $\mathbf{d}(s)$*. The entry* $\mathbf{Q}^{\mathbf{d}}(s, s) = -\sum_{s' \neq s} \mathbf{Q}^{\mathbf{d}}(s, s')$*.*

A CTMDP $\mathcal{M} = (S, \mathcal{D}, \mathbf{Q})$ with $|\mathcal{D}| = 1$, i.e., when only a unique action can be taken in each state, is called a *continuous-time Markov chain* (CTMC) and is simply denoted by the tuple $(S, \mathbf{Q})$, and with abuse of notation, we also write $\mathbf{Q}$ for the unique generator matrix. The CTMDP $\mathcal{M}$ reduces to a CTMC whenever a decision vector $\mathbf{d}$ is fixed for all time on the CTMDP.

Intuitively, $\mathbf{Q}^{\mathbf{d}}(s, s') > 0$ indicates that by fixing a decision vector $\mathbf{d}$, a transition from s to s' is possible and that the timing of the transition is exponentially distributed with rate $\mathbf{Q}^{\mathbf{d}}(s, s')$. If there are several states s' such that $\mathbf{Q}^{\mathbf{d}}(s, s') > 0$, more than one transition will be possible, and there will be a *race condition* among the exponentially distributed transition times associated with the potential successor states.

For each decision vector $\mathbf{d} \in \mathcal{D}$ and any $s \in S$, the total rate of taking an outgoing transition from state s when $\mathbf{d}$ is fixed is given by $E_{\mathbf{d}}(s) = \sum_{s' \neq s} \mathbf{Q}^{\mathbf{d}}(s, s')$, By fixing this decision vector $\mathbf{d}$, a transition from a state s into s' occurs within time t with probability

$$\mathbf{P}(s, s', t) = \frac{\mathbf{Q}^{\mathbf{d}}(s, s')}{E_{\mathbf{d}}(s)} . (1 - e^{-E_{\mathbf{d}}(s)t}), \quad t \geq 0.$$

Intuitively, $1 - e^{-E_{\mathbf{d}}(s)t}$ is the probability of taking an outgoing transition at s within time t (exponentially distributed with rate $E_{\mathbf{d}}(s)$) and $\mathbf{Q}^{\mathbf{d}}(s, s')/E_{\mathbf{d}}(s)$ is the probability of taking transition to s' among possible next states at s. Thus, the total probability of moving from s to s' under the decision $\mathbf{d}$ in one transition, written $\mathbf{P}_{\mathbf{d}}(s, s')$ is $\mathbf{Q}^{\mathbf{d}}(s, s')/E_{\mathbf{d}}(s)$. A state $s \in S$ is called *absorbing* if and only if $\mathbf{Q}^{\mathbf{d}}(s, s') = 0$ for all $s' \in S$ and all decision vectors $\mathbf{d} \in \mathcal{D}$. For an absorbing state, we have $E_{\mathbf{d}}(s) = 0$ for any decision vector $\mathbf{d}$ and no transitions are enabled. The initial state of a CTMDP is either fixed deterministically or selected randomly according to a probability distribution α over the set of states S.

Consider a time interval $[0, B]$ with time bound $B > 0$. Let Ω denote the set of all right-continuous step functions $f : [0, B] \to S$, i.e., there are time points $t_0 = 0 < t_1 < t_2 < \ldots < t_m = B$ such that $f(t') = f(t'')$ for all $t', t'' \in [t_i, t_{i+1})$ for all $i \in \{0, 1, \ldots, m-1\}$. Let $\mathcal{F}$ denote the sigma-algebra of the *cylinder sets*

$$\mathsf{Cyl}(s_0, I_0, \ldots, I_{m-1}, s_m) := \{f \in \Omega \,|\, \forall 0 \leq i \leq m \cdot f(t_i) = s_i \wedge i < m \Rightarrow (t_{i+1} - t_i) \in I_i\}. \tag{3.1}$$

for all m, $s_i \in S$ and non-empty time intervals $I_0, I_1, \ldots, I_{m-1} \subset [0, B]$.

Definition 3.1.2 *A* policy π *is a function from* $[0, B]$ *into* $\mathcal{D}$*, which is assumed to be Lebesgue measurable. Any policy gives a decision vector* $\pi_t \in \mathcal{D}$ *at time* t *such that the action* $\pi_t(s)$ *is taken when the CTMDP is at state* s *at time* t*. The set of all such polices is denoted by* Π_B*.*

Any policy π together with an initial distribution α induces the probability space $(\Omega, \mathcal{F}, \mathbf{P}^\pi_\alpha)$. If the initial distribution is chosen deterministically as $s \in S$, we denote the probability measure by $\mathbf{P}^\pi_s$ instead of $\mathbf{P}^\pi_\alpha$.

A policy $\pi : [0, B] \to \mathcal{D}$ is *piecewise constant* if there exist a number $m \in \mathbb{N}$ and time points $t_0 = 0 < t_1 < t_2 < \ldots < t_m = B$ such that $\pi_{t'} = \pi_{t''}$ for all $t', t'' \in (t_i, t_{i+1}]$ and all $i \in \{0, 1, \ldots, m-1\}$. The policy is *stationary* if $m = 1$. We denote the class of stationary policies by $\Pi_{\mathfrak{st}}$; observe that a stationary policy is given by a fixed decision vector, so $\Pi_{\mathfrak{st}}$ is isomorphic with the set of decision vectors $\mathcal{D}$. In particular, it is a finite set.

Remark 12 *The policies in Def. 3.1.2 are called* timed positional *policies since the action is selected deterministically as a function of time and the state of the CTMDP at that time. A stationary policy is only positional since the selected action is independent of time.*

3.2. Decidability of Time-Bounded Reachability for CTMDPs

As mentioned earlier, the decidability of the time-bounded reachability problem for CTMDPs is open. Existing papers either consider *time-abstract* policies [15, 145, 28, 186, 134] or focus on numerical approximation schemes [31, 133, 8, 61, 32, 158, 160]. However, policies that depend on time are strictly more powerful and the *decision problem* has remained open. For the special case of continuous-time Markov chains (CTMCs), where each state has a unique action, the time-bounded reachability problem is decidable [9]. The proof uses tools from transcendental number theory, specifically, the Lindemann-Weierstrass theorem. One might expect that a similar argument might be used to show decidability for CTMDPs as well.

In this section, we show *conditional* decidability. Our result uses, like several other conditional results on dynamical systems, Schanuel's conjecture from transcendental number theory (see, e.g., [101]). Our proof has the following ingredients. First, we use the fact that the optimal policy for the time-bounded reachability problem is a timed, *piecewise constant* function with a *finite* number of switches [127, 135, 144]. We show that each switch point of an optimal policy corresponds to a non-tangential zero of an associated linear dynamical system. Second, we use the result of Macintyre and Wilkie [113, 114] that Schanuel's conjecture implies the decidability of the real-closed field together with the exponential, sine, and cosine functions over a bounded domain. The existence of non-tangential zeros of linear dynamical systems can be encoded in this theory. Third, for each natural number $k \in \mathbb{N}$, we write a sentence in this theory whose

validity implies there is an optimal strategy with exactly k switch points. By enumerating over k, we find the exact number of switches in an optimal strategy. Finally, we write another sentence in the theory that checks if the reachability probability attained by (an encoding of) the optimal policy is greater than the given bound.

We also study the related decision problem whether there is a *stationary* (i.e., time independent) optimal policy. We show that there is a "direct" conditional decision procedure for this problem based on Schanuel's conjecture and recent results on zeros of exponential polynomials [38], which avoids the result of Macintyre and Wilkie.

At the same time, we show that an *unconditional* decidability result is likely to be very difficult. We show that the bounded continuous-time Skolem problem [21, 38] reduces to checking if there is an optimal stationary policy in the time-bounded CTMDP problem. The bounded continuous Skolem problem is a long-standing open problem about linear dynamical systems [38, 21]; it asks if a linear dynamical system in continuous time has a non-tangential zero in a bounded interval. Our reduction, in essence, demonstrates that CTMDPs can "simulate" any linear dynamical system: a non-tangential zero in the dynamics corresponds to a policy switch point in the simulating CTMDP.

Our result is in the same spirit as several recent results providing conditional decision procedures, based on Schanuel's conjecture, or hardness results, based on variants of the Skolem problem, for problems on probabilistic systems. For example, Daviaud et al. [46] showed conditional decidability of subcases of the containment problem for probabilistic automata subject to the conditional decidability of the theory of real closed fields with the exponential function [115, 185]. For lower bounds, Akshay et al. [4] showed a reduction from the (unbounded, discrete) Skolem problem to reachability on discrete time Markov chains and Piribauer and Baier [143] show that the positivity problem in discrete time can be reduced into several decision problems corresponding to optimization tasks over discrete time MDPs.

The content of this section is based on our paper [119]. In summary, we summarize the main result of this section as the following theorem.

Theorem 3.2.1 *(1) The time-bounded reachability problem for CTMDPs is decidable assuming Schanuel's conjecture. (2) Whether the time-bounded reachability problem has a stationary optimal policy is decidable assuming Schanuel's conjecture. (3) The bounded continuous Skolem problem reduces to checking if the time-bounded reachability problem has a stationary optimal policy.*

3.2.1. Problem Statement

Let us define the event

$$\mathit{Reach} := \cup\{f \in \Omega \mid f(t) = \mathbf{good} \text{ for some } t \in [0, B]\}. \tag{3.2}$$

The event *Reach* defined in (3.2) is written as a union of an uncountable number of functions but it is measurable in the probability space $(\Omega, \mathcal{F}, \mathbf{P}^{\pi}_{\alpha})$ for any α. Since the state space is finite, *Reach* can be written as a countable union of cylinder sets in the form of (3.1) by taking all the time intervals to be $[0, B]$ and enumerating over all possible

sequence of states (which is countable) [13]. Now, we are able to define the time-bounded reachability problem.

Problem 3.1 (Time-bounded reachability for CTMDPs) ***Inputs:*** *a CTMDP* $\mathcal{M} = (\{1, \ldots, n\} \uplus \{\mathbf{good}\}, \mathcal{D}, \mathbf{Q})$ *with a distinguished absorbing state named* **good**, *a time bound* $B > 0$ *and a vector* $r \in [0,1]^n$

Question: *Decide whether we have*

$$\sup_{\pi \in \Pi_B} \mathbf{P}_s^{\pi}(\mathit{Reach}) > r(s), \quad \text{for all } s \in \{1, \ldots, n\}.$$

A policy $\pi^* \in \Pi_B$ is *optimal* if $P_s^{\pi^*}(\mathit{Reach}) = \sup_{\pi \in \Pi_B} \mathbf{P}_s^{\pi}(\mathit{Reach})$. Note that there are more general classes of policies that may depend also on the history of the states in the previous time points and which map the history to a distribution over $\mathcal{D}$. It is shown that piecewise constant timed positional policies are sufficient for the optimal reachability probability [127, 135, 144]. That is, if there is an optimal policy from the larger class of policies, there is already one from the class of piecewise constant, timed, positional policies.

A closely related problem is the existence of *stationary* optimal policies; here, it is possible that the optimal stationary policy performs strictly worse than an optimal policy.

Problem 3.2 ***Inputs:*** *A CTMDP* $\mathcal{M} = (\{1, \ldots, n\} \uplus \{\mathbf{good}\}, \mathcal{D}, \mathbf{Q})$ *and a time bound* $B > 0$.
Question: *Decide whether there is an optimal policy* π^* *that is stationary, namely*

$$\exists \pi^* \in \Pi_{\mathfrak{st}} \text{ s.t. } \sup_{\pi \in \Pi_B} \mathbf{P}_s^{\pi}(\mathit{Reach}) = \mathbf{P}_s^{\pi^*}(\mathit{Reach}), \quad \text{for all } s \in \{1, \ldots, n\}.$$

3.2.2. Characterizing the Optimal Policy

In the following, we shall assume that the CTMDPs and all bounds in the above decision problems are given using rational numbers. That is, rates of transitions in each generator matrix is a rational number, and the time bound B is a rational number.

Theorem 3.2.2 ([31, 127]) *A policy* $\pi \in \Pi_B$ *is optimal if* $\mathbf{d}_t$, *the decision vector taken by* π *at time* $B - t$, *maximizes for almost all* $t \in [0, B]$

$$\max_{\mathbf{d}_t} (\mathbf{Q}^{\mathbf{d}_t} W_t^{\pi}) \text{ with } \frac{d}{dt} W_t^{\pi} = \mathbf{Q}^{\mathbf{d}_t} W_t^{\pi}, \tag{3.3}$$

with the initial condition $W_0^{\pi}(\mathbf{good}) = 1$ *and* $W_0^{\pi}(s) = 0$ *for all* $s \in \{1, 2, \ldots, n\}$. *There exists a piecewise constant policy* π *that maximizes the equations.*

The maximization in Equation (3.3) above is performed element-wise. Equation (3.3) should be solved forward in time to construct the policy π backward in time due to the definition $\mathbf{d}_t = \pi_{B-t}$. One can alternatively write down (3.3) directly backward in time based on π_t.

The proof of Theorem 3.2.2 is constructive [31, 127] and is based on the following sets for any vector W:

$$\begin{aligned}
&\mathcal{F}_1(W) = \{\mathbf{d} \in \mathcal{D} \,|\, \mathbf{d} \text{ maximizes } \mathbf{Q}^{\mathbf{d}} W\},\\
&\mathcal{F}_2(W) = \{\mathbf{d} \in \mathcal{F}_1(W) \,|\, \mathbf{d} \text{ maximizes } [\mathbf{Q}^{\mathbf{d}}]^2 W\},\\
&\cdots\\
&\mathcal{F}_j(W) = \{\mathbf{d} \in \mathcal{F}_{j-1}(W) \,|\, \mathbf{d} \text{ maximizes } [\mathbf{Q}^{\mathbf{d}}]^j W\}.
\end{aligned} \tag{3.4}$$

The sets $\mathcal{F}_j(W)$ form a sequence of decreasing sets such that $\mathcal{F}_1(W) \supseteq \mathcal{F}_2(W) \supseteq \ldots \supseteq \mathcal{F}_{n+2}(W) = \mathcal{F}_{n+k}(W)$ for all $k > 2$. An optimal piecewise constant policy is the one that satisfies the condition $\mathbf{d}_t \in \mathcal{F}_{n+2}(W_t^\pi)$ for all $t \in [0, B]$. Note that if $\mathcal{F}_j(W_t^\pi)$ has only one element for some j, $\mathcal{F}_k(W_t^\pi) = \mathcal{F}_j(W_t^\pi)$ for all $k \geq j$ and that element is the optimal decision vector. The next proposition shows that when $\mathcal{F}_{n+2}(W_t^\pi)$ has more than one element, we can pick any one (and in fact, switch between them arbitrarily).

Proposition 3.2.1 *Let π be an optimal policy satisfying Equation (3.3). Take any t^* such that $\mathcal{F}_{n+2}(W_{t^*}^\pi) \neq \lim_{t \uparrow t^*} F_{n+2}(W_t^\pi)$. If $\mathcal{F}_{n+2}(W_{t^*}^\pi) = \{\mathbf{d}^1, \mathbf{d}^2, \ldots, \mathbf{d}^p\}$ for some $p > 1$ and*

$$\Delta_i := \sup\{\delta > 0 \,|\, \mathbf{d}^i \in \mathcal{F}_{n+2}(W_t^\pi) \text{ for all } t \in [t^*, t^* + \delta)\}, \quad \forall i \in \{1, 2, \ldots, p\}$$

Then, $\Delta_1 = \Delta_2 = \cdots = \Delta_p$.
Suppose there are points δ_1, δ_2 such that $t^* \leq \delta_1 < \delta_2 < t^* + \Delta_1$ and for all $t \in [\delta_1, \delta_2)$, we have $\pi_{B-t} = \mathbf{d}$ for some $\mathbf{d} \in \mathcal{F}_{n+1}(W_{t^*}^\pi)$. If π' is a policy that agrees with π on $[0, \delta_1)$ but for all $t \in [\delta_1, \delta_2)$, we have $\pi'_{B-t} = \mathbf{d}'$ for some $\mathbf{d}' \in \mathcal{F}_{n+1}(W_{t^*}^\pi) \setminus \{\mathbf{d}\}$, then π' also satisfies Equation (3.3) for almost all $t \in [0, \delta_2)$.

Proof *Since $\mathcal{F}_{n+2}(W_{t^*}^\pi) = \mathcal{F}_{n+k}(W_{t^*}^\pi)$ for all $k > 2$, for any $\mathbf{d}^i$ and $\mathbf{d}^j$ belonging to the set $\mathcal{F}_{n+2}(W_{t^*}^\pi)$, we have $[\mathbf{Q}^{\mathbf{d}^i}]^l W_{t^*}^\pi = [\mathbf{Q}^{\mathbf{d}^j}]^l W_{t^*}^\pi$ for all $l \geq 0$. Pick $\delta > 0$ sufficiently small such that $\{\mathbf{d}^1, \mathbf{d}^2, \ldots, \mathbf{d}^p\} \subseteq \mathcal{F}_{n+2}(W_t^\pi)$ for all $t \in [t^*, t^* + \delta)$. If the policy π selects $\mathbf{d}^i$ for all $t \in [t^*, t^* + \delta)$, we can write*

$$W_t^\pi = e^{[\mathbf{Q}^{\mathbf{d}^i}](t - t^*)} W_{t^*}^\pi \text{ for } t \in [t^*, t^* + \delta),$$

where $e^\Gamma := \sum_{k=0}^\infty \frac{1}{k!} \Gamma^k$ denotes the exponential of a matrix Γ. Therefore, using the fact that $[\mathbf{Q}^{\mathbf{d}^i}]^l W_{t^*}^\pi = [\mathbf{Q}^{\mathbf{d}^j}]^l W_{t^*}^\pi$ for all $l \geq 0$ we have

$$e^{[\mathbf{Q}^{\mathbf{d}^i}](t - t^*)} W_{t^*}^\pi = e^{[\mathbf{Q}^{\mathbf{d}^j}](t - t^*)} W_{t^*}^\pi, \quad \forall\, t \geq t^*. \tag{3.5}$$

Similarly, we have

$$[\mathbf{Q}^{\mathbf{d}^i}]^l e^{[\mathbf{Q}^{\mathbf{d}^i}]\Delta} W_{t^*}^\pi = [\mathbf{Q}^{\mathbf{d}^j}]^l e^{[\mathbf{Q}^{\mathbf{d}^j}]\Delta} W_{t^*}^\pi, \quad \forall\, l \geq 0 \text{ and } \Delta \geq 0. \tag{3.6}$$

Now take any $i = \arg\min_j \Delta_j$, thus $\Delta_i \leq \Delta_j$ for all j. Also take $\mathbf{d}' \in \mathcal{F}_{n+2}(W_{t^* + \Delta_i}^\pi)$ and $\mathbf{d}' \neq \mathbf{d}^i$ (this is possible due to the definition of Δ_i). Denote by h the smallest integer for which $1 \leq h \leq n + 2$ and

$$[\mathbf{Q}^{\mathbf{d}'}]^h W_{t^* + \Delta_i}^\pi > [\mathbf{Q}^{\mathbf{d}^i}]^h W_{t^* + \Delta_i}^\pi \Rightarrow [\mathbf{Q}^{\mathbf{d}'}]^h e^{[\mathbf{Q}^{\mathbf{d}^i}]\Delta_i} W_{t^*}^\pi > [\mathbf{Q}^{\mathbf{d}^i}]^h e^{[\mathbf{Q}^{\mathbf{d}^i}]\Delta_i} W_{t^*}^\pi.$$

Combining the above expression with Equation (3.6)*, we get*

$$[\mathbf{Q}^{\mathbf{d}'}]^h e^{[\mathbf{Q}^{\mathbf{d}^i}]\Delta_i} W^\pi_{t^*} > [\mathbf{Q}^{\mathbf{d}^j}]^h e^{[\mathbf{Q}^{\mathbf{d}^j}]\Delta_i} W^\pi_{t^*} \;\Rightarrow\; [\mathbf{Q}^{\mathbf{d}'}]^h W^\pi_{t^*+\Delta_i} > [\mathbf{Q}^{\mathbf{d}^j}]^h W^\pi_{t^*+\Delta_i},$$

which implies that $\Delta_j \leq \Delta_i$ *for any* j*. The particular selection of* i *results in* $\Delta_j = \Delta_i$ *for all* i, j*. The second part of the proposition is obtained by setting* $\Delta = (\delta_2 - \delta_1)$ *in Equation* (3.6) *and using the definition of the exponential of a matrix.*

The above proposition highlights the fact that whenever $\mathcal{F}_{n+2}(W^\pi_t)$ contains more than one decision vector over a time interval, one can construct infinitely many optimal policies by arbitrarily switching between such decision vectors. In the rest of this section, we restrict our attention to optimal policies that take only mandatory switches: the optimal policy will take an element of $\mathcal{F}_{n+2}(W^\pi_t)$ as long as possible. This does not influence Problems 3.1 and 3.2.

The major challenge in the computation of the optimal policy, thus answering the reachability problem, is the computation of the largest time $t \in [0, B)$ such that $\mathcal{F}_{n+2}(W^\pi_t) \neq \mathcal{F}_{n+2}(W^\pi_{t^-})$, where $W^\pi_{t^-}$ denotes the value of $W^\pi_{t-\delta}$ with δ converging to zero from the right. Suppose a decision vector $\mathbf{d}_0 \in \mathcal{F}_{n+2}(W^\pi_0)$ is selected. The optimal policy will change at the following time point:

$$t'' := \sup\{t \,|\, \mathbf{d}_0 \in \mathcal{F}_{n+2}(W^\pi_{t'}) \text{ for all } t' \in [0, t)\}.$$

3.2.3. Conditional Decidability Results

Schanuel's Conjecture and its Implications

Our decidability results will assume Schanuel's Conjecture for the complex numbers, a unifying conjecture in transcendental number theory (see, e.g., [101]). Recall that a *transcendence basis* of a field extension L/K is a subset $S \subseteq L$ such that S is algebraically independent over K and L is algebraic over $K(S)$. The *transcendence degree* of L/K is the (unique) cardinality of some basis.

Conjecture 3.2.1 (Schanuel's Conjecture (SC)) *Let* $a_1, \ldots, a_n$ *be complex numbers that are linearly independent over rational numbers* $\mathbb{Q}$*. Then the field* $\mathbb{Q}(a_1, \ldots, a_n, e^{a_1}, \ldots, e^{a_n})$ *has transcendence degree at least* n *over* $\mathbb{Q}$*.*

An important consequence of Schanuel's conjecture is that the theory of reals $(\mathbb{R}, 0, 1, +, \cdot, \leq)$ remains decidable when extended with the exponential and trigonometric functions over bounded domains.[1]

Theorem 3.2.3 (Macintyre and Wilkie (see [113, 114])) *Assume* ***SC****. For any* $n \in \mathbb{N}$*, the theory* $\mathbb{R}_{\mathsf{MW}} := (\mathbb{R}, \exp \restriction [0, n], \sin \restriction [0, n], \cos \restriction [0, n])$ *is decidable.*

[1] We note that while the result is claimed in several papers [113, 114], a complete proof of this result has never been published. Thus, it would be nice to have a "direct" proof of our main theorem (Theorem 3.2.1) starting with Schanuel's conjecture. We do not know such a proof.

Our main result will show that Problems 3.1 and 3.2 can be decided based on Theorem 3.2.3. In fact, Problem 3.1 can be decided directly from Schanuel's conjecture and recent results on exponential polynomials [38].

Theorem 3.2.4 *Assume **SC**. Then Problems 3.1 and 3.2 are decidable.*

In contrast, solving the time-bounded reachability problem for *stationary* policies is decidable unconditionally. This is because fixing a stationary policy reduces the time-bounded reachability problem to one on CTMCs, and one can use the decision procedure from [9].

Non-tangential Zeros

Recall that the solution to a first-order linear ODE of dimension n:

$$\frac{d}{dt}X_t = AX_t, \quad z_t = CX_t$$

with real matrices A and C and real initial condition $X_0 \in \mathbb{R}^n$, can be written as $z_t = Ce^{At}X_0$ where e^Γ denotes the exponential of a square matrix Γ, and defined as the infinite sum $e^\Gamma := \sum_{k=0}^{\infty} \frac{1}{k!}\Gamma^k$ that is guaranteed to converge for any matrix Γ. The function can be expressed as an exponential polynomial $z_t = \sum_{j=1}^{k} P_t(j)e^{\lambda_j t}$, where $\lambda_1, \ldots, \lambda_k$ are the distinct (real or complex) eigenvalues of A. Each function $P_t(j)$ is a polynomial function of t possibly with complex coefficients and has a degree one less than the multiplicity of the eigenvalue λ_j. Since the eigenvalues come in conjugate pairs, we can write the real-valued function z as

$$z_t = \sum_{j=1}^{k} e^{a_j t} \sum_{l=0}^{m_j-1} c_{j,l} t^l \cos(b_j t + \varphi_{j,l}), \tag{3.7}$$

where the eigenvalues are $a_j \pm \mathbf{i}b_j$ with multiplicity m_j. If A, X_0, and C are over the rational numbers, then a_j, b_j, $c_{j,l}$ are real algebraic and $\varphi_{j,l}$ is such that $e^{\mathbf{i}\varphi_{j,l}}$ is algebraic for all j and l. We can symbolically compute derivatives of z which also become functions with a similar closed-form as in (3.7).

Definition 3.2.1 *The function z_t has a* zero *at $t = t^*$ if $z_{t^*} = 0$. The zero is said to be* non-tangential *if there is an $\varepsilon > 0$ such that $z_{t_1} z_{t_2} < 0$ for all $t_1 \in (t^* - \varepsilon, t^*)$ and all $t_2 \in (t^*, t^* + \varepsilon)$. The zero is called* tangential *if there is an $\varepsilon > 0$ such that $z_{t_1} z_{t_2} > 0$ for all $t_1 \in (t^* - \varepsilon, t^*)$ and all $t_2 \in (t^*, t^* + \varepsilon)$.*

Note that there are functions with zeros that are neither tangential nor non-tangential. Consider the function $z_t = t \sin\left(\frac{1}{t}\right)$ for $t \neq 0$ and $z_0 = 0$. The function does not satisfy the conditions of being tangential or non-tangential. For any $\varepsilon > 0$, there are $t_1 \in (-\varepsilon, 0)$ and $t_2 \in (0, \varepsilon)$, such that $z_{t_1} z_{t_2} = t_1 t_2 \sin\left(\frac{1}{t_1}\right) \sin\left(\frac{1}{t_2}\right)$ is positive. There are also t_1 and t_2 in the respective intervals that make $z_{t_1} z_{t_2}$ negative. We only work with functions of the form (3.7) that are analytic thus infinitely differentiable. Therefore, the first non-zero derivative of z_t at t^* will decide if t^* is tangential or not.

Proposition 3.2.2 *For any function z_t of the form (3.7) such that $z_{t^*} = 0$ and $z \not\equiv 0$, there is a k_0 such that $\frac{d^k}{dt^k} z_t\big|_{t=t^*} = 0$ for all $k < k_0$ and $\frac{d^{k_0}}{dt^{k_0}} z_t\big|_{t=t^*} \neq 0$. Moreover, t^* is tangential if k_0 is an even number and is non-tangential if k_0 is an odd number.*

Proof *The proof is based on the Taylor series of z_t at $t = t^*$. Take k_0 the order of the first non-zero derivative of z_t at $t = t^*$. This k_0 always exists since otherwise $z \equiv 0$. The Taylor series of z_t will be*

$$z_t = \sum_{k=k_0}^{\infty} \frac{(t-t^*)^k}{k!} \frac{d^k}{dt^k} z_t\big|_{t=t^*} = (t-t^*)^{k_0} \frac{d^{k_0}}{dt^{k_0}} z_t\big|_{t=t^*} \sum_{k=0}^{\infty} \alpha_k (t-t^*)^k, \tag{3.8}$$

for some $\{\alpha_0, \alpha_1, \ldots\}$ with $\alpha_0 = \frac{1}{k_0!}$. Define the function g by $g_t := \frac{z_t}{(t-t^*)^{k_0}}$ for $t \neq t^*$ and $g_{t^*} := \frac{1}{k_0!} \frac{d^{k_0}}{dt^{k_0}} z_t\big|_{t=t^*}$. Using (3.8), we get that g is continuous at t^* with $g_{t^*} \neq 0$. Therefore, there is an interval $(t^* - \varepsilon, t^* + \varepsilon)$ over which the function has the same sign as g_{t^*}. For all $t_1 \in (t^* - \varepsilon, t^*)$ and $t_2 \in (t^*, t^* + \varepsilon)$

$$\begin{aligned}
&g_{t_1} g_{t^*} > 0 \Rightarrow \frac{z_{t_1}}{(t_1 - t^*)^{k_0}} g_{t^*} > 0 \Rightarrow (-1)^{k_0} z_{t_1} g_{t^*} > 0 \\
&g_{t_2} g_{t^*} > 0 \Rightarrow \frac{z_{t_2}}{(t_2 - t^*)^{k_0}} g_{t^*} > 0 \Rightarrow z_{t_2} g_{t^*} > 0 \\
&\Rightarrow (-1)^{k_0} z_{t_1} g_{t^*} z_{t_2} g_{t^*} > 0 \Rightarrow (-1)^{k_0} z_{t_1} z_{t_2} > 0.
\end{aligned}$$

This means $z_{t_1} z_{t_2} > 0$ for even k_0 and t^* becomes tangential, and $z_{t_1} z_{t_2} < 0$ for odd k_0 and t^* becomes non-tangential.

For any function $z_t = Ce^{At}X_0$, the predicate $\mathsf{NonTangentialZero}(z, l, u)$ stating the existence of a non-tangential zero in an interval (l, u) is expressible in $\mathbb{R}_{\mathsf{MW}}$:

$$\exists t^* \,.\, l < t^* < u \wedge z_{t^*} = 0 \wedge [\exists \varepsilon > 0 \,.\, \forall t_1 \in (t^* - \varepsilon, 0), t_2 \in (0, t^* + \varepsilon) \,.\, z_{t_1} z_{t_2} < 0]$$

Switch Points are Non-Tangential Zeroes

Given a CTMDP $\mathcal{M}$ and a piecewise constant optimal policy $\pi : [0, B] \to \mathcal{D}$ for the time-bounded reachability problem, a *switch point* t^* is a point of discontinuity of π. Consider a switch point t^* such that the optimal policy takes the decision vector $\mathbf{d}$ in the time interval $(t^* - \varepsilon)$ and then switches to another decision vector $\mathbf{d}'$ at time t^* for some $\varepsilon > 0$:

$$\begin{aligned}
&\mathbf{d} \in \mathcal{F}_{n+2}(W_t^\pi) \text{ and } \mathbf{d}' \notin \mathcal{F}_{n+2}(W_t^\pi) \quad \forall t \in (t^* - \varepsilon, t^*), \\
&\mathbf{d} \notin \mathcal{F}_{n+2}(W_t^\pi) \text{ and } \mathbf{d}' \in \mathcal{F}_{n+2}(W_t^\pi) \quad \forall t \in (t^*, t^* + \varepsilon).
\end{aligned}$$

Consider a (not necessarily unique) state $s \in S$ with actions $a, b \in \mathcal{D}_s$ such that $a \neq b$ and $\mathbf{d}(s) = a$, $\mathbf{d}'(s) = b$. Define the following set of first-order ODEs

$$\Sigma : \begin{cases} \frac{d}{dt} W_t^\pi = \mathbf{Q}^{\mathbf{d}} W_t^\pi \\ z_t = (q^a - q^b) W_t^\pi \end{cases} \tag{3.9}$$

for $t \in (t^* - \varepsilon, t^* + \varepsilon)$, where q^a and q^b denote the s^{th} row of the matrices $\mathbf{Q}^{\mathbf{d}}$ and $\mathbf{Q}^{\mathbf{d}'}$, respectively. The optimal decision vector on an interval before t^* is $\mathbf{d}$, thus for all $t \in (t^* - \varepsilon, t^*)$,

$$\mathbf{d} \in \mathcal{F}_1(W_t^\pi) \Rightarrow \mathbf{Q}^{\mathbf{d}} W_t^\pi \geq \mathbf{Q}^{\mathbf{d}'} W_t^\pi \Rightarrow (\mathbf{Q}^{\mathbf{d}} - \mathbf{Q}^{\mathbf{d}'}) W_t^\pi \geq 0 \Rightarrow (q^a - q^b) W_t^\pi \geq 0 \Rightarrow z_t \geq 0.$$

The next lemma states that the switch point t^* corresponds to a non-tangential zero for z_t.

Lemma 3.2.1 *Let π be an optimal piecewise constant policy for the time-bounded reachability problem with bound B. Suppose $\pi(B - t) = \mathbf{d}_t$ for all $t \in [0, B]$. Suppose that for a time point t^*, $\mathbf{d} \in \mathcal{D}$ is an optimal decision before t^* and $\mathbf{d}' \neq \mathbf{d}$ is optimal right after t^*. There is an ε such that for any $s \in S$ with $\mathbf{d}(s) \neq \mathbf{d}'(s)$, $z_t < 0$ for all $t \in (t^*, t^* + \varepsilon)$ with z_t defined in (3.9).*

Proof *Take k_0 to be the smallest index $k \leq n$ with $\mathbf{d} \notin \mathcal{F}_{k+1}(W_{t^*}^\pi)$ and $\mathbf{d}' \in \mathcal{F}_{k+1}(W_{t^*}^\pi)$. Since $\mathbf{d}'$ is optimal at t^*, we have $\mathbf{d}, \mathbf{d}' \in \mathcal{F}_{k+1}(W_{t^*}^\pi)$ for all $k < k_0$. We show inductively that*

$$[\mathbf{Q}^{\mathbf{d}}]^{k+1} W_{t^*}^\pi = [\mathbf{Q}^{\mathbf{d}'}]^{k+1} W_{t^*}^\pi \text{ and } \frac{d^k}{dt^k} z_{t^*} = 0 \text{ for all } 0 \leq k < k_0. \tag{3.10}$$

The claim is true for $k = 0$:

$$\begin{aligned} &\mathbf{d}, \mathbf{d}' \in \mathcal{F}_1(W_{t^*}^\pi) \Rightarrow \mathbf{Q}^{\mathbf{d}} W_{t^*}^\pi = \mathbf{Q}^{\mathbf{d}'} W_{t^*}^\pi \\ &\Rightarrow (\mathbf{Q}^{\mathbf{d}} - \mathbf{Q}^{\mathbf{d}'}) W_{t^*}^\pi = \begin{bmatrix} \cdots \\ q^a - q^b \\ \cdots \end{bmatrix} W_{t^*}^\pi = 0 \Rightarrow (q^a - q^b) W_{t^*}^\pi = 0 \Rightarrow z_{t^*} = 0. \end{aligned}$$

Now suppose (3.10) holds for $(k-1)$ with $k < k_0$. Then

$$\begin{aligned} &\mathbf{d}, \mathbf{d}' \in \mathcal{F}_{k+1}(W_{t^*}^\pi) \Rightarrow [\mathbf{Q}^{\mathbf{d}}]^{k+1} W_{t^*}^\pi = [\mathbf{Q}^{\mathbf{d}'}]^{k+1} W_{t^*}^\pi \\ &\Rightarrow \mathbf{Q}^{\mathbf{d}} [\mathbf{Q}^{\mathbf{d}}]^k W_{t^*}^\pi = \mathbf{Q}^{\mathbf{d}'} [\mathbf{Q}^{\mathbf{d}'}]^k W_{t^*}^\pi \Rightarrow^{(*)} \mathbf{Q}^{\mathbf{d}} [\mathbf{Q}^{\mathbf{d}}]^k W_{t^*}^\pi = \mathbf{Q}^{\mathbf{d}'} [\mathbf{Q}^{\mathbf{d}}]^k W_{t^*}^\pi \\ &\Rightarrow [\mathbf{Q}^{\mathbf{d}} - \mathbf{Q}^{\mathbf{d}'}] [\mathbf{Q}^{\mathbf{d}}]^k W_{t^*}^\pi = 0 \Rightarrow^{(**)} [\mathbf{Q}^{\mathbf{d}} - \mathbf{Q}^{\mathbf{d}'}] \frac{d^k}{dt^k} X_{t^*} = 0 \\ &\Rightarrow (q^a - q^b) \frac{d^k}{dt^k} X_{t^*} = 0 \Rightarrow \frac{d^k}{dt^k} z_{t^*} = 0, \end{aligned}$$

where $(*)$ holds due to the induction assumption and $(**)$ is true due to the differential equation (3.9). Finally, we show that $\frac{d^{k_0}}{dt^{k_0}} z_{t^*} < 0$.

$$\begin{aligned} &\mathbf{d} \notin \mathcal{F}_{k_0+1}(W_{t^*}^\pi) \text{ and } \mathbf{d}' \in \mathcal{F}_{k_0+1}(W_{t^*}^\pi) \Rightarrow [\mathbf{Q}^{\mathbf{d}}]^{k_0+1} W_{t^*}^\pi < [\mathbf{Q}^{\mathbf{d}'}]^{k_0+1} W_{t^*}^\pi \\ &\Rightarrow \mathbf{Q}^{\mathbf{d}} [\mathbf{Q}^{\mathbf{d}}]^{k_0} W_{t^*}^\pi < \mathbf{Q}^{\mathbf{d}'} [\mathbf{Q}^{\mathbf{d}'}]^{k_0} W_{t^*}^\pi \Rightarrow^{(\imath)} \mathbf{Q}^{\mathbf{d}} [\mathbf{Q}^{\mathbf{d}}]^{k_0} W_{t^*}^\pi < \mathbf{Q}^{\mathbf{d}'} [\mathbf{Q}^{\mathbf{d}}]^{k_0} W_{t^*}^\pi \\ &\Rightarrow [\mathbf{Q}^{\mathbf{d}} - \mathbf{Q}^{\mathbf{d}'}] \frac{d^{k_0}}{dt^{k_0}} W_{t^*}^\pi < 0 \Rightarrow (q^a - q^b) \frac{d^{k_0}}{dt^{k_0}} W_{t^*}^\pi < 0 \Rightarrow \frac{d^{k_0}}{dt^{k_0}} z_{t^*} < 0, \end{aligned}$$

where $(\imath)$ holds due to (3.10) for $k_0 - 1$.

Since $z_{t^*} = 0$, we can select ε such that $z_t > 0$ for all $t \in (t^* - \varepsilon, t^*)$. Using Taylor expansion (3.8) *and the facts that $\frac{d^{k_0}}{dt^{k_0}} z_{t^*} < 0$ and $z_t > 0$ for $t \in (t^* - \varepsilon, t^*)$, we have that k_0 must be an odd number, which means t^* is non-tangential by Prop. 3.2.2. The function z_t changes sign from positive to negative at t^*.*

Conditional Decidability

The decision procedure for Problem 3.1 is as follows. Fix a CTMDP $\mathcal{M} = (\{1, \ldots, n\} \uplus \{\mathbf{good}\}, \mathcal{D}, \mathbf{Q})$ and a bound B. We inductively construct a piecewise constant optimal policy, going forward in time. To begin, we set the initial decision vector to $\mathbf{d}^1$, where $\mathbf{d}^1$ is selected such that $\mathbf{d}^1 \in \mathcal{F}_{n+2}(W_0^\pi)$ (Equation (3.4)) with W_0^π set to the indicator vector that is 1 at the **good** state and 0 in other states.

Note that in general $\mathcal{F}_{n+2}(W_t^\pi)$ in (3.4) may have finitely many elements and the choice of optimal decision at time t, $\mathbf{d}_t \in \mathcal{F}_{n+2}(W_t^\pi)$ is not unique. Based on results of Proposition 3.2.1, any arbitrary element of $\mathcal{F}_{n+2}(W_t^\pi)$ can be chosen; but, we do not alter this choice until the picked decision vector does not belong to $\mathcal{F}_{n+2}(W_t^\pi)$ anymore. We know that there is a piecewise constant optimal policy π with finitely many switches obtained from the charactrization in Theorem 3.2.2. Denote the (unknown) number of switches by $k \in \mathbb{N}$.

We find k as follows. We inductively check the existence of a sequence of decision vectors $\mathbf{d}^1, \ldots, \mathbf{d}^k$ and time points $t_1, \ldots, t_{k-1}$ such that the optimal policy (given a lexicographical order on $\mathcal{D}$) switches from $\mathbf{d}^i$ to $\mathbf{d}^{i+1}$ at time t_i but does not have any switch between the time points. Then, we check if the optimal policy makes at least one additional switch point in the interval (t_k, B). The check reduces the question to a number of satisfiability questions in $\mathbb{R}_{\mathsf{MW}}$. If we find an additional switch, we know that the optimal strategy has at least $k+1$ switches and continue to check if there are further switch points. If not, we know that the optimal policy has k switch points.

We need some notation. A *prefix* $\sigma_k = (\mathbf{d}^1, t_1, \mathbf{d}^2, t_2, \ldots, t_{k-1}, \mathbf{d}^k) \in (\mathcal{D} \times (0, B))^* \times \mathcal{D}$ is a finite sequence of decision vectors from $\mathcal{D}$ and strictly increasing time points $0 < t_1 < t_2 < \ldots < t_{k-1} < B$ such that $\mathbf{d}^i \neq \mathbf{d}^{i+1}$ for $i \in \{1, \ldots, k-1\}$. Intuitively, it represents the prefix of a piecewise constant policy with the first $k-1$ switches. For two decision vectors $\mathbf{d}, \mathbf{d}'$, let $\Delta(\mathbf{d}, \mathbf{d}') := \{s \mid \mathbf{d}(s) \neq \mathbf{d}'(s)\}$ be the states at which the actions suggested by the decision vectors differ. For a decision vector $\mathbf{d}$, let $\mathbf{d}[s \mapsto b]$ denote the decision vector that maps state s to action b but agrees with $\mathbf{d}$ otherwise.

For a prefix $\sigma_k = (\mathbf{d}^1, t_1, \mathbf{d}^2, t_2, \ldots, t_{k-1}, \mathbf{d}^k)$, a state $s \in S$, and an action $b \in \mathcal{D}_s$, define

$$y_t^{s,b}(\sigma_k) = \mathbf{u}^\top(s)([\mathbf{Q}^{\mathbf{d}^k}] - [\mathbf{Q}^{\mathbf{d}^k[s \mapsto b]}]) e^{[\mathbf{Q}^{\mathbf{d}^k}](t - t_{k-1})} e^{[\mathbf{Q}^{\mathbf{d}^{k-1}}](t_{k-1} - t_{k-2})} \cdots e^{[\mathbf{Q}^{\mathbf{d}^1}] t_1} \mathbf{u}(\mathbf{good}),$$

where $\mathbf{u}(s)$ is a vector of dimension $n+1$ that assigns one to s and zero to every other entry. Observe that $y_t^{s,b}(\sigma_k)$ is a solution of a set of linear ODEs similar to z_t in Equation (3.9):

$$\begin{cases} \frac{d}{dt} W_t = [\mathbf{Q}^{\mathbf{d}^k}] W_t \\ y_t^{s,b}(\sigma_k) = \mathbf{u}^\top(s)([\mathbf{Q}^{\mathbf{d}^k}] - [\mathbf{Q}^{\mathbf{d}^k[s \mapsto b]}]) W_t, \end{cases} \tag{3.11}$$

with the condition $W_{t_{k-1}} = e^{[\mathbf{Q}^{\mathbf{d}^{k-1}}](t_{k-1}-t_{k-2})} \dots e^{[\mathbf{Q}^{\mathbf{d}^1}]t_1}\mathbf{u}(\mathbf{good})$.

We shall use (variants of) the predicate $\mathsf{NonTangentialZero}(y^{\cdot,\cdot}, t_1, t_2)$, but write the predicates informally for readability. We need two additional predicates $\mathsf{Switch}(\sigma_k, t^*, \mathbf{d}')$ and $\mathsf{NoSwitch}(\sigma_{k+1})$. The predicate Switch states that, given a prefix σ_k, the first switch from $\mathbf{d}^k$ to a new decision vector $\mathbf{d}'$ occurs at time point $t^* > t_{k-1}$. This new switch requires three conditions. First, there is a simultaneous non-tangential zero at t^* for all dynamical systems of the form (3.11) associated with $y_t^{s,\mathbf{d}'(s)}(\sigma_k)$, $s \in \Delta(\mathbf{d}^k, \mathbf{d}')$. Second, t^* is the first time after t_{k-1} that any of the dynamical systems have a non-tangential zero. Finally, none of the states in $S \setminus \Delta(\mathbf{d}^k, \mathbf{d}')$ whose action remains the same before and after the switch, have a non-tangential zero in $(t_{k-1}, t^*]$ (up to and including t^*):

$$
\begin{aligned}
&\mathsf{Switch}(\underbrace{(\mathbf{d}^1, t_1, \dots, t_{k-1}, \mathbf{d}^k)}_{\sigma_k}, t^*, \mathbf{d}') \equiv \\
&\quad 0 < t_1 < \dots < t_{k-1} < B \wedge (B > t^* > t_{k-1}) \wedge (\Delta(\mathbf{d}^k, \mathbf{d}') \neq \emptyset) \wedge \\
&\quad \bigwedge_{s \in \Delta(\mathbf{d}^k, \mathbf{d}')} \left(\begin{array}{c} \text{“}y_t^{s,\mathbf{d}'(s)}(\sigma_k) \text{ has a non-tangential zero at } t^*\text{”} \wedge \\ \text{“}y_t^{s,\mathbf{d}'(s)}(\sigma_k) \text{ has no non-tangential zero in } (t_{k-1}, t^*)\text{”} \end{array} \right) \wedge \\
&\quad \bigwedge_{s \in S \setminus \Delta(\mathbf{d}^k, \mathbf{d}')} \text{“}y_t^{s,\mathbf{d}'(s)}(\sigma_k) \text{ has no non-tangential zero in } (t_{k-1}, t^*]\text{”}
\end{aligned}
$$

The predicate $\mathsf{NoSwitch}(\sigma_{k+1})$ states that, given a prefix σ_{k+1}, the last decision vector $\mathbf{d}^{k+1}$ of (σ_{k+1}) stays optimal and does not switch to another decision vector within the interval (t_k, B). This is equivalent to stating that none of the dynamical systems of the from (3.11) associated with $y_t^{s,b}(\sigma_{k+1})$ for $s \in S, b \in \mathcal{D}_s \setminus \mathbf{d}^{k+1}(s)$ has a non-tangential zero in (t_k, B):

$$
\mathsf{NoSwitch}(\sigma_{k+1}) \equiv \bigwedge_{s, b \neq \mathbf{d}^{k+1}(s)} \text{“}y_t^{s,b}(\sigma_{k+1}) \text{ has no non-tangential zero in } (t_k, B)\text{”}
$$

We can now check if the optimal strategy has exactly k switches. The first part of the predicate written below sets up a proper σ and the last conjunct states that there is no further switch after the last one.

$$
\exists t_1, \dots, t_k . (0 < t_1 < t_2 \dots < t_k < B) \wedge \bigwedge_{i=1}^{k} \mathsf{Switch}(\underbrace{\mathbf{d}^1, t_1, \dots, \mathbf{d}^i, t_i, \mathbf{d}^{i+1}}_{\sigma_{i+1}}) \wedge \mathsf{NoSwitch}(\sigma_{k+1}).
$$

We can enumerate these formulas with increasing k over all choices of decision vectors and stop when the above formula is valid. At this point, we know that there is a piecewise constant optimal policy with k switches, which plays the decision vectors $\mathbf{d}^1, \dots, \mathbf{d}^k$. We can make one more query to check if the probability of reaching **good** when playing this

strategy is at least a given rational vector $r \in [0,1]^n$:

$$\exists t_1, \ldots, t_k.(0 < t_1 < \ldots, t_k < B) \wedge \bigwedge_{i=1}^{k} \mathsf{Switch}(\mathbf{d}^1, t_1, \ldots, \mathbf{d}^i, t_i, \mathbf{d}^{i+1}) \wedge \mathsf{NoSwitch}(\sigma_{k+1}) \\ \wedge \bigwedge_{s=1}^{n} \mathbf{u}^\top(s) e^{[\mathbf{Q}^{\mathbf{d}^{k+1}}](B-t_k)} e^{[\mathbf{Q}^{\mathbf{d}^k}](t_k - t_{k-1})} \cdots e^{[\mathbf{Q}^{\mathbf{d}^1}]t_1} \mathbf{u}(\mathbf{good}) > r(s) \tag{3.12}$$

This completes the proof of conditional decidability of Problem 3.1.

Conditional Decidability for Problem 3.2

A stationary policy $\mathbf{d}$ is not optimal if there is a switch point. Using the Switch predicate and conditional decidability of $\mathbb{R}_{\mathsf{MW}}$, this shows conditional decidability of Problem 3.2.

In fact, to check the presence of a single non-tangential zero, one can avoid Theorem 3.2.3 and get a direct construction based on Schanuel's conjecture. This construction is similar to [38] and is provided in Section A.2. Unfortunately, when there are multiple switch points, we have to existentially quantify over previous switch points. Thus, the techniques of [38] cannot be straightforwardly extended to find a direct conditional decision procedure for Problem 3.1.

We do not know if there is a numerical procedure that only uses an oracle for non-tangential zeros. The problem is that, while numerical techniques can be used to bound each non-tangential zero with rational intervals with arbitrary precision as well as compute the reachability probability to arbitrary precision, we do not know how to numerically detect whether the reachability probability in (3.12) is exactly equal to a given r. By the Lindemann-Weierstrass Theorem [102], we already know that for CTMDPs with stationary optimal strategies, the value of reachability probability for any rational time bound $B > 0$ is transcendental and hence $\sup_{\pi \in \Pi_B} \mathbf{P}_s^\pi(\mathit{Reach}) \neq r(s)$ for all $s \in S$. However, we cannot prove that the reachability probability remains irrational in the general case.

3.2.4. Lower Bound: Continuous Skolem Problem

Problem 3.3 (Bounded Continuous-Time Skolem Problem) *Given a linear ordinary differential equation (ODE)*

$$\frac{d^n}{dt^n} z_t + a_{n-1} \frac{d^{n-1}}{dt^{n-1}} z_t + \cdots + a_1 \frac{d}{dt} z_t + a_0 z_t = 0 \tag{3.13}$$

with rational initial conditions $z_0, \frac{dz_t}{dt}|_{t=0}, \ldots, \frac{d^{n-1}z_t}{dt^{n-1}}|_{t=0} \in \mathbb{Q}$ *and rational coefficients* $a_{n-1}, a_{n-2}, \ldots, a_0 \in \mathbb{Q}$ *and a time bound* $B \in \mathbb{Q}$*, the* bounded continuous Skolem problem

asks whether there exists $0 < t^* < B$ *such that it is a non-tangential zero for* z_t*. Further, we can assume w.l.o.g. that* $z_0 = 0$ *in the initial condition.*[1]

We note that our definition is slightly different from the usual definition of the problem, e.g., in [21, 38], which simply asks for any zero (i.e., $z_{t^*} = 0$), not necessarily a non-tangential one. Our version of the bounded continuous Skolem problem is also decidable assuming **SC** [38]. However, there is no unconditional decidability result known for this problem, even though we only look for a non-tangential zero.

We can encode any given linear ODE of order n in the form of (3.13) into a set of n first-order linear ODE on $X : [0, B] \to \mathbb{R}^n$ with

$$\begin{cases} \frac{d}{dt}X_t = AX_t, \quad X_0 = \left[z_0, \frac{dz_t}{dt}\Big|_{t=0}, \ldots, \frac{d^{n-1}z_t}{dt^{n-1}}\Big|_{t=0}\right]^\top \\ z_t = CX_t, \end{cases} \tag{3.14}$$

with the state matrix A and output matrix C are

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}. \tag{3.15}$$

Using the representation (3.14), the solution of the linear ODE (3.13) can be written as $z_t = Ce^{At}X_0$. Therefore, the bounded continuous-time Skolem problem can be restated as whether the expression $Ce^{At}X_0$ has a non-tangential zero in the interval $(0, B)$.

We now reduce the bounded continuous-time Skolem problem to Problem 3.2. Given an instance (3.14)-(3.15) of the Skolem problem of dimension n, we shall construct a CTMDP over states $\{1, \ldots, 2n\} \cup \{\mathbf{good}, \mathbf{bad}\}$ and bound B, and just two decision vectors $\mathbf{d}^a$ and $\mathbf{d}^b$ that only differ in the available actions (a or b) at state 1. Our reduction will ensure that the answer of the Skolem problem has a non-tangential zero iff there is a switch in the optimal policy in the time-bounded reachability problem for bound B, and thus, iff stationary policies are not optimal.

Theorem 3.2.5 *For every instance of the bounded continuous-time Skolem problem with dynamics* $\frac{d}{dt}X_t = AX_t$*,* $z_t = CX_t$*, initial condition* X_0*, and time bound* B*, there is a CTMDP* $\mathcal{M}$ *such that the dynamical system has a non-tangential zero in* $(0, B)$ *iff the optimal strategy of the CTMDP in the time-bounded reachability problem is not stationary.*

We sketch the main ideas of the proof here. Consider the linear differential equation described by the state space representation in (3.14) with the initial condition X_0 that has its first element equal to zero $X_0(1) = 0$. Given the time bound $B > 0$, to solve the

[1] The assumption is w.l.o.g. because given a linear ODE whose solution is z_t, one can construct another linear ODE whose solution is $y_t = tz_t$. Clearly, $y_0 = 0$ and there is a non-tangential zero of z in $(0, B)$ iff there is a non-tangential zero of y in $(0, B)$.

bounded continuous Skolem problem, we are looking for the existence of a time $0 < t^* < B$ such that $z_{t^*} = 0$ is non-tangential. Equivalently, we want to find a non-tangential zero for the function $Ce^{At}X_0$, where $C = \begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix}$.

There are three obstacles to go from (3.14) to generator matrices for a CTMDP. Each generator matrix must have non-diagonal entries that are non-negative. The sum of each row of the matrix must be zero. Moreover, the last state of the CTMDP must be absorbing. None of these properties may hold for a general A. We show a series of transformations that take the matrix A to a matrix P that is sub-stochastic. Then we construct the generator matrices of the CTMDP using P that include the required absorbing state. We denote by $\mathbf{0}_{\mathsf{m}}$ and $\mathbf{1}_{\mathsf{m}}$ as row vectors of dimension m with all elements equal to zero and one, respectively.

Theorem 3.2.6 *Suppose $A \in \mathbb{Q}^{n\times n}$, $X_0 \in \mathbb{Q}^n$ and $C = [1, \mathbf{0}_{n-1}]$ are given with $X_0(1) = 0$. There are positive constants γ, λ and a* generator *matrix $P \in \mathbb{Q}^{(2n+1)\times(2n+1)}$ such that*

$$Ce^{At}X_0 = \gamma e^{\lambda t}\left[C'e^{Pt}Y_0\right], \quad C' = [1, -1, \mathbf{0}_{2n-1}], \quad Y_0 = [\mathbf{0}_{2n}, 1]^\top. \tag{3.16}$$

Remark 13 *The first equality in* (3.16) *ensures that nature of zeros of the two functions $Ce^{At}X_0$ and $C'e^{Pt}Y_0$ are the same. If one of them has a non-tangential zero at t^* the other one will also have a non-tangential zero at t^*. To see this, suppose $Ce^{At^*}X_0 = 0$ and $Ce^{At}X_0$ changes sign at t^*. The same things happen to $C'e^{Pt}Y_0$ due to the fact that the two functions are different with only a positive factor of $\gamma e^{\lambda t}$.*

Without loss of generality, we assume the element A_{11} is negative. This assumption is needed when constructing the CTMDP in the sequel. If the assumption does not hold, we can always replace A with $A-\lambda_0\mathbb{I}_n$ for a sufficiently large λ_0 and merge λ_0 with λ in (3.16). Define the map $\phi_1 : \cup_n\mathbb{Q}^{n\times n} \to \cup_n\mathbb{Q}^{2n\times 2n}_{\geq 0}$ such that $\phi_1(A)$ is obtained by replacing each entry A_{ij} with the matrix $\begin{bmatrix}\alpha_{ij} & \beta_{ij}\\ \beta_{ij} & \alpha_{ij}\end{bmatrix}$, where $\alpha_{ij} = max(A_{ij}, 0)$ and $\beta_{ij} = max(-A_{ij}, 0)$. The map ϕ_1 maps any square matrix to another matrix with non-negative entries ([4]). Also define the map $\phi_2 : \cup_n\mathbb{Q}^n \to \cup_n\mathbb{Q}^{2n}$ such that $\phi_2(X)$ replaces each entry $X(i)$ with two entries $[X(i), 0]^\top$.

Proposition 3.2.3 *We have $C''e^{\phi_1(A)t}Y_2 = Ce^{At}X_0$ with $Y_2 := \phi_2(X_0)$ and $C'' := [1, -1, \mathbf{0}_{2n-2}]$.*

Proof *We can show inductively that for any $k \in \{0, 1, 2, \ldots\}$, $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$, and $[\beta_1, \beta_2, \ldots, \beta_n] := [\alpha_1, \alpha_2, \ldots, \alpha_n]A^k$, we have*

$$[\alpha_1, -\alpha_1, \alpha_2, -\alpha_2, \ldots, \alpha_n, -\alpha_n]\phi_1(A)^k = [\beta_1, -\beta_1, \beta_2, -\beta_2, \ldots, \beta_n, -\beta_n].$$

Substitute $[\alpha_1, \alpha_2, \ldots, \alpha_n]$ by C and $[\beta_1, \beta_2, \ldots, \beta_n] = CA^k$ to get

$$\begin{aligned}
&C''\phi_1(A)^kY_2 = C''\phi_1(A)^k\phi_2(X_0) = [\beta_1, -\beta_1, \beta_2, -\beta_2, \ldots, \beta_n, -\beta_n]\phi_2(X_0)\\
&= [\beta_1, \beta_2, \ldots, \beta_n]X_0 = CA^kX_0\\
&\Rightarrow C''e^{\phi_1(A)t}Y_2 = \sum_{k=0}^{\infty}\frac{t^k}{k!}C''\phi_1(A)^kY_2 = \sum_{k=0}^{\infty}\frac{t^k}{k!}CA^kX_0 = Ce^{At}X_0.
\end{aligned}$$

Next, we define $\lambda := \max_i \sum_{j=1}^{n} |A_{ij}| + 1$, $P_2 := \phi_1(A) - \lambda \mathbb{I}_n$, and the vector $\boldsymbol{\beta} \in \mathbb{Q}^{2n}$ with

$$\boldsymbol{\beta}(2i-1) = \boldsymbol{\beta}(2i) = \max(0, -P_2 Y_2(2i-1), -P_2 Y_2(2i)) \quad 1 \leq i \leq n.$$

Note that the row sum of P_2 is at most -1 and $\boldsymbol{\beta} + P_2 Y_2$ is element-wise non-negative with the maximum element

$$\gamma := \max_i P_2 Y_2(i) + \boldsymbol{\beta}(i).$$

Proposition 3.2.4 *The above choices of λ, γ and the matrix*

$$P := \begin{bmatrix} P_2 & \vdots & (P_2 Y_2 + \boldsymbol{\beta})/\gamma \\ \dots & \dots & \dots \\ \mathbf{0} & \vdots & 0 \end{bmatrix}$$

satisfy (3.16) *in Theorem 3.2.6. Moreover, P is row sub-stochastic.*

Proof *We can easily show by induction that*

$$P^k Y_0 = \begin{bmatrix} P_2^{k-1}(P_2 Y_2 + \boldsymbol{\beta})/\gamma \\ 0 \end{bmatrix}, \quad \forall k \in \{1, 2, \dots\}.$$

$$C' e^{Pt} Y_0 = \sum_{k=0}^{\infty} \frac{t^k}{k!} C' P^k Y_0 = C' Y_0 + C'' \sum_{k=1}^{\infty} \frac{t^k}{k!} P_2^{k-1}(P_2 Y_2 + \boldsymbol{\beta})/\gamma,$$

where $C'' := [1, -1, \mathbf{0}_{2n-2}]$ is the same vector as C' but the last element is eliminated.

$$C' e^{Pt} Y_0 = C' Y_0 + C'' e^{P_2 t} Y_2/\gamma - C'' Y_2/\gamma + \sum_{k=1}^{\infty} \frac{t^k}{k!} C'' P_2^{k-1} \boldsymbol{\beta}/\gamma.$$

The term $C' Y_0$ is zero by simple multiplication of the two vectors. $C'' Y_2 = C'' \phi_2(X_0) = X_0(1)$, which is also assumed to be zero. Finally, we see by induction that for all $k \in \{1, 2, \dots\}$, the elements $(2i-1)$ and $2i$ of the matrix $P_2^{k-1} \boldsymbol{\beta}$ are equal due to the particular structure of P_2 and $\boldsymbol{\beta}$. Therefore, the last sum in the above is also zero and we get

$$\begin{aligned} C' e^{Pt} Y_0 &= C'' e^{P_2 t} Y_2/\gamma = C'' e^{\phi_1(A)t - \lambda \mathbb{I} t} \phi_2(X_0)/\gamma \\ &= C'' e^{\phi_1(A)t} \phi_2(X_0) e^{-\lambda t}/\gamma = C e^{AtX_0} e^{-\lambda t}/\gamma. \end{aligned}$$

To show that P is a sub-stochastic matrix, we recall that $P_2 Y_2 + \boldsymbol{\beta} \geq 0$ with maximum element γ. Then

$$P_2 \times \mathbf{1}_{2n} + (P_2 Y_2 + \boldsymbol{\beta})/\gamma \leq \phi_1(A)\mathbf{1}_{2n} - \lambda \mathbf{1}_{2n} + \mathbf{1}_{2n} = \phi_1(A)\mathbf{1}_{2n} - \max_i \sum_j |A_{ij}| \leq 0.$$

As the last step, we add an additional row and column to P to make it stochastic:

$$\mathbf{Q}^a := \begin{bmatrix} P_2 & \vdots & \Theta & \vdots & (P_2 Y_2 + \boldsymbol{\beta})/\gamma \\ \cdots & & \cdots & & \cdots \\ \mathbf{0}_{2\times 2n} & \vdots & \mathbf{0}_{2\times 1} & \vdots & \mathbf{0}_{2\times 1} \end{bmatrix}, \bar{C} = \begin{bmatrix} 1 & -1 & \mathbf{0}_{2n} \end{bmatrix}, \bar{Y}_0 = \begin{bmatrix} \mathbf{0}_{2n+1} \\ 1 \end{bmatrix},$$

where Θ has non-negative entries and is such that $\mathbf{Q}^a$ is stochastic (sum of elements of each row is zero). The added row and column correspond to an absorbing state for a CTMDP with no effect on reachability probability: $\bar{C} e^{t\mathbf{Q}^a} \bar{Y}_0 = C' e^{Pt} Y_0$.

Next, we obtain a second generator matrix for the CTMDP. Define $\mathbf{Q}^b := \mathbf{Q}^a + K$ with

$$K := \begin{bmatrix} -r & r & \mathbf{0}_{2n} \\ \mathbf{0}_{(2n+1)\times 1} & \mathbf{0}_{(2n+1)\times 1} & \mathbf{0}_{(2n+1)\times 2n} \end{bmatrix},$$

Note that $\mathbf{Q}^b$ has exactly the same transition rates as in $\mathbf{Q}^a$ except the transition from state 1 to state 2, which is changed by r.

Remark 14 *We assumed w.l.o.g. that A_{11} is negative. The construction of $P_2, P, \mathbf{Q}^a$ results in a positive value for $\mathbf{Q}^a_{12}$. Therefore, it is possible to select both negative and positive values for r such that $\mathbf{Q}^b_{12} = \mathbf{Q}^a_{12} + r \geq 0$.*

Construction of the CTMDP. The CTMDP $\mathcal{M}$ has $2n+2$ states, corresponding to the rows of $\mathbf{Q}^a$ and $\mathbf{Q}^b$, with the absorbing state $2n+2$ associated with the **good** state and the absorbing state $2n+1$ with reachability probability equal to zero. We shall set the time bound to be B. $\mathcal{D}_s$ the set of actions that can be taken in state $s \in \{2, 3, \ldots, 2n+2\}$ is singleton and $\mathcal{D}_1 = \{a, b\}$. The set of decision vectors has two elements $\mathcal{D} = \{\mathbf{d}^a, \mathbf{d}^b\}$ corresponding to the actions a, b taken at state 1. For simplicity, we denote the generator matrices of these decision vectors by $\mathbf{Q}^a$ and $\mathbf{Q}^b$, respectively. Moreover, the two actions a, b have the same transition rates for jumping from state 1 to other states, except giving different rates r_a, r_b for jumping from 1 to 2 such that $r_b - r_a = r$.

The optimal policy π takes decision vector $\mathbf{d}_t \in \mathcal{D}$ at time $B-t$ such that $\mathbf{d}_t \in \mathcal{F}_{n+2}(W_t^\pi)$ for all $t \in [0, B]$ as defined in (3.4).

Proposition 3.2.5 *Let r have the same sign of the first non-zero element of the set $\{\bar{C}\bar{Y}_0, \bar{C}\mathbf{Q}^a\bar{Y}_0, \bar{C}(\mathbf{Q}^a)^2\bar{Y}_0, \ldots\}$ and such that $\mathbf{Q}^a_{12} + r \geq 0$. This particular selection of r results in the optimality of $\mathbf{d}^a$ at $t = 0$.*

Proof *We have $W_0^\pi = \bar{Y}_0$ and $\mathcal{F}_k(W_0^\pi) = \arg\max_{\mathbf{d}} [\mathbf{Q}^{\mathbf{d}}]^k \bar{Y}_0$. Then, we need to compare $[\mathbf{Q}^a]^k \bar{Y}_0$ with $[\mathbf{Q}^b]^k \bar{Y}_0$ for different values of k and see which one gives the first highest value. These two are the same for $k = 1$ and $\mathcal{F}_1(W_0^\pi) = \arg\max_{\mathbf{d}} \mathbf{Q}^{\mathbf{d}} \bar{Y}_0 = \{\mathbf{d}^a, \mathbf{d}^b\}$. Suppose For $k_0 > 1$ is the smallest index such that $\bar{C}[\mathbf{Q}^a]^{k_0}\bar{Y}_0 \neq 0$. It can be shown inductively that $[\mathbf{Q}^b]^k \bar{Y}_0 = [\mathbf{Q}^a]^k \bar{Y}_0$ for all $1 \leq k \leq k_0$:*

$$\begin{aligned} [\mathbf{Q}^b]^k \bar{Y}_0 &= \mathbf{Q}^b [\mathbf{Q}^b]^{k-1} \bar{Y}_0 = (\mathbf{Q}^a + K)[\mathbf{Q}^b]^{k-1} \bar{Y}_0 = (\mathbf{Q}^a + K)[\mathbf{Q}^a]^{k-1} \bar{Y}_0 \\ &= [\mathbf{Q}^a]^k \bar{Y}_0 + K[\mathbf{Q}^a]^{k-1} \bar{Y}_0 = [\mathbf{Q}^a]^k \bar{Y}_0 - r \begin{bmatrix} \bar{C}[\mathbf{Q}^a]^{k-1} \bar{Y}_0 \\ \mathbf{0}_{(2n+1)\times 2n} \end{bmatrix} = [\mathbf{Q}^a]^k \bar{Y}_0. \end{aligned}$$

This means $\mathcal{F}_k(W_0^\pi) = \arg\max_{\mathbf{d}}[\mathbf{Q}^{\mathbf{d}}]^k\bar{Y}_0 = \{\mathbf{d}^a, \mathbf{d}^b\}$ *for all* $1 \leq k \leq k_0$. *We have for* $k = k_0 + 1$

$$[\mathbf{Q}^b]^{k_0+1}\bar{Y}_0 = [\mathbf{Q}^a]^{k_0+1}\bar{Y}_0 - r\begin{bmatrix}\bar{C}[\mathbf{Q}^a]^{k_0}\bar{Y}_0\\ \mathbf{0}_{(2n+1)\times 2n\cdot}\end{bmatrix}$$

The first element of $[\mathbf{Q}^b]^{k_0+1}\bar{Y}_0$ *is strictly less than the first element of* $[\mathbf{Q}^a]^{k_0+1}\bar{Y}_0$ *since* r *has the same sign as* $\bar{C}[\mathbf{Q}^a]^{k_0}\bar{Y}_0$. *Thus* $\mathcal{F}_{k_0+1}(W_0^\pi) = \arg\max_{\mathbf{d}}[\mathbf{Q}^{\mathbf{d}}]^{k_0+1}\bar{Y}_0 = \{\mathbf{d}^a\}$.

Note that the Skolem problem is trivial with the solution $z_t = 0$ for all $t \in [0, B]$ if all the elements of the set $\{\bar{C}\bar{Y}_0, \bar{C}\mathbf{Q}^a\bar{Y}_0, \bar{C}(\mathbf{Q}^a)^2\bar{Y}_0, \ldots\}$ are zero.

Prop. 3.2.5 guarantees existence of an $\varepsilon \in (0, B)$ such that W_t^π satisfies

$$\frac{d}{dt}W_t^\pi = \mathbf{Q}^a W_t^\pi \quad \forall t \in (0, \varepsilon),$$

with the initial condition $W_0^\pi(2n+2) = 1$ and $W_0^\pi(s) = 0$ for all $s \in \{1, 2, \ldots, 2n+1\}$.

To check if the optimal policy switches to $\mathbf{d}^b$ at some time point, we should check if there is $t^* < B$ such that $\mathbf{d}^b \in \mathcal{F}_{n+2}(W_{t^*}^\pi)$. This is equivalent to having t^* being non-tangential for the maximization in $\mathcal{F}_1(W_t^\pi)$, which means t^* is non-tangential for the equation

$$\mathbf{Q}^a W_t^\pi = \mathbf{Q}^b W_t^\pi \Leftrightarrow KW_t^\pi = 0 \Leftrightarrow \bar{C}W_t^\pi = 0.$$

Summarizing the above derivations, we have the following set of ODEs

$$\frac{d}{dt}W_t^\pi = \mathbf{Q}^a W_t^\pi \quad , W_0^\pi = \bar{Y}_0, \quad z_t = \bar{C}W_t^\pi. \tag{3.17}$$

The optimal policy for CTMDP $\mathcal{M}$ switches from $\mathbf{d}^a$ to $\mathbf{d}^b$ at some time point t^* if and only if z_t in (3.17) has a non-tangential zero in $(0, B)$ if and only if the original dynamics $Ce^{At}X_0$ has a non-tangential zero in $(0, B)$. This completes the proof of Theorem 3.2.5.

3.3. Enhancing the Scalability of Time-Bounded Reachability of CTMCs and CTMDPs

Existing approaches for approximating the time-bounded reachability problem are based on discretization-based, and in practice, are expensive computational procedures, especially as the time bound increases. The standard state-space reduction technique is probabilistic bisimulation [94, 103, 30, 14]: a probabilistic bisimulation is an equivalence relation on the states that allows "lumping" together the equivalence classes without changing the value of time-bounded reachability properties, or indeed of any CSL property [14]. Unfortunately, probabilistic bisimulation is a strong notion and small perturbations to the transition rates can change the relation drastically. Thus, in practice, it is often of limited use.

In this section, we take a control-theoretic view to state space reductions of CTMCs and CTMDPs. Our starting point is that the forward Chapman-Kolmogorov equations characterizing time-bounded reachability define a linear dynamical system for CTMCs and

a switched affine dynamical system for CTMDPs; moreover, one can transform the problem so that the dynamics is stable. Our first observation is a generalization of probabilistic bisimulation to a quantitative setting. We show that probabilistic bisimulation can be viewed as a projection matrix that relates the original dynamical system with its bisimulation reduction. We then relax bisimulation to a quantitative notion, using a *generalized projection* operation between two linear systems. The content of this section is based on our papers [158, 160].

CTMCs. A generalized projection does not maintain a linear relationship between the original and the reduced linear systems. However, our second result shows how the difference between the states of the two linear dynamical systems can be bounded by an exponentially decreasing function of time. The key to this result is finding an appropriate Lyapunov function for the difference between the two dynamics, which demonstrates an exponential convergence over time. We focus the presentation on irreducible CTMCs (i.e., those with the property that it is possible with some positive probability to get from any state to any other state in finite time) and show that the search for a suitable Lyapunov function can be reduced to a system of matrix inequalities, which have a simple solution. This leads to an error bound of the form $L_0 e^{-\kappa t}$, where L_0 depends on the matrices defining the dynamics, and κ is related to the eigenvalues of the dynamics. Clearly, the error goes to zero exponentially as $t \to \infty$. Hence, by solving the reduced linear system, one can approximate the time-bounded reachability probability in the original system, with a bound on the error that converges to zero as a function of the reachability horizon. For reducible CTMCs (i.e., those that are not irreducible), we show that the same approach is applicable by preprocessing the structure of CTMC and eliminating those bottom strongly connected components that do not influence the reachability probability.

The Lyapunov approach suggests a systematic procedure to reduce the state space of a CTMC. If the original dynamical system has dimension m, we show, using Schur decomposition, that we can compute an r-dimensional linear system for each $r \leq m$ as well as a Lyapunov-based bound on the error between the dynamics. Thus, for a given tolerance ε, one can iterate this procedure to find an appropriate r. This r-dimensional system can be solved using existing techniques, e.g., computing the exponential of upper-triangular matrices.

CTMDPs. For CTMDPs, we generalize the approach for CTMCs using Lyapunov stability theorems for switched systems. Once again, the objective is to use multiple Lyapunov functions as a way to demonstrate stability, and derive an error bound from the multiple Lyapunov functions. For this we construct a piecewise quadratic Lyapunov function for a switched affine dynamical system. Then we synthesize a policy for the CTMDP via its reduced-order switched system in order to have time-bounded reachability probability above a threshold. We provide error bounds that depend on the minimum dwell time of the policy.

The notion of *behavioral pseudometrics* on stochastic systems has been studied extensively [10, 51] as a quantitative measure of dissimilarity between states, but mainly for

discrete time Markov models and mostly for providing an upper bound on the difference between *all* formulas in a logic; by necessity, this makes the distance too pessimistic for a single property. In contrast, our approach considers a notion of distance for a specific time-bounded reachability property, and provides a time-varying error bound.

We have implemented our state space reduction approach and evaluated its performance on a queuing system benchmark. Fixing time horizon and error bound, our reduction algorithm computes a reduced order system, the analysis of which requires a significantly less computational effort. We show that, as the time horizon increases, we get significant reductions in the dimension of the linear system while providing tight bounds on the quality of the approximation.

3.3.1. Time-Bounded Reachability on CTMCs

Let $\mathcal{C} = (S \uplus \{\mathbf{good}, \mathbf{bad}\}, \mathbf{Q})$ be a CTMC, with $|S| = n$, and two states **good** and **bad** and $B > 0$ denotes a finite time bound. We are interested in approximating the probability of reaching the **good** state while avoiding the **bad** state before the time B for a given subset $S_0 \subseteq S$ of states. Defining $n_0 = |S_0|$, we denote solution to this problem as an $n_0 \times 1$ vector $\mathrm{Prob}^{\mathcal{C}}(C, T)$, where C is an $n_0 \times (n+2)$ matrix with n_0 ones on its main diagonal, corresponding to the states in S_0. If $S_0 = S$, then C is the $(n+2) \times (n+2)$ identity matrix. For $s \in S_0$, the value $\mathrm{Prob}^{\mathcal{C}}(C, T)(s)$ corresponds to the probability with which the **good** state is visited while avoiding the **bad** state before the time bound B, when the initial distribution is $\alpha = \mathbf{1}(s)$. Note that $\mathbf{Q}$ is characterized such that the two states **good** and **bad** are made absorbing by removing all of their outgoing edges. The solution to the time-bounded reachability problem for a projection matrix C can be obtained as:

$$\begin{aligned} &\frac{d}{dt} Z_t = \mathbf{Q} Z_t, \quad Z_0 = \mathbf{1}(\mathbf{good}), \\ &\mathrm{Prob}^{\mathcal{C}}(C, t) = C Z_t \end{aligned} \tag{3.18}$$

where $Z_t \in \mathbb{R}^{n+2}$ is a column vector with elements $Z_t(i) = \mathrm{Prob}^{\mathcal{C}}(\mathbf{1}(s_i), t)$. Notice that in this formulation, we have let time "run backward": we start with a initial vector which is zero except for corresponding element to the state **good** and compute "backward" up to the time B. By reordering states, if necessary, the generator matrix Q in (3.18) can be written as:

$$\mathbf{Q} = \begin{bmatrix} A & \vdots & \boldsymbol{\chi} & \vdots & \mathcal{B} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{0} & \vdots & \mathbf{0} & \vdots & \mathbf{0} \end{bmatrix} \tag{3.19}$$

with $A \in \mathbb{R}^{n \times n}$, $\boldsymbol{\chi} \in \mathbb{R}^{n \times 1}$, and $\boldsymbol{\beta} \in \mathbb{R}^{n \times 1}$. Vectors $\boldsymbol{\chi}$ and $\boldsymbol{\beta}$ contain the rates corresponding to the incoming transitions to the states **bad** and **good**, respectively. With this reordering of the states, it is obvious that in (3.18), $Z_t(\mathbf{bad}) = 0$ and $Z_t(\mathbf{good}) = 1$, thus we assume states **good** and **bad** are not included in C. We write Z_t^S for the vector

(in $\mathbb{R}^n$) restricting Z to states in S. These variables should satisfy

$$\frac{d}{dt}Z_t^S = AZ_t^S + \boldsymbol{\beta}, \quad Z_0^S = 0,$$
$$\text{Prob}^{\mathcal{C}}(C,t) = CZ_t^S. \tag{3.20}$$

Equation (3.20) can be seen as model of a linear dynamical system with unit input. Our aim here is to compute an approximate solution of (3.20) using reduction techniques from control theory while providing guarantees on the accuracy of the computation and to interpret the solution as the probability for time-bounded reachability.

Let $\gamma := max_{i=1:n}|a_{ii}|$, the maximal diagonal element of A, and define matrix H as:

$$H = \frac{A}{\gamma} + \mathbb{I}_n, \tag{3.21}$$

where $\mathbb{I}_n$ is the $n \times n$ identity matrix. In the following, we fix the following assumption.

Assumption 2 *H is an irreducible matrix, i.e., its associated directed graph is strongly connected. Moreover, $\boldsymbol{\beta} + \boldsymbol{\chi} \neq 0$. That is, either* **good** *or* **bad** *is reachable in one step from some state in S.*

Remark 15 *The above assumption is "WLOG." First, if there is no edge from S to* **good** *or* **bad***, the problem is trivial. Second, the general case, when H is not irreducible can be reduced to the assumption in polynomial time (see Appendix A.3.2). Thus, the assumption restricts attention to the core technical problem. Throughout the rest of this section, we only consider models for which the above assumption holds.*

Recall that a matrix A is stable if every eigenvalue of A has negative real part. The *spectral radius* of a matrix is the largest absolute value of its eigenvalues. We also denote the real part of the eigenvalues of a complex number by $\text{Re}(\cdot)$.

Proposition 3.3.1 *Assumption 2 implies that matrix A is invertible and stable.*

Proof *Due to the definition of H in* (3.21)*, we have $\lambda(H) = 1 + \lambda(A)/\gamma$, where $\lambda(\cdot)$ denotes the eigenvalues of a matrix. We use ϱ for the spectral radius of H. For irreducible matrix H, the Perron-Frobenius theorem implies that ϱ is positive and it is a simple eigenvalue of H. There are left eigenvectors associated with eigenvalue ϱ such that their entries are all positive. Without loss of generality, we denote one of these left eigenvectors by ν that is normalized such that sum of its entries is equal to one. The aim is to show that $\varrho < 1$. Since the sum of every row of H is less than or equal to one, ϱ cannot be greater than one. The following reasoning shows that $\varrho = 1$ gives a contradiction. Define a diagonal matrix*

$$\Delta := diag(\boldsymbol{\chi} + \boldsymbol{\beta})/\gamma$$

and let $\tilde{H} := H + \Delta$. This matrix is a row stochastic matrix and is irreducible. Then it can be considered as an irreducible probability transition matrix of a discrete-time Markov chain. Note that

$$\nu\tilde{H} = \nu(H + \Delta) = \varrho\nu + \nu\Delta = \nu + \nu\Delta.$$

We can show by induction that the following inequality

$$\nu\tilde{H}^k \geq \nu + \nu\Delta + \nu\Delta^2 + \ldots + \nu\Delta^k \tag{3.22}$$

holds element-wise for all $k \in \mathbb{N}$. *This can be seen using the inductive step*

$$\nu\tilde{H}^{k+1} = (\nu\tilde{H}^k)\tilde{H} \geq (\nu + \nu\Delta + \ldots + \nu\Delta^k)(H + \Delta) \geq \nu + \nu\Delta + \nu\Delta^2 + \ldots \nu\Delta^{k+1}.$$

The last inequality is true since all the additional terms in its left-hand side have non-negative entries (all elements of H, Δ, ν *are non-negative).*

Taking the sum of all entries of both sides of (3.22)*, we get*

$$\sum_i \nu_i \geq \sum_i \nu_i(1 + \Delta_{ii} + \Delta_{ii}^2 + \ldots + \Delta_{ii}^k),$$

which is a contradiction since at least one diagonal element of Δ *is positive. Then we have* $\varrho < 1$*, which results in* $\mathrm{Re}(\lambda(A)) < 0$ *due to the relation* $\lambda(H) = 1 + \lambda(A)/\gamma$*. Therefore,* A *is stable and invertible.*

Since the input to (3.20) is fixed, we try to transform it to a set of differential equations without input but with initial value. Let us take a transformation that translates Z_t by the offset vector $A^{-1}\boldsymbol{\beta}$:

$$X_t := Z_t^S + A^{-1}\boldsymbol{\beta}. \tag{3.23}$$

The evolution of $X(\cdot)$ is:

$$\begin{aligned} &\frac{d}{dt}X_t = AX_t, \quad X_0 = A^{-1}\mathcal{B}, \\ &\mathrm{Prob}^{\mathcal{C}}(C, t) = CX_t + d. \end{aligned} \tag{3.24}$$

where $d = -CA^{-1}\mathcal{B}$. The *dimension* (number of variables) of dynamical system (3.24) is n, the size of the state space S.

Remark 16 *Under Assumption 2, the solution of infinite horizon reachability problem is* $-A^{-1}\mathcal{B}$*, which can be computed efficiently as the solution of a system of linear equations. Elements of* X_t *defined in* (3.23) *contain the values of finite-horizon reachability.*

In the following, we show how the solution of this dynamical system can be approximated by a dynamical system of lower dimension. Our approach relies on stability property of matrix A, and gives an upper bound on the approximation error that converges exponentially to zero as a function of time. Thus our approach is beneficial for long time horizons when previous techniques fail to provide tight bounds.

Bisimulation and Projections

Probabilistic bisimulation or lumpability is a classical technique to reduce the size of the state space of a CTMC [94, 103, 30, 14]. For CTMC $\mathcal{C} = (S, \mathbf{Q})$ with space $S_{\mathcal{C}} = S \uplus \{\mathbf{good}, \mathbf{bad}\}$, a bisimulation on $\mathcal{C}$ is an equivalence relation $\cong$ on $S_{\mathcal{C}}$ such that **good** and **bad** are singleton equivalence classes and for any two states $s_1, s_2 \in S$, $s_1 \cong s_2$ implies $\mathbf{Q}(s_1, \Theta) = \mathbf{Q}(s_2, \Theta)$ for every equivalence class Θ of $\cong$, where $\mathbf{Q}(s, \Theta) := \sum_{s' \in \Theta} \mathbf{Q}(s, s')$. Given a bisimulation relation $\cong$ on $\mathcal{C}$, we can construct a CTMC $\bar{\mathcal{C}} = (\bar{S}, \bar{\mathbf{Q}})$ of smaller size such that probabilities are preserved over paths of $\mathcal{C}$ and $\bar{\mathcal{C}}$. In particular, $s_1 \cong s_2$, implies that

$$\mathrm{Prob}^{\mathcal{C}}(\mathbf{1}(s_1), t) = \mathrm{Prob}^{\bar{\mathcal{M}}}(\mathbf{1}(s_2), t), \quad \forall t \in \mathbb{R}_{\geq 0}.$$

The CTMC $\bar{\mathcal{C}}$ has the quotient state space $\{[s]_{\cong} \mid s \in S\} \uplus \{\mathbf{good}, \mathbf{bad}\}$, where $[s]_{\cong}$ is the equivalence class of $s \in S$, rate function $\bar{\mathbf{Q}}([s]_{\cong}, \Theta) = \mathbf{Q}(s, \Theta)$ for any $\Theta \in \bar{S}$.

We now show how the differential equation (3.24) for $\mathcal{C}$ and $\bar{\mathcal{C}}$ relate. Assume that the state space of $\bar{\mathcal{C}}$ is $\bar{S} \cup \{\mathbf{good}, \mathbf{bad}\}$, where $|\bar{S}| = r$. We have

$$\begin{aligned} &\frac{d}{dt}\bar{X}_t = \bar{A}\bar{X}_t, \quad \bar{X}_0 = \bar{A}^{-1}\bar{\mathcal{B}}, \\ &Prob^{\bar{\mathcal{C}}}(\bar{C}, t) = d + \bar{C}\bar{X}_t, \end{aligned} \tag{3.25}$$

where $\bar{A}$ and $\bar{\mathcal{B}}$ are computed similarly to that of $\mathcal{C}$ according to the generator matrix of $\bar{\mathcal{C}}$. Note that $\bar{A}$ is an $r \times r$ matrix. Matrix $\bar{C}$ is $n_0 \times r$ constructed according to S_0, with $|S_0|$ ones corresponding to the quotient states $\{[s]_{\cong} \mid s \in S_0\}$. We now define a *projection matrix* $P_{\cong} \in \mathbb{R}^{n \times r}$ as $P_{\cong}(i, j) = 1$ if $s_i \in [j]$, i.e., s_i belongs to the equivalence class $[j] \in \bar{S}$, and zero otherwise. This projection satisfies $CP_{\cong} = \bar{C}_S$, and together with the definition of $\cong$ implies the following proposition.

Proposition 3.3.2 *For every bisimulation $\cong$, the projection matrix $P_{\cong}$ satisfies the following*

$$AP_{\cong} = P_{\cong}\bar{A}, \quad \mathcal{B} = P_{\cong}\bar{\mathcal{B}}. \tag{3.26}$$

Conversely, every projection matrix satisfying (3.26) *defines a bisimulation relation. In particular,*

$$X_t = P_{\cong}\bar{X}_t, \quad \forall t \in \mathbb{R}_{\geq 0}. \tag{3.27}$$

Example 2 *As an example, consider the CTMC in Figure 3.1 with $\Lambda_{31} = 0$ and $\Lambda_{42} = 1$ without any state* **bad**, *and assume first that $\varepsilon_{ij} = 0$ for all i, j. We are interested in computing the probability of reaching state* **good**, *which is made absorbing by removing its outgoing links. It is easy to see that the bisimulation classes are $\{s_1, s_2\}$, $\{s_3, s_4\}$, and $\{\mathbf{good}\}$. The bisimulation reduction and the corresponding projection matrix $P_{\cong}$ are shown on the right-hand side. The differential equation for the reduced CTMC has dimension 2.*

Unfortunately, as is well known, bisimulation is a strong condition, and small perturbations in the rates can cause two states to not be bisimilar. Consider a perturbed version of

the CTMC by setting $\varepsilon_{23} = -\varepsilon_{13} = 0.05$, which will give the following generator matrix:

$$\mathbf{Q} = \begin{bmatrix} -3.95 & 0 & 1.95 & 0 & 2 \\ 0 & -4.05 & 1.05 & 1 & 2 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Here, $\varepsilon_{ij} \neq 0$ for some transitions, and the CTMC on the right-hand side of Figure 3.1 is not a bisimulation reduction. Let us also consider a perturbed version of the CTMC on the right-hand side of Figure 3.1 with the generator matrix

$$\mathbf{Q}_r = \begin{bmatrix} -4.05 & 2.05 & 2 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Clearly, these two perturbed CTMCs are not bisimilar according to the usual definition of bisimulation relation, but the following real matrix

$$P = \begin{bmatrix} \frac{390}{469} & \frac{39}{469} & \frac{40}{469} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

satisfies the equality $\mathbf{Q}P = P\mathbf{Q}_r$. Note that P is no longer a projection matrix but has entries in $[0, 1]$, which sum up to 1 for each row. This particular P satisfies $AP = P\bar{A}$ but not $\mathcal{B} = P\bar{\mathcal{B}}$ (see (3.26)). Thus the original dynamics of X_t and their lower-dimensional version $\bar{X}_t$, reduced with P, do not satisfy the equality (3.27).

However, since A is a stable matrix, we expect the trajectories of the original and the reduced dynamics to converge, that is, the error between the trajectories to go to zero as time goes to infinity. In the following, we generalize projection matrices as above, and formalize this intuition.

Generalized Projections and Reduction

Suppose we are given CTMCs $\mathcal{C}$ and $\bar{\mathcal{C}}$, with corresponding dynamical systems (3.24) and (3.25), and a matrix P with entries in $[0, 1]$ whose rows add up to 1, such that $AP = P\bar{A}$. We call such a P a *generalised projection*. Define vector $\bar{C} := CP$. In general, the equality $\mathcal{B} = P\bar{\mathcal{B}}$ does not hold for generalized projections. In the following we provide a method based on Lyapunov stability theory to quantify an upper bound ε_t such that

$$\left| Prob^{\mathcal{C}}(C, t) - Prob^{\bar{\mathcal{C}}}(\bar{C}, t) \right| \leq \varepsilon_t \tag{3.28}$$

for all $t \geq 0$, where ε_t depends linearly on the mismatch $\mathcal{B} - P\bar{\mathcal{B}}$ and decays exponentially with t.

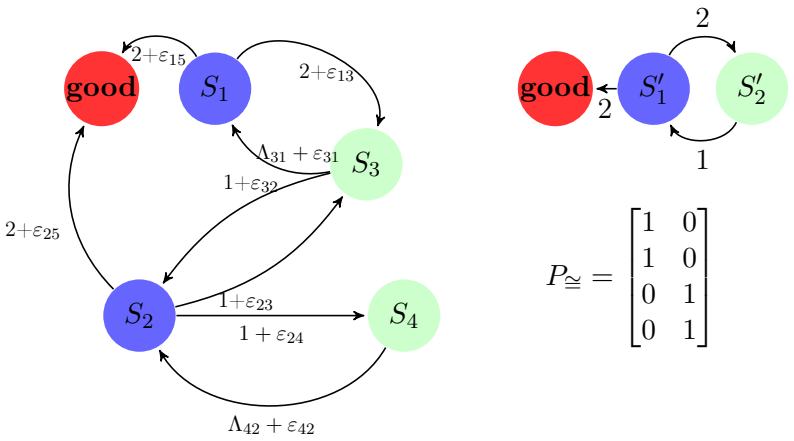

Figure 3.1.: Full state ε-perturbed CTMC (left), reduced-order CTMC (right), and projection matrix (right, below) computed for the unperturbed CTMC ($\varepsilon_{ij} = 0$) with $\Lambda_{31} = 0$ and $\Lambda_{42} = 1$.

First, we recall some basic results for linear dynamical systems (see, e.g., [54]). The dynamics of these systems are represented by a set of linear differential equations of the form

$$\frac{d}{dt}Y_t = AY_t, \quad Y_t \in \mathbb{R}^n, \quad Y_0 = Y_0. \tag{3.29}$$

We call the system stable if A is a stable matrix. In this case, it is known that $\lim_{t\to\infty} Y_t = 0$ for any initial state $Y_0 = Y_0 \in \mathbb{R}_0$.

Definition 3.3.1 *A continuous scalar function $V : \mathbb{R}^n \to \mathbb{R}$ is called a* Lyapunov function *for the dynamical system* (3.29) *if $V(y) = 0$ for $y = 0$; $V(y) > 0$ for all $y \in \mathbb{R}^n\backslash\{0\}$; and $dV(Y_t)/dt < 0$ along trajectories of the dynamical system with $Y_t \neq 0$.*

A matrix $M \in \mathbb{R}^{n\times n}$ is symmetric if $M^\top = M$. A symmetric matrix M satisfying the condition $Y^\top MY > 0$ for all $Y \in \mathbb{R}^n\backslash\{0\}$ is called *positive definite*, and written as $M \succ 0$. Any symmetric matrix M satisfying $Y^\top MY \geq 0$ for all $Y \in \mathbb{R}^n$ is called *positive semi-definite*, written as $M \succeq 0$. Similarly, we can define *negative definite* matrices $M \prec 0$ and negative semi-definite matrices $M \preceq 0$. We write $M_1 \succ M_2$ if and only if $M_1 - M_2 \succ 0$ and $M_1 \succeq M_2$ if and only if $M_1 - M_2 \succeq 0$. $M_1 \prec M_2$ and $M_1 \preceq M_2$ are defined similarly. The eigenvalues of a symmetric positive definite matrix M are always positive. We denote the largest eigenvalue of the positive definite matrix M by $\lambda_{max}(M)$. Any positive definite matrix M satisfies $Y^\top MY \leq \lambda_{max}(M)\|Y\|_2^2$ for any $Y \in \mathbb{R}^n$, where $\|Y\|_2$ indicates the two norm of Y. The following is standard.

Theorem 3.3.1 *[98] Linear dynamical system* (3.29) *is stable iff there exists a quadratic Lyapunov function $V(Y) = Y^\top MY$ such that $M \succ 0$ and $A^\top M + MA \prec 0$. Moreover, for any constant $\kappa > 0$ such that $A^\top M + MA + 2\kappa M \preceq 0$, we have*

$$\|Y_t\|_2 \leq Le^{-\kappa t}\|Y_0\|_2, \quad \forall Y_0 \in \mathbb{R}^n, \forall t \in \mathbb{R}_{\geq 0},$$

for some constant $L \geq 0$, where $\|\cdot\|_2$ indicates the two norm of a vector.

Note that in our setting, we are not interested in the study of asymptotic stability of systems, but we are given two dynamical systems (3.24) and (3.25), and we would like to know how close their trajectories are as a function of time. In this way we can use one of them as an approximation of the other one with guaranteed error bounds. For this reason, we define Lyapunov function $V : \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}$ of the form

$$V(X, \bar{X}) = (X - P\bar{X})^\top M (X - P\bar{X}), \tag{3.30}$$

where $M \succ 0$ is a positive definite matrix. The value of $V(X_t, \bar{X}_t)$ at $t = 0$ can be calculated as

$$\begin{aligned} V(X_0, \bar{X}_0) &= (A^{-1}\mathcal{B} - P\bar{A}^{-1}\bar{\mathcal{B}})^\top M (A^{-1}\mathcal{B} - P\bar{A}^{-1}\bar{\mathcal{B}}) \\ &= (\mathcal{B} - P\bar{\mathcal{B}})^\top {A^{-1}}^\top M A^{-1} (\mathcal{B} - P\bar{\mathcal{B}}), \end{aligned} \tag{3.31}$$

where the second equality is obtained using $AP = P\bar{A}$ which implies $P\bar{A}^{-1} = A^{-1}P$. The next theorem shows that the function (3.30) is indeed a Lyapunov function that satisfies the conditions of Definition 3.3.1 but for the dynamical equations of $(X_t - P\bar{X}_t)$.

Theorem 3.3.2 *Consider dynamical systems* (3.24) *and* (3.25) *with invertible matrix A, and let P be a generalized projection satisfying $AP = P\bar{A}$. If there exist matrix M and constant $\kappa > 0$ satisfying the following set of matrix inequalities:*

$$\begin{cases} M \succ 0 \\ C^\top C \preceq M \\ MA + A^\top M + 2\kappa M \preceq 0, \end{cases} \tag{3.32}$$

then we have $\|\mathrm{Prob}^{\mathcal{C}}(C, t) - \mathrm{Prob}^{\bar{\mathcal{C}}}(\bar{C}, t)\|_2 \leq \varepsilon_t$, for all $t \geq 0$, with

$$\varepsilon_t = \xi \|\Gamma\|_2 e^{-\kappa t}, \tag{3.33}$$

where $\Gamma := \mathcal{B} - P\bar{\mathcal{B}}$ is the mismatch induced by the generalized membership functions and $\xi^2 := \lambda_{max}({A^{-1}}^\top M A^{-1})$.

The error in (3.33) is exponentially decaying with decay factor κ and increases linearly with mismatch Γ. A different version of the result, is proved in Appendix A.3.1.

Proof *With the abuse of notation, let us denote $V(X_t, \bar{X}_t)$ under the dynamics of X_t and $\bar{X}_t$ in* (3.24) *and* (3.25) *also by V_t:*

$$V_t := V(X_t, \bar{X}_t), \quad \forall t \geq 0.$$

We assume the argument of V can be inferred from the context, which is either a time instance t or the pair $(X, \bar{X})$. We compute derivative of V_t with respect to time:

$$\begin{aligned}\frac{d}{dt}V_t = \frac{d}{dt}V(X_t, \bar{X}_t) &= \frac{dV(X,\bar{X})}{d(X-P\bar{X})}\frac{d(X-P\bar{X})}{dt} \\ &\frac{d(X-P\bar{X})^\top}{dt}M(X-P\bar{X}) + (X-P\bar{X})^\top M\frac{d(X-P\bar{X})}{dt} \\ &= X^\top MAX + X^\top A^\top MX - X^\top MP\bar{A}\bar{X} \\ &- X^\top A^\top MP\bar{X} - \bar{X}^\top \bar{A}^\top P^\top MX - \bar{X}^\top P^\top MAX \\ &+ \bar{X}^\top P^\top MP\bar{A}\bar{X} + \bar{X}^\top \bar{A}^\top P^\top MP\bar{X}.\end{aligned}$$

Because of equality $AP = P\bar{A}$, we can factorize $\frac{d}{dt}V + 2\kappa V$ as

$$\frac{d}{dt}V + 2\kappa V = [X^\top \bar{X}^\top]\begin{bmatrix}K_{11} & K_{12}\\ K_{21} & K_{22}\end{bmatrix}\begin{bmatrix}X\\ \bar{X}\end{bmatrix}, \tag{3.34}$$

where

$$K_{11} = MA + A^\top M + 2\kappa M \tag{3.35}$$
$$K_{12} = K_{21}^\top = -MP\bar{A} - A^\top MP - 2\kappa MP \tag{3.36}$$
$$K_{22} = P^\top MP\bar{A} + \bar{A}^\top P^\top MP + 2\kappa P^\top MP. \tag{3.37}$$

We can decompose the weight matrix in (3.34) *as*

$$\begin{bmatrix}K_{11} & K_{12}\\ K_{21} & K_{22}\end{bmatrix} = \begin{bmatrix}K_{11} & -K_{11}P\\ -P^\top K_{11}^\top & P^\top K_{11}P\end{bmatrix} = \begin{bmatrix}\mathbb{I}\\ -P^\top\end{bmatrix} K_{11}\begin{bmatrix}\mathbb{I} & -P\end{bmatrix}.$$

Recall from inequalities of (3.32) *that K_{11} satisfies $K_{11} = MA + A^\top M + 2\kappa M \preceq 0$, which implies $\frac{d}{dt}V + 2\kappa V \leq 0$. This inequality guarantees that $V_t \leq V_0 e^{-2\kappa t}$. Note that since $V_t = V(X_t, \bar{X}_t)$ is a quadratic function of $X_t - P\bar{X}_t$, the inequality $V_t \leq V_0 e^{-2\kappa t}$ means $X_t - P\bar{X}_t$ will go to zero exponentially in time with decaying factor κ. To get a precise upper bound on error between outputs of the two systems, we first bound V_0. Notice that V_0 is obtained in* (3.31)*, which satisfies*

$$V_0 = V(X_0, \bar{X}_0) = \Gamma^\top ({A^{-1}}^\top M A^{-1})\Gamma \leq \lambda_{max}({A^{-1}}^\top M A^{-1})\|\Gamma\|_2^2.$$

The inequality holds since M is positive definite which makes ${A^{-1}}^\top M A^{-1}$ also positive

definite. Now recall $\bar{C} := CP$ *and write*

$$\begin{aligned}
\|\mathrm{Prob}^{\mathcal{C}}(C,t) - \mathrm{Prob}^{\bar{\mathcal{C}}}(\bar{C}_S,t)\|_2 &= \|CX_t - \bar{C}X_t\|_2 \\
&= \|C_S(X_t - P\bar{X}_t)\|_2 && \textit{(using } \bar{C} := CP\textit{)} \\
&= \left[(X_t - P\bar{X}_t)^\top C^\top C(X_t - P\bar{X}_t)\right]^{1/2} && \textit{(using equality } \|Y\|_2 = [Y^\top Y]^{1/2}\textit{)} \\
&\le \left[(X_t - P\bar{X}_t)^\top M(X_t - P\bar{X}_t)\right]^{1/2} && \textit{(using } C^\top C \preceq M\textit{)} \\
&= V_t^{1/2} && \textit{(using definition of } V_t \\
&\le V_0^{1/2} e^{-\kappa t} && \textit{(using the exponential bound on } V_t\textit{)} \\
&\le \lambda_{max}(A^{-1^\top} M A^{-1})^{1/2} \|\Gamma\|_2 e^{-\kappa t} && \textit{(using the bound on } V_0\textit{)} \\
&= \xi\|\Gamma\|_2 e^{-\kappa t} = \varepsilon_t && \textit{(using definitions of } \xi \textit{ and } \varepsilon_t\textit{).}
\end{aligned}$$

This completes the proof.

Matrix inequalities (3.32) in Theorem 3.3.2 are bilinear in terms of unknowns (entries of M and constant κ) due to the multiplication between κ and M, thus are difficult to solve. Under Assumption 2, there exists M and κ such that (3.32) is satisfied. In the following we show how to obtain a solution efficiently when A is stable.

Theorem 3.3.3 *Assumption 2 implies that matrix* A *has a simple eigenvalue equal to* $\bar{\rho} := \max_i \mathrm{Re}(\lambda_i(A))$ *and its left eigenvector* ν *can be selected such that all its entries are strictly positive. A feasible solution of* (3.32) *can be selected by letting* κ *be any positive constant*

$$\kappa \le -\frac{1}{2}\bar{\rho} = -\frac{1}{2}\max_i \mathrm{Re}(\lambda_i(A)), \tag{3.38}$$

and choosing the diagonal matrix $M = diag(\nu)$ *with entries of* ν *normalized to have them greater or equal to one.*

Proof *The matrix* $H = \frac{A}{\gamma} + \mathbb{I}_m$ *is sub-stochastic and irreducible. According to Perron-Frobenius theorem,* H *has a simple real eigenvalue* ρ*, which is its largest eigenvalue in absolute sense, and its associated left eigenvector* ν *having strictly positive entries. Without loss of generality, we assume that* ν *is normalized such that it has all entries greater or equal to one. We also proved in Proposition 3.3.1 that* $\rho < 1$*. The definition of* H *implies that* $\lambda_i(H) = \lambda_i(A)/\gamma + 1$*. Thus we get* $\bar{\rho} := \max_i \mathrm{Re}(\lambda_i(A)) = -\gamma(1-\rho)$ *is a simple eigenvalue of* A *with the same left eigenvector* ν*.*
Matrix $(A + \kappa\mathbb{I}_m)$ *has exactly the same eigenvalues as that of* A *but increased by* κ*. Selecting* $\kappa < -\bar{\rho}$ *implies* $(A + \kappa\mathbb{I}_m)$ *still has all its eigenvalues with negative real parts. Therefore,* $(A + \kappa\mathbb{I}_m)$ *is stable and according to Theorem 3.3.1, there is a matrix* M *satisfying* $(A^\top + \kappa\mathbb{I}_m)M + M(A + \kappa\mathbb{I}_m) \prec 0$*, which means* $A^\top M + MA + 2\kappa M \preceq 0$*.*
We show that $M = diag(\nu)$ *is a solution for this inequality. Denote by* $\mathbf{1}_m$ *the column vector of dimension* m *with all entries equal to one. We have*

$$\begin{aligned}
&(A^\top + 2\kappa\mathbb{I}_m)M\mathbf{1}_m = (A^\top + 2\kappa\mathbb{I}_m)\nu^\top = (\bar{\rho} + 2\kappa)\nu^\top, \\
&MA\mathbf{1}_m = M(A\mathbf{1}_m) = (\nu^\top)\cdot(A\mathbf{1}_m) \quad \textit{(entry-wise product of } \nu^\top \textit{ and } A\mathbf{1}_m\textit{).}
\end{aligned}$$

Since $\nu^\top$ has positive entries, $(\bar{\rho} + 2\kappa) \leq 0$, and $(A\mathbf{1}_m)$ has non-positive entries, we have that both matrices $(A^\top + 2\kappa\mathbb{I}_m)M$ and MA are right sub-stochastic satisfying Assumption 2. Therefore, $(A^\top + 2\kappa\mathbb{I}_m)M + MA$ is symmetric and stable, its eigenvalues will be negative, thus it is semi-definite negative. This concludes the proof.

Next, we show that for a given $r \leq m$, we can find a suitable $\bar{A}$ and P such that $AP = P\bar{A}$.

Theorem 3.3.4 *Given the matrix $A \in \mathbb{R}^{n\times n}$, for each $r \leq m$, there is a $m \times r$ matrix P and an $r \times r$ matrix $\bar{A}$, computable in polynomial time in m, such that $AP = P\bar{A}$.*

Proof *Every matrix A can be decomposed as*

$$A = UNU^{-1}, \tag{3.39}$$

in which N is an upper triangular matrix, called the Schur form *of A, and U is a unitary matrix [77]. Schur decomposition of A can be performed iteratively with $\mathcal{O}(m^3)$ arithmetic operations using QR decomposition [49]. We choose $\bar{A}$ as the first r rows and columns of N and P as first r columns of U. Since N is upper triangular, the equality $AP = P\bar{A}$ holds for this choice of $\bar{A}$ and P.*

Once κ is fixed, constraints (3.32) become matrix inequalities that are linear in terms of entries of M and can be solved using convex optimization [62] and developed tools for linear matrix inequalities [69, 111]. In particular, the diagonal matrix M defined in Theorem 3.3.3 is a feasible solution to the matrix inequalities. However, when C is not full rank, which is the case when $S_0 \neq S$, solving the matrix inequalities for M can result in better error bounds.

Notice that $V_0 = (X_0 - P\bar{X}_0)^\top M(X_0 - P\bar{X}_0)$ and using (3.24), we have $X_0 = A^{-1}\mathcal{B}$. Therefore, it is important to find $\bar{X}_0$ that results in the least V_0. We can compute $\bar{X}_0$ by minimizing V_0:

$$\min_{\bar{X}_0} \left[X_0 - P\bar{X}_0\right]^\top M \left[X_0 - P\bar{X}_0\right], \tag{3.40}$$

which is a weighted least square optimization and has the closed-form solution

$$\bar{X}_0 = (P^\top MP)^{-1}P^\top M(A^{-1}\mathcal{B}). \tag{3.41}$$

Choosing this initial state $\bar{X}_0$ will provide a tighter initial error bound. Knowing $\bar{A}$ and $\bar{X}_0$, one can find $\bar{\mathcal{B}} = \bar{A}\bar{X}_0$.

Theorems 3.3.3-3.3.4 give an algorithm, shown in Algorithm 8, to find lower dimensional approximations to the dynamical system (3.24), and Theorem 3.3.2 provides a quantitative error bound for the approximation. The procedure is summarized in Algorithm 8. Given a time-bounded reachability problem and an error bound ε, we iteratively compute reduced order dynamical systems of dimension $r = 1, \ldots, n-1$ using Theorems 3.3.3-3.3.4. Then, we check if the error bound in Theorem 3.3.2 is at most ε. If so, we solve the dynamical system of dimension r (using, e.g., exponential of an upper-triangular matrix) to compute an ε-approximation to the time-bounded reachability problem. If not, we increase r and search again.

Example 3 *Consider the CTMC in Figure 3.1 with* $\Lambda_{31} = 1$, $\Lambda_{42} = 2$ *and* $\varepsilon_{ij} = 0$ *for all* i, j *(the CTMC is unperturbed). The generator matrix for the CTMC is*

$$\mathbf{Q} = \begin{bmatrix} -4 & 0 & 2 & 0 & 2 \\ 0 & -4 & 1 & 1 & 2 \\ 1 & 1 & -2 & 0 & 0 \\ 0 & 2 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

As in Example 3.1, we are interested in computing the probability of reaching the state **good**. *Using the partition defined in Eq.* (3.19)*, we get*

$$A = \begin{bmatrix} -4 & 0 & 2 & 0 \\ 0 & -4 & 1 & 1 \\ 1 & 1 & -2 & 0 \\ 0 & 2 & 0 & -2 \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} 2 \\ 2 \\ 0 \\ 0 \end{bmatrix}.$$

Note that A *is reducible with* $\bar{\rho} = -0.7639$. *All the values are reported by rounding to 4 decimal digits. We select the decay rate* $\kappa = 0.3820$ *using Eq.* (3.38). *Then we compute* U *and* N *based on the Schur decomposition of* A*:*

$$N = \begin{bmatrix} -5.2361 & 0 & 0.1602 & -0.9871 \\ 0 & -0.7639 & -0.9871 & -0.1602 \\ 0 & 0 & -4.4142 & 0 \\ 0 & 0 & 0 & -1.5858 \end{bmatrix}, \quad U = \begin{bmatrix} 0.6015 & 0.3717 & 0.6533 & -0.2706 \\ 0.6015 & 0.3717 & -0.6533 & 0.2706 \\ -0.3717 & 0.6015 & -0.2706 & -0.6533 \\ -0.3717 & 0.6015 & 0.2706 & 0.6533 \end{bmatrix}.$$

Using Theorem 3.3.3 we find matrix M *as*

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3.2361 & 0 \\ 0 & 0 & 0 & 1.6180 \end{bmatrix}.$$

Selecting the order $r = 2$*, we find* $\bar{A}$ *as the first* (2×2) *block of* N *and* P *the first* 2 *columns of* U*:*

$$\bar{A} = \begin{bmatrix} -5.2361 & 0 \\ 0 & -0.7639 \end{bmatrix}, \quad P = \begin{bmatrix} 0.6015 & 0.3717 \\ 0.6015 & 0.3717 \\ -0.3717 & 0.6015 \\ -0.3717 & 0.6015 \end{bmatrix}.$$

Using (3.41)*, we compute the initial state of the reduced-order system as*

$$\bar{X}_0 = \begin{bmatrix} 0.4595 \\ 1.9465 \end{bmatrix}.$$

The above selection results in $\varepsilon_B = 0$ *for any arbitrary time bound* T. *Therefore, the order of the set of differential equations that we need to solve reduces from four into two without*

incurring any error. In this case, our approach retrieves the reduction originating from the exact bisimulation.
We now consider a perturbed version of the CTMC with the generator matrix

$$\mathbf{Q} = \begin{bmatrix} -3.95 & 0 & 1.95 & 0 & 2 \\ 0 & -4.05 & 1.05 & 1 & 2 \\ 1 & 1 & -2 & 0 & 0 \\ 0 & 2 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.42}$$

By performing the same computations as above, we find

$$\kappa = 0.3730 \qquad M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1.9047 & 0 & 0 \\ 0 & 0 & 3.1887 & 0 \\ 0 & 0 & 0 & 1.5376 \end{bmatrix},$$

and

$$\bar{A} = \begin{bmatrix} -5.2580 & -0.0770 \\ 0 & -0.7613 \end{bmatrix}, \quad P = \begin{bmatrix} 0.5436 & 0.3753 \\ 0.6443 & 0.3864 \\ -0.3646 & 0.5922 \\ -0.3955 & 0.5993 \end{bmatrix}, \quad \bar{X}_0 = \begin{bmatrix} 0.4165 \\ 1.9454 \end{bmatrix}.$$

For example, we have $\varepsilon_B = 0.0008e^{-0.3730T}$ *according to Theorem 3.3.2, which is* 0.0005 *for time bound* $B = 1$.

Symbolic Computation on the Reduced Model

Based on the construction of $\bar{A}$ of the reduced system according to the Schur form (3.39), matrix $\bar{A}$ is upper-triangular as

$$\bar{A} = \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} & \bar{A}_{13} & \cdots & \bar{A}_{(1)(r-1)} & \bar{A}_{1r} \\ 0 & \bar{A}_{22} & \bar{A}_{23} & \cdots & \bar{A}_{(2)(r-1)} & \bar{A}_{2r} \\ 0 & 0 & \bar{A}_{33} & \cdots & \bar{A}_{(3)(r-1)} & \bar{A}_{3r} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & \bar{A}_{(r-1)(r-1)} & \bar{A}_{(r-1)(r)} \\ 0 & 0 & \cdots & \cdots & 0 & \bar{A}_{rr} \end{bmatrix}.$$

This property of $\bar{A}$ can be exploited to make the computation of reachability probability more efficient. In fact, solution of the differential equation $\dot{\bar{X}}_t = \bar{A}\bar{X}_t$ in (3.25) can be written as $\bar{X}_t = e^{\bar{A}t}\bar{X}_0$. Let us first assume all diagonal elements of $\bar{A}$ are distinct. Denote the i^{th} element of $\bar{X}_t$ by $\bar{X}_t(i)$. The last element of $\bar{X}_t$ can be easily computed as

$$\dot{\bar{X}}^r_t = \bar{A}_{rr}\bar{X}^r_t \Rightarrow \bar{X}^r_t = e^{\bar{A}_{rr}t}\bar{X}^r_0.$$

Algorithm 8: Order reduction of CTMCs

Input: CTMC $\mathcal{C} = (S_{\mathcal{C}}, \mathbf{Q})$, time bound B, maximum error bound ε

1. Compute A, $\boldsymbol{\beta}$ and κ, based on (3.19) and (3.38)
2. Compute M using Theorem 3.3.3
3. Compute the Schur decomposition of A and save the matrices U and N using (3.39)
4. $r \leftarrow 0$
5. **Do**
   $r \leftarrow r + 1$
   Set $\bar{A}$ as the first r rows and columns of N
   Set P as first r columns of U
   Compute $\bar{X}_0$ according to (3.41)
   Compute error bound ε_r using (3.33) for time bound B and $\bar{\mathcal{B}} = \bar{A}\bar{X}_0$
   While $(\varepsilon_r > \varepsilon)$

Result: Reduced-order system (3.25)

In general, it is possible to perform the computations bottom-up. Once we solve the equations for $\bar{X}_t^r, \bar{X}_t^{r-1}, \ldots, \bar{X}_t^{i+1}$, we use their explicit form to solve the differential equation for $\bar{X}_t^i$. This gives the solution in closed-form as

$$\bar{X}_t^i = \sum_{j=i}^{r} \alpha_{ij} e^{\bar{A}_{jj} t}, \tag{3.43}$$

where

$$\alpha_{ij} = \begin{cases} \sum_{k=i+1}^{j} \frac{\bar{A}_{ik}\alpha_{kj}}{-\bar{A}_{ii}+\bar{A}_{jj}} & \text{for } \; j > i, \\ -\sum_{j=i+1}^{r} \alpha_{ij} + \bar{X}_0^i & \text{for } \; j = i. \end{cases}$$

This closed-form solution can be verified inductively. Note that the computation of α_{ij} is performed sequentially and backward with respect to the index i. To make these computations clear, let us define the matrix $\boldsymbol{\alpha} := [\alpha_{ij}]_{i,j}$, which is upper triangular. The last row of this matrix has one non-zero element, which is simply $\alpha_{rr} = \bar{X}_0^r$. The computation of the i^{th} row of $\boldsymbol{\alpha}$ is performed as follows. The non-diagonal elements in the i^{th} row will need the entries from previously computed rows which are the $(i+1)^{st}, (i+2)^{nd}, \ldots, r^{th}$ rows. The diagonal element in the i^{th} row needs its non-diagonal elements.

For the case that $\bar{A}$ has eigenvalues with multiplicities $\mathsf{m} > 1$, the closed-form solution (3.43) becomes a linear combination of functions $t^l e^{\bar{A}_{ii} t}$ for $0 \leq l \leq \mathsf{m} - 1$, and the coefficients can be computed in a similar way. The details of such computations can be found in general text books on control theory, e.g., [138].

Example 4 *Let us consider the CTMC with the generator matrix given in* (3.42)*. The reduced system for this CTMC was computed in Example 3. We use our symbolic computation method described above to find the solution to the time-bounded reachability problem. Based on Eq.* (3.43)*, the closed form solution to the time-bounded reachability problem over the reduced system with time bound B will be*

$$\bar{X}_1(B) = -0.0332e^{-0.7613T} + 0.4498e^{-5.2580T}$$
$$\bar{X}_2(B) = 1.9454e^{-0.7613T}.$$

3.3.2. Time-Bounded Reachability on CTMDPs

Let $\mathcal{M} = (S \uplus \{\mathbf{good}, \mathbf{bad}\}, \mathcal{D}, \mathbf{Q})$ be a CTMDP with two absorbing states **good** and **bad**, where $|S| = n$, and let $B \in \mathbb{R}_{\geq 0}$ be a time bound. Define the event

$$\mathcal{RA} := \cup\{f \in \Omega \mid \exists t \in [0, B] \text{ s.t. } f_t = \mathbf{good} \text{ and } f_{t'} \neq \mathbf{bad} \text{ for all } t' \in [0, t)\}. \quad (3.44)$$

Let $S_0 \subseteq S$ with $|S_0| = n_0$ be a set of desired initial states. For a fixed timed positional policy $\pi \in \Pi_B$ and an initial state $s \in S_0$, we define

$$\text{Prob}^{\mathcal{M}(\pi)}(\mathbf{1}(s), B) = \mathbf{P}^{\pi}_{s}(\mathcal{RA}).$$

Furthermore, let C be an $n_0 \times (n + 2)$ matrix with n_0 ones on its main diagonal, corresponding to the states in S_0. We denote $\text{Prob}^{\mathcal{M}(\pi)}(C, B)$ as the n_0-dimensional vector containing all the values of $\text{Prob}^{\mathcal{M}(\pi)}(\mathbf{1}(s), B)$ for $s \in S_0$. If $S_0 = S$, then C is the $(n + 2) \times (n + 2)$ identity matrix. Intuitively, for $s \in S_0$, the value $\text{Prob}^{\mathcal{M}(\pi)}(\mathbf{1}(s), B)(s)$ corresponds to the probability with which the **good** state is visited within the time bound $[0, B]$, while the **bad** state is avoided, starting from the initial distribution $\alpha = \mathbf{1}(s)$ and under the policy π.

For an initial set of states S_0, a rational vector $r \in [0, 1]^{n_0}$ and a time bound $B > 0$, we are interested in synthesizing a policy $\pi \in \Pi_B$ such that:

$$\text{Prob}^{\mathcal{M}(\pi)}(\mathbf{1}(s), B) \geq r(s), \quad (3.45)$$

for all $s \in S_0$. Synthesizing such a policy can be done by maximizing the left-hand side of (3.45) on the set of policies and then comparing the optimal value with r. Characterization of the optimal policy is performed as follows [32]. We partition any generator matrix $\mathbf{Q}^{\mathbf{d}}$ corresponding to decision vector $\mathbf{d} \in \mathcal{D}$, as

$$\mathbf{Q}^{\mathbf{d}} = \begin{bmatrix} A^{\mathbf{d}} & \vdots & \boldsymbol{\chi}^{\mathbf{d}} & \vdots & \boldsymbol{\beta}^{\mathbf{d}} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{0} & \vdots & \mathbf{0} & \vdots & \mathbf{0} \end{bmatrix} \quad (3.46)$$

with $A^{\mathbf{d}} \in \mathbb{R}^{n \times n}$, $\boldsymbol{\chi}^{\mathbf{d}} \in \mathbb{R}^{n \times 1}$, and $\boldsymbol{\beta}^{\mathbf{d}} \in \mathbb{R}^{n \times 1}$. Then for a CTMDP $\mathcal{M}$ with matrix C indicating a subset of initial states $S_0 \subseteq S$ for which we would like to satisfy (3.45),

$\max_\pi \text{Prob}^{\mathcal{M}(\pi)}(C, B)$ can be characterized backward in time as the solution of the following set of nonlinear differential equations

$$\begin{aligned} &\frac{d}{dt}W_t = \max_{\mathbf{d}_t \in \mathcal{D}} \mathbf{Q}^{\mathbf{d}_t} W_t, \quad W(0) = \mathbf{1}(\mathbf{good}), \\ &\max_\pi \text{Prob}^{\mathcal{M}(\pi)}(C, B) = CW_t, \end{aligned} \tag{3.47}$$

where W_t is a column vector containing probabilities $\max_\pi \text{Prob}^{\mathcal{M}(\pi)}(\mathbf{1}(s), B)$ as a function of initial state s.

With respect to the partitioning (3.46), it is obvious that in (3.47), $W_t(\mathbf{bad}) = 0$ and $W_t(\mathbf{good}) = 1$ for all $t \in \mathbb{R}_{\geq 0}$. The remaining state variables W_t should satisfy

$$\begin{aligned} &\frac{d}{dt}W_t = \max_{\mathbf{d}_t \in \mathcal{D}} (A^{\mathbf{d}_t} W_t + \boldsymbol{\beta}^{\mathbf{d}_t}), \quad W(0) = 0, \\ &\max_\pi Prob^{\mathcal{M}(\pi)}(C, t) = CW_t. \end{aligned} \tag{3.48}$$

The optimal policy is the one maximizing the right-hand side of differential equation in (3.48), and thus it is time-dependent and is only a function of state of the CTMDP at time t. In [144], it is shown that the policy that maximizes time-bounded reachability probability of CTMDPs contains only finitely many switches. However, finding the optimal policy is computationally expensive for CTMDPs with large number of states. The current state of the art solutions are based on breaking the time interval $[0, B]$ into smaller intervals of length δ, and then computing (approximate) optimal decisions in each interval of length $\frac{T}{\delta}$ sequentially (see [61, 33]). Thus, a set of linear differential equations must be solved in each interval, which is computationally expensive.

In the following, we will develop a new way of synthesizing a policy that satisfies (3.45) by approximating the solution of (3.48) via generalized projections and reductions. We treat (3.48) as a *switched affine system* [68]. We are given a collection of $|\mathcal{D}|$ affine dynamical systems, characterized by the pairs $(A^{\mathbf{d}}, \boldsymbol{\beta}^{\mathbf{d}})$, and the role of any policy $\pi = \{\mathbf{d}_t \in \mathcal{D},\, t \geq 0\}$ is to switch from one dynamical system to another by picking a different pair. The main underlying idea of our approximate computation is to consider the reduced order version of these dynamical systems and find a switching policy π. We provide guarantees on the closeness to the exact reachability probability when this policy is applied to the original CTMDP. For this we require the following assumption.

Assumption 3 *Matrices* $\{A^{\mathbf{d}},\, \mathbf{d} \in \mathcal{D}\}$ *are all stable.*

Note that this assumption is satisfied if for each choice of actions, the resulting CTMC is irreducible (Prop. 3.3.1) and the time-bounded reachability problem does not have a trivial solution.

Under Assumption 3, we can find matrix $M^{\mathbf{d}}$ and constant $\kappa^{\mathbf{d}} > 0$, for any $\mathbf{d} \in \mathcal{D}$, such that the following matrix inequalities hold:

$$\begin{cases} M^{\mathbf{d}} \succ 0 \\ C^\top C \preceq M^{\mathbf{d}} \\ M^{\mathbf{d}} A^{\mathbf{d}} + [A^{\mathbf{d}}]^\top M^{\mathbf{d}} + 2\kappa^{\mathbf{d}} M^{\mathbf{d}} \preceq 0, \end{cases} \tag{3.49}$$

We need the following lemma that gives us a bound on the solution of reduced order systems.

Lemma 3.3.1 *Suppose generalized projections $P^{\mathbf{d}}$ and matrices $\bar{A}^{\mathbf{d}}$ satisfy $A^{\mathbf{d}}P^{\mathbf{d}} = P^{\mathbf{d}}\bar{A}^{\mathbf{d}}$ for any $\mathbf{d} \in \mathcal{D}$. Then $V(\bar{X}^{\mathbf{d}}) = [\bar{X}^{\mathbf{d}}]^\top \bar{M}^{\mathbf{d}} \bar{X}^{\mathbf{d}}$ with $\bar{M}^{\mathbf{d}} = [P^{\mathbf{d}}]^\top M^{\mathbf{d}} P^{\mathbf{d}}$ and $M^{\mathbf{d}}$ satisfying* (3.49)*, is a Lyapunov function for $d\bar{X}_t/dt = \bar{A}^{\mathbf{d}}\bar{X}_t^{\mathbf{d}}$ for each $\mathbf{d} \in \mathcal{D}$. Moreover,*

$$\|\bar{X}_{t_1}^{\mathbf{d}}\|_{\bar{M}^{\mathbf{d}}} \leq \|\bar{X}_{t_0}^{\mathbf{d}}\|_{\bar{M}^{\mathbf{d}}} e^{-\kappa^{\mathbf{d}}(t_1-t_0)}, \quad \forall t_1 \geq t_0, \tag{3.50}$$

where $\|Y\|_G := \sqrt{Y^\top G Y}$ is the weighted two-norm of a vector Y.

Proof *We prove* (3.50) *via a bound on the Lyapunov function $V(\bar{X}^{\mathbf{d}})$:*

$$\begin{aligned}
\frac{d}{dt}V(\bar{X}^{\mathbf{d}}) &= (\bar{A}^{\mathbf{d}}\bar{X}^{\mathbf{d}})^\top \bar{M}^{\mathbf{d}}\bar{X}^{\mathbf{d}} + [\bar{X}^{\mathbf{d}}]^\top \bar{M}^{\mathbf{d}}(\bar{A}^{\mathbf{d}}\bar{X}^{\mathbf{d}}) \\
&= [\bar{X}^{\mathbf{d}}]^\top([\bar{A}^{\mathbf{d}}]^\top \bar{M}^{\mathbf{d}} + \bar{M}^{\mathbf{d}}\bar{A}^{\mathbf{d}})\bar{X}^{\mathbf{d}} \\
&= [\bar{X}^{\mathbf{d}}]^\top([\bar{A}^{\mathbf{d}}]^\top [P^{\mathbf{d}}]^\top M^{\mathbf{d}} P^{\mathbf{d}} + [P^{\mathbf{d}}]^\top M^{\mathbf{d}} P^{\mathbf{d}}\bar{A}^{\mathbf{d}})\bar{X}^{\mathbf{d}} \\
&= [\bar{X}^{\mathbf{d}}]^\top([P^{\mathbf{d}}]^\top [A^{\mathbf{d}}]^\top M^{\mathbf{d}} P^{\mathbf{d}} + [P^{\mathbf{d}}]^\top M^{\mathbf{d}} A^{\mathbf{d}} P^{\mathbf{d}})\bar{X}^{\mathbf{d}} \\
&= [\bar{X}^{\mathbf{d}}]^\top [P^{\mathbf{d}}]^\top([A^{\mathbf{d}}]^\top M^{\mathbf{d}} + M^{\mathbf{d}} A^{\mathbf{d}})P^{\mathbf{d}}\bar{X}^{\mathbf{d}} \\
&\leq -2\kappa^{\mathbf{d}}[\bar{X}^{\mathbf{d}}]^\top [P^{\mathbf{d}}]^\top M^{\mathbf{d}} P^{\mathbf{d}}\bar{X}^{\mathbf{d}} = -2\kappa^{\mathbf{d}} V(\bar{X}^{\mathbf{d}}),
\end{aligned}$$

thus $V(\bar{X}_t^{\mathbf{d}}) \leq V(\bar{X}_{t_0}^{\mathbf{d}})e^{-2\kappa^{\mathbf{d}}_{(t-t_0)}}$, for all $t \geq t_0$, which gives (3.50).

Consider an arbitrary time-dependent Markov policy $\pi = \{\mathbf{d}_t \in \mathcal{D},\, t \geq 0\}$. Then there is a sequence of decision vectors $(\mathbf{d}^1, \mathbf{d}^2, \mathbf{d}^3, \ldots)$ with switching times $(t_1, t_2, t_2, \ldots)$ such that actions in $\mathbf{d}^i$ are selected over time interval $[t_{i-1}, t_i)$ depending on the state of $\mathcal{M}$, for any $i = 1, 2, \ldots$ with $t_0 = 0$. We first study time-bounded reachability for $\mathcal{M}$ under policy π, which can be characterized as the switched system:

$$\frac{d}{dt}W_t = A^{\mathbf{d}^i}W_t + \boldsymbol{\beta}^{\mathbf{d}^i},\ \forall t \in [t_{i-1}, t_i),\ i = 1, 2, \ldots \tag{3.51}$$

Similar to our discussion on CTMC, we prefer to move constant inputs $\boldsymbol{\beta}^{\mathbf{d}^i}$ in (3.51) into initial states. Therefore, we define the following piecewise translation

$$X_t := W_t + [A^{\mathbf{d}^i}]^{-1}\boldsymbol{\beta}^{\mathbf{d}^i},\ \forall t \in [t_{i-1}, t_i),\ i = 1, 2, \ldots \tag{3.52}$$

that depends also on π. Note that $[A^{\mathbf{d}^i}]^{-1}\boldsymbol{\beta}^{\mathbf{d}^i}$ is exactly the solution of the unbounded reachability probability (steady state solution of (3.51) when matrix $A^{\mathbf{d}^i}$ is selected for all time instances). Thus the evolution of X_t becomes

$$\frac{d}{dt}X_t = A^{\mathbf{d}^i}X_t,\ \forall t \in [t_{i-1}, t_i),\ i = 1, 2, \ldots, \tag{3.53}$$

with state X_t having jumps at switching time instances t_i that are equal to

$$\Delta X_{t_i} := X_{t_i} - X_{t_i^-} = [A^{\mathbf{d}^{i+1}}]^{-1}\boldsymbol{\beta}^{\mathbf{d}^{i+1}} - [A^{\mathbf{d}^i}]^{-1}\boldsymbol{\beta}^{\mathbf{d}^i}, \tag{3.54}$$

where $X_{t_i^-}$ denotes the left-sided limit of X_t at t_i, i.e., $X_{t_i^-} := \lim_{t \uparrow t_i} X_t$. The quantity ΔX_{t_i} is exactly the difference between unbounded reachability probability if one of the decision vectors $\mathbf{d}^i$ and $\mathbf{d}^{i+1}$ is taken independent of time. Similarly, we define

$$\Delta^{ij} := [A^{\mathbf{d}^j}]^{-1}\boldsymbol{\beta}^{\mathbf{d}^j} - [A^{\mathbf{d}^i}]^{-1}\boldsymbol{\beta}^{\mathbf{d}^i}, \tag{3.55}$$

which will be used later in Theorem 3.3.5. Note that W_t is a continuous function of time no matter what decision vectors $\{\mathbf{d}^1, \mathbf{d}^2, \ldots\}$ are selected, but it converges to different steady state vectors depending on the chosen decision vectors. On the other hand, when we change the variables to X_t using the affine transformation (3.52), X_t becomes a discontinuous function of time, with discontinuity at time instances t_i and jumps equal to ΔX_{t_i} defined in (3.54), but it will always converge to zero independent of the chosen decision vectors $\{\mathbf{d}^1, \mathbf{d}^2, \ldots\}$.

Now we construct the reduced order switched system

$$\frac{d}{dt}\bar{X}_t = \bar{A}^{\mathbf{d}^i}\bar{X}_t, \ \forall t \in [t_{i-1}, t_i), \ i = 1, 2, \ldots, \tag{3.56}$$

with $\bar{A}^{\mathbf{d}}$ satisfying $A^{\mathbf{d}}P^{\mathbf{d}} = P^{\mathbf{d}}\bar{A}^{\mathbf{d}}$ for all $\mathbf{d} \in \mathcal{D}$. We choose the values of jumps $\Delta\bar{X}_{t_i} := \bar{X}_{t_i} - \bar{X}_{t_i^-}$ so that the behavior of (3.56) is as close as possible to (3.53). For this, we have

$$\bar{X}_{t_i} := \arg\min_{\bar{X}} \left\| \Delta X_{t_i} - P^{\mathbf{d}^{i+1}}\bar{X} + P^{\mathbf{d}^i}\bar{X}_{t_i^-} \right\|_{M^{\mathbf{d}^{i+1}}}, \tag{3.57}$$

which can be computed for any value of $\bar{X}_{t_i^-}$.

Define the *dwell time* of a policy π by $\tau = min_i(t_i - t_{i-1})$, i.e., the minimum time between two consecutive switches of decision vectors in π. The paper [132] shows that for any epsilon-optimal policy there is a bound on the minimum dwell time. The next theorem quantifies the error between the two switched systems using the dwell time of the policy.

Theorem 3.3.5 *Given a CTMDP $\mathcal{M}$, a policy π with dwell time τ, switching time instances $t_0 = 0 \le t_1 \le t_2 \le \cdots$, and bounded-time reachability over $[0, B]$. Suppose there exist $M^{\mathbf{d}^i}, \kappa^{\mathbf{d}^i}$ satisfying (3.49), constant μ satisfying $M^{\mathbf{d}^i} \preceq \mu M^{\mathbf{d}^j}$ for all $\mathbf{d}^i, \mathbf{d}^j \in \mathcal{D}$, and matrices $\bar{A}^{\mathbf{d}^i}, P^{\mathbf{d}^i}$ such that $A^{\mathbf{d}^i}P^{\mathbf{d}^i} = P^{\mathbf{d}^i}\bar{A}^{\mathbf{d}^i}$. Then we have*

$$\|X_B - P^{\mathbf{d}^{n+1}}\bar{X}_B\|_{M^{\mathbf{d}^{n+1}}} \le \varepsilon_n e^{-\kappa(B-t_n)}, \tag{3.58}$$

where t_n is the last switching time instance before the time bound B and $\kappa := \min_d \kappa^{\mathbf{d}}$ is the minimum decay rate. The quantity ε_n is obtained from the difference equations

$$\begin{aligned} \bar{\varepsilon}_i &= \mu g \bar{\varepsilon}_{i-1} + \Delta_{max} \\ \varepsilon_i &= \mu g \varepsilon_{i-1} + 2\mu g \bar{\varepsilon}_{i-1} + 2\Delta_{max}, \quad i \in \{1, 2, \ldots\}, \end{aligned} \tag{3.59}$$

where $g := e^{-\kappa\tau}$ $\Delta_{max} := \max_{i,j} \|\Delta^{ij}\|_{M_j}$ with Δ^{ij} defined in (3.55), initial conditions $\varepsilon_0 := \|[A^{\mathbf{d}^0}]^{-1}\boldsymbol{\beta}^{\mathbf{d}^0} - P^{\mathbf{d}^0}\bar{X}_0\|_{M^{\mathbf{d}^0}}$, and $\bar{\varepsilon}_0 = \|\bar{X}(0)\|_{\bar{M}^{\mathbf{d}^1}}$. The states $\bar{X}_{t_i}$ at switching time instances are reset to a value according to the weighted least square method similar to (3.41).

Proof *We show that the following inequalities hold with $\bar{\varepsilon}_i, \varepsilon_i$ satisfying* (3.59)*:*

$$||\bar{X}_{t_i}||_{\bar{M}^{\mathbf{d}^{i+1}}} \leq \bar{\varepsilon}_i \quad \text{and} \quad ||X_{t_i} - P^{\mathbf{d}^i}\bar{X}_{t_i}||_{M^{\mathbf{d}^{i+1}}} \leq \varepsilon_i.$$

Note that ε_i and $\bar{\varepsilon}_i$ are defined inductively in (3.59) *and depend on each other. ε_i bounds the norm of $X_{t_i} - P^{\mathbf{d}^i}\bar{X}_{t_i}$ weighted by $M^{\mathbf{d}^{i+1}}$ but $\bar{\varepsilon}_i$ bounds the norm of $\bar{X}_{t_i}$ weighted by $\bar{M}^{\mathbf{d}^{i+1}}$. In order to establish the relation between these two quantities inductively, we have to use the appropriate weight and change it using the definition $\bar{M}^{\mathbf{d}} = [P^{\mathbf{d}}]^\top M^{\mathbf{d}} P^{\mathbf{d}}$ whenever necessary.*

At the i^{th} switching time instance, we have $X_{t_i} = X_{t_i^-} + \Delta^{i,i+1}$. By adding and subtracting the term $P^{\mathbf{d}^i}\bar{X}_{t_i^-}$ and noting that $M^{\mathbf{d}^{i+1}} \leq \mu M^{\mathbf{d}^i}$, we can write:

$$\begin{aligned}
&\|X_{t_i^-} + \Delta^{i,i+1} - P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_{M^{\mathbf{d}^{i+1}}} \\
&\quad = \|X_{t_i^-} - P^{\mathbf{d}^i}\bar{X}_{t_i^-} + P^{\mathbf{d}^i}\bar{X}_{t_i^-} + \Delta^{i,i+1} - P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_{M^{\mathbf{d}^{i+1}}} \\
&\quad \leq \|X_{t_i^-} - P^{\mathbf{d}^i}\bar{X}_{t_i^-}\|_{M^{\mathbf{d}^{i+1}}} + \|P^{\mathbf{d}^i}\bar{X}_{t_i^-} + \Delta^{i,i+1} - P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_{M^{\mathbf{d}^{i+1}}} \\
&\quad \leq \mu\|X_{t_i^-} - P^{\mathbf{d}^i}\bar{X}_{t_i^-}\|_{M^{\mathbf{d}^i}} + \|P^{\mathbf{d}^i}\bar{X}_{t_i^-} + \Delta^{i,i+1} - P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_{M^{\mathbf{d}^{i+1}}}. \qquad (3.60)
\end{aligned}$$

For the time interval $[t_{i-1}, t_i)$ we already know that

$$\|X_{t_i^-} - P^{\mathbf{d}^i}\bar{X}_{t_i^-}\|_{M^{\mathbf{d}^i}} \leq \|X_{t_{i-1}} - P^{\mathbf{d}^i}\bar{X}_{t_{i-1}}\|_{M^{\mathbf{d}^i}} e^{-\kappa^{\mathbf{d}^i}(t_i - t_{i-1})} \leq g\varepsilon_{i-1},$$

since the policy has dwell time τ. Now we deal with the second term in (3.60)*. As a consequence of picking columns of $P^{\mathbf{d}^i} \in \mathbb{R}^n \times \mathbb{R}^r$ from the corresponding unitary matrix, one can easily notice that $[P^{\mathbf{d}^i}]^\top P^{\mathbf{d}^i} = \mathbb{I}_r$ and $P^{\mathbf{d}^i}[P^{\mathbf{d}^i}]^\top \leq \mathbb{I}_n$ for every i. Therefore, using the triangle inequality we get*

$$\begin{aligned}
&\|P^{\mathbf{d}^i}\bar{X}_{t_i^-} + \Delta^{i,i+1} - P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_{M^{\mathbf{d}^{i+1}}} \\
&\quad \leq \|P^{\mathbf{d}^i}\bar{X}_{t_i^-}\|_{M^{\mathbf{d}^{i+1}}} + \|\Delta^{i,i+1}\|_{M^{\mathbf{d}^{i+1}}} + \|P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_{M^{\mathbf{d}^{i+1}}} \\
&\quad \leq \mu\|\bar{X}_{t_i^-}\|_{\bar{M}^{\mathbf{d}^i}} + \Delta_{max} + \|\bar{X}_{t_i}\|_{\bar{M}^{\mathbf{d}^{i+1}}}. \qquad (3.61)
\end{aligned}$$

The last inequality is due to $M^{\mathbf{d}^{i+1}} \leq \mu M^{\mathbf{d}^i}$, the definition of Δ_{max} in Theorem 3.3.5, and the definition $\bar{M}^{\mathbf{d}} = [P^{\mathbf{d}}]^\top M^{\mathbf{d}} P^{\mathbf{d}}$ in Lemma 3.3.1. $\bar{X}_{t_i}$ is selected as the minimizer of the expression

$$\|P^{\mathbf{d}^i}\bar{X}_{t_i^-} + \Delta^{i,i+1} - P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_2, \qquad (3.62)$$

which is

$$\bar{X}_{t_i} = [P^{\mathbf{d}^{i+1}}]^\top (P^{\mathbf{d}^i}\bar{X}_{t_i^-} + \Delta^{i,i+1}). \qquad (3.63)$$

Therefore,

$$\begin{aligned}
\|\bar{X}_{t_i}\|_{\bar{M}^{\mathbf{d}^{i+1}}}]^2 &= (P^{\mathbf{d}^i}X_{t_i^-} + \Delta^{i,i+1})^\top P^{\mathbf{d}^{i+1}}[P^{\mathbf{d}^{i+1}}]^\top M^{\mathbf{d}^{i+1}} P^{\mathbf{d}^{i+1}}[P^{\mathbf{d}^{i+1}}]^\top (P^{\mathbf{d}^i}X_{t_i^-} + \Delta^{i,i+1}) \\
&\leq (P^{\mathbf{d}^i}X_{t_i^-} + \Delta^{i,i+1})^\top M^{\mathbf{d}^{i+1}} (P^{\mathbf{d}^i}X_{t_i^-} + \Delta^{i,i+1}).
\end{aligned}$$

Based on (3.50) *and taking dwell time* τ *into account, we know that*

$$\|\bar{X}_{t_i^-}\|_{\bar{M}^{\mathbf{d}^i}} \leq \|\bar{X}_{t_{i-1}}\|_{\bar{M}^{\mathbf{d}^i}} e^{-\kappa\tau}.$$

Then,

$$\|\bar{X}_{t_i}\|_{M^{\mathbf{d}^{i+1}}} \leq \mu\|\bar{X}_{t_i^-}\|_{\bar{M}^{\mathbf{d}^i}} + \Delta_{max} \leq \mu g\|\bar{X}_{t_{i-1}}\|_{\bar{M}^{\mathbf{d}^i}} + \Delta_{max} \tag{3.64}$$

Putting (3.64) *into* (3.61) *we have:*

$$\|\bar{X}_{t_i^-} + \Delta^{i,i+1} - P^{\mathbf{d}^{i+1}}\bar{X}_{t_i}\|_{M^{\mathbf{d}^i}} \leq 2\mu g\|\bar{X}_{t_{i-1}}\|_{M^{\mathbf{d}^{i+1}}} + 2\Delta_{max} = 2\bar{\varepsilon}_i + 2\Delta_{max}. \tag{3.65}$$

Combining the two computed upper bounds, we get the difference equations (3.59).

Remark 17 *(1) The precision of the bound in* (3.59) *can be increased in two ways. First, the bound will be lower for policies with larger dwell time* τ *(smaller* g*). Second, if we increase the order of reduced system,* ε_0 *will become smaller. (2) The gain* g *solely depends on the CTMDP* $\mathcal{M}$ *and dwell time of policy* π*. In order to have a meaningful error bound, dwell time should satisfy* $\tau > \frac{\log \mu}{\kappa}$*. This condition is already true if we find a* common Lyapunov function *for the CTMDP* $\mathcal{M}$*, i.e., if there is one matrix* M *independent of the decision vector* d *satisfying* (3.49)*. In that case,* $\mu = 1$ *and dwell time can take any positive value.*

corollary 3.3.1 *The error* ε_i *in* (3.59) *converges to the constant value* $\gamma\Delta_{max}$ *for* $\mu g < 1$*, where*

$$\gamma := \frac{2 - 4\mu g}{(1 - \mu g)^2}. \tag{3.66}$$

Proof *We can rewrite* (3.59) *into a discrete time state space representation as*

$$\begin{bmatrix}\varepsilon_i \\ \bar{\varepsilon}_i\end{bmatrix} = \begin{bmatrix}\mu g & 2\mu g \\ 0 & \mu g\end{bmatrix}\begin{bmatrix}\varepsilon_{i-1} \\ \bar{\varepsilon}_{i-1}\end{bmatrix} + \begin{bmatrix}2 \\ 1\end{bmatrix}\Delta_{max}, \tag{3.67}$$

We consider (3.67) *as a dynamical system in discrete time (index* i *plays the role of time, which is discrete). Such a discrete-time dynamical system is asymptotically stable if all eigenvalues of its state matrix are in the unit circle. Since the state matrix of* (3.67) *is upper triangular, its eigenvalues are the same as the diagonal elements of the state matrix, which are both* μg*. Therefore, the system is asymptotically stable iff* $\mu g < 1$*. Hence, we can compute the steady state value of* ε *using the expression below:*

$$\lim_{i\to\infty} \varepsilon_i = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}1 - \mu g & 2\mu g \\ 0 & 1 - \mu g\end{bmatrix}^{-1}\begin{bmatrix}2 \\ 1\end{bmatrix}\Delta_{max} = \frac{2 - 4\mu g}{(1 - \mu g)^2}\Delta_{max}.$$

Remark 18 *For the case of having no* **bad** *states, we get* $A^{-1}\beta^{\mathbf{d}} = -\mathbf{1}$ *and* $\Delta_{max} = 0$*. Corollary 3.3.1 implies that for CTMDP* $\mathcal{M}$ *with no* **bad** *states, the error bound will converge to zero as a function of time.*

So far we discussed reduction and error computation for a given policy π. Our proposed CTMDP reduction scheme is outlined in Algorithm 9. Notice that the statement of Theorem 3.3.5 holds for any policy as long as it has a dwell time at least τ. Therefore, we can find a policy using a reduced system and apply it to the original CTMDP $\mathcal{M}$ with the goal of maximizing reachability probability. For a given CTMDP $\mathcal{M}$, time horizon B, probability threshold θ, and error bound ε, we select a dwell time τ and order of the reduced system such that $\varepsilon_n e^{-\kappa(B-t_n)} \leq \varepsilon$ according to (3.58) with $n = \lfloor T/\tau \rfloor$. Then we construct a policy π using the reduced order system (3.56) by setting $\mathbf{d}^0 = \arg\max_d A^{\mathbf{d}} X^{\mathbf{d}}(0)$ where $X^{\mathbf{d}}(0) = [A^{\mathbf{d}}]^{-1}\beta^{\mathbf{d}}$. The next selection of policies are done by respecting dwell time and $\mathbf{d}^{i+1} = \arg\max_d P^d \bar{A}^{\mathbf{d}} \bar{X}_t^{\mathbf{d}}$ for $t \geq t_i + \tau$ with t_i being the previous switching time. Policy synthesis over the reduced order system can be implemented as it is shown in Algorithm 10. Note that the computed policy may not be optimal because we fix a dwell time and a discretization time step. If the computed interval for reachability probability is not above θ, we go back and improve the results by increasing the order of the reduced system.

Algorithm 9: Order reduction of CTMDPs

Input: CTMDP $\mathcal{M}$, time bound B, maximum error bound ε, policy π with dwell time τ

1. Compute $A^{\mathbf{d}}$, $\beta^{\mathbf{d}}$ and $\kappa^{\mathbf{d}}$ for all d, based on (3.46) and (3.38)
2. Set $\kappa = \min_d \kappa^{\mathbf{d}}$ and $M^{\mathbf{d}} = \mathbb{I}_{|S|}$
3. Compute the maximum number of switches as $n = \lfloor \frac{T}{\tau} \rfloor$
4. Initialise the order $r = 0$
5. **Do**
   $r \leftarrow r + 1$
   Compute $\bar{A}^{\mathbf{d}}$ and $P^{\mathbf{d}}$ for all $\mathbf{d} \in \mathcal{D}$ using (3.39)
   Compute $\bar{X}^{\mathbf{d}}(0)$ for all $\mathbf{d} \in \mathcal{D}$ using (3.41)
   Compute error bound ε_r as (3.58) using (3.59)
   While $(\varepsilon_r \geq \varepsilon)$

Result: Reduced order system of (3.56) with matrices ($\bar{A}^{\mathbf{d}}$, $P^{\mathbf{d}}$ for $\mathbf{d} \in \mathcal{D}$)

Example 5 *Consider a CTMDP described by the following generator matrices corresponding to two decisions* $\mathbf{d}^1$ *and* $\mathbf{d}^2$*,*

$$\mathbf{Q}^{\mathbf{d}^1} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0.01 & -3.01 & 0.5 & 0.5 & 2 \\ 0 & 0.01 & -1.01 & 0 & 1 \\ 0 & 0.01 & 0.05 & -1.06 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{Q}^{\mathbf{d}^2} = \begin{bmatrix} -1.5 & 0 & 0.75 & 0.75 & 0 \\ 0.01 & -3.01 & 0.5 & 0.5 & 2 \\ 0 & 0.01 & -1.01 & 0 & 1 \\ 0 & 0.01 & 0.05 & -1.06 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

This means there are two actions available in the first state, and each induces outgoing rates specified by the first rows of $\mathbf{Q}^{\mathbf{d}^1}$ *and* $\mathbf{Q}^{\mathbf{d}^2}$*. The other states have only one action available. The last state is* **good***, which is absorbing. We set the time bound* $B = 10$*. Using the partition defined in Eq.* (3.46)*, we get*

$$A^{\mathbf{d}^1} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0.01 & -3.01 & 0.5 & 0.5 \\ 0 & 0.01 & -1.01 & 0 \\ 0 & 0.01 & 0.05 & -1.06 \end{bmatrix}, \quad \boldsymbol{\beta}^{\mathbf{d}^1} = \begin{bmatrix} 0 \\ 2 \\ 1 \\ 1 \end{bmatrix},$$

$$A^{\mathbf{d}^2} = \begin{bmatrix} -1.5 & 0 & 0.75 & 0.75 \\ 0.01 & -3.01 & 0.5 & 0.5 \\ 0 & 0.01 & -1.01 & 0 \\ 0 & 0.01 & 0.05 & -1.06 \end{bmatrix}, \quad \boldsymbol{\beta}^{\mathbf{d}^2} = \begin{bmatrix} 0 \\ 2 \\ 1 \\ 1 \end{bmatrix}.$$

Both $A^{\mathbf{d}^1}$ *and* $A^{\mathbf{d}^2}$ *are irreducible. Thus, Assumption 3 holds. We compute the decay rates* $\kappa^{\mathbf{d}^1}$ *and* $\kappa^{\mathbf{d}^1}$ *using Eq.* (3.38) *and set* $\kappa = \min(\kappa^{\mathbf{d}^1}, \kappa^{\mathbf{d}^2}) = 0.4965$*. Furthermore, Eq.* (3.49) *can be satisfied by setting* $M^{\mathbf{d}^1} = M^{\mathbf{d}^2} = \mathbb{I}_4$*. This allows us to choose* $\mu = 1$*. Hence, the dwell time* τ *can take any positive value since* $\frac{\log \mu}{\kappa} = 0$*. We set the dwell time* $\tau = 2.3$*.*
For the reduced order $r = 3$*, we use Theorem 3.3.4 and get*

$$\bar{A}^{\mathbf{d}^1} = \begin{bmatrix} -3.0199 & 0.9859 & 0.6244 \\ 0 & -0.993 & -0.3137 \\ 0 & 0 & -1.0071 \end{bmatrix}, \qquad P^{\mathbf{d}^1} = \begin{bmatrix} -0.4437 & -0.8962 & -0.0059 \\ 0.8962 & -0.4437 & 0.0041 \\ -0.0045 & -0.0024 & 0.7071 \\ -0.0045 & -0.0024 & 0.7071 \end{bmatrix}$$

that correspond to the decision vector $\mathbf{d}^1$*, and*

$$\bar{A}^{\mathbf{d}^2} = \begin{bmatrix} -3.0149 & -0.0174 & 0.6982 \\ 0 & -1.5 & -1.0571 \\ 0 & 0 & -1.0049 \end{bmatrix}, \qquad P^{\mathbf{d}^2} = \begin{bmatrix} 0.0049 & -1 & -0.0001 \\ 1 & 0.0049 & 0.0071 \\ -0.005 & 0.0001 & 0.7071 \\ -0.005 & 0.0001 & 0.7071 \end{bmatrix}$$

that correspond to $\mathbf{d}^2$*. We initialize the set of differential equations with* $\bar{X}^{\mathbf{d}^1}(0)$ *and* $\bar{X}^{\mathbf{d}^2}(0)$ *computed using Eq.* (3.41) *as*

$$\bar{X}^{\mathbf{d}^1}(0) = \begin{bmatrix} -0.4436 \\ 1.3447 \\ -1.4125 \end{bmatrix} \quad \textit{and} \quad \bar{X}^{\mathbf{d}^2}(0) = \begin{bmatrix} -0.9949 \\ 0.9949 \\ -1.4124 \end{bmatrix}.$$

Note that $g = e^{-\kappa\tau} = 0.007$*,* $\Delta^{12} = \Delta^{21} = 0$*, and* $\Delta_{max} = 0$*. We compute the error of order reduction using equations* (3.58)-(3.59) *with* $n = \lfloor \frac{T}{\tau} \rfloor = 4$ *and* $t_n = n\tau = 9.2$*. This gives the error bound* 0.1396*.*

Our formulated error bound depends on the order r of the reduced system and the dwell time τ. There is a tradeoff between r and τ for having a guaranteed error bound. The error bound depends on r implicitly and is selected recursively. Computation of the sub-optimal policy depends also on the discretization step δ. The overall complexity of such a computation for a CTMDP with m states, l decision vectors, and time bound B is $\mathcal{O}(lm^3) + \mathcal{O}(\frac{Blr^2}{\delta})$, where the first and second terms are the computational complexities for the reduced system and the sub-optimal policy, respectively.

Algorithm 10: Sub-optimal policy synthesis for CTMDPs

Input: Reduced system ($\bar{A}^{\mathbf{d}}$, $P^{\mathbf{d}}$ for $d \in \mathcal{D}$), time bound B, dwell time τ, discretization step δ

1. $\mathbf{d}^0 = \arg\max_{d \in \mathcal{D}}(\mathbf{Q}^{\mathbf{d}} X^{\mathbf{d}}(0))$
2. $k = \lfloor \frac{\tau}{\delta} \rfloor + 1$
3. $\pi_t = \mathbf{d}^0$ for $t \in [0, k\delta)$
4. **While** $k < \lfloor \frac{T}{\delta} \rfloor + 1$
   Compute a possibly sub-optimal policy using:

$$\mathbf{d}^k = arg\max_{\mathbf{d} \in \mathcal{D}}(\mathbf{Q}^{\mathbf{d}} P^{\mathbf{d}} \bar{X}^{\mathbf{d}}(k\delta))$$

   If $\mathbf{d}^k \neq \mathbf{d}^{k-1}$
   $\pi_t = \mathbf{d}^k$ for $t \in [k\delta, (k + \lfloor \frac{\tau}{\delta} \rfloor + 1)\delta)$
   $k \leftarrow k + \lfloor \frac{\tau}{\delta} \rfloor + 1$
   Compute $\bar{X}^{\mathbf{d}}(k\delta)$ using (3.56) and (3.63) for all $\mathbf{d} \in \mathcal{D}$
   Else
   $\pi_t = \mathbf{d}^k$ for $t \in [k\delta, (k+1)\delta)$
   $k \leftarrow k + 1$
   Compute $\bar{X}^{\mathbf{d}}(k\delta)$ using (3.56) for all $\mathbf{d} \in \mathcal{D}$
   End
End

Result: Sub-optimal policy π_t for $t \in [0, B]$

3.3.3. Experimental Evaluation

Here, we first use our method for reachability analysis of two queuing systems, namely $M/M/1$ and tandem networks. We then evaluate the performance of our proposed symbolic computation on randomly generated models.

The $M/M/1$ queue consists of only one queue with a specific capacity denoted by **cap**. Jobs arrive with the rate $\bar{\lambda}$ and are processed with the rate μ. The $M/M/1$ queue can be modeled as a CTMC with a state space of size $(\mathbf{cap} + 1)$. We find the probability of

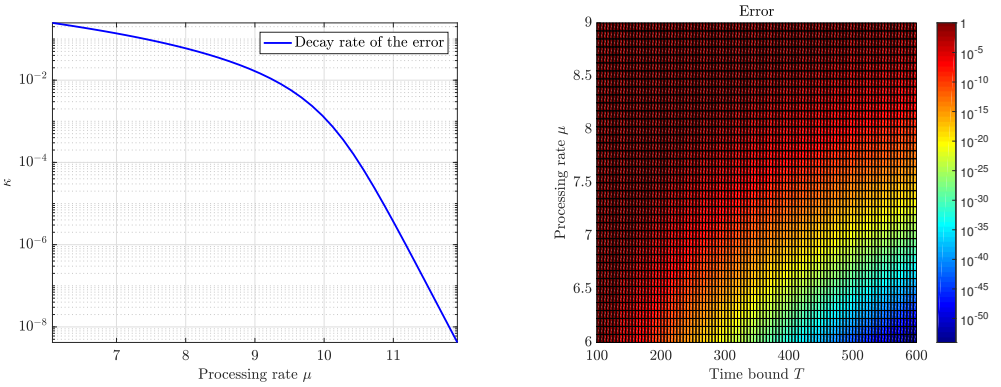

Figure 3.2.: Error analysis for the state reduction for $M/M/1$ queuing system. **left:** decay rate of the error as a function of processing rate μ. **right:** error of the state reduction as a function of time bound B and processing rate μ. The error is very small for larger time bounds B and smaller μ.

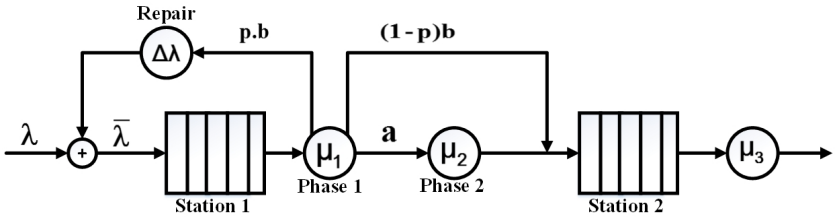

Figure 3.3.: A typical tandem network

reaching the configuration in which the queue is at its full capacity from a configuration in which the queue is empty. The generator matrix of this CTMC is tridiagonal, with upper diagonal entries $\bar{\lambda}$, lower diagonal entries μ, and main diagonal entries $-(\bar{\lambda}+\mu)$.

We choose **cap** $= 100$ (size of the state space is 101) and fix the size of the reduced system to $r = 10$. We also fix the arrival rate $\bar{\lambda} = 10$ and study the behavior of our formulated error bound for state reduction with respect to the processing rate μ. Figure 3.2 (**left**) demonstrates the variations of the decay rate κ defined in Eq. (3.38) as a function of processing rate μ. The decay rate is larger for smaller values of μ and become very close to zero for larger values of μ, which makes our approach very efficient for smaller values of μ. This fact is also visible from Figure 3.2 (**right**), where the error defined formally in Eq. (3.33) is shown as a function of the time bound B and μ in logarithmic scale. It can be observed that the error is very small for larger time bounds B and smaller μ.

We now apply our results to the *tandem network* shown in Figure 3.3. The network is a queuing system that consists of a $M/Cox2/1$ queue composed with a $M/M/1$ queue [75].

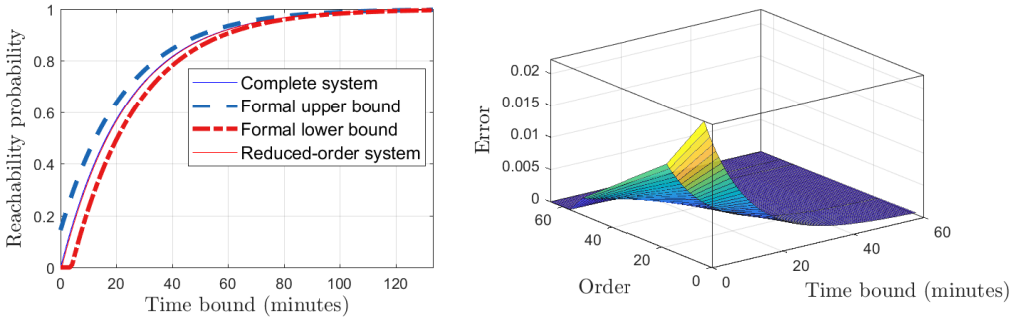

Figure 3.4.: **left:** approximate reachability probability for tandem network as a function of time horizon with guaranteed error bounds. **right:** error bound as a function of time horizon and order of the reduced system;

Both queuing stations have a capacity of **cap**. The first queuing station has two phases for processing jobs while the second queuing station has only one phase. Processing phases are indicated by circles in Figure 3.3. Jobs arrive at the first queuing station with rate $\bar{\lambda}$ and are processed in the first phase with rate μ_1. After this phase, jobs are passed through the second phase with probability a, which are then processed with rate μ_2. Alternatively, jobs will be sent directly to the second queuing station with probability b, a percent of which will have to undergo a repair phase and will go back to the first station with rate $\Delta\lambda$ to be processed again. This percentage is denoted by p. Processing in the second station has rate μ_3.

The tandem network can be modeled as a CTMC with a state space of size determined by **cap**. We find the probability of reaching to the configurations in which both stations are at their full capacity (blocked state) starting from a configuration in which both stations are empty (empty state). We consider **cap** $= 5$ which results in a CTMC with 65 states. We have chosen values $\mu_1 = \mu_2 = 2$, $\mu_3 = \lambda = 4$, $a = 0.1$, and $b = 0.9$. We also set $p = 0$ and $\Delta\lambda = 0$, which means no job is going to the repair phase. Matrix inequalities (3.32) are satisfied with M being identity and $\kappa = 0.001$. Using the reduction technique of Subsection 3.3.1, we can find approximate solution of reachability with only 3 state variables. Figure 3.4 (**left**) shows reachability probability computed over the tandem network and the reduced order system together with the error bound as a function of time horizon. The error has the initial value 0.02, computed via the choice of initial reduced state in (3.41), and converges to zero exponentially with rate 0.0013. It can also be noticed that the outputs of the full and reduced-order systems cannot be distinguished in the figure. This is due to the fact that their actual difference is very small compared to the formal error bound characterized by our method.

Figure 3.4 (**right**) gives the error bound as a function of time horizon of reachability and order of the reduced system. As discussed, the error goes to zero exponentially as a function of time horizon. It also converges to zero by increasing the order of reduced system.

Now consider a scenario that the network can operate in *fast* or *safe* modes. In fast mode,

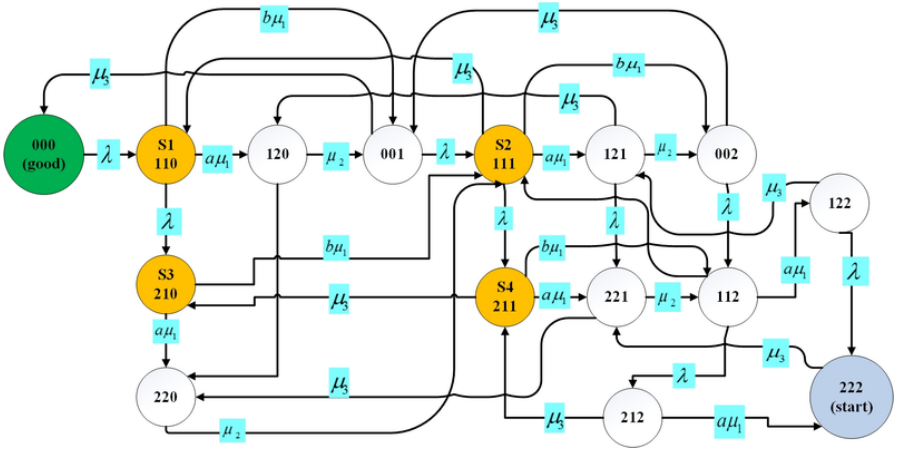

Figure 3.5.: State diagram of a CTMDP with 16 states and 16 decision vectors corresponding to a tandem network with capacity 2. States S_1, S_2, S_3, S_4 have two modes with rates $a \in \{0.6, 0.7\}$.

fewer jobs are sent through the second phase (corresponding to a smaller value of a); this, in turn, increases the probability that jobs which did not pass second phase, need to be processed again. We model influence of returned jobs as an increase in $\Delta\lambda$.
We consider the case that there are two possible rates $a \in \{0.6, 0.7\}$ corresponding respectively to fast and safe modes. If fast mode is chosen, 10% of jobs will be returned ($p = 0.1$) with rate $\Delta\lambda = 0.05$. In the safe mode, only 5% of jobs ($p = 0.05$) will be returned with the same rate $\Delta\lambda$. We set $\mu_1 = \mu_2 = 2.5$ and $\mu_3 = \lambda = 3$.

A tandem network with capacity $\mathbf{cap} = 2$ and these two modes can be modeled as a CTMDP with 16 states and 16 decision vectors. Figure 3.5 depicts state diagram of this CTMDP with states S_1, S_2, S_3, S_4 having two modes with the corresponding value of rate a. We assume the tandem network is initially at the state 220 of Figure 3.5, which means there are two jobs in the first station, both are being served in the second phase, and there is no job in the second station. We consider synthesizing a strategy with respect to the probability of having both queuing stations becoming empty by time B. We have implemented the approach of Subsection 3.3.2 and obtained a reduced system of order 6 with $\varepsilon_0 = 0.14$. Figure 3.6 (**left**) demonstrates reachability probabilities as a function of time for both the tandem network and its reduced counterpart together with the error bound. Intuitively, choosing the fast mode in the beginning will result in faster progress of the tasks, especially when queues are more loaded; however, if this selection is continued, it will result in a high number of returned jobs, which is not desired. This behavior is observed depending on the state in the form of three switches in states S_2, S_3, S_4. In Figure 3.6 (**left**) the green trajectory corresponds to the reachability probability of the original CTMDP under the non-restricted optimal piecewise constant policy. Figure 3.6 (**right**) demonstrates the impact of dwell time on the optimization error (in blue) and on the guaranteed error bound (in red) for time bound $B = 100$ seconds. The reduction error bounds are computed formally using the results of Theorem 3.3.5, by solving (3.59)

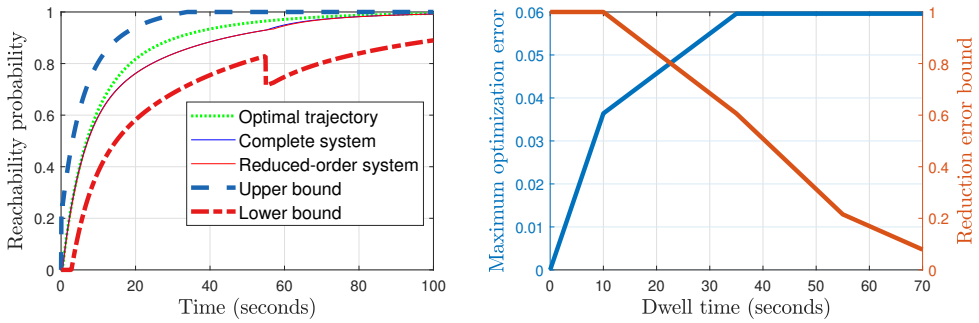

Figure 3.6.: **left:** approximate reachability probability for tandem network with 16 decision vectors including and the formal bounds ($\tau = 55$ seconds). **right:** error in the optimal reachability probability and the reduction error bound with dwell time ($B = 100$ seconds).

and using it in (3.58). The optimization error is computed numerically. For each dwell time, we compute optimal reachability probability corresponding to the full-order system running with non-restricted policy as well as the reachability probability corresponding to the reduced-order system with policy restricted with the chosen dwell time. The optimization error is defined as the difference between these two values.

Finally, we assess the performance of symbolic computation on randomly generated models. Table 3.1 compares runtime of the reachability probability computation using three different methods: adaptive implementation of the uniformization technique presented in [32] (RT_u), symbolic computation presented in our work without state reduction (RT_s) using only Algorithm 10 of Abschnitt 3.3.1, and symbolic computation with state reduction (RT_{sr}) by running both Algorithms 9 and 10.

Note that the method presented in [32] is developed for sub-optimal policy synthesis of CTMDPs and tunes the length of the time discretization adaptively. According to our experiments, the adaptive selection of time discretization makes it more efficient also for reachability computation of CTMCs in comparison with the uniform discretization proposed in [14]. Therefore, we compare our results with the approach of [32].

The experiments are done using MATLAB R2017a on a 3.3 GHz Intel Core i5 processor. For each experiment, 10 stochastic matrices are generated randomly as infinitesimal generator matrix corresponding to a CTMC without imposing any sparsity assumption. To implement the uniformization, the step time is tuned adaptively with maximum truncation error bound 0.01. The maximum number of terms in the Maclaurin expansion is set to 5 and the time bound is fixed at 5 seconds, while the minimum time step for uniformization is chosen to be 10^{-4} seconds. Note that RT_{sr} also includes the time for running Algorithm 9. As it can be observed from Table 3.1, RT_{sr} is smaller than RT_u and RT_s by at least two and one orders of magnitude, respectively.

Table 3.1.: Comparison of runtime (in seconds) for the reachability probability computation using the uniformization technique of [32] (RT_u), symbolic computation without state reduction (RT_s) by running only Algorithm 10, and symbolic computation with state reduction (RT_{sr}) by running both Algorithms 9 and 10.

Number of states	RT_u	RT_s	RT_{sr}
100	3.132	0.0781	0.0112
200	7.295	0.5483	0.0362
500	94.55	8.247	0.2371
800	461.8	35.31	0.9968
1000	831.8	68.61	1.788
1200	1444.2	114.73	2.4911
1500	3384.1	226.21	4.8538

3.4. Conclusion

In this chapter, we studied the time-bounded reachability problem for CTMDPs, examining it from the angles of scalability and decidability. First, we established a conditional decidability outcome for the time-bounded reachability problem of CTMDPs, harnessing tools from number theory. Following this, we tackled the scalability challenge of approximating the time-bounded reachability value for both CTMCs and CTMDPs. We introduced a control-theoretic approach aimed at reducing the state space size while maintaining formal error bounds.

4

Reachability and Pseudo-Reachability in Linear Dynamical Systems

A (discrete-time) linear dynamical system in m dimensions is defined by a linear map $x \mapsto Ax$ for an $n \times n$ rational matrix A. The map specifies how an individual state (a real-valued vector in m dimensions) evolves over time; a trajectory starting from a state s is given by the sequence $(s, As, A^2s, \ldots)$. Linear dynamical systems are fundamental models in many different domains of science and engineering, and the computability and complexity of decision problems for linear dynamical systems are of both theoretical and practical interest.

The *orbit* of a point s is the smallest transitive set containing s and closed under the dynamic map. The *orbit problem* for linear dynamical systems asks, given s and t, if t is in the orbit of s [73]. In a seminal paper, Kannan and Lipton [91] showed that the orbit problem can be decided in polynomial time. However, a natural generalization of the orbit problem, the *Skolem problem*, in which we ask whether the orbit of a given state s intersects a given hyperplane, turns out to be notoriously difficult and remains open after many decades [173, 140]. A breakthrough occurred in the mid-1980s, when Mignotte *et al.* [126] and Vereshchagin [179] independently showed decidability in dimension 4 or less. These deep results make essential use of Baker's theorem on linear forms in logarithms (which earned Baker the Fields Medal in 1970), as well as a p-adic analogue of Baker's theorem due to van der Poorten. Unfortunately, little progress on that front has since been recorded.

Orbit and Skolem problems are defined for the exact dynamics, i.e., when there is no disturbance affecting the system's evolution. In practice, one is often interested in answering whether a specific target set is reachable under the influence of disturbances constrained to values from a bounded set. In this chapter, we study two such problems. First, we study the point-to-point reachability problem for perturbed linear dynamical systems with hypercubic disturbance sets. We show that this problem is at least as hard as the *positivity* problem—a famous longstanding open problem in the theory of linear dynamical systems. Next, we shift our focus to reachability under *pseudo-orbits* as generalizations of the orbits, and present exciting decidability results for point and hyperplane target sets.

4.1. Preliminaries

4.1.1. Notation

The sets of natural numbers (including zero), rational numbers, real numbers, and algebraic numbers are denoted by $\mathbb{N}$, $\mathbb{Q}$, $\mathbb{R}$, and $\bar{\mathbb{Q}}$, respectively. We assume a standard representation of algebraic numbers in terms of their defining polynomials, by which we can perform arithmetic operations and test equality in polynomial time in their representation (see, e.g., [41]).

For any column vector $x = [x_1, x_2, \ldots, x_m]^\top \in \mathbb{R}^m$, we use the notations $\|x\|_2 := \sqrt{x^\top x}$ and $\|x\|_\infty := \max_i |x_i|$ to indicate respectively the two norm and infinity norm of x. For any matrix $A = [a_{ij}]_{i,j} \in \mathbb{R}^{m\times m}$, we define $\|A\|_2$ and $\|A\|_\infty$ to indicate respectively the (induced) two norm and infinity norm of A. Note that $\|Ax\|_2 \leq \|A\|_2 \|x\|_2$ and $\|Ax\|_\infty \leq \|A\|_\infty \|x\|_\infty$ for all $x \in \mathbb{R}^m$. We write $\mathbf{0} \in \mathbb{R}^m$ for the zero vector and $\mathbf{1} \in \mathbb{R}^m$ for the all-ones vector. We denote by $\rho(A)$ the spectral radius of a matrix A, which is the largest absolute value of the eigenvalues of A. For any $A \in \mathbb{R}^{m\times m}$ and any $\gamma > \rho(A)$, recall that there is a constant $c > 0$ such that $\|A^n\|_2 \leq c\gamma^n$ for all $n \in \mathbb{N}$.

4.1.2. Discrete-Time Linear Dynamical Systems

An m-dimensional discrete-time linear dynamical system is specified by an $m \times m$ matrix A of rational numbers. The *trajectory* determined by an initial state $x_0 \in \mathbb{R}^m$ is the sequence $(x_n)_{n\geq 0}$ given by

$$x_{n+1} = Ax_n, \quad (n \in \mathbb{N}).$$

We call the set $\mathcal{O}(A, x_0) := \{x_n \mid n \in \mathbb{N}\}$ the *orbit* of x_0.

For any $\epsilon > 0$, an ϵ-perturbed linear dynamical system has state trajectories $(x_n)_{n\geq 0}$ such that

$$x_{n+1} = Ax_n + d_n, \quad (n \in \mathbb{N}),$$

where A is as before and $d_n \in [-\epsilon, \epsilon]^m$ for all n. For an initial state $x_0 \in \mathbb{R}^m$, we define the *ϵ-pseudo-orbit* $\widetilde{\mathcal{O}}_\epsilon(A, x_0)$ of the dynamics as the set of states reachable in the perturbed dynamics. More formally, define

- for $n = 0$, $\widetilde{\mathcal{O}}_\epsilon^{(n)}(A, x_0) := \{x_0\}$,
- for all $n \in \mathbb{N}$, $\widetilde{\mathcal{O}}_\epsilon^{(n+1)}(A, x_0) := \{Ax + d \in \mathbb{R}^m \mid x \in \widetilde{\mathcal{O}}_\epsilon^{(n)}(A, x_0), d \in [-\epsilon, \epsilon]^m\}$, and
- $\widetilde{\mathcal{O}}_\epsilon(A, x_0) := \bigcup_{n\geq 0} \widetilde{\mathcal{O}}_\epsilon^{(n)}(A, x_0)$.

Finally, we define the *pseudo-orbit* $\widetilde{\mathcal{O}}(A, x_0) := \bigcap_{\epsilon>0} \widetilde{\mathcal{O}}_\epsilon(A, x_0)$ as the intersection of all the ϵ-pseudo-orbits of x_0, for all $\epsilon > 0$. Clearly, $\mathcal{O}(A, x) \subseteq \widetilde{\mathcal{O}}(A, x)$ for any A and x. We will make use of the following characterization, which follows directly from the definition: Any $t \in \widetilde{\mathcal{O}}_\epsilon(A, s)$ is of the form $t = A^n s + \sum_{i=0}^{n-1} A^i d_{n-i-1}$ for some $n \in \mathbb{N}$ and some sequence of perturbations d_i with $\|d_i\|_\infty \leq \epsilon$.

4.1.3. Decision Problems for Linear Dynamical Systems

Here, we present a collection of significant decision problems related to linear dynamical systems that have received extensive attention. We start with the orbit problem.

Problem 4.1 (Orbit Problem) *Given $A \in \mathbb{Q}^{m\times m}$ and $s,t \in \mathbb{Q}^m$, decide whether $t \in \mathcal{O}(A,s)$.*

A celebrated result of Kannan and Lipton [91] shows that the orbit problem is decidable in polynomial time. Orbit problem corresponds to the point-to-point reachability in linear dynamical systems under the exact dynamics. Below, we describe the positivity problem, which corresponds to the reachability of half-spaces under the exact dynamics.

Problem 4.2 (Positivity Problem) *Given $A \in \mathbb{Q}^{m\times m}$ and $s \in \mathbb{Q}^m$, decide if there exists N such that $\mathbf{e}_1 A^N s > 0$, where $\mathbf{e}_1 = [1\ 0\ \cdots\ 0]$.*

Below, we define a problem which will be related to the positivity problem later (Lemma 4.1.1).

Problem 4.3 *Given a rational matrix A and a rational vector s, decide if there exists $N \geq 0$ such that $-\mathbf{1} < A^N s < \mathbf{1}$.*

Lemma 4.1.1 *Problem 4.3 is at least as hard as the positivity problem.*

Proof *We show how the positivity problem can be decided using a decision procedure for Problem 4.3. First we show how we can decide, using an oracle for Problem 4.3, the following problem. Given $A \in \mathbb{Q}^{m\times m}$ and $s \in \mathbb{Q}^m$, does there exist N such that $A^N s > \mathbf{0}$? We call this* Reachability in the Positive Quadrant Problem*.*

Given such M and x_0, first divide both numbers by a sufficiently large positive numbers to obtain $A' \in \mathbb{Q}^{m\times m}$ and $s' \in \mathbb{Q}^d$ such that $\mathbf{-2} < A'^N s' < \mathbf{2}$ for all N. Next, construct $M \in \mathbb{Q}^{m+1\times m+1}$ and $v \in \mathbb{Q}^{m+1}$ such that

- *$v = [\leftarrow s'^\top \rightarrow \ \mid 0.5]^\top$, that is s' obtained by adjoining 0.5 to s';*
- *$M[\leftarrow x^\top \rightarrow \ \mid 0.5]^\top = [\leftarrow (A'x - \mathbf{1})^\top \rightarrow \ \mid 0.5]^\top$ for all $x \in \mathbb{Q}^m$.*

This way, $A^N s > \mathbf{0} \iff \mathbf{0} < A'^N s' < \mathbf{2} \iff -\mathbf{1} < M^N v < \mathbf{1}$.

Next, let us show how we can decide the positivity problem using an oracle for the Reachability in the Positive Quadrant Problem. Let $w \in \{<,>\}^m$ specify the quadrant $S_w = \{\mathbf{x} : \mathbf{x}(i)w(i)0\}$. Observe that the reachability in a quadrant S_w (given A, s, does there exist N such that $A^N \in S_w$?) can be reduced to the Reachability in the Positive Quadrant Problem by a change of basis.

Finally, let an instance of positivity problem be given by $A \in \mathbb{Q}^{m\times m}$ and $s \in \mathbb{Q}^m$, and let $S_{w_1}, \ldots, S_{w_{2^m-1}}$ be all the quadrants with $w_i(1)$ equal to $>$ (i.e. with positive first coordinate). Then $\exists N.\mathbf{e}_1 A^N s > 0 \iff \bigvee_{i=1}^{2^m-1} \exists N.A^N s \in S_{w_i}$. Now observe that whether there exists N such that $A^N s \in S_{w_i}$ can be decided by using the reduction to the Reachability in the Positive Quadrant Problem described above.

4.1.4. Jordan Normal Form

First we establish that pseudo-orbits can be translated with change of bases.

Proposition 4.1.1 *For matrices $A, B, Q \in \mathbb{R}^{m\times m}$ with $A = QBQ^{-1}$ and for any $x \in \mathbb{R}^m$, we have $Q\,\widetilde{\mathcal{O}}_{\gamma_2}(B, Q^{-1}x) \subseteq \widetilde{\mathcal{O}}_{\epsilon}(A, x) \subseteq Q\,\widetilde{\mathcal{O}}_{\gamma_1}(B, Q^{-1}x)$, where $\gamma_1 = \epsilon \left\|Q^{-1}\right\|_\infty$ and $\gamma_2 = \epsilon / \|Q\|_\infty$. Moreover, $\widetilde{\mathcal{O}}(A, x) = Q\,\widetilde{\mathcal{O}}(B, Q^{-1}x)$.*

We will use Proposition 4.1.1 with matrix A represented using the *Jordan canonical form*.

Jordan Decomposition. For a given rational square matrix A one can compute *change of basis matrix* Q and *Jordan normal form* J so that $A = QJQ^{-1}$ and $J = \mathrm{diag}(J_1, J_2, \cdots, J_z)$ with J_i representing the i^{th} Jordan block taking the following form

$$J_i = \begin{bmatrix} \Lambda_i & 1 & 0 & \dots & 0 & 0 \\ 0 & \Lambda_i & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \Lambda_i & 1 \\ 0 & 0 & 0 & \dots & 0 & \Lambda_i \end{bmatrix}, \tag{4.1}$$

where Λ_i denotes the i^{th} eigenvalue of A. The size of J_i is equal to the multiplicity of the eigenvalue Λ_i and is denoted by $\kappa(\Lambda_i)$.

Real Jordan form. For any $A \in \mathbb{R}^{m\times m}$ having complex eigenvalues, matrices Q and J in the Jordan normal form could have complex entries. In this case, the complex eigenvalues form complex conjugate pairs and give a *real Jordan form*: there are *real* matrices Q and J such that $A = QJQ^{-1}$ and $J = \mathrm{diag}(J_1, J_2, \cdots, J_z)$. The matrix J_i represents the i^{th} real Jordan block corresponding to either a real eigenvalue Λ_i or a complex pair $\Lambda_i = a_i \pm jb_i$. It is equal to (4.1) for real Λ_i and has the following form for the complex pair $\Lambda_i = a_i \pm jb_i$,

$$J_i = \begin{bmatrix} \Lambda_i & I_{2\times2} & 0_{2\times2} & \dots & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & \Lambda_i & I_{2\times2} & \dots & 0_{2\times2} & 0_{2\times2} \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & \dots & \Lambda_i & I_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & \dots & 0_{2\times2} & \Lambda_i \end{bmatrix}, \tag{4.2}$$

where with abuse of notation, we have indicated $\Lambda_i = \begin{bmatrix} a_i & -b_i \\ b_i & a_i \end{bmatrix}$. $I_{2\times2}$ and $0_{2\times2}$ denote identity and fully zero matrices of size 2 by 2.

The real Jordan normal form and the change of basis matrices Q and Q^{-1} can be computed in polynomial time (see [34] and also Appendix A.4.9).

Computing matrix powers. If $A = QJQ^{-1}$, then we have $A^n = QJ^nQ^{-1}$ for $n \in \mathbb{N}$, where $J^n = \mathrm{diag}(J_1^n, J_2^n, \dots, J_z^n)$, and for the i^{th} block J_i corresponding to the eigenvalue

Λ_i with multiplicity $\kappa(\Lambda_i)$ we have

$$J_i^n = \begin{bmatrix} \Lambda_i^n & n\Lambda_i^{n-1} & \binom{n}{2}\Lambda_i^{n-1} & \cdots & \binom{n}{\kappa(\Lambda_i)-1}\Lambda_i^{n-k+1} \\ 0 & \Lambda_i^n & n\Lambda_i^{n-1} & \cdots & \binom{n}{\kappa(\Lambda_i)-2}\Lambda_i^{n-k+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & n\Lambda_i^{n-1} \\ 0 & 0 & 0 & \cdots & \Lambda_i^n \end{bmatrix}$$

4.2. Hardness of the Reachability Problem

Motivated by decidability of the orbit problem, a very natural question would be about decidability of point-to-point reachability under perturbed dynamics with bounded disturbances. Fijalkow et al. studied this problem for the case that the allowable set of disturbances is definable by boolean combinations of linear inequalities, and presented several undecidability and hardness results for different instances of the problem [63]. In this section, we consider hypercubic sets as the allowable set of disturbances and prove that the corresponding point-to-point reachability problem is hard by relating it to the positivity problem—a famous longstanding open problem.

4.2.1. Problem Statement

Let $\Sigma\colon x \mapsto Ax + d$ describe a perturbed linear dynamical system with $A \in \mathbb{Q}^{m\times m}$ and $d \in \mathcal{D}$, where $\mathcal{D}$ denotes a hypercubic set $[b - \epsilon\mathbf{1}, b + \epsilon\mathbf{1}]$ with $b \in \mathbb{Q}^m$ and $\epsilon \in \mathbb{R}_{>0}$. In order to prove our hardness result, we notice that we can substitute the linear dynamical system Σ with an affine dynamical system $\Sigma_a\colon x \mapsto Ax + b + d$, with $d \in \mathcal{D}'$, where $\mathcal{D}' = [-\epsilon, \epsilon]^m$. Therefore, it is enough to study the hardness of the ϵ-pseudo-reachability problem for affine dynamical systems with fixed ϵ.

Problem 4.4 ***Inputs:*** *A matrix $A \in \mathbb{Q}^{m\times m}$, $b \in \mathbb{Q}^m$, $s, t \in \mathbb{Q}^m$, a fixed $\epsilon > 0$.*

Question: *Decide whether there exists an ϵ-pseudo-orbit from s to t under the mapping $x \mapsto Ax + b$.*

In the rest of this subsection, we prove that Problem 4.4 is *hard*.

4.2.2. Hardness Proof

In order to prove our results, we relate Problem 4.4 to the positivity problem, which has been open for many years and hence considered as a hard problem.

Theorem 4.2.1 *Problem 4.4 is at least as hard as the positivity problem.*

Proof *Consider an instance of the ϵ-pseudo-reachability problem characterized by a (perturbed) linear dynamical system with state matrix*

$$A = \begin{bmatrix} M & 0 & 0 \\ -M & 0 & 0 \\ I & I & I \end{bmatrix},$$

an affine term $b = (\epsilon\mathbf{1}, \epsilon\mathbf{1}, \epsilon\mathbf{1})$, starting at $s = (\epsilon x_0, -\epsilon x_0, \mathbf{0})$ with additive disturbance vector taking values from the set $\mathcal{D} = \{d \in \mathbb{R}^{3m} \mid |d(i)| < \epsilon \text{ for } 1 \leq i \leq 3m\}$. For the state and disturbance vectors, we write $x = (x^1, x^2, x^3)$ and $d = (d^1, d^2, d^3)$, where $x^k \in \mathbb{R}^m$ and $d^k \in [-\epsilon, \epsilon]^m$ for $1 \leq k \leq 3$. Given the point $t = (\epsilon\mathbf{1}, \epsilon\mathbf{1}, \mathbf{0})$, the decision problem is to check whether t is in the orbit of s under the dynamics $x_n = Ax_{n-1} + b + d_n$. We want to show a reduction from Problem 4.3 (characterized with the same matrix M as used in blocks of A) to the described instance of ϵ-pseudo-reachability. Note that we have already shown that Problem 4.3 is at least as hard as the positivity problem (Lemma 4.1.1).

First, assume that there exists a positive integer N s.t. $-\mathbf{1} < M^N x_0 < \mathbf{1}$. Choosing $d_0 = \ldots = d_{N-2} = (-\epsilon\mathbf{1}, -\epsilon\mathbf{1}, -\epsilon\mathbf{1})$ and $d_{N-1} = (-\epsilon M^N x_0, \epsilon M^N x_0, -\epsilon\mathbf{1})$, starting from $x_0 = s$, we have that $x_N = t$. Note that $d_{N-1} \in D$ since $-\mathbf{1} < M^N x_0 < \mathbf{1}$ holds.

Conversely, assume that there exists a disturbance sequence so that $x_N = t$. We show that the only possible sequence is $d_0 = \ldots = d_{N-2} = (-\epsilon\mathbf{1}, -\epsilon\mathbf{1}, -\epsilon\mathbf{1})$ and $d_{N-1} = (-\epsilon M^N x_0, \epsilon M^N x_0, -\epsilon\mathbf{1})$. To that end, we open up the update rule for x^3 and write

$$\begin{aligned} x_n^3 &= x_{n-1}^1 + x_{n-1}^2 + x_{n-1}^3 + d_{n-1}^3 + 3\epsilon\mathbf{1} \\ &= Mx_{n-2}^1 + d_{n-2}^1 - Mx_{n-2}^1 + d_{n-2}^2 + x_{n-1}^3 + d_{n-1}^3 + 3\epsilon\mathbf{1} \\ &= x_{n-1}^3 + d_{n-2}^1 + d_{n-2}^2 + d_{n-1}^3 + 3\epsilon\mathbf{1}. \end{aligned} \tag{4.3}$$

Based on Eq. 4.3, for every choice of $d \in D$ we have $x_n^3 \geq x_{n-1}^3$ and in particular $x_n^3 = x_{n-1}^3$ only when $d_{n-2}^1 = d_{n-2}^2 = d_{n-1}^3 = -\epsilon\mathbf{1}$. Note that since $x_N = t$, $x_N^3 = \mathbf{0}$, but once $x_n^3 > 0$ for $0 < n < N$, there is no possibility of bringing x^3 back to the origin. Therefore, the only possible sequence is $d_0 = \ldots = d_{N-2} = (-\epsilon\mathbf{1}, -\epsilon\mathbf{1}, -\epsilon\mathbf{1})$ and $d_{N-1} = (-\epsilon M^N x_0, \epsilon M^N x_0, -\epsilon\mathbf{1})$. This time we know that $d_{N-1} \in D$; hence, $-\mathbf{1} \leq M^N \mathbf{x}_0 \leq \mathbf{1}$.

4.3. Decidability of Pseudo-Reachability Problems

The orbit and Skolem problems are defined on the exact dynamics of the linear system. In dynamical systems theory, one is often interested in "rough" dynamics of a system—in topological terms, we wish to study closed sets containing the orbit. Orbits arising from linear dynamics are usually not closed sets. Indeed, the orbit of the dynamics $x \mapsto \frac{1}{2}x$ does not contain the limit point 0. One way to retain closure is through *pseudo-orbits* [45], a concept going back several decades. A pseudo-orbit generalizes the orbit by allowing arbitrarily small imprecisions throughout the dynamics. For a precision $\epsilon > 0$, we say t is in the ϵ-pseudo-orbit of s if there is a sequence of points $(s = s_0, s_1, \ldots, s_n = t)$ with $n > 0$ such that $\|As_i - s_{i+1}\|_\infty < \epsilon$ for each $i \in \{0, \ldots, n-1\}$. That is, an ϵ-pseudo-orbit contains the sequence of points that would be an orbit if each state were known only up to precision ϵ. Finally, t is in the pseudo-orbit of s if it is in the ϵ-pseudo-orbit of s *for all* $\epsilon > 0$.

One can provide a computational analogue of pseudo-orbits (see [142]). Alice is simulating the trajectory of a dynamical system but in every iteration, her computation has a rounding error ϵ. An infinitely powerful adversary, Bob, rounds Alice's result in an arbitrary fashion to a new state within a distance of ϵ of the actual outcome. A state t is

pseudo-reachable from s iff Bob can fool Alice into believing that t is reachable in the simulation no matter how accurate her simulation is.

The study of pseudo-orbits go back to Anosov, Bowen, and Conley [7, 25, 45]. Conley [45] formulated the fundamental theorem of dynamical systems: the iteration of any continuous, possibly non-linear, map on a compact metric space decomposes the space into a chain-recurrent part (the pseudo-orbit analogue of a period orbit) and a gradient-like part.

In linear systems theory, *controllability* is a fundamental property of linear systems [164]. Controllability states that the system can be controlled from any point to any other point. However, this may require unboundedly large control actions. A pseudo-orbit can be seen as a stronger notion, where we ask if the dynamics can be controlled from a starting point to an ending point no matter how small the control input is: if a state belongs to the pseudo-orbit, then for every ϵ, there is a sequence of control inputs each bounded in norm by ϵ that steers the system to that state.

In this section, we consider the *pseudo-orbit* and *pseudo-Skolem* problems, corresponding to pseudo-reachability for point and hyperplane target sets, respectively. In short we show that *the pseudo-orbit problem is decidable in polynomial time* and that *the Skolem problem is decidable in full generality on pseudo-orbits.* We proceed in two steps. First, we generalize Kannan and Lipton's analysis to show that the pseudo-orbit problem can be decided in polynomial time. Our proof involves a careful examination of the eigenvalues of the matrix A, similar to Kannan and Lipton's proof. More generally, we show that pseudo-reachability to a bounded semi-algebraic set is decidable. Next, we consider the hyperplane pseudo-reachability (a.k.a. pseudo-Skolem) problem. Our proof again proceeds by a case analysis on the eigenvalues of A. The most interesting case is when there is an eigenvalue of modulus greater than 1. We analyze a series whose terms are polynomial-exponential functions of $n \in \mathbb{N}$ associated with the dynamics. We show that the infimum of this sum can be effectively computed. The proof of effective computability uses tools from Diophantine approximation as well as a reduction to the theory of reals. We show that the dynamics pseudo-reaches the hyperplane in case the infimum of the above sum is 0. If the infimum is non-zero, we prove that we can find an effective bound N such that the dynamics pseudo-reaches the hyperplane iff, for sufficiently small ϵ, it pseudo-reaches the hyperplane within N steps. Putting everything together, we conclude that the pseudo-Skolem problem is decidable.

The content of this section is based on our paper [47]. The following summarizes our main theorem in this section.

Theorem 4.3.1

1. *The pseudo-orbit problem is decidable in polynomial time.*

2. *The hyperplane pseudo-reachability problem is decidable.*

The rest of the section is dedicated to the proof of this theorem.

4.3.1. Problem Statement

The problems that we consider in this section are extensions of the original orbit problem.

Problem 4.5 (Pseudo-orbit problem) ***Inputs:*** *A matrix $A \in \mathbb{Q}^{m\times m}$ and $s, t \in \mathbb{Q}^m$*

Question: *Decide whether $t \in \widetilde{\mathcal{O}}(A, s)$.*

Problem 4.6 (Hyperplane pseudo-reachability problem) ***Inputs:*** *A matrix $A \in \mathbb{Q}^{m\times m}$, $s \in \mathbb{Q}^m$, and a hyperplane $c^\top \cdot x = v$ for $c, v \in \mathbb{Q}^m$*

Question: *Decide whether $\widetilde{\mathcal{O}}_\epsilon(A, s)$ intersects the hyperplane for all $\epsilon > 0$.*

4.3.2. Decidability of the Pseudo-Orbit Problem

Here, we show that Problem 4.5 is decidable in polynomial time. Fix a matrix A and let J be the real Jordan form for A. Proposition 4.1.1 shows that $\widetilde{\mathcal{O}}(A, x)$ can be obtained from the pseudo-orbit $\widetilde{\mathcal{O}}(J, x)$. Our decidability proof involves a case analysis on the modulus of the eigenvalues of J. We first consider the cases where J is a single block, i.e.,

$$J = \begin{bmatrix} \Lambda & I & & \\ & \Lambda & \ddots & \\ & & \ddots & I \\ & & & \Lambda \end{bmatrix} \text{ with } \Lambda = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \text{ and } I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \text{ or } \Lambda = [r] \text{ and } I = [1], \tag{4.4}$$

with real matrix entries $a, b, r \in \mathbb{R}$.

We shall case split on the spectral radius $\rho(J)$, which is the absolute value of the unique eigenvalue of the Jordan block J. We consider three cases: $\rho(J) < 1$, $\rho(J) = 1$ and $\rho(J) > 1$. The following lemma will be useful in relating the first and third cases. Its proof is simply by reversing time.

Lemma 4.3.1 (Reversibility Lemma) *For any invertible matrix $A \in \mathbb{R}^{m\times m}$, $x \in \widetilde{\mathcal{O}}_\epsilon(A, s)$ implies $s \in \widetilde{\mathcal{O}}_\gamma(A^{-1}, x)$ with $\gamma = \epsilon \left\|A^{-1}\right\|_\infty$. Moreover,*

$$x \in \widetilde{\mathcal{O}}(A, s) \iff s \in \widetilde{\mathcal{O}}(A^{-1}, x). \tag{4.5}$$

Lemma 4.3.2 (Eigenvalues inside the unit circle) *Let $J \in \mathbb{R}^{m\times m}$ be a Jordan block of the form (4.4) with $\rho(J) < 1$. For every $s \in \mathbb{R}^m$,*

$$\widetilde{\mathcal{O}}(J, s) = \mathcal{O}(J, s) \cup \{\mathbf{0}\} = \overline{\mathcal{O}(J, s)},$$

where $\overline{\mathcal{O}(J, s)}$ denotes the closure of the orbit.

Proof *We prove the lemma by showing there is a constant $C > 0$ satisfying*

$$\overline{\mathcal{O}(J, s)} \overset{*}{=} \mathcal{O}(J, s) \cup \{\mathbf{0}\} \overset{**}{\subseteq} \widetilde{\mathcal{O}}(J, s) \overset{\dagger}{\subseteq} \bigcap_{\epsilon>0} \bigcup_{z\in\mathcal{O}(J,s)} \mathcal{B}(z, C\epsilon) \overset{\S}{\subseteq} \overline{\mathcal{O}(J, s)}, \tag{4.6}$$

where $\mathcal{B}(z,\epsilon) := \{y \in \mathbb{R}^m \mid \|z-y\|_2 \leq \epsilon\}$ *is the closed ball with respect to two norm with center* z *and radius* ϵ*. It is easy to see that equality (*) holds since all the eigenvalues of* J *are inside the unit circle,* $\lim_{n\to\infty} J^n = 0$*, and* 0 *is the only limiting point of any state trajectory.*

It is also easy to see that inclusion (**) is correct. Note that for any $\epsilon > 0$*,* $\mathcal{O}(J,s) \subseteq \widetilde{\mathcal{O}}_\epsilon(J,s)$ *and the set* $\widetilde{\mathcal{O}}_\epsilon(J,s)$ *is closed by definition. Taking intersection over* $\epsilon > 0$*, we get* $\mathcal{O}(J,s) \subseteq \widetilde{\mathcal{O}}(J,s)$ *with* $\widetilde{\mathcal{O}}(J,s)$ *being a closed set. Therefore,* $\overline{\mathcal{O}(J,s)} \subseteq \widetilde{\mathcal{O}}(J,s)$*.*

We now choose a value of C *which allows us to prove inclusion* (†)*. First pick* γ *such that* $\rho(J) < \gamma < 1$*. Next choose* c_1 *to be a constant (which is guaranteed to exist) satisfying* $\|J^n\|_2 \leq c_1\gamma^n$ *for all* $n \in \mathbb{N}$*, and finally set* $C := c_1 m/(1-\gamma)$*. We show that* $\widetilde{\mathcal{O}}_\epsilon(J,s) \subseteq \bigcup_{z\in\mathcal{O}(J,s)} \mathcal{B}(z, C\epsilon)$ *for any* $\epsilon > 0$*. Take any* $x \in \widetilde{\mathcal{O}}_\epsilon(J,s)$*. Then there is a sequence* $(d_0, d_1, \ldots)$ *and* $n \in \mathbb{N}$ *such that* $\|d_i\|_\infty \leq \epsilon$ *and* $x = J^n s + \sum_{i=0}^{n-1} J^i d_{n-i-1}$*. Now*

$$\|x - J^n s\|_2 = \left\|\sum_{i=0}^{n-1} J^i d_{n-i-1}\right\|_2 \leq \sum_{i=0}^{n-1} \left\|J^i\right\|_2 \|d_{n-i-1}\|_2 \leq \sum_{i=0}^{n-1} c_1\gamma^i m\epsilon \leq \frac{c_1 m\epsilon}{1-\gamma} = C\epsilon,$$

We then get $x \in \mathcal{B}(z, C\epsilon)$ *for* $z := J^n s \in \mathcal{O}(J,s)$*.*

The inclusion § *can be proven by taking an arbitrary point* $y \notin \overline{\mathcal{O}(J,s)}$ *and showing that there is an* $\epsilon > 0$ *for which* $y \notin \mathcal{B}(z, C\epsilon)$ *for all* $z \in \mathcal{O}(J,s)$*. Note that the complement of* $\overline{\mathcal{O}(J,s)}$ *is an open set, which means there is a* $\theta > 0$ *such that* $\mathcal{B}(y,\theta) \cap \overline{\mathcal{O}(J,s)} = \emptyset$*. Taking* ϵ *such that* $C\epsilon < \theta$ *will give the intended result.*

Additionally, we prove the following lemma (that will be useful later) about the behaviour of pseudo-orbits when all eigenvalues are inside the unit circle.

Lemma 4.3.3 *Let* $A \in \mathbb{R}^{m\times m}$ *and* $s \in \mathbb{R}^m$*. If* $\rho(A) < 1$*, then for every* $\delta > 0$ *there exists an effectively computable* $N \in \mathbb{N}$ *and* $\epsilon > 0$ *such that after time* N*, all* ϵ*-pseudo-orbits are contained inside the ball* $\mathcal{B}(\mathbf{0}, \delta)$*.*

Proof *Let* $(x_n)_{n\in\mathbb{N}}$ *denote an* ϵ*-pseudo-orbit starting from* s *with a sequence of disturbances* $(d_n)_{n\in\mathbb{N}}$*. Suppose* $\rho(A) < 1$ *and let* $\gamma \in (\rho(A), 1)$*. There is a constant* $c > 0$ *satisfying* $\|A^n\|_2 \leq c\gamma^n$ *for all* n*. Then we get*

$$\begin{aligned}\|x_n\|_2 = \left\|A^n s + \sum_{k=0}^{n-1} A^k d_{n-k-1}\right\|_2 &\leq \|A^n\|_2 \|s\|_2 + \sum_{k=0}^{n-1} \left\|A^k\right\|_2 \|d_{n-k-1}\|_2 \\ &\leq c\gamma^n \|s\|_2 + \sum_{k=0}^{n-1} m\,\epsilon\, c\gamma^k \leq c\gamma^n \|s\|_2 + \frac{m\,\epsilon\, c}{1-\gamma}.\end{aligned}$$

Taking $\epsilon = \delta(1-\gamma)/(2mc)$ *and* N *with* $\gamma^N \|s\|_2 \leq \delta/(2c)$ *gives the intended result.*

Lemma 4.3.4 (Eigenvalues outside the unit circle) *Let* $J \in \mathbb{R}^{m\times m}$ *be a Jordan block of the form* (4.4) *with* $\rho(J) > 1$*. For every* $s \in \mathbb{R}^m$*, we have* $\widetilde{\mathcal{O}}(J, \mathbf{0}) = \mathbb{R}^m$ *and* $\widetilde{\mathcal{O}}(J,s) = \mathcal{O}(J,s)$ *if* $s \neq \mathbf{0}$*.*

Proof *In this case, J is invertible and all eigenvalues of J^{-1} are inside the unit circle. We apply the Reversibility Lemma 4.3.1 and Lemma 4.3.2.*

$$x \in \widetilde{\mathcal{O}}(J,s) \iff s \in \widetilde{\mathcal{O}}(J^{-1},x) \iff s \in \mathcal{O}(J^{-1},x) \cup \{\mathbf{0}\} \iff s = \mathbf{0} \text{ or } x \in \mathcal{O}(J,s).$$

Therefore, any x is in $\widetilde{\mathcal{O}}(J,s)$ if $s = 0$, and $\widetilde{\mathcal{O}}(J,s) = \mathcal{O}(J,s)$ for $s \neq \mathbf{0}$.

Lemma 4.3.5 (Eigenvalues on the unit circle) *Let $J \in \mathbb{R}^{m\times m}$ be a Jordan block of the form (4.4) with $\rho(J) = 1$. For every $s \in \mathbb{R}^m$, we have $\widetilde{\mathcal{O}}(J,s) = \mathbb{R}^m$.*

Proof *The key part of the proof is to show that $\mathbf{0} \in \widetilde{\mathcal{O}}(A,s)$ for any s and for any A having the eigenvalues on the unit circle. Once we show this, we know that $s \in \widetilde{\mathcal{O}}(A^{-1},\mathbf{0})$ is true for any s and any matrix A due to the Reversibility lemma. Stated for the inverse of A and any x, we get $x \in \widetilde{\mathcal{O}}(A,\mathbf{0})$. Since pseudo-orbits are transitive, we have $x \in \widetilde{\mathcal{O}}(A,s)$ for any x and s, which is the intended result.*

We show $\mathbf{0} \in \widetilde{\mathcal{O}}(A,s)$ equivalently by replacing A with its Jordan form J and doing induction on the structure of J. The proof has two stages. The first stage is to show that $\mathbf{0} \in \widetilde{\mathcal{O}}(J,s)$ for all s when J has a single block simple eigenvalues. The second stage is to show that we can sequentially increase the multiplicity of eigenvalues and multiple blocks.

Base case: *Suppose $J = \begin{bmatrix} a & -b \\ b & a \end{bmatrix}$ with $a^2 + b^2 = 1$ or $J = r$ with $|r| = 1$. Observe that the multiplication by J does not increase the two norm of a vector. Hence setting*

$$d_n = \begin{cases} -\epsilon \cdot \frac{Jx_n}{\|Jx_n\|_2} & \text{if } ||Jx_n||_\infty > \epsilon, \\ -Jx_n & \text{otherwise,} \end{cases}$$

we obtain the ϵ-pseudo-orbit $(x_0 = s, x_1, x_2, \ldots, x_m, \mathbf{0}, \mathbf{0}, \ldots)$ from any s where $\|x_k\|_2 = \|x_{k-1}\|_2 - \epsilon$ for $k \leq m$, which gives $\mathbf{0} \in \widetilde{\mathcal{O}}(J,s)$.

Inductive case: *We show that if $\mathbf{0} \in \widetilde{\mathcal{O}}(J_1,s_1)$ and $\mathbf{0} \in \widetilde{\mathcal{O}}(J_2,s_2)$ for all s_1 and s_2 of appropriate dimensions, we also have $\mathbf{0} \in \widetilde{\mathcal{O}}(J,s)$ with $J = \begin{bmatrix} J_1 & B \\ 0 & J_2 \end{bmatrix}$ for any B and any s with appropriate dimensions. Let us partition any state $x = (x^1, x^2)$ according to the dimensions of J_1 and J_2. Let $\epsilon > 0$ and $s = (s^1, s^2)$. By the assumption, there exist ϵ-perturbations $(d_0^2, d_1^2, \ldots, d_{N-1}^2)$ that bring s^2 to $\mathbf{0}$ under J_2. Let $d_n = (\mathbf{0}, d_n^2)$ for $0 \leq n < N$ be a sequence of ϵ-perturbations for the linear system with mapping J. We obtain the sequence $(x_0 = s, x_1, \ldots, x_N)$ with $x_N^2 = \mathbf{0}$: the ϵ-perturbations $d_0, \ldots, d_{N-1}$ have brought the second coordinate to $\mathbf{0}$. By the assumption, we also have $\mathbf{0} \in \widetilde{\mathcal{O}}_\epsilon(J_1, x_N^1)$, which gives ϵ-perturbations $(d_0^1, \ldots, d_M^1)$ that bring x_N^1 to $\mathbf{0}$ under J_1. Let us expand the sequence of perturbations for the linear system J with $d_{n+N} = (d_n^1, \mathbf{0})$ for $0 \leq n \leq M$. It is easy to see that $(d_0, \ldots, d_{N+M})$ bring the system from s to $\mathbf{0}$ due to the structure of J that is upper triangular.*

We now consider the general case where J has multiple blocks.

Definition 4.3.1 *Let $J \in \mathbb{R}^{m\times m}$ be a real Jordan block matrix and $s \in \mathbb{R}^m$. We define*

$$\Delta(J,s) := \begin{cases} \mathbb{R}^m & \text{if } \rho(J) = 1 \text{ or, } \rho(J) > 1 \text{ and } s = \mathbf{0}, \\ \{\mathbf{0}\} & \text{if } \rho(J) < 1, \\ \emptyset & \text{otherwise.} \end{cases}$$

The following lemma states that certain points in the pseudo-orbit of real Jordan blocks are ϵ-pseudo-reachable exactly at any sufficiently large time step, for every $\epsilon > 0$. The lemma provides the flexibility to "synchronize" reaching parts of the state for different Jordan blocks.

Lemma 4.3.6 (Synchronization Lemma) *Let $J \in \mathbb{R}^{m\times m}$ be a Jordan block with eigenvalue λ. For $s \in \mathbb{R}^m$, $t \in \Delta(J,s)$ if and only if for every $\epsilon > 0$ there exists $N_\epsilon \in \mathbb{N}$ such that for all $N > N_\epsilon$, there exists an ϵ-pseudo-orbit $(x_i)_{i\in\mathbb{N}}$ of s under J such that $x_N = t$.*

Proof

- *$|\lambda| < 1$ and $\Delta(J,s) = \{\mathbf{0}\}$. By Lemma 4.3.2, $\mathbf{0} \in \widetilde{\mathcal{O}}(J,s)$ and hence for every $\epsilon > 0$, there exists N_ϵ such that $t = \mathbf{0}$ can be ϵ-pseudo reached at time N_ϵ. Now simply observe that once an ϵ-pseudo-orbit reaches $\mathbf{0}$, it can remain there forever by setting all future perturbations to zero. To prove the other direction, suppose $t \neq \mathbf{0}$. By Lemma A.4.3, there must exist a time bound T such that for sufficiently small ϵ, all ϵ-pseudo-orbits of s after time T are contained in $\mathcal{B}(\mathbf{0}, \frac{\|t\|_2}{2})$. Hence for sufficiently small ϵ no N_ϵ with the the specified property can exist.*
- *$|\lambda| = 1$ and $\Delta(J,s) = \mathbb{R}^m$. In the proof of Lemma 4.3.5, for every $t \in \mathbb{R}^m$ and $\epsilon > 0$ we construct an ϵ-pseudo-orbit from s that visits $\mathbf{0}$ followed by t. Let N_ϵ be the number of steps required to ϵ-reach t. We can postpone visiting t to any time step $N > N_\epsilon$ by simply waiting at the point $\mathbf{0}$ for $N - N_\epsilon$ steps.*
- *$|\lambda| > 1$, $s = \mathbf{0}$ and $\Delta(J,s) = \mathbb{R}^m$. Similarly to the case above, in Lemma 4.3.4 for each ϵ we construct an ϵ-pseudo-orbit that visits t at time N_ϵ, and reaching t can be delayed arbitrarily by spending a necessary number of steps at $\mathbf{0}$ at the beginning.*
- *$|\lambda| > 1$, $s \neq \mathbf{0}$ and $\Delta(J,s) = \emptyset$. Let $t \in \mathbb{R}^m$. In this case, observe that there must exist a time bound T such that for sufficiently small ϵ, all ϵ-pseudo-orbits of s after time T are contained outside $\mathcal{B}(\mathbf{0}, 2\|t\|_2)$. Hence for sufficiently small ϵ no N_ϵ with the the specified property can exist.*

There are two modes of pseudo reachability: via orbit, or at larger and larger time steps for smaller ϵ.

Lemma 4.3.7 *Let $A \in \mathbb{R}^{m\times m}$ and $s, t \in \mathbb{R}^m$. If there exists N such that for every ϵ, t is ϵ-pseudo-reachable from s within the first N steps, then $t \in \mathcal{O}(A,s)$.*

Proof *Suppose such N exists. By continuity of the map $x \mapsto Ax$, for every $\delta > 0$ there exists $\epsilon > 0$ such that for every $\epsilon' < \epsilon$ and ϵ'-pseudo-orbit $(x_i)_{i\in\mathbb{N}}$, $\left\|x_i - A^i s\right\|_2 < \delta$ for $0 \leq i < N$. Hence the intersection of the first N elements of all ϵ-pseudo-orbits is exactly $\{s, As, \ldots, A^{N-1}s\}$.*

Lemma 4.3.8 *For $J = \mathrm{diag}(J_1, \ldots, J_l)$ in real Jordan normal form and $s \in \mathbb{R}^m$,*

$$\widetilde{\mathcal{O}}(J, s) = \mathcal{O}(J, s) \cup \Pi_{i=1}^{l} \Delta(J_i, s_i).$$

Proof *Suppose $t = (t_1, \ldots, t_l) \in \Pi_{i=1}^{l} \Delta(J_i, s_i)$. That is, for every ϵ and $1 \leq i \leq l$ there exists an ϵ-pseudo-orbit $(x_j^i)_{j \in \mathbb{N}}$ of s_i under J_i that reaches t_i. By Lemma 4.3.6, for every ϵ there exist ϵ-pseudo-orbits $(y_j^i)_{j \in \mathbb{N}}$ of $s_1, \cdots, s_l$ that reach $t_1, \ldots, t_l$, respectively, at the same time N. That is, $y_N^i = t_i$ for $1 \leq i \leq m$. Hence $(y_i^1, \ldots, y_i^z)_{i \in \mathbb{N}}$ is an ϵ-pseudo-orbit of s under J that reaches t.*

Now suppose $t \in \widetilde{\mathcal{O}}(J, s) \setminus O(J, s)$. We prove, by a case analysis on J_i, that $t_i \in \Delta(J_i, s_i)$ for $1 \leq i \leq l$. The main idea is that if t is pseudo-reachable but not reachable, then in order to reach it via an ϵ-pseudo-orbit one will need longer and longer time horizons as $\epsilon \to 0$ (Lemma 4.3.7).

1. *$\rho(J_i) < 1$. Since t is not in the orbit, we can find a sequence $N_1 < N_2 < \cdots$ of time steps and $\epsilon_1 > \epsilon_2 > \cdots$ of perturbations such that t is ϵ_j-reachable from s earliest at time N_j. In particular, t_i is ϵ_j reachable from s_i at time N_j for every j. But by Lemma A.4.3 this means that $|t_i| < \delta$ for every $\delta > 0$. Hence $t_i = \mathbf{0} \in \Delta(J_i, s_i)$.*
2. *$\rho(J_i) = 1$. Since in this case $\Delta(J_i, s_i) = \mathbb{R}^{\kappa(i)}$, trivially $t_i \in \Delta(J_i, s_i)$.*
3. *$\rho(J_i) > 1$ and $s_i = \mathbf{0}$. Since in this case too $\Delta(J_i, s_i) = \mathbb{R}^{\kappa(i)}$, trivially $t_i \in \Delta(J_i, s_i)$.*
4. *$\rho(J_i) > 1$ and $s_i \neq \mathbf{0}$. This case cannot arise, as similarly to Case 1, one can argue that if pseudo-reaching t_i requires larger and larger time steps as $\epsilon \to 0$, then $|t_i| > \delta$ for every δ. But in this case no such t_i can exist.*

Proof *(of Theorem 4.3.1(1)). We now put everything together to show the pseudo-orbit problem is decidable in polynomial time. Given $A \in \mathbb{Q}^{m \times m}$, and $s, t \in \mathbb{Q}^m$, we compute (in polynomial time) matrices $Q, J, Q^{-1} \in (\mathbb{R} \cap \bar{\mathbb{Q}})^{m \times m}$ such that $A = QJQ^{-1}$ and J is in real Jordan normal form [34]. Then, we compute $t' = Q^{-1}t$ and $s' = Q^{-1}s$, and by Proposition 4.1.1 we have that $t \in \widetilde{\mathcal{O}}(A, s)$ if and only if $t' \in \widetilde{\mathcal{O}}(J, s')$. It remains to decide whether $t' \in \widetilde{\mathcal{O}}(J, s')$. For this we use the characterization described in Lemma 4.3.8. To decide whether $t' \in \mathcal{O}(J, s')$, observe that $Q^{-1}t \in \mathcal{O}(J, Q^{-1}s) \iff t \in \mathcal{O}(A, s)$, and whether $t \in \mathcal{O}(A, s)$ is an instance of the orbit problem and can be decided in polynomial time.[1] Finally, it remains to check whether $t_i \in \Delta(J_i, s_i)$ for each block J_i, which can be done easily given the simplicity of $\Delta(J_i, s_i)$.*

We end the part with an application of Theorem 4.3.1(1). A set S is *pseudo-reachable* from s under A if for every $\epsilon > 0$, there exists a point $x_\epsilon \in S$ that is ϵ-pseudo-reachable from s under A. An *algebraic* set is the set of zeros of a collection of polynomials. A *semialgebraic* set is a union of algebraic sets and projections of algebraic sets. We show that we can decide if a bounded semialgebraic set is pseudo-reachable, by reducing the problem to the pseudo-orbit problem.

[1] Technically, [91] consider the orbit problem for rational inputs and we require the orbit problem where the input can contain algebraic numbers. However, a polynomial time algorithm is still possible.

Theorem 4.3.2 *Given $A \in \mathbb{Q}^{m\times m}$, $x_0 \in \mathbb{Q}^m$, and a bounded semialgebraic set S, it is decidable if S is pseudo-reachable from x_0 under A.*

4.3.3. Decidability of the Pseudo-Skolem Problem

In this subsection, we prove Theorem 4.3.1(2). First we consider the case where we are given:

- a hyperplane $H = \{x \in \mathbb{R}^m : c^\top x = v\}$ with $(c, v) \in (\mathbb{R} \cap \bar{\mathbb{Q}})^m \times (\mathbb{R} \cap \bar{\mathbb{Q}})$,
- $J = \mathrm{diag}(J_1, \ldots, J_z) \in (\mathbb{R} \cap \bar{\mathbb{Q}})^{m\times m}$ in real Jordan normal form, and
- a starting point $x_0 \in (\mathbb{R} \cap \bar{\mathbb{Q}})^m$.

We show how to decide if for every $\epsilon > 0$ there exists an ϵ-pseudo-orbit $(x_i)_{i\in\mathbb{N}}$ of x_0 under J that *hits* the hyperplane H, i.e. $c^\top x_N - v = 0$ for some $N \in \mathbb{N}$.

A block J_i is *relevant* with respect to hyperplane $H = \{x : c^\top x = v\}$ if the coefficients of c at the coordinates corresponding to J_i are not all 0. Intuitively, dimensions corresponding to blocks that are not relevant can simply be omitted from the analysis as they do not play a role in determining whether a point is in H or not. *Relevant eigenvalues* of J are the eigenvalues of relevant blocks. The *relevant spectral radius*, written $\rho_H(J)$, is the largest modulus of all relevant eigenvalues. Our proof is based on a case analysis on the relevant spectral radius of J. We shall see that the proof is simple when the relevant spectral radius is ≤ 1 but requires more technical ideas when it is > 1.

Lemma 4.3.9 (Case $\rho_H(J) \leq 1$) *Fix a matrix J in real Jordan normal form, a starting state x_0, and a hyperplane $H = \{x : c^\top x = v\}$.*

1. *If $\rho_H(J) = 1$, then H is pseudo-reachable.*
2. *If $\rho_H(J) < 1$ and $\mathbf{0} \in H$ then H is pseudo-reachable. If $\rho_H(J) < 1$ and $\mathbf{0} \notin H$, there exists an effectively computable time bound N such that H is pseudo-reachable if and only if there exists $0 \leq i \leq N$ such that $J^i x_0 \in H$ (that is, H is reachable from x_0 under J after at most N steps).*

Proof *First suppose $\rho_H(J) = 1$. We write $J = diag(J_h, J_r)$, where $\rho_H(J_h) = 1$ and $\rho_H(J_r) < 1$ (observe that wlog we can assume the blocks of J have non-decreasing spectral radius when listed from top to bottom) and correspondingly set $s = (s_h, s_r)$, $c = (c_h, c_r)$. Note that $c_h \neq 0$ by the relevance of at least one of eigenvalues of modulus 1.*

By Lemma 4.3.2 we know $\mathbf{0} \in \widetilde{\mathcal{O}}(J_r, s_r)$. By Lemma 4.3.5, we can select y such that $c_h^\top y - v = 0$ and $y \in \widetilde{\mathcal{O}}(J_h, s_h)$. Therefore, invoking Lemma 4.3.6, for every $\epsilon > 0$ we can find $N \in \mathbb{N}$ and construct ϵ-pseudo-orbits $(x_n^h)_{n\in\mathbb{N}}$ and $(x_n^r)_{n\in\mathbb{N}}$ such that $x_N^h = y$ and $x_N^r = \mathbf{0}$, which implies that for the ϵ-pseudo-orbit $x_n = (x_n^h, x_n^r)$, $c^\top x_N - v = c_h^\top y + c_r^\top \mathbf{0} - v = 0$ as desired.

Now suppose $\rho_H(J) < 1$.

Case 1: $v = 0$. *Since* $\mathbf{0} \in H$ *and the origin is pseudo-reachable from* x_0 *(Lemma 4.3.2),* H *is pseudo-reachable.*

Case 2: $v \neq 0$. *Using Lemma A.4.3, and setting* $\delta = |v|/(2\|c\|_2)$*, we can find* $\epsilon > 0$ *and horizon* $N \in \mathbb{N}$ *after which every* ϵ*-pseudo-orbit is trapped in* $\mathcal{B}(0, \delta)$*. Thus, the hyperplane cannot be pseudo-reached after time* N*, as the hyperplane does not intersect with* $\mathcal{B}(0, \delta)$*. It remains to check if the hyperplane is pseudo-reachable at any of the first* N *time-steps. In fact, for a bounded time interval, a hyperplane is pseudo-reachable iff it is reachable. This is because the effect of finitely many disturbance terms* $(d_0, \ldots, d_{N-1})$ *can be made arbitrarily small for small enough* ϵ*. Therefore, decidability in this case only requires checking if the bounded orbit* $(y_n)_{0 \leq n \leq N}$ *hits the hyperplane before the time horizon* N*, that is, if there exists a time-step* $0 \leq n \leq N$ *such that* $c^\top y_n - v = 0$*, which is clearly decidable.*

We now consider the case $\rho_H(J) > 1$. The main ideas of our proof are as follows:

1. A point x_n in the ϵ-pseudo-orbit belongs to the hyperplane (c, v) if $c^\top x_n - v = 0$. In particular, $c^\top x_n - v$ can be written as a sum over exponential polynomials in eigenvalues of different sizes.

2. We factor out the scaling factor corresponding to the top eigenvalues, leaving a sum over normalized eigenvalues, together with a sum over disturbances (of order ϵ) and additional terms which go to zero with large n.

3. We relate hyperplane pseudo-reachability to the limit inferior of the sum over normalized eigenvalues. If the limit is zero, we show the hyperplane is pseudo-reachable. If the limit is positive, we show there is an effective bound N such that if the hyperplane is pseudo-reachable, it is reachable within N steps.

4. We apply results from Diophantine approximation and the theory of reals to compute the limit inferior of the sum over normalized eigenvalues.

Fix $J = \operatorname{diag}(J_1, \ldots, J_l) \in (\mathbb{R} \cap \bar{\mathbb{Q}})^{m \times m}$, a starting point $x_0 \in (\mathbb{R} \cap \bar{\mathbb{Q}})^m$, and a hyperplane $H = \{x \in \mathbb{R}^m \mid c^\top x = v\}$ with $c, v \in (\mathbb{R} \cap \bar{\mathbb{Q}})^m$. We assume without loss of generality that all blocks are relevant.

Step 1: Analyzing $c^\top x_n - v$. Let $L = \rho_H(J) > 1$ be the largest modulus of a relevant eigenvalue of J and suppose the blocks are arranged in non-increasing order of the modulus of eigenvalues. In particular, let $t \leq l$ be such that the first t blocks (t for "top") have $\rho(J_1) = \cdots = \rho(J_t) = L > 1$. We call the eigenvalues of these blocks the *top eigenvalues*. The remaining blocks satisfy $L > \rho(J_{t+1}) \geq \cdots \geq \rho(J_l)$.

Let $(d_i)_{i \in \mathbb{N}}$ be a sequence of perturbations and $(x_i)_{i \in \mathbb{N}}$ the resulting pseudo-orbit. We have that for all time steps n,

$$c^\top x_n - v = c^\top \left(J^n x_0 + \sum_{k=0}^{n-1} J^k d_{n-k-1} \right) - v = \sum_{i=1}^{l} \left(c^i J_i^n x_0^i + c^i \sum_{k=0}^{n-1} J_i^k d_{n-k-1}^i \right) - v,$$

where for all $1 \leq i \leq l$, c^i, x_n^i d_n^i are projections of $c^\top$, x_n and d_n, respectively, onto the coordinates governed by J_i. Observe that c^i is a row vector for every i.

Step 2: Normalized sum. We define a normalized version of this sum by factoring out L^n (the size of the top eigenvalues) and n^D, where we define D in such a way that we normalize polynomials in n that appear in the sum. Observe that for $1 \leq i \leq t$ (the top eigenvalues),

$$c^i J_i^n = \begin{cases} \left[p_1^i(n)\lambda^n + \overline{p_1^i(n)\lambda^n} \quad \cdots \quad p_{2\kappa(i)}^i(n)\lambda^n + \overline{p_{2\kappa(i)}^i(n)\lambda^n} \right] & \text{if } J_i \text{ has eigenvalues } \lambda, \overline{\lambda} \\ \left[p_1^i(n)\rho^n \quad \cdots \quad p_{\kappa(i)}^i(n)\rho^n \right] & \text{if } J_i \text{ has a single eigenvalue } \rho \end{cases}$$

for polynomials $p_1^i, \ldots, p_{\kappa(i)}^i$ (with algebraic coefficients) where $\kappa(i)$ is the multiplicity of the block J_i.

We define D to be the largest number such that the monomial n^D appears with a non-zero coefficient in at least one of $c^i J_i^n$ for $1 \leq i \leq t$. (Note that if all entries of c are non-zero $D+1$ is equal to the largest multiplicity of a top eigenvalue block of J, as can be seen from the description of powers of a Jordan block in Subsection 4.1.4.)

We can now define

$$f(n) := \frac{c^\top \cdot x_n - v}{L^n n^D} = \sum_{i=1}^{l} \left(c^i \frac{J_i^n}{L^n n^D} x_0^i + c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^D} d_{n-k-1}^i \right) - \frac{v}{L^n n^D}$$

For notational convenience we define vector-valued functions $g^i(n) := c^i \frac{J_i^n}{L^n n^D}$ for $1 \leq i \leq l$. The following technical lemma summarizes the relevant properties of these scaled terms.

Lemma 4.3.10 (Normalization Lemma)

1. *For $1 \leq i \leq t$ (top eigenvalues), $\left\|g^i(n)\right\|_\infty = O(1)$ (with respect to n).*
2. *For $t+1 \leq i \leq l$ (non-top eigenvalues), $\lim_{n\to\infty} \left\|g^i(n)\right\|_\infty = 0$.*
3. *There exists $1 \leq j \leq t$ and effectively computable $N \in \mathbb{N}$ and $C > 0$ such that $n > N \implies \left\|g^j(n)\right\|_\infty > C$.*

Proof *We address each point individually.*

1: For $1 \leq i \leq t$ let J_i have eigenvalues λ and $\overline{\lambda}$ (the case where J_i has a single real eigenvalue is similar but simpler) and observe that

$$g^i(n) = \left[\frac{p_1^i(n)}{n^D} \left(\frac{\lambda}{L}\right)^n + \overline{\frac{p_1^i(n)}{n^D}} \left(\frac{\overline{\lambda}}{L}\right)^n \quad \cdots \quad \frac{p_{2\kappa(i)}^i(n)}{n^D} \left(\frac{\lambda}{L}\right)^n + \overline{\frac{p_{2\kappa(i)}^i(n)}{n^D}} \left(\frac{\overline{\lambda}}{L}\right)^n \right].$$

By the definition of top eigenvalues, $|\lambda| = L$ and thus $\frac{\lambda}{L}$ and $\frac{\overline{\lambda}}{L}$ have modulus 1. By construction of n^D, the polynomials $p_1^i(n), \ldots, p_{2\kappa(i)}^i$ all have degree at most D and hence the terms $\frac{p_1^i(n)}{n^D}, \ldots, \frac{p_{2\kappa(i)}^i(n)}{n^D}$ are bounded from above by a constant.

2: *For $t+1 \leq i \leq l$ let J_i have eigenvalues λ and $\overline{\lambda}$ and observe that*

$$g^i(n) = \left[\frac{p_1^i(n)}{n^D} \left(\frac{\lambda}{L}\right)^n + \frac{\overline{p_1^i(n)}}{n^D} \left(\frac{\overline{\lambda}}{L}\right)^n \quad \cdots \quad \frac{p_{\kappa(i)}^i(n)}{n^D} \left(\frac{\lambda}{L}\right)^n + \frac{\overline{p_{\kappa(i)}^i(n)}}{n^D} \left(\frac{\overline{\lambda}}{L}\right)^n \right].$$

By construction $|\lambda| < L$ and thus $\gamma := \frac{\lambda}{L}$ and $\overline{\gamma}$ have moduli $|\gamma|, |\overline{\gamma}| < 1$. The polynomials $p_1^i(n), \ldots, p_{2\kappa(i)}^i(n)$ may not be asymptotically bounded by n^D (since n^D was constructed only considering top eigenvalues). However, it is clear that the exponentially vanishing $\left(\frac{\lambda}{L}\right)^n$ and $\left(\frac{\overline{\lambda}}{L}\right)^n$ will dominate the polynomials and all entries of $g^i(n)$ will thus vanish.

3: *Observe that by construction of n^D, there must exist a top eigenvalue block J_j ($1 \leq j \leq t$) for which at least one polynomial in $c^j J_j^n$ has degree D. Let $r > D$ be the multiplicity of the block J_j, which has the form of a real Jordan matrix with a single block (Eq. (4.4)) with sub-blocks Λ. One can write*

$$c^j J_j^n = \begin{bmatrix} c_r^j & c_{r-1}^j & \cdots & c_0^j \end{bmatrix} \begin{pmatrix} \Lambda^n & n\Lambda^{n-1} & \binom{n}{2}\Lambda^{n-1} & \cdots & \binom{n}{r-1}\Lambda^{n-r+1} \\ 0 & \Lambda^n & n\Lambda^{n-1} & \cdots & \binom{n}{r-2}\Lambda^{n-r+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & n\Lambda^{n-1} \\ 0 & 0 & 0 & \cdots & \Lambda^n \end{pmatrix}, \tag{4.7}$$

where c_k^j for $1 \leq k \leq r$ corresponds to a row vector of size two or one, respectively, when Λ is a 2×2 or 1×1 matrix. Analyzing this product, we see that $c_r^j, \ldots, c_{D+1}^j = \mathbf{0}$, $c_D^j \neq \mathbf{0}$ and the single entry of $c^j J_j^n$ whose polynomial component has degree D is exactly $c_D^j \binom{n}{D} \Lambda^{n-D}$.

We define $\hat{\Lambda} := \Lambda / L$. Note that $\left\| \hat{\Lambda} \right\|_2 = 1$. Now observe that for this block J_j, we have

$$g^j(n) = \frac{1}{L^n n^D} c^j J_i^n = c_D^j \frac{1}{D!} \hat{\Lambda}^n + \frac{1}{n}(O(1)).$$

Therefore, there exists sufficiently large N such that for all $n \in \mathbb{N}$,

$$n > N \implies \left\| \frac{1}{L^n n^D} c^j J_j^n \right\|_\infty > \frac{1}{2} \left\| c_D^j \frac{1}{D!} \hat{\Lambda}^n \right\|_\infty > \frac{\left\| c_D^j \right\|_2}{4D!}.$$

Thus we have shown Point 3 with $C = \frac{\left\| c_D^j \right\|_2}{4D!}$.

Step 3: Conditions for reachability and non-reachability. Now we are ready to attack our original problem. Going back, H is ϵ-pseudo-reachable if and only if $f(n) = 0$ for some disturbance sequence $(d_i)_{i \in \mathbb{N}}$ with $d_i \in [-\epsilon, \epsilon]^m$ for all i. We analyze how $f(n)$ can be brought to 0 in this way.

Lemma 4.3.11 *Let*

$$D = \liminf_{n\to\infty} \left| \sum_{i=1}^{t} g^i(n) x_0^i \right|. \tag{4.8}$$

If $D = 0$, then H is pseudo-reachable. If $D > 0$, there exists a computable time bound N such that H is pseudo-reachable if and only if it is reachable (in the standard sense) within the first N steps.

Proof *Suppose $D = 0$. Take an arbitrary $\epsilon > 0$. We argue that H is ϵ-pseudo-reachable. Recall that*

$$\begin{aligned} f(n) &= \sum_{i=1}^{l} \left(c^i \frac{J_i^n}{L^n n^D} x_0^i + c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^D} d_{n-k-1}^i \right) - \frac{v}{L^n n^D} \\ &= \sum_{i=1}^{l} \left(g^i(n) x_0^i + c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^D} d_{n-k-1}^i \right) - \frac{v}{L^n n^D}. \end{aligned}$$

Let I be such that $\left\|g^I(n)\right\|_\infty > C$, for $C > 0$ and sufficiently large n (Point 3 of the Normalization Lemma). We construct a pseudo-orbit with all perturbations set to zero except d_0^i and obtain

$$f(n) = c^I \frac{J_I^{n-1}}{L^n n^D} d_0^I + \sum_{i=1}^{t} g^i(n) x_0^i + \sum_{i=t+1}^{l} c^i \frac{J_i^n}{L^n n^D} x_0^i - \frac{v}{L^n n^D}.$$

Intuitively, we will use the term $c^I \frac{J_I^{n-1}}{L^n n^D} d_0^I$ to cancel out the remaining summands above, but we have to argue that this can be done using a disturbance of size at most ϵ. Moreover, observe that $c^I \frac{J_I^{n-1}}{L^n n^D}$ is very close to $g^I(n)$. Formally, we first find N large enough such that

- $\left\|g^I(N)\right\|_\infty > C$,
- $\left\| \sum_{i=t+1}^{l} c^i \frac{J_i^N}{L^N N^D} x_0^i - \frac{v}{L^N N^D} \right\|_\infty < \frac{C^2}{\|J_i\|_\infty} \frac{\epsilon}{2}$ *(possible because for $t+1 \le i \le l$, $\rho(J_i) < 1$ and $L > 1$), and*
- $\left\| \sum_{i=1}^{t} g^i(N) x_0^i \right\|_\infty < \frac{C^2}{\|J_i\|_\infty} \frac{\epsilon}{2}$ *(possible because $\liminf_{n\to\infty} \left\| \sum_{i=1}^{t} g^i(n) x_n^i \right\| = 0$).*

Finally, we determine the value of d_0^i. Without loss of generality, assume that $g^I(N)$ is of the form $\begin{bmatrix} C' & \cdots \end{bmatrix}$ where $|C'| > C$, that is the first entry of $g^I(N)$ is large. We then observe that $c^I \frac{J_I^{N-1}}{L^N N^D} d_0^I = g^I(N) J_I^{-1} d_0^I$ and set

$$d_0^I = J_I \cdot \left[-\frac{1}{C'} \left(\sum_{i=1}^{t} g^i(N) x_0^i + \sum_{i=t+1}^{l} c^i \frac{J_i^N}{L^N N^D} x_0^i - \frac{v}{L^N N^D} \right) \quad 0 \quad 0 \quad \cdots \quad 0 \right]^\top$$

to obtain

$$c^I \frac{J_I^{N-1}}{L^N N^D} d_0^I = - \left(\sum_{i=1}^{t} g^i(N) x_0^i + \sum_{i=t+1}^{l} c^i \frac{J_i^N}{L^N N^D} x_0^i - \frac{v}{L^N N^D} \right)$$

and hence $f(N) = 0$.

Now suppose $D > 0$. *Recall*

$$\begin{aligned} f(n) &= \sum_{i=1}^{l} \left(g^i(n)x_0^i + c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^D} d_{n-k-1}^i \right) - \frac{v}{L^n n^D} \\ &= \sum_{i=1}^{t} g^i(n)x_0^i + \sum_{i=t+1}^{l} c^i \frac{J_i^n}{L^n n^D} x_0^i + \sum_{i=1}^{l} c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^D} d_{n-k-1}^i - \frac{v}{L^n n^D}. \end{aligned}$$

In this case we shall construct a time bound N *after which for all sufficiently small value of* ϵ, *the term* $\sum_{i=1}^{t} g^i(n)x_0^i$ *will dominate the other summands. Let* $2\Delta > 0$ *be a lower bound on* $\liminf_{n\to\infty} |\sum_{i=1}^{t} g^i(n)x_0^i| > 0$. *We shall see how to obtain such a bound effectively later (Lemma 4.3.12). We compute* N *with the following properties.*

- *For all* $n > N$, $\left|\sum_{i=1}^{t} g^i(n)x_0^i\right| > \Delta$. *Possible because* $\liminf_{n\to\infty} \left|\sum_{i=1}^{t} g^i(n)x_n^i\right| > 2\Delta$.
- *For all* $n > N$, $\left|\sum_{i=t+1}^{l} c^i \frac{J_i^n}{L^n n^d} x_0^i\right|, \left|\frac{v}{L^n n^d}\right| \ll \Delta$. *The former is possible because for* $t+1 \le i \le l$, $\rho(J_i) < L$.
- *For sufficiently small* ϵ, *for all* $n > N$, $\left|c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^d} d_{n-k-1}^i\right| \ll \Delta$ *for* $1 \le i \le l$. *To see that this is always possible, observe that*

  $$\left|c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^d} d_{n-k-1}^i\right| \le \sum_{k=0}^{n-1} \left\|c^i \frac{J_i^k}{L^n n^d}\right\|_\infty M\epsilon$$ *(where fixed* M *bounds the matrix dimension)*

  and

  $$\lim_{n\to\infty} \sum_{k=0}^{n} \left\|c^i \frac{J_i^k}{L^n n^d}\right\|_\infty \le \lim_{n\to\infty} \sum_{k=0}^{n} \left\|c^i \frac{1}{L^{n-k}} \frac{J_i^k}{L^k k^d}\right\|_\infty = \lim_{n\to\infty} \sum_{k=0}^{n} \left\|\frac{1}{L^{n-k}} g^i(k)\right\|_\infty.$$

  Recalling Point 1 of the Normalization Lemma, $\left\|g^i(n)\right\|_\infty = O(1)$ *and hence*

  $$\lim_{n\to\infty} \sum_{k=0}^{n} \left\|\frac{1}{L^{n-k}} g^i(k)\right\|_\infty = O(1),$$

  by bounding the sum $\sum_{k=0}^{n} \left\|\frac{1}{L^{n-k}} g^i(k)\right\|_\infty$ *from above by a geometric sequence. Therefore,* $\sum_{k=0}^{n-1} \left\|c^i \frac{J_i^k}{L^n n^d}\right\|_\infty M\epsilon$ *can be made* $\ll \Delta$ *by choosing* ϵ *to be sufficiently small.*

Once we have chosen N, *by the properties above we will have that for all* $n > N$, *for sufficiently small* ϵ,

$$|f(n)| \ge \left|\sum_{i=1}^{t} g^i(n)x_0^i\right| - \left|\sum_{i=t+1}^{l} c^i \frac{J_i^n}{L^n n^d} x_0^i + \sum_{i=1}^{l} c^i \sum_{k=0}^{n-1} \frac{J_i^k}{L^n n^d} d_{n-k-1}^i - \frac{v}{L^n n^d}\right| > 0.$$

Therefore, H is pseudo-reachable if and only if for every $\epsilon > 0$, H is ϵ-pseudo-reachable within the first N steps. By Lemma 4.3.7, this is the case if and only if H is reachable within the first N steps.

Step 4: Analyzing $\liminf_{n\to\infty} |\sum_{i=1}^{t} g^i(n)x_0^i|$. Consider a single term $g^i(n)x_0^i$. Writing $x_0^i = \begin{bmatrix} X_0 & X_1 & \cdots & X_z \end{bmatrix}^\top$, where $X_1, \ldots, X_z \in \mathbb{R}$, we have

$$g^i(n)x_0^i = \sum_{r=1}^{z} \left(\frac{p_r^i(n)}{n^D} \left(\frac{\lambda}{L}\right)^n + \frac{\overline{p_r^i(n)}}{n^D} \left(\frac{\overline{\lambda}}{L}\right)^n \right) X_z.$$

Let $\gamma_i = \frac{\lambda}{L}$. Note that $|\gamma_i| = 1$. By the construction of n^D, none of the polynomials have a term of degree higher than D. Therefore, we can absorb the constants X_r and the monomial n^D into the polynomials, sum the terms up, and write them as polynomials in $\frac{1}{n}$. That is,

$$g^i(n)x_0^i = q^i(1/n)\gamma_i^n + \overline{q^i(1/n)}\overline{\gamma_i}^n$$

for suitable polynomials q^i with algebraic coefficients. Thus

$$\liminf_{n\to\infty} \left| \sum_{i=1}^{t} g^i(n)x_n^i \right| = \liminf_{n\to\infty} \left| \sum_{i=1}^{t} q^i(1/n)\gamma_i^n + \overline{q^i(1/n)}\overline{\gamma_i}^n \right|$$

We defer the proof of the following lemma, which requires tools from Diophantine analysis and the theory of reals, to the Appendix A.4.6.

Lemma 4.3.12 *Let $\gamma_1, \ldots, \gamma_t$ be algebraic numbers with modulus 1. Let $q^1, \ldots, q^t$ be polynomials with algebraic coefficients. The quantity*

$$\liminf_{n\to\infty} \left| \sum_{i=1}^{t} q^i(1/n)\gamma_i^n + \overline{q^i(1/n)}\overline{\gamma_i}^n \right|$$

can be effectively computed. If it is greater than zero, there is an effectively computable N satisfying the requirement of Lemma 4.3.11.

Proof of Theorem 4.3.1(2). We are now ready to aggregate our case analysis into the proof the pseudo-reachability in hyperplanes is decidable. Given $A \in \mathbb{Q}^{m\times m}$, $x_0 \in \mathbb{Q}^m$ and $H = \{x : c^\top \cdot x = 0\}$, we first convert A to real Jordan normal form as described in Section 4.1.4 to obtain $J = Q^{-1}AQ$. We then perform a coordinate transform on x_0 and H to obtain $H' = \{x : c^\top Qx = 0\}$ and $x_0' = Q^{-1}x_0$. The original problem is now equivalent to pseudo-reachability of H' from x_0' under J.

Next, we remove dimensions from $x_0', c^\top Q$ and J that do not correspond to relevant blocks and determine the relevant spectral radius $\rho_H(J)$ of J. If $\rho_H(J) = 1$ then H' is reachable by Lemma 4.3.9(1). If $\rho_H(J) < 1$, then by Lemma 4.3.9(2), H' is pseudo-reachable if and only if $\mathbf{0} \in H'$ or $x_0', Jx_0', \ldots, J^N x_0'$ hits H', where N is the computable bound in the Lemma.

Finally, we consider the case where $\rho_H(J) > 1$. Let $J_1, \ldots, J_t$ be the blocks of J with $\rho(J) = \rho_H(J)$ and $c^1, \ldots, c^t$, $x_0^1, \ldots, x_0^t$ be the corresponding coordinates of $c^\top Q$ and x'_0, respectively. Finally, compute the value of $\liminf_{n\to\infty} \left|\sum_{i=1}^{t} g^i(n) x_n^i\right|$ using Lemma 4.3.12 and use Lemma 4.3.11 to either immediately conclude reachability or to compute the bound N and determine reachability by checking if $x'_0, Jx'_0, \ldots, J^N x'_0$ hits H'.

4.4. Conclusion

In this chapter, we began with establishing the computational complexity of the point-to-point reachability problem for discrete-time linear dynamical systems featuring hypercubic disturbance (or control) sets. Motivated by the hardness results of this problem, we turned our attention to the pseudo-Skolem problem for linear dynamical systems, which asks whether a hyperplane target set can be reached under *all* of the hypercubic disturbance sets that are centered at the origin and have non-zero volumes. Our findings revealed that this problem is indeed decidable, in contrast to the enduring mystery of the decidability of the well-known Skolem problem, which has eluded resolution despite decades of dedicated research. In [48], we extend the results of Section 4.3.3 and show decidability of pseudo-reachability problem for arbitrary semi-algebraic sets when the state matrix is diagonalizable.

5

Conclusion and Future Work

In this thesis, we have considered the controller synthesis problem for different classes of dynamical systems and specifications. For more general classes, we focused on improving scalability of the existing sound controller synthesis methods by proposing novel ideas. For restricted classes of dynamics and specifications, we proposed novel sound and complete decision procedures which enabled us to tackle unsolved problems. Below, we provide a brief description of outcomes presented in this thesis.

Broadening the scope of abstraction-based controller design for nonlinear dynamical systems. In Chapter 2, we considered the general class of non-linear dynamical systems with bounded disturbances and focused on broadening the scope of abstraction-based controller synthesis for infinite-horizon temporal specifications. We identified three key shortcoming of ABCD method, that are (1) requirement for knowing the analytical dynamics, (2) huge memory requirements, and (3) inapplicability of ABCD for multi-agent scenarios. In Section 2.2, we proposed a data-driven method that could learn an abstraction which was correct with respect to the given confidence and could be used for synthesizing controllers against the infinite-horizon temporal specifications. In Section 2.3, we proposed a memory-efficient method which was based on training neural representations for the finite abstraction and also the computed controller. Through extensive experiments, we empirically illustrated that our method was very successful in reducing the memory requirements for ABCD method. Finally, in Section 2.4, we considered the setting wherein a heterogeneous population of agents need to fulfill a given joint reach-avoid task while no communication among them is allowed. Our proposed method consists of a centralized planning step for computing nominal temporal-spatial (open-loop) trajectories for all of the agents, and a decentralized tracking step that uses ABCD to synthesize feedback controllers providing guaranteed track of the open-loop trajectories computed in the first step. We empirically illustrated that our method can efficiently solve different multi-agent scenarios, including formation control, lane merging and multi-agent reach-avoid tasks.

Decidability and scalability of time-bounded reachability for CTMDPs. In Chapter 3, we considered the time-bounded reachability problem for CTMDPs from both scalability and decidability perspectives. For the first time, we provided a conditional decidability result for the time-bounded reachability problem of CTMDPs by using tools from number theory. Subsequently, we considered the scalability issue for approximating the value of time-bounded reachability for both CTMCs and CTMDPs, and provided a

control theoretic method for reducing the state space size, while providing formal error bounds.

Reachability-like specifications for linear dynamical systems. In Chapter4, we first proved the hardness of the point-to-point reachability problem for discrete time linear dynamical systems with hypercubic disturbance (equivalently, control) sets. Motivated by this result, in Section 4.3, we considered the pseudo-Skolem problem for linear dynamical systems, which asks whether a hyperplane target set can be reached under *all* of the hypercubic disturbance sets that are centered at the origin and have non-zero volumes. We proved that this problem is indeed decidable, although the decidability of the famous Skolem problem remains open despite decades of continuous effort.

Future Work

In this thesis, we studied several interesting questions that are related to guaranteed controller design for cyber-physical systems. But, many more are remained open. Following is a list of interesting problems that can be pursued in future.

Further development of ABCD for stochastic systems. In this thesis, we mainly considered application of ABCD for systems with bounded disturbances. As a result of considering worst-case disturbances, the outcome controller would become relatively conservative. The level of conservativeness can be reduced significantly. However, the methods we developed in this thesis cannot be directly used to tackle the corresponding bottlenecks of ABCD when applied to the stochastic systems. Therefore, one meaningful direction for future research would be to develop solutions that are tailored for application to the stochastic systems.

Reducing the conservativeness resulted from overapproximations. A usual complaint about ABCD is that it cannot (successfully) be applied to solving synthesis problems for high-dimensional systems. Although the exponential growth in number of states and transitions is an important computational burden, increasing computational resources (e.g., through parallel implementations) can occasionally be very helpful. A more fundamental problem is resulted by conservative overapproximations of the dynamics. Majority of the existing methods compute a growth bound that is valid across the whole state space. However, such a growth bound is often extremely conservative and drastically lowers the chances for getting a useful controller. In order to reduce the level of conservativeness, one suggestion would be to apply local over-approximations as much as possible. This increases the size of winning domain and hence can potentially eliminate excessive reduction in size of discretizations, which can also lower the computational costs.

Improving scalability of ABCD by utilizing structural sparsity in dynamics. Structure of many systems of interest is sparse, meaning that evolution of some states only influences a few other states immediately. We believe that this feature can be used in order to enhance scalability of ABCD for many of the systems.

Investigating possible security consequences of the decidability results for the pseudo-orbit and pseudo-Skolem problems. In Chapter 4, we illustrated that for

certain cases, even when the target is far from the trajectories of the unperturbed system, an adversary can use an arbitrarily small sequence of perturbations that steer the trajectory of the system to actually hit the target that was unreachable by the original dynamics. This can potentially raise security concerns and definitely is worth to be carefully studied.

Bibliography

[1] 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, 2008.

[2] M. J. Ablowitz and A. S. Fokas. *Complex Variables: Introduction and Applications*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2003.

[3] A. Ahmad, C. Belta, and R. Tron. Adaptive sampling-based motion planning with control barrier functions. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 4513–4518. IEEE, 2022.

[4] S. Akshay, T. Antonopoulos, J. Ouaknine, and J. Worrell. Reachability problems for Markov chains. *Inf. Process. Lett.*, 115(2):155–158, 2015.

[5] J. Alonso-Mora, E. Montijano, T. Nägeli, O. Hilliges, M. Schwager, and D. Rus. Distributed multi-robot formation control in dynamic environments. *Autonomous Robots*, 43(5):1079–1100, 2019.

[6] R. Alur, S. Moarref, and U. Topcu. Pattern-based refinement of assume-guarantee specifications in reactive synthesis. In *TACAS*. Springer, 2015.

[7] D. V. Anosov. Geodesic flows on closed Riemannian manifolds of negative curvature. *Proc. Steklov Inst. Math.*, 90, 1967.

[8] A. M. Ayala, S. B. Andersson, and C. Belta. Formal synthesis of control policies for continuous time Markov processes from time-bounded temporal logic specifications. *IEEE Trans. Automat. Contr.*, 59(9):2568–2573, 2014.

[9] A. Aziz, K. Sanwal, V. Singhal, and R. K. Brayton. Model-checking continous-time Markov chains. *ACM Trans. Comput. Log.*, 1(1):162–170, 2000.

[10] G. Bacci, G. Bacci, G. Larsen, and R. Mardare. On the total variation distance of semi-Markov chains. In *FoSSaCS, Lecture Notes in Computer Science*, pages 185–199. Springer Berlin Heidelberg, 2015.

[11] T. S. Badings, A. Abate, N. Jansen, D. Parker, H. A. Poonawala, and M. Stoelinga. Sampling-based robust control of autonomous systems with non-Gaussian noise. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(9):9669–9678, June 2022.

[12] T. S. Badings, L. Romao, A. Abate, D. Parker, H. A. Poonawala, M. Stoelinga, and N. Jansen. Robust control for dynamical systems with non-gaussian noise via formal abstractions. *J. Artif. Intell. Res.*, 76:341–391, 2023.

[13] C. Baier and J.-P. Katoen. *Principles of model checking*. MIT press, 2008.

[14] C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Transactions on Software Engineering*, 29(6):524–541, 2003.

[15] C. Baier, H. Hermanns, J. Katoen, and B. R. Haverkort. Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. *Theor. Comput. Sci.*, 345(1):2–26, 2005.

[16] A. Bajcsy, S. Bansal, E. Bronstein, V. Tolani, and C. J. Tomlin. An efficient reachability-based framework for provably safe autonomous navigation in unknown environments. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 1758–1765. IEEE, 2019.

[17] S. Bansal, M. Chen, J. F. Fisac, and C. J. Tomlin. Safe sequential path planning of multi-vehicle systems under presence of disturbances and imperfect information. In *ACC*, 2017.

[18] G. Banusic, R. Majumdar, M. Pirron, A. Schmuck, and D. Zufferey. PGCD: robot programming and verification with geometry, concurrency, and dynamics. In *ICCPS 2019, Montreal, QC, Canada*, 2019.

[19] A. Barto, R. Sutton, and C. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE TSMCS*, SMC-13(5):834–846, Sept. 1983.

[20] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer, 2006.

[21] P. C. Bell, J. Delvenne, R. M. Jungers, and V. D. Blondel. The continuous Skolem-Pisot problem. *Theor. Comput. Sci.*, 411(40-42):3625–3634, 2010.

[22] C. Belta, B. Yordanov, and E. A. Gol. *Formal methods for discrete-time dynamical systems*, volume 15. Springer, 2017.

[23] G. O. Berger, R. M. Jungers, and Z. Wang. Data-driven invariant subspace identification for black-box switched linear systems. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 32–37. IEEE, 2022.

[24] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[25] R. Bowen. *Equilibrium States and the Ergodic Theory of Anosov Diffeomorphisms*, volume 470 of *Lecture Notes in Mathematics*. Springer-Verlag, 1975.

[26] J. Boyan and A. Moore. Generalization in reinforcement learning: Safely approximating the value function. In *Advances in Neural Information Processing Systems*, volume 7. MIT Press, 1994.

[27] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[28] T. Brázdil, V. Forejt, J. Krcál, J. Kretínský, and A. Kucera. Continuous-time stochastic games with time-bounded reachability. *Inf. Comput.*, 224:46–70, 2013.

[29] Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, 1986.

[30] P. Buchholz. Exact performance equivalence: An equivalence relation for stochastic automata. *Theoretical Computer Science*, 215(1-2):263–287, Feb. 1999.

[31] P. Buchholz and I. Schulz. Numerical analysis of continuous time Markov decision processes over finite horizons. *Comput. Oper. Res.*, 38(3):651–659, 2011.

[32] P. Buchholz, E. M. Hahn, H. Hermanns, and L. Zhang. Model checking algorithms for CTMDPs. In *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*, pages 225–242. Springer, 2011.

[33] Y. Butkova, H. Hatefi, H. Hermanns, and J. Krčál. Optimal continuous time Markov decisions. In *Automated Technology for Verification and Analysis*, pages 166–182. Springer International Publishing, 2015.

[34] J. Cai. Computing Jordan normal forms exactly for commuting matrices in polynomial time. *Int. J. Found. Comput. Sci.*, 5(3/4):293–302, 1994.

[35] J.-Y. Cai, R. J. Lipton, and Y. Zalcstein. The complexity of the A B C problem. *SIAM J. Comput.*, 29(6), 2000.

[36] G. C. Calafiore and M. C. Campi. The scenario approach to robust control design. *IEEE Transactions on automatic control*, 51(5):742–753, 2006.

[37] J. Chen, S. Moarref, and H. Kress-Gazit. Verifiable control of robotic swarm from high-level specifications. In *AAMAS*, pages 568–576. ACM, 2018.

[38] V. Chonev, J. Ouaknine, and J. Worrell. On the Skolem problem for continuous linear dynamical systems. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 100:1–100:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[39] H. Choset, S. Hutchinson, K. Lynch, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[40] J. Chow and K. Cheung. A toolbox for power system dynamics and control engineering education and research. *IEEE Transactions on Power Systems*, 7(4): 1559–1564, 1992.

[41] H. Cohen. *A Course in Computational Algebraic Number Theory*. Springer, 1993.

[42] M. H. Cohen and C. Belta. Model-based reinforcement learning for approximate optimal control with temporal logic specifications. In *HSCC '21: 24th ACM International Conference on Hybrid Systems: Computation and Control, Nashville, Tennessee, May 19-21, 2021*, pages 12:1–12:11. ACM, 2021.

[43] M. H. Cohen, Z. Serlin, K. Leahy, and C. Belta. Temporal logic guided safe model-based reinforcement learning: A hybrid systems approach. *Nonlinear Analysis: Hybrid Systems*, 47:101295, 2023.

[44] G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decompostion. In *Automata Theory and Formal Languages*, pages 134–183, Berlin, Heidelberg, 1975. Springer Berlin Heidelberg. ISBN 978-3-540-37923-2.

[45] C. C. Conley. *Isolated invariant sets and the Morse index*, volume 25 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1978.

[46] L. Daviaud, M. Jurdzinski, R. Lazic, F. Mazowiecki, G. A. Pérez, and J. Worrell. When is containment decidable for probabilistic automata? In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 121:1–121:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[47] J. D'Costa, T. Karimov, R. Majumdar, J. Ouaknine, M. Salamati, S. Soudjani, and J. Worrell. The pseudo-Skolem problem is decidable. In *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*, volume 202 of *LIPIcs*, pages 34:1–34:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[48] J. D'Costa, T. Karimov, R. Majumdar, J. Ouaknine, M. Salamati, and J. Worrell. The pseudo-reachability problem for diagonalisable linear dynamical systems. In *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPIcs*, pages 40:1–40:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[49] J. W. Demmel. *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics, 1997.

[50] A. Desai, I. Saha, J. Yang, S. Qadeer, and S. Seshia. DRONA: A framework for safe distributed mobile robotics. In *ICCPS*, 2017.

[51] J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Metrics for labelled Markov processes. *Theoretical Computer Science*, 318(3):323–354, 2004.

[52] A. Devonport, A. Saoud, and M. Arcak. Symbolic abstractions from data: A pac learning approach. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 599–604, 2021. doi: 10.1109/CDC45484.2021.9683316.

[53] F. Djeumou, A. P. Vinod, E. Goubault, S. Putot, and U. Topcu. On-the-fly control of unknown systems: From side information to performance guarantees through reachability. *IEEE Transactions on Automatic Control*, 68(8):4857–4872, 2023.

[54] J. Doyle, B. Francis, and A. Tannenbaum. *Feedback control theory*. Macmillan Publishing Co., 1990.

[55] S. Dutta, X. Chen, and S. Sankaranarayanan. Reachability analysis for neural feedback systems using regressive polynomial rule inference. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*. ACM, Apr. 2019.

[56] P. M. Esfahani, T. Sutter, and J. Lygeros. Performance bounds for the scenario approach and an extension to a class of non-convex programs. *IEEE Transactions on Automatic Control*, 60(1):46–58, 2014.

[57] C. Fan, B. Qi, S. Mitra, and M. Viswanathan. Dryvr: Data-driven verification and compositional reasoning for automotive systems. In *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pages 441–461. Springer, 2017.

[58] C. Fan, U. Mathur, S. Mitra, and M. Viswanathan. Controller synthesis made real: reach-avoid specifications and linear dynamics. In *CAV*, pages 347–366, 2018.

[59] C. Fan, K. Miller, and S. Mitra. Fast and guaranteed safe controller synthesis for nonlinear vehicle models. In *CAV*, pages 629–652, 2020.

[60] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas. Efficient and accurate estimation of Lipschitz constants for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[61] J. Fearnley, M. N. Rabe, S. Schewe, and L. Zhang. Efficient approximation of optimal control for continuous-time Markov games. *Inf. Comput.*, 247:106–129, 2016.

[62] W. Feller. *An Introduction to probability theory and its applications*. John Wiley & Sons, 1968.

[63] N. Fijalkow, J. Ouaknine, A. Pouly, J. S. Pinto, and J. Worrell. On the decidability of reachability in linear time-invariant systems. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2019, Montreal, QC, Canada, April 16-18, 2019*, pages 77–86. ACM, 2019.

[64] I. Gavran, R. Majumdar, and I. Saha. Antlab: A multi-robot task server. *ACM TECS*, 16(5s):1–19, Oct. 2017.

[65] G. Ge. *Algorithms Related to Multiplicative Representations of Algebraic Numbers*. PhD thesis, U.C. Berkeley, 1993.

[66] A. Girard. Low-complexity switching controllers for safety using symbolic models. *IFAC Proceedings Volumes*, 45(9):82–87, 2012. ISSN 1474-6670. 4th IFAC Conference on Analysis and Design of Hybrid Systems.

[67] A. Girard and G. J. Pappas. Approximation metrics for discrete and continuous systems. *IEEE Transactions on Automatic Control*, 52(5):782–798, 2007.

[68] A. Girard, G. Pola, and P. Tabuada. Approximately bisimilar symbolic models for incrementally stable switched systems. *IEEE Transactions on Automatic Control*, 55(1):116–126, 2009.

[69] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In *Recent Advances in Learning and Control*, pages 95–110. Springer-Verlag Limited, 2008.

[70] K. Grover, F. dos Santos Barbosa, J. Tumova, and J. Kretınsky. Semantic abstraction-guided motion planningfor scltl missions in unknown environments. In *Robotics: Science and Systems*, 2021.

[71] L. Haan and A. Ferreira. *Extreme value theory: an introduction*, volume 3. Springer, 2006.

[72] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford University Press, 1999.

[73] M. A. Harrison. Lectures on linear sequential machines. Technical report, DTIC Document, 1969.

[74] S. Herbert, M. Chen, S. Han, S. Bansal, J. Fisac, and C. Tomlin. FaSTrack: A modular framework for fast and guaranteed safe motion planning. In *CDC*, 2017.

[75] H. Hermanns, J. Meyer-Kayser, and M. Siegle. Multi terminal binary decision diagrams to represent and analyse continuous time Markov chains. In *Proc. 3rd International Workshop on the Numerical Solution of Markov Chains (NSMC'99)*, pages 188–207, 2 1999.

[76] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer. Learning an approximate model predictive controller with guarantees. *IEEE Control. Syst. Lett.*, 2(3):543–548, 2018.

[77] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press., Cambridge, 1985.

[78] T. Howell, B. Jackson, and Z. Manchester. Altro: A fast solver for constrained trajectory optimization. In *IROS*, 2019.

[79] K. Hsu, R. Majumdar, K. Mallik, and A.-K. Schmuck. Multi-layered abstraction-based controller synthesis for continuous-time systems. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control*. ACM, Apr. 2018.

[80] K.-C. Hsu, V. Rubies-Royo, C. J. Tomlin, and J. F. Fisac. Safety and liveness guarantees through reach-avoid reinforcement learning. In *Robotics: Science and Systems*, 2021.

[81] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu. ReachNN. *ACM Transactions on Embedded Computing Systems*, 18(5s):1–22, Oct. 2019.

[82] C. Huang, J. Fan, X. Chen, W. Li, and Q. Zhu. POLAR: A polynomial arithmetic framework for verifying neural-network controlled systems. In *Automated Technology for Verification and Analysis*, pages 414–430. Springer International Publishing, 2022.

[83] J. W. Huang, S. J. Roberts, and J.-P. Calliess. On the sample complexity of Lipschitz constant estimation. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

[84] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee. Verisig. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*. ACM, Apr. 2019.

[85] E. Ivanova, A. Saoud, and A. Girard. Lazy controller synthesis for monotone transition systems and directed safety specifications. *Autom.*, 135:109993, 2022.

[86] B. Jackson, T. Howell, K. Shah, M. Schwager, and Z. Manchester. Scalable cooperative transport of cable-suspended loads with UAVs using distributed trajectory optimization. *IEEE Robot. Autom. Lett.*, 5(2):3368–3374, 2020.

[87] M. Jordan and A. Dimakis. Provable Lipschitz certification for generative models. In *International Conference on Machine Learning*, pages 5118–5126. PMLR, 2021.

[88] K. D. Julian, J. Lopez, J. S. Brush, M. P. Owen, and M. J. Kochenderfer. Policy compression for aircraft collision avoidance systems. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–10, 2016.

[89] K. D. Julian, M. J. Kochenderfer, and M. P. Owen. Deep neural network compression for aircraft collision avoidance systems. *Journal of Guidance, Control, and Dynamics*, 42(3):598–608, Mar. 2019.

[90] K. C. Kalagarla, R. Jain, and P. Nuzzo. Model-free reinforcement learning for optimal control of markov decision processes under signal temporal logic specifications. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 2252–2257, 2021.

[91] R. Kannan and R. J. Lipton. Polynomial-time algorithm for the orbit problem. *J. ACM*, 33(4):808–821, 1986.

[92] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE TRA*, 12(4): 566–580, 1996.

[93] M. Kazemi, R. Majumdar, M. Salamati, S. Soudjani, and B. Wooding. Data-driven abstraction-based control synthesis. *CoRR*, abs/2206.08069, 2022.

[94] J. G. Kemeny and J. L. Snell. *Finite Markov Chains: With a New Appendix "Generalization of a Fundamental Matrix"*. Undergraduate Texts in Mathematics. Springer, 1976. ISBN 978-0-387-90192-3.

[95] M. Khaled and M. Zamani. pFaces: an acceleration ecosystem for symbolic control. In *HSCC*, pages 252–257, 2019.

[96] M. Khaled and M. Zamani. pfaces: an acceleration ecosystem for symbolic control. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2019, Montreal, QC, Canada, April 16-18, 2019*, pages 252–257. ACM, 2019.

[97] M. Khaled, E. S. Kim, M. Arcak, and M. Zamani. Synthesis of symbolic controllers: A parallelized and sparsity-aware approach. In *Tools and Algorithms for the Construction and Analysis of Systems - 25th International Conference, TACAS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings, Part II*, volume 11428 of *Lecture Notes in Computer Science*, pages 265–281. Springer, 2019.

[98] H. K. Khalil. *Nonlinear systems*. Prentice Hall Upper Saddle River, NJ, 1996.

[99] M. J. Kochenderfer and N. Monath. Compression of optimal value functions for Markov decision processes. In *2013 Data Compression Conference*, pages 501–501, 2013.

[100] S. Kousik, S. Vaskov, F. Bu, M. Johnson-Roberson, and R. Vasudevan. Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots. *Int. J. Robot. Res.*, 2020.

[101] S. Lang. *Introduction to transcendental numbers*. Addison-Wesley series in mathematics. Addison-Wesley Pub. Co., 1966.

[102] S. Lang. Transcendental numbers and Diophantine approximations. *Bull. Amer. Math. Soc.*, 77(5):635–677, 09 1971.

[103] K. G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991.

[104] A. Lavaei, M. Khaled, S. Soudjani, and M. Zamani. AMYTISS: Parallelized automated controller synthesis for large-scale stochastic systems. In *Computer Aided Verification: 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21–24, 2020, Proceedings, Part II 32*, pages 461–474. Springer, 2020.

[105] A. Lavaei, A. Nejati, P. Jagtap, and M. Zamani. Formal safety verification of unknown continuous-time systems: A data-driven approach. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, HSCC '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383394.

[106] A. Lavaei, S. Soudjani, A. Abate, and M. Zamani. Automated verification and synthesis of stochastic hybrid systems: A survey. *Automatica*, 146:110617, 2022.

[107] A. Lavaei, S. Soudjani, E. Frazzoli, and M. Zamani. Constructing MDP abstractions using data with formal guarantees. *IEEE Control Systems Letters*, 7:460–465, 2022.

[108] S. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[109] S. LaValle and S. Hutchinson. Optimal motion planning for multiple robots having independent goals. *IEEE Trans. Robot. Autom.*, 14(6):912–925, 1998.

[110] B. Legat, R. M. Jungers, and J. Bouchat. Abstraction-based branch and bound approach to q-learning for hybrid optimal control. In *Learning for Dynamics and Control*, pages 263–274. PMLR, 2021.

[111] J. Lofberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *International Conference on Robotics and Automation*, 2004.

[112] M. Ma and L. Fan. Implementing consensus based distributed control in power system toolbox. In *2016 North American Power Symposium (NAPS)*, pages 1–6, 2016.

[113] A. Macintyre. Model theory of exponentials on Lie algebras. *Math. Struct. Comput. Sci.*, 18(1):189–204, 2008.

[114] A. Macintyre. Turing meets Schanuel. *Ann. Pure Appl. Log.*, 167(10):901–938, 2016.

[115] A. Macintyre and A. J. Wilkie. On the decidability of the real exponential field. In *Kreiseliana. About and Around Georg Kreisel*, pages 441–467. A K Peters, 1996.

[116] E. Macoveiciuc and G. Reissig. On-the-fly symbolic synthesis with memory reduction guarantees. *IEEE Transactions on Automatic Control*, pages 1–8, 2022.

[117] R. Majumdar, K. Mallik, A. Schmuck, and D. Zufferey. Assume-guarantee distributed synthesis. In *EMSOFT*, 2020.

[118] R. Majumdar, N. Ozay, and A. Schmuck. On abstraction-based controller design with output feedback. In *HSCC '20: 23rd ACM International Conference on Hybrid Systems: Computation and Control, Sydney, New South Wales, Australia, April 21-24, 2020*, pages 15:1–15:11. ACM, 2020.

[119] R. Majumdar, M. Salamati, and S. Soudjani. On decidability of time-bounded reachability in CTMDPs. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 133:1–133:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[120] R. Majumdar, K. Mallik, M. Salamati, S. Soudjani, and M. Zareian. Symbolic reach-avoid control of multi-agent systems. In *ICCPS '21: ACM/IEEE 12th International Conference on Cyber-Physical Systems, Nashville, Tennessee, USA, May 19-21, 2021*, pages 209–220. ACM, 2021.

[121] R. Majumdar, M. Salamati, and S. Soudjani. Neural abstraction-based controller synthesis and deployment. *ACM Trans. Embed. Comput. Syst.*, 22(5s):141:1–141:25, 2023.

[122] A. Makdesi, A. Girard, and L. Fribourg. Efficient data-driven abstraction of monotone systems with disturbances. In *7th IFAC Conference on Analysis and Design of Hybrid Systems, ADHS 2021, Brussels, Belgium, July 7-9, 2021*, volume 54 of *IFAC-PapersOnLine*, pages 49–54. Elsevier, 2021.

[123] A. Makdesi, A. Girard, and L. Fribourg. Efficient data-driven abstraction of monotone systems with disturbances. *IFAC-PapersOnLine*, 54(5):49–54, 2021. 7th IFAC Conference on Analysis and Design of Hybrid Systems ADHS 2021.

[124] D. W. Masser. Linear relations on algebraic groups. In *New Advances in Transcendence Theory*. Camb. Univ. Press, 1988.

[125] P.-J. Meyer, H. Yin, A. Brodtkorb, M. Arcak, and A. Sørensen. Continuous and discrete abstractions for planning, applied to ship docking, 2019.

[126] M. Mignotte, T. N. Shorey, and R. Tijdeman. The distance between terms of an algebraic recurrence sequence. *J. für die reine und angewandte Math.*, 349, 1984.

[127] B. L. Miller. Finite state continuous time Markov decision processes with a finite planning horizon. *SIAM Journal on Control*, 6(2):266–280, 1968.

[128] I. Mitsioni, P. Tajvar, D. Kragic, J. Tumova, and C. Pek. Safe data-driven contact-rich manipulation. In *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, pages 120–127. IEEE, 2021.

[129] P. Mohajerin Esfahani, T. Sutter, and J. Lygeros. Performance bounds for the scenario approach and an extension to a class of non-convex programs. *IEEE Transactions on Automatic Control*, 60(1):46–58, 2015.

[130] A. M. Mood. *Introduction to the Theory of Statistics*. McGraw-hill, 1950.

[131] R. M. Murray, M. Rathinam, and W. Sluis. Differential flatness of mechanical control systems: A catalog of prototype systems. In *Proceedings of the 1995 ASME International Congress and Exposition*, 1995.

[132] M. R. Neuhausser and L. Zhang. Time-bounded reachability probabilities in continuous-time Markov decision processes. In *2010 Seventh International Conference on the Quantitative Evaluation of Systems*, pages 209–218, Sep. 2010.

[133] M. R. Neuhäußer and L. Zhang. Time-bounded reachability probabilities in continuous-time Markov decision processes. In *QEST 2010, Seventh International Conference on the Quantitative Evaluation of Systems, Williamsburg, Virginia, USA, 15-18 September 2010*, pages 209–218. IEEE Computer Society, 2010.

[134] M. R. Neuhäußer, M. Stoelinga, and J. Katoen. Delayed nondeterminism in continuous-time Markov decision processes. In *Foundations of Software Science and Computational Structures, 12th International Conference, FOSSACS 2009, York, UK, March 22-29, 2009. Proceedings*, volume 5504 of *Lecture Notes in Computer Science*, pages 364–379. Springer, 2009.

[135] M. R. Neuhäußer. *Model checking nondeterministic and randomly timed systems*. PhD thesis, Enschede, 2010.

[136] A. Nikou and D. V. Dimarogonas. Decentralized tube-based model predictive control of uncertain nonlinear multiagent systems. *Int. J. Robust Nonlinear Control*, 29 (10):2799–2818, Mar. 2019.

[137] P. Nilsson and A. Ames. Barrier functions: Bridging the gap between planning from specifications and safety-critical control. In *CDC*, 2018.

[138] K. Ogata. *Modern control engineering*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 4th edition, 2001. ISBN 0130609072.

[139] J. Ouaknine and J. Worrell. Positivity problems for low-order linear recurrence sequences. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 366–379, 2014.

[140] J. Ouaknine and J. Worrell. On linear recurrence sequences and loop termination. *ACM SIGLOG News*, 2(2):4–13, 2015.

[141] Y. Pant, H. Abbas, R. Quaye, and R. Mangharam. Fly-by-Logic: Control of multi-drone fleets with temporal logic objectives. In *ICCPS*, 2018.

[142] C. H. Papadimitriou and G. Piliouras. From nash equilibria to chain recurrent sets: An algorithmic solution concept for game theory. *Entropy*, 20(10):782, 2018.

[143] J. Piribauer and C. Baier. On Skolem-hardness and saturation points in Markov decision processes. In *ICALP*, 2020.

[144] M. N. Rabe and S. Schewe. Finite optimal control for time-bounded reachability in CTMDPs and continuous-time Markov games. *Acta Inf.*, 48(5-6):291–315, 2011.

[145] M. N. Rabe and S. Schewe. Optimal time-abstract schedulers for CTMDPs and continuous-time Markov games. *Theor. Comput. Sci.*, 467:53–67, 2013.

[146] G. Reissig, A. Weber, and M. Rungger. Feedback refinement relations for the synthesis of symbolic controllers. *IEEE TAC*, 62(4):1781–1796, 2016.

[147] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. *J. Symb. Comp.*, 1992.

[148] A. Rodionova, Y. Pant, K. Jang, H. Abbas, and R. Mangharam. Learning-to-Fly: Learning-based collision avoidance for scalable urban air mobility. *ArXiv*, 2020.

[149] P. Roy, P. Tabuada, and R. Majumdar. Pessoa 2.0. In *Proceedings of the 14th international conference on Hybrid systems: computation and control*. ACM, Apr. 2011.

[150] S. Ruder. An overview of gradient descent optimization algorithms. *ArXiv*, abs/1609.04747, 2016.

[151] M. Rungger and O. Stursberg. On-the-fly model abstraction for controller synthesis. In *2012 American Control Conference (ACC)*, pages 2645–2650, 2012.

[152] M. Rungger and M. Zamani. SCOTS: A tool for the synthesis of symbolic controllers. In *HSCC*, 2016.

[153] S. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach*. Pearson Education, 2010.

[154] S. Sadraddini and C. Belta. Formal guarantees in data-driven model identification and control synthesis. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control, HSCC 2018, Porto, Portugal, April 11-13, 2018*, pages 147–156. ACM, 2018.

[155] I. Saha, R. Rattanachai, V. Kumar, G. J. Pappas, and S. Seshia. Implan: Scalable incremental motion planning for multi-robot systems. In *ICCPS*, 2016.

[156] Y. Sahin, P. Nilsson, and N. Ozay. Provably-correct coordination of large collections of agents with counting temporal logic constraints. In *ICCPS*, 2017.

[157] A. Salamati, A. Lavaei, S. Soudjani, and M. Zamani. Data-driven safety verification of stochastic systems via barrier certificates. *IFAC-PapersOnLine*, 54(5):7–12, 2021. 7th IFAC Conference on Analysis and Design of Hybrid Systems ADHS 2021.

[158] M. Salamati, S. Soudjani, and R. Majumdar. Approximate time bounded reachability for CTMCs and CTMDPs: A Lyapunov approach. In *Quantitative Evaluation of Systems - 15th International Conference, QEST 2018, Beijing, China, September 4-7, 2018, Proceedings*, volume 11024 of *Lecture Notes in Computer Science*, pages 389–406. Springer, 2018.

[159] M. Salamati, R. Salvia, E. Darulova, S. Soudjani, and R. Majumdar. Memory-efficient mixed-precision implementations for robust explicit model predictive control. *ACM Trans. Embed. Comput. Syst.*, 18(5s):100:1–100:19, 2019.

[160] M. Salamati, S. Soudjani, and R. Majumdar. A Lyapunov approach for time-bounded reachability of CTMCs and CTMDPs. *ACM Trans. Model. Perform. Evaluation Comput. Syst.*, 5(1):2:1–2:29, 2020.

[161] S. Samuel, K. Mallik, A. Schmuck, and D. Neider. Resilient abstraction-based controller design. In *HSCC '20: 23rd ACM International Conference on Hybrid Systems: Computation and Control, Sydney, New South Wales, Australia, April 21-24, 2020*, pages 33:1–33:2. ACM, 2020.

[162] Y. D. Sergeyev. An information global optimization algorithm with local tuning. *SIAM Journal on Optimization*, 5(4):858–870, 1995.

[163] S. Singh, M. Chen, S. Herbert, C. Tomlin, and M. Pavone. Robust tracking with model mismatch for fast and safe planning: an SOS optimization approach. In *WAFR*, pages 545–564, 2018.

[164] E. D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*. Springer-Verlag, Berlin, Heidelberg, 1998. ISBN 0387984895.

[165] S. Soudjani and R. Majumdar. Concentration of measure for chance-constrained optimization. *IFAC-PapersOnLine*, 51(16):277–282, 2018.

[166] M. Srinivasan, S. Coogan, and M. Egerstedt. Control of multi-agent systems with finite time control barrier certificates and temporal logic. In *CDC*, 2018.

[167] R. Strongin. On the convergence of an algorithm for finding a global extremum. *Eng. Cybernetics*, 11:549–555, 1973.

[168] R. Strongin, K. Barkalov, and S. Bevzuk. Acceleration of global search by implementing dual estimates for Lipschitz constant. In *International Conference on Numerical Computations: Theory and Algorithms*, pages 478–486. Springer, 2019.

[169] D. Sun, S. Jha, and C. Fan. Learning certified control using contraction metric. In *Conference on Robot Learning*, 2020.

[170] X. Sun, R. Nambiar, M. Melhorn, Y. Shoukry, and P. Nuzzo. DoS-resilient multi-robot temporal logic motion planning. In *ICRA*, pages 6051–6057, 2019.

[171] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Second edition, 2018.

[172] P. Tabuada. *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer Publishing Company, Incorporated, 1st edition, 2009. ISBN 1441902236.

[173] T. Tao. *Structure and randomness: pages from year one of a mathematical blog*. American Mathematical Society, 2008.

[174] R. Tedrake, I. Manchester, M. Tobenkin, and J. W. Roberts. LQR-trees: Feedback motion planning via sums-of-squares verification. *Int. J. Robot. Res*, 2010.

[175] H.-D. Tran, F. Cai, M. L. Diego, P. Musau, T. T. Johnson, and X. Koutsoukos. Safety verification of cyber-physical systems with reinforcement learning control. *ACM Transactions on Embedded Computing Systems*, 18(5s):1–22, Oct. 2019.

[176] H.-D. Tran, X. Yang, D. Manzanas Lopez, P. Musau, L. V. Nguyen, W. Xiang, S. Bak, and T. T. Johnson. Nnv: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *Computer Aided Verification*, pages 3–17, Cham, 2020. Springer International Publishing.

[177] J. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.

[178] C. F. Verdier, N. Kochdumper, M. Althoff, and M. Mazo. Formal synthesis of closed-form sampled-data controllers for nonlinear continuous-time systems under stl specifications. *Automatica*, 139:110184, 2022. ISSN 0005-1098.

[179] N. K. Vereshchagin. The problem of appearance of a zero in a linear recurrence sequence (in russian). *Mat. Zametki*, 38(2), 1985.

[180] A. Virmaux and K. Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

[181] X. Wang, S. Nair, and M. Althoff. Falsification-based robust adversarial reinforcement learning. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 205–212. IEEE, 2020.

[182] Z. Wang and R. M. Jungers. Probabilistic guarantees on the objective value for the scenario approach via sensitivity analysis. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 5668–5673. IEEE, 2022.

[183] K. Watanabe, N. Renninger, S. Sankaranarayanan, and M. Lahijanian. Probabilistic specification learning for planning with safety constraints. In *Intelligent Robots and Systems (IROS)*, pages 6558–6565. IEEE, 2021.

[184] T.-W. Weng, H. Zhang, P.-Y. Chen, J. Yi, D. Su, Y. Gao, C.-J. Hsieh, and L. Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. In *International Conference on Learning Representations*, 2018.

[185] A. J. Wilkie. Schanuel's conjecture and the decidability of the real exponential field. In *Algebraic Model Theory*, pages 223–230. Springer Netherlands, Dordrecht, 1997.

[186] N. Wolovick and S. Johr. A characterization of meaningful schedulers for continuous-time Markov decision processes. In *Formal Modeling and Analysis of Timed Systems, 4th International Conference, FORMATS 2006, Paris, France, September 25-27, 2006, Proceedings*, volume 4202 of *Lecture Notes in Computer Science*, pages 352–367. Springer, 2006.

[187] T. Wongpiromsarn, U. Topcu, and R. Murray. Receding horizon temporal logic planning. *IEEE TAC*, 57(11):2817–2830, 2012.

[188] G. Wood and B. Zhang. Estimation of the Lipschitz constant of a function. *Journal of Global Optimization*, 8(1):91–103, 1996.

[189] W. Xiao and C. Cassandras. Decentralized optimal merging control for connected and automated vehicles with optimal dynamic resequencing. In *ACC*, 2020.

[190] W. Xiao, C. Cassandras, and C. Belta. Decentralized merging control in traffic networks with noisy vehicle dynamics: a joint optimal control and barrier function approach. In *ITSC*, 2019.

[191] B. Xue, M. Zhang, A. Easwaran, and Q. Li. PAC model checking of black-box continuous-time dynamical systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(11):3944–3955, 2020.

[192] L. Yang and N. Ozay. Provably-correct fault tolerant control with delayed information. In *CDC*, 2017.

[193] I. S. Zapreev, C. Verdier, and M. Mazo. Optimal symbolic controllers determinization for bdd storage. *IFAC-PapersOnLine*, 51(16):1–6, 2018. ISSN 2405-8963. 6th IFAC Conference on Analysis and Design of Hybrid Systems ADHS 2018.

[194] B. Zhong, M. Zamani, and M. Caccamo. Synthesizing safety controllers for uncertain linear systems: A direct data-driven approach. In *2022 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1278–1284, 2022.

A
Appendices

A.1. Additional Data for Experiments of Section 2.4

Multi-Drone Path Planning

Every drone is modeled as a control system $\Sigma^i = (X, x^i_{\mathsf{in}}, U, W, f^d_\tau)$, where f^d_τ is the sampled-time abstraction of the following continuous dynamics:

$$f^d(x_t, u_t) := \begin{bmatrix} \dot{x}(1) \\ \dot{x}(2) \\ \dot{x}(3) \end{bmatrix} = \begin{bmatrix} u(1)cos(x(3)) \\ u(1)sin(x(3)) \\ u(2) \end{bmatrix}.$$

where $x(1)$ and $x(2)$ denote the drone's position in two-dimensional space, $x(3)$ denotes the rotational angle, and $u(1)$ and $u(2)$ represent control inputs for each drone. Choosing a sampling time $\tau = 0.1$ s, the nominal dynamics f^d_τ can be characterized uniquely. We consider state and input spaces to be $X = [-1, 11]^2 \times [-2, 3.3]$ and $U = [-2.4, 2.4]^2$, respectively. The disturbance set and robustness margin are chosen as $|W| \leq (0, 0.025, 0.025)$ and $\varepsilon = (0.20, 0.20, 0.24)$.

Recall that we consider 10 drones, i.e., $N = 10$. Selecting the horizon length $T = 104$ and minimum safe distance $\delta = 0.24$ m, ALTRO computes a valid open-loop trajectory in 77.8 seconds for the product system with 30 state and 20 input variables. Figure 2.29 gives time-space illustration for the safe tubes around the nominal trajectories.

For ABCD, we set $\eta_X = (0.025, 0.025, 0.03)$ and $\eta_U = (0.3, 0.3)$. Table 2.10 shows the run times and number of state-input pairs corresponding to both local and global ABCD. Noticeably, already when $N = 2$, memory requirement for global ABCD exceeds memory limits, even 1.5 TB RAM on a cluster machine is not sufficient to synthesize a controller.

On the other hand, using ALTRO alone would not provide guarantee against bounded disturbance. Figure A.2 illustrates the performance of open-loop and feedback controllers in regulating distance between two particular the drones with and without disturbances. As expected, in the absence of disturbances, the open-loop controllers suffice and the distance between the two drones (shown in solid blue) does not go below the defined threshold. Next, we consider the case when constant additive disturbances $(0, 0.025, 0.025)$ and $(0, -0.025, -0.025)$ are being applied to the two drones throughout the whole horizon. It can be noticed that applying the open-loop controller causes that distance between the two drones (shown in solid yellow) to go below the predefined threshold. However, the

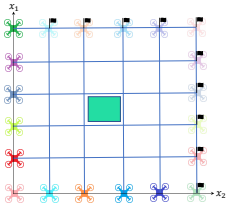

Figure A.1.: The mission map for the multi-drone path planning example

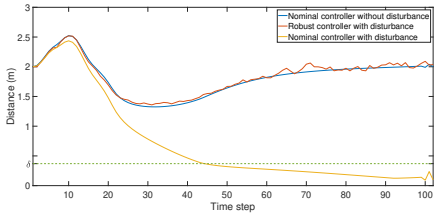

Figure A.2.: Performance of open-loop and feedback controllers in regulating distance between two selected drones for disturbance-free and perturbed situations for the multi-drone path planning example

feedback controller is capable of maintaining distance (shown in solid red) within the safe region when the same disturbance is being applied.

Crane and Vehicle

We model the crane and vehicle as control systems $\Sigma^1 = (X^1, x^1_{\mathsf{in}}, U^1, W^1, f^c_\tau)$ and $\Sigma^2 = (X^2, x^2_{\mathsf{in}}, U^2, W^2, f^l_\tau)$, respectively. The dynamics are obtained by discretizing the following continuous-time dynamics.

The crane is modeled as cart-pole system [19]:

$$\ddot{\theta} = \frac{M_t g \sin(\theta) - \cos(\theta)(F + M_p l \dot{\theta}^2 \sin(\theta))}{l(4/3 M_t - M_p \cos^2(\theta))} = f^c_1(\theta, \dot{\theta}, F)$$
$$\ddot{z} = \frac{F + M_p l \dot{\theta}^2 \sin(\theta) - M_p l \ddot{\theta} \cos(\theta)}{M_t} = f^c_2(\theta, \dot{\theta}, F),$$

where $g = -9.8$ m/s^2 is the acceleration of gravity, $M_c = 1$ kg is the cart mass, $M_p = 0.1$ kg is the pole mass, $M_t = M_c + M_p$ denotes the total mass, and $l = 0.5$ m is the half-pole length. Further, the cart's position, the pole's angle, and input force to the cart are denoted by $x^{(1)}(1) = z$, $x^{(1)}(3) = \theta$, and $u^{(1)} = F$, respectively. The continuous-time dynamics of the crane is of the following form:

$$f^c(x^{(1)}_t, u^{(1)}_t) := \begin{bmatrix} \dot{x}^{(1)}(1) \\ \dot{x}^{(1)}(2) \\ \dot{x}^{(1)}(3) \\ \dot{x}^{(1)}(4) \end{bmatrix} = \begin{bmatrix} \dot{z} \\ \ddot{z} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} x^{(1)}(2) \\ f^c_1(x^{(1)}(3), x^{(1)}(4), u^{(1)}) \\ x^{(1)}(4) \\ f^c_2(x^{(1)}(3), x^{(1)}(4), u^{(1)}) \end{bmatrix}.$$

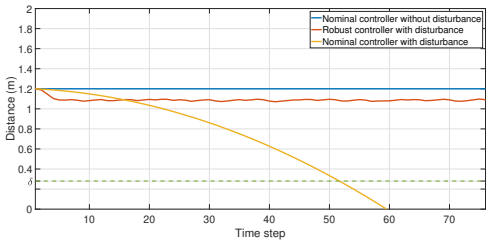

Figure A.3.: Comparison of open-loop and feedback controllers for the lane merging example

The vehicle's continuous-time dynamics takes the form of

$$f^l(x_t^{(2)}, u_t^{(2)}) = \begin{bmatrix} \dot{x}^{(2)}(1) \\ \dot{x}^{(2)}(2) \end{bmatrix} = \begin{bmatrix} x^{(2)}(2) \\ u^{(2)} \end{bmatrix},$$

where $x^{(2)}(1)$ and $x^{(2)}(2)$ denote the vehicle's position and speed, and $u^{(2)}$ represents the vehicle's control input (acceleration). On fixing the sampling time $\tau = 0.1$ s, one can derive f_τ^c and f_τ^l. For the crane, the disturbance set and robustness margin are chosen as $|W^1| \leq (0, 0.05, 0, 0)$ and $\varepsilon^1 = (0.135, 0.385, 0.176, 0.768)$. Similarly, for the vehicle, disturbance set and robustness margin are chosen as $|W^2| \leq (0, 0.1)$ and $\varepsilon^2 = (0.08, 0.12)$.

There is no obstacle for this example and for minimum distance between the crane and the vehicle we choose $\delta = 0.035$ m. Fixing the horizon length to $T = 70$, ALTRO was capable of generating a valid nominal trajectory in 0.65 seconds. Figure 2.26 (**left**) demonstrates snapshots of the produced trajectory. As before, under the nominal open-loop controllers, applying (constant) additive disturbance $W = (0, 0.05, 0, 0)$ (to the cart-pole system) causes a collision between the crane and the vehicle before the end of the mission (Figure 2.26 (**right**)).

In the next step, we use SCOTS in order to compute a feedback controller tolerating disturbances. We choose state and input spaces for the crane to be $X^1 = [-0.195, 5.49] \times [-1.99, 4.37] \times [1.20, 4.68] \times [-5.44, 5.28]$ and $U^1 = [-7, 7]$, respectively. For the vehicle, we set $X^2 = [3, 9] \times [-3, 1.995]$ and $U^2 = [-3, 3]$. We choose state and input partition sizes $\eta_X^1 = (0.015, 0.035, 0.016, 0.064)$, $\eta_U^1 = 0.2$, $\eta_X^2 = (0.01, 0.015)$ and $\eta_U^2 = 0.1$. Table 2.10 shows the run times and number of state-input pairs corresponding to local and global ABCD. As before, for the cart-pole model, global ABCD exceeds our 1.5 TB memory limit. Note that computing feedback controllers for the crane and vehicle takes 511 seconds and 0.3 seconds, respectively. The large difference is due to the difference in the size of transition systems for the two dynamics.

Lane Merging

The nominal dynamics for each of the vehicles is the same as the one for modeling drones (given in Section A.1). The disturbance set and robustness margin are chosen as $|W| \leq (0.03, 0.03, 0.03)$ and $\varepsilon = (0.16, 0.16, 0.16)$. For collision and obstacle avoidance, we choose $\delta = 0.37$ m. The horizon length is fixed to $T = 110$. Given these settings,

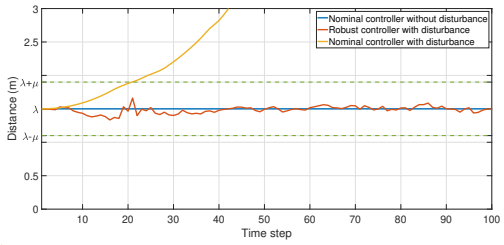

Figure A.4.: Comparison of open-loop and feedback controllers for the formation control example

ALTRO generates a valid nominal trajectory in 89.02 seconds. Next, we use ABCD in order to compute feedback controllers tolerating additive disturbance W. We choose state and input spaces for each vehicle's model to be $X = [-0.5, 15] \times [0.1, 7.4] \times [-1, 0.4]$ and $U = [-0.9, 3] \times [-2.1, 2.1]$, respectively. State and input partition sizes are chosen as $\eta_X = (0.02, 0.02, 0.02)$ and $\eta_U = (0.3, 0.15)$. Table 2.10 shows run times and number of state-input pairs corresponding to local and global ABCD. For $N > 1$, memory requirement for global ABCD exceeds memory limits. Figure 2.30 demonstrates snapshots of one sample trajectory when feedback controllers are employed under the presence of disturbance. It should be noticed that using ALTRO alone would not provide guarantee against bounded disturbance. Figure A.3 illustrates the fact that open-loop controller fails in keeping one of the vehicles away from the road's sides under perturbed situation when constant additive disturbance vector $(-0.03, 0.03, -0.03)$ is being applied throughout the whole horizon. In contrast, employing a feedback controller results in successful lane merging.

Multi-Drone Formation Control

In the multi-drone formation control case study, the nominal dynamics for each of the drones is the same as that in Section A.1. The disturbance set and robustness margin are chosen as $|W| \leq (0.03, 0.03, 0.03)$ and $\varepsilon = (0.24, 0.24, 0.24)$. Distance between each pair of drones positioned at the diamond's vertices is set to be $\lambda^{i,j} = \frac{3\sqrt{2}}{2}$ m for $i, j \in \{1, 2, 3, 4\}$, while the drone positioned at the center is supposed to keep distance $\lambda^{5,j} = 1.5$ m for $j \in \{1, 2, 3, 4\}$. Setting the minimum distance for obstacle avoidance to $\delta = 0.4$ m and horizon length $T = 100$, ALTRO finds a valid solution over the product system with 15 state and 10 input variables within 114.3 seconds. Next, we synthesize local controllers for every drone such that the specifications hold for the perturbed models with $\mu = 0.5$ m. We consider state and input spaces to be $X = [-2, 17] \times [-2, 17] \times [0.6, 1.6]$ and $U = [-0.9, 4.8] \times [-3, 3]$, respectively. We select $\eta_X = (0.03, 0.03, 0.03)$ and $\eta_U = (0.3, 0.15)$. Table 2.10 shows the run times and number of state-input pairs corresponding to local and global ABCD. Already for two drones, the memory requirement for global ABCD exceeds the available memory of 1.5 TB of RAM. Figure 2.31 illustrates four sequential frames of a sample perturbed trajectory generated by employing feedback controllers. Notice that both relative position and

orientation between drones are kept (almost) constant throughout the journey. On the other hand, using ALTRO alone would not provide guarantee against bounded disturbance. Figure A.4 illustrates performance of open-loop and feedback controllers on regulating distance between two specific drones with and without disturbances. As expected, in the absence of disturbances, the open-loop controllers suffice and the distance between the two drones (shown in solid blue) does not go below the threshold line (showed as the dotted line). However, when constant additive disturbance vectors $(0, 0.03, 0.03)$ and $(0, -0.03, -0.03)$ are being applied to the two drones throughout the whole horizon, open-loop controller fails, whereas the feedback controller is still capable of maintaining distance above the given threshold.

A.2. A Direct Algorithm for Problem 3.2

We now show a "direct" method for decidability of Problem 3.2 based on Schanuel's conjecture but without relying on the decidability of $\mathbb{R}_{\mathsf{MW}}$. As stated before, a switch point in a strategy corresponds to the existence of a non-tangential zero for the functions $y_t^{s,b}(\mathbf{d}^1)$ for $s \in S$ and $b \in \mathcal{D}_s \setminus \mathbf{d}^1(s)$. We know $y_t^{s,b}(\mathbf{d}^1)$ is an exponential polynomial of the form (3.7). Thus, deciding Problem 3.2 reduces to checking if an exponential polynomial of the form (3.7) in one free variable t has a non-tangential zero in a bounded interval. We use the following result from [38].

Theorem A.2.1 ([38]) *Assume **SC**. It is decidable whether an exponential polynomial of the form* (3.7) *has a zero in the interval* (t_1, t_2) *with* $t_1, t_2 \in \mathbb{Q}$.

Theorem A.2.1 decides whether a zero, not necessarily a non-tangential one, exists. We shall use the characterization of Proposition 3.2.2 to check if a non-tangential zero of $y_t := y_t^{s,b}(\mathbf{d}^1)$ exists in $(0, B)$. Define the functions

$$z_t^k = y_t^2 + \sum_{j=1}^{k} \left(\frac{d^j}{dt^j} y_t \right)^2, \quad k \in \{0, 1, 2, \ldots\}. \tag{A.1}$$

Theorem A.2.2 *Fix rational numbers* $t_1 < t_2$. *Suppose* y_t *has a zero in the interval* (t_1, t_2) *and* y_t *is not identically zero over this interval. There is* k_0 *as the smallest* k *such that* z_t^k *in* (A.1) *does* not *have any zero in* (t_1, t_2). *Moreover, the zero of* y_t *in* (t_1, t_2) *is non-tangential if* k_0 *is odd and is tangential if* k_0 *is even.*

Intuitively, the above theorem states that if y_t has at least one zero in (t_1, t_2), we can check for the existence of a tangential or non-tangential zero by a finite number of applications of Theorem A.2.1 to functions z_t^k in (A.1). Note that y_t may have both tangential and non-tangential zeros; Theorem A.2.2 gives a way of identifying the type of one of the zeros (the one with the largest order).

Proof (Proof of Theorem A.2.2) *Since y_t is an exponential polynomial, so is z_t^k for all k. Thus, we can use Theorem A.2.1 to check if z_t^k has a zero in (t_1, t_2). Note that z_t^k is the sum of squares of $\frac{d^j}{dt^j} y_t$, which means*

$$z_{t^*}^k = 0 \;\Rightarrow\; y_{t^*} = \frac{dy_t}{dt}\Big|_{t=t^*} = \cdots = \frac{d^k y_t}{dt^k}\Big|_{t=t^*} = 0. \tag{A.2}$$

The first part of the theorem is proved by showing that if for each k, z_t^k has a zero in (t_1, t_2), then y_t is identically zero. Suppose $z_t^k = 0$ for some $t = t_k^*$ in the interval (t_1, t_2), for any $k \in \{0, 1, 2, \ldots\}$. Using (A.2)*, we get that $y_t = 0$ for all $t \in \{t_0^*, t_1^*, t_2^*, \ldots\}$. If the set $\{t_0^*, t_1^*, t_2^*, \ldots\}$ is not finite, we get that y_t is identically zero according to the identity theorem [2]. If the set of zeros is finite, there is some t^* that appears infinitely often in the sequence $(t_0^*, t_1^*, t_2^*, \ldots)$. Therefore, $z_{t^*}^k = 0$ for infinitely many indices, which means $\frac{d^k y_t}{dt^k}\big|_{t=t^*} = 0$ for all k. Having y_k as an analytic function, this again implies that y_t is identically zero.*

Since y_t is not identically zero, take k_0 such that $z_t^{k_0}$ does not have a zero in (t_1, t_2) but $z_t^{k_0-1}$ does. Then, there is $t^* \in (t_1, t_2)$ such that y_t and all its derivatives up to order $k_0 - 1$ are zero at t^* but $\frac{d^{k_0}}{dt^{k_0}} y_t\big|_{t=t^*} \neq 0$. This t^* and k_0 satisfy the conditions of Proposition 3.2.2. Thus, t^* is a non-tangential zero for y_t if k_0 is odd and a tangential zero if k_0 is even.

To check if there is a non-tangential zero in an interval $(0, B)$, we apply Theorem A.2.2 to each zero of y_t individually. Suppose y_t has at least one zero. We can localize all zeros of y_t as follows:

1. Set $(t_1, t_2) := (0, B)$;
2. Set k_0 to be the smallest index such that z_t^k in (A.1) does not have any zero in (t_1, t_2);
3. If $k_0 > 0$, do the next steps:
   - Use bisection to find an interval $(t', t'') \subset (t_1, t_2)$ such that over this interval, $z_t^{k_0-1}$ has a zero and $z_t^{k_0}$ and $\frac{d^{k_0}}{dt^{k_0}} y_t$ do not have any zero;
   - Store (t', t'');
   - Repeat Steps 2-3 with $(t_1, t_2) := (t_1, t')$;
   - Repeat Steps 2-3 with $(t_1, t_2) := (t'', t_2)$.

The bisection used in the above algorithm sequentially splits the interval into two sub-intervals and picks the one that contains the zero of $z_t^{k_0-1}$. It stops when $\frac{d^{k_0}}{dt^{k_0}} y_t$ does not have any zero over the selected sub-interval. The splitting terminates after a finite number of iterations due to the fact that $\frac{d^{k_0}}{dt^{k_0}} y_t$ is a continuous function and non-zero at the zero of y_t. The whole algorithm terminates after a finite number of iterations since y_t has a finite number of zeros in $(0, B)$ (note that if y_t has infinite number of zeros in $(0, B)$, it will be identically zero according to the identity theorem [2]). The output of the algorithm is a set of intervals. Within each interval, y_t has a single zero. Applying Theorem A.2.2 to each such interval will decide whether the zero is tangential or non-tangential.

A.3. Proofs Related to Reduction for CTMCs and CTMDPs

A.3.1. Error Bounds for ε-Bisimilar CTMCs

Given matrices A and $\bar{A}$ corresponding to stochastic matrices $\mathbf{Q}$ and $\bar{Q}$, suppose that there exists a matrix P_b such that $AP_b = P_b\bar{A} + \Delta AP_b$ and $\boldsymbol{\beta} = P_b\bar{\boldsymbol{\beta}} + \Delta\boldsymbol{\beta}$, where all elements of ΔA and $\Delta\boldsymbol{\beta}$ are bounded by ε in the absolute value sense. Hence, a CTMC with $\hat{A} = A - \Delta A$ and $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta} - \Delta\boldsymbol{\beta}$ can be reduced based on the notion of exact bisimulation. ΔA and $\Delta\boldsymbol{\beta}$ include all rate mismatches with respect to the equivalence classes specified by P_b. Defining the error vector as $e_t = X_t - P_b\bar{X}_t$, dynamics of error would be as the following:

$$\begin{aligned} \dot{e}_t &= Ae_t + \Delta AP_b\bar{X}_t \\ \dot{\bar{X}}_t &= \bar{A}\bar{X}_t \end{aligned} \tag{A.3}$$

Since A and $\bar{A}$ are both stable matrices (extracted from the stochastic matrices $\mathbf{Q}$ and $\bar{\mathbf{Q}}$), steady state value of the vector e_t would be zero. The next theorem gives a bound on e_t for the case that absolute value of elements of ΔA and $\Delta\boldsymbol{\beta}$ do not exceed a certain threshold ε.

Theorem A.3.1 *Suppose that elements of ΔA and $\Delta\boldsymbol{\beta}$ are bounded by ε. The elements of the error $e_t \in \mathcal{R}^n$ defined in* (A.3) *are bounded by*

$$|e_t(i)| \leq (m\varepsilon + \rho)\Lambda_i$$

where, $\rho = ||e_0||_\infty$, $\Lambda = -A^{-1}$ and $\Lambda_i = \sum_{j=1}^{n} \Lambda(i,j)$.

Proof *Let us denote state transition matrix $G(t) := e^{At}$ and write its i^{th} row as $g_t(i)$. We also denote the i^{th} column of ΔA by ΔA_i. For $e_t(i)$ we can write:*

$$e_t(i) = g_t(i)\Delta A * P_b\bar{X}_t + g_t(i)e_0 = \sum_{j=1}^{n} \int_0^t g_{t-\tau}(i)\Delta A_j F_\tau(j)d\tau + g_t(i)e_0$$

where, $*$ operator stands for convolution of two signals in time domain and $F_t(i)$ is a scalar and obtained by multiplying j^{th} row of P_b by vector $\bar{X}_t$ which is bounded by 1. Therefore:

$$|e_t(i)| \leq \varepsilon n \int_0^t ||g_i(\tau)||_1 d\tau + g_t(i)e_0$$

Moreover, for every arbitrary time $t \geq 0$ we have $\|e_t(i)\| \leq (\varepsilon n + \rho)\int_0^\infty ||g_\tau(i)||_1 d\tau$. However, this bound cannot be easily found since it requires computing $G_t = e^{At}$. To avoid the computation of G_t, we use the uniformized form of $\mathbf{Q}$ defined as $H_0 := \frac{\mathbf{Q}}{\gamma_0} + \mathbb{I}_{n+2}$. H_0

is a row stochastic matrix and γ_0 is the maximum of absolute value of diagonal elements of $\mathbf{Q}$. Using H_0 one can compute state transition matrix corresponding to $\mathbf{Q}$ as [32]:

$$e^{\mathbf{Q}t} = \sum_{k=0}^{\infty} H_0^k e^{(-\gamma_0 t)} \frac{(\gamma_0 t)^k}{k!}$$

It is easy to notice that for every k, inner argument in the above summation is (element-wise) non-negative. We can also expand e^{Qt} in the following form:

$$e^{\mathbf{Q}t} = \begin{bmatrix} e^{At} & \vdots & (e^{At} - I)A^{-1}\boldsymbol{\beta} \cdots \\ \cdots & & \cdots \\ \mathbf{0} & \vdots & 1 \end{bmatrix}$$

It can be seen that e^{At} is one of the blocks inside e^{Qt}. Therefore, e^{At} is (element-wise) a non-negative matrix for all $t \geq 0$. Using the definition of the Fourier transform of a function [138], we get

$$\int_0^\infty |G_{ij}(\tau)| d\tau = \int_0^\infty G_{ij}(\tau) d\tau = -A_{ij}^{-1}$$

where, A_{ij}^{-1} denotes the ij^{th} element of A^{-1}. Setting $\Lambda := -A^{-1}$ and $\Lambda_i := \sum_{j=1}^{n} \Lambda(i,j)$, we get

$$|e_t(i)| \leq (\varepsilon n + \rho)\Lambda_i.$$

A.3.2. Reducible CTMC Case

Throughout Section 3.3, irreducibility of models is assumed. In this section, we show that our results are applicable to reducible CTMCs. The only assumption required for validity of the results of Subsection 3.3.1 is the stability of the matrix A. We prove in the sequel that this assumption holds also for reducible CTMCs by preprocessing its structure and eliminating bottom strongly connected components (BSCCs) that do not affect the reachability probability.

Remark 19 *For any given time bound, the reachability probabilities corresponding to the BSCCs of the CTMC $\mathcal{C}$ are zero except for the BSCC containing the single state* **good**. *Therefore, these BSCCs can be eliminated from the generator matrix. Thus we obtain a dynamical system for which the only BSCC is* {**good**}.

Proposition A.3.1 *For a reducible CTMC $\mathcal{C}$, after eliminating all the BSCCs except* {**good**} *and the states that can never reach* {**good**}, *the matrix A in* (3.19) *will be stable.*

Proof *If the CTMC is reducible, we first eliminate all the BSCCs except* {**good**}. *We also eliminate states that can never reach* {**good**}. *Therefore, the modified CTMC consists of only transient states and* {**good**}. *The transient states can be partitioned into strongly*

connected components. The canonical form of matrix A for such a CTMC will have the following structure:

$$A' = \begin{bmatrix} A'_{11} & A'_{12} & A'_{13} & \cdots & \cdots & A'_{1n} \\ 0 & A'_{22} & A'_{23} & A'_{24} & \cdots & A'_{2n} \\ 0 & 0 & A'_{33} & A'_{34} & \cdots & A'_{3n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & A'_{(n-1)(n-1)} & A'_{(n-1)n} \\ 0 & 0 & 0 & \cdots & 0 & A'_{nn}, \end{bmatrix} \tag{A.4}$$

where A'_{ii}s correspond to different strongly connected components. Since it is possible to reach from any state to $\{\mathbf{good}\}$*, A'_{ii}s satisfy Assumption 2 are stable.*

Equation (3.24) with the block upper-diagonal matrix A' in (A.4) can be solved bottom-up while the order reduction can be utilized in each step.

A.4. Proofs for Decidability of Pseudo-Skolem

A.4.1. Proof of Proposition 4.1.1

We want to show

$$Q\,\widetilde{\mathcal{O}}_{\gamma_2}(B, Q^{-1}x) \subseteq \widetilde{\mathcal{O}}_{\epsilon}(A, x) \subseteq Q\,\widetilde{\mathcal{O}}_{\gamma_1}(B, Q^{-1}x), \tag{A.5}$$

where $\gamma_1 = \epsilon \left\|Q^{-1}\right\|_\infty$ and $\gamma_2 = \epsilon / \left\|Q\right\|_\infty$.

Take any $y \in \widetilde{\mathcal{O}}_{\epsilon}(A, x)$. We show that $Q^{-1}y \in \widetilde{\mathcal{O}}_{\gamma_1}(B, Q^{-1}x)$ to get the right-hand side of (A.5). Since $y \in \widetilde{\mathcal{O}}_{\epsilon}(A, x)$, there is a state trajectory $(x_0, x_1, \ldots)$ and a sequence $(d_0, d_1, \ldots)$ such that $x_0 = x$, $x_{n+1} = Ax_n + d_n$, $d_n \in [-\epsilon, \epsilon]^m$ for all $n \in \mathbb{N}$, and y appears in the state trajectory. We construct a new state trajectory $(y_0, y_1, \ldots)$ and the sequence $(\bar{d}_0, \bar{d}_1, \ldots)$ with the transformation $x_n = Qy_n$ and $d_n = Q\bar{d}_n$. Then we have $y_{n+1} = Q^{-1}AQy_n + Q^{-1}d_n = By_n + \bar{d}_n$. Note that $\left\|\bar{d}_n\right\|_\infty = \left\|Q^{-1}d_n\right\|_\infty \leq \left\|Q^{-1}\right\|_\infty \left\|d_n\right\|_\infty \leq \gamma_1$. Since y appears in the state trajectory $(x_0, x_1, \ldots)$, $Q^{-1}y$ appears in the state trajectory $(y_0, y_1, \ldots)$ with $y_0 = Q^{-1}x_0 = Q^{-1}x$. Therefore, $Q^{-1}y \in \widetilde{\mathcal{O}}_{\gamma_1}(B, Q^{-1}x)$ which results in $y \in Q\widetilde{\mathcal{O}}_{\gamma_1}(B, Q^{-1}x)$.

To prove the left-hand side of (A.5), We invoke the right-hand side by replacing (A, B, Q, x, ϵ) with $(B, A, Q^{-1}, Q^{-1}x, \gamma_2)$. This gives $\widetilde{\mathcal{O}}_{\gamma_2}(B, Q^{-1}x) \subseteq Q^{-1}\,\widetilde{\mathcal{O}}_{\gamma_1'}(A, x)$ with $\gamma_1' = \gamma_2 \left\|Q\right\|_\infty$. Setting $\gamma_1' = \epsilon$ proves the left-hand side of (A.5).

To prove that $\widetilde{\mathcal{O}}(A, x) = Q\widetilde{\mathcal{O}}(B, Q^{-1}x)$, we take intersection of all the sides in (A.5) over $\epsilon > 0$:

$$\bigcap_{\epsilon>0} Q\,\widetilde{\mathcal{O}}_{\gamma_2}(B, Q^{-1}x) \subseteq \bigcap_{\epsilon>0} \widetilde{\mathcal{O}}_{\epsilon}(A, x) \subseteq \bigcap_{\epsilon>0} Q\,\widetilde{\mathcal{O}}_{\gamma_1}(B, Q^{-1}x).$$

Due to the linear relation between γ_1 and γ_2 with ϵ, we get

$$Q\,\widetilde{\mathcal{O}}(B, Q^{-1}x) \subseteq \widetilde{\mathcal{O}}(A, x) \subseteq Q\,\widetilde{\mathcal{O}}(B, Q^{-1}x) \quad \Rightarrow \quad \widetilde{\mathcal{O}}(A, x) = Q\,\widetilde{\mathcal{O}}(B, Q^{-1}x).$$

A.4.2. Proof of Lemma 4.3.1

Any $t \in \widetilde{\mathcal{O}}_\epsilon(A,s)$ is of the form $t = A^n s + \sum_{i=0}^{n-1} A^i d_{n-i-1}$ for some $n \in \mathbb{N}$ and some d_i with $\|d_i\|_\infty \leq \epsilon$. This means $s = A^{-n}t + \sum_{i=0}^{n-1} A^{-i} d'_{n-i-1}$ with $d'_{n-1-i} = A^{-1}d_i$. Since $\left\|d'_{n-1-i}\right\|_\infty \leq \left\|A^{-1}\right\|_\infty \epsilon$, we get $s \in \widetilde{\mathcal{O}}_\gamma(A^{-1},t)$. To get (4.5), notice that

$$t \in \widetilde{\mathcal{O}}(A,s) \Rightarrow t \in \bigcap_{\epsilon>0} \widetilde{\mathcal{O}}_\epsilon(A,s) \Rightarrow s \in \bigcap_{\gamma>0} \widetilde{\mathcal{O}}_\gamma(A^{-1},t) \Rightarrow s \in \widetilde{\mathcal{O}}(A^{-1},t).$$

Applying the same argument to the matrix A^{-1} will give the other side of (4.5).

A.4.3. Proof of Lemma 4.3.2

We prove the lemma by showing there is a constant $C > 0$ satisfying

$$\overline{\mathcal{O}(J,s)} \overset{*}{=} \mathcal{O}(J,s) \cup \{\mathbf{0}\} \overset{**}{\subseteq} \widetilde{\mathcal{O}}(J,s) \overset{\dagger}{\subseteq} \bigcap_{\epsilon>0} \bigcup_{z\in\mathcal{O}(J,s)} \mathcal{B}(z,C\epsilon) \overset{\S}{\subseteq} \overline{\mathcal{O}(J,s)}, \tag{A.6}$$

where $\mathcal{B}(z,\epsilon) := \{y \in \mathbb{R}^m \mid \|z-y\|_2 \leq \epsilon\}$ is the closed ball with respect to two norm with center z and radius ϵ. It is easy to see that equality (*) holds since all the eigenvalues of J are inside the unit circle, $\lim_{n\to\infty} J^n = 0$, and 0 is the only limiting point of any state trajectory.

It is also easy to see that inclusion (**) is correct. Note that for any $\epsilon > 0$, $\mathcal{O}(J,s) \subseteq \widetilde{\mathcal{O}}_\epsilon(J,s)$ and the set $\widetilde{\mathcal{O}}_\epsilon(J,s)$ is closed by definition. Taking intersection over $\epsilon > 0$, we get $\mathcal{O}(J,s) \subseteq \widetilde{\mathcal{O}}(J,s)$ with $\widetilde{\mathcal{O}}(J,s)$ being a closed set. Therefore, $\overline{\mathcal{O}(J,s)} \subseteq \widetilde{\mathcal{O}}(J,s)$.

We now choose a value of C which allows us to prove inclusion (†). First pick γ such that $\rho(J) < \gamma < 1$. Next choose c_1 to be a constant (which is guaranteed to exist) satisfying $\|J^n\|_2 \leq c_1\gamma^n$ for all $n \in \mathbb{N}$, and finally set $C := c_1 m/(1-\gamma)$. We show that $\widetilde{\mathcal{O}}_\epsilon(J,s) \subseteq \bigcup_{z\in\mathcal{O}(J,s)} \mathcal{B}(z,C\epsilon)$ for any $\epsilon > 0$. Take any $x \in \widetilde{\mathcal{O}}_\epsilon(J,s)$. Then there is a sequence $(d_0,d_1,\ldots)$ and $n \in \mathbb{N}$ such that $\|d_i\|_\infty \leq \epsilon$ and $x = J^n s + \sum_{i=0}^{n-1} J^i d_{n-i-1}$. Now

$$\|x - J^n s\|_2 = \left\|\sum_{i=0}^{n-1} J^i d_{n-i-1}\right\|_2 \leq \sum_{i=0}^{n-1} \left\|J^i\right\|_2 \|d_{n-i-1}\|_2 \leq \sum_{i=0}^{n-1} c_1\gamma^i m\epsilon \leq \frac{c_1 m\epsilon}{1-\gamma} = C\epsilon,$$

We then get $x \in \mathcal{B}(z,C\epsilon)$ for $z := J^n s \in \mathcal{O}(J,s)$.

The inclusion § can be proven by taking an arbitrary point $y \notin \overline{\mathcal{O}(J,s)}$ and showing that there is an $\epsilon > 0$ for which $y \notin \mathcal{B}(z,C\epsilon)$ for all $z \in \mathcal{O}(J,s)$. Note that the complement of $\overline{\mathcal{O}(J,s)}$ is an open set, which means there is a $\theta > 0$ such that $\mathcal{B}(y,\theta) \cap \overline{\mathcal{O}(J,s)} = \emptyset$. Taking ϵ such that $C\epsilon < \theta$ will give the intended result.

A.4.4. Proof of Lemma 4.3.5

The key part of the proof is to show that $\mathbf{0} \in \widetilde{\mathcal{O}}(A,s)$ for any s and for any A having the eigenvalues on the unit circle. Once we show this, we know that $s \in \widetilde{\mathcal{O}}(A^{-1},\mathbf{0})$ is true for

any s and any matrix A due to the Reversibility lemma. Stated for the inverse of A and any x, we get $x \in \widetilde{\mathcal{O}}(A, \mathbf{0})$. Since pseudo-orbits are transitive, we have $x \in \widetilde{\mathcal{O}}(A, s)$ for any x and s, which is the intended result.

We show $\mathbf{0} \in \widetilde{\mathcal{O}}(A, s)$ equivalently by replacing A with its Jordan form J and doing induction on the structure of J. The proof has two stages. The first stage is to show that $\mathbf{0} \in \widetilde{\mathcal{O}}(J, s)$ for all s when J has a single block simple eigenvalues. The second stage is to show that we can sequentially increase the multiplicity of eigenvalues and multiple blocks.

Base case: Suppose $J = \begin{bmatrix} a & -b \\ b & a \end{bmatrix}$ with $a^2 + b^2 = 1$ or $J = r$ with $|r| = 1$. Observe that the multiplication by J does not increase the two norm of a vector. Hence setting

$$d_n = \begin{cases} -\epsilon \cdot \frac{Jx_n}{\|Jx_n\|_2} & \text{if } ||Jx_n||_\infty > \epsilon, \\ -Jx_n & \text{otherwise,} \end{cases}$$

we obtain the ϵ-pseudo-orbit $(x_0 = s, x_1, x_2, \ldots, x_m, \mathbf{0}, \mathbf{0}, \ldots)$ from any s where $\|x_k\|_2 = \|x_{k-1}\|_2 - \epsilon$ for $k \leq m$, which gives $\mathbf{0} \in \widetilde{\mathcal{O}}(J, s)$.

Inductive case: We show that if $\mathbf{0} \in \widetilde{\mathcal{O}}(J_1, s_1)$ and $\mathbf{0} \in \widetilde{\mathcal{O}}(J_2, s_2)$ for all s_1 and s_2 of appropriate dimensions, we also have $\mathbf{0} \in \widetilde{\mathcal{O}}(J, s)$ with $J = \begin{bmatrix} J_1 & B \\ 0 & J_2 \end{bmatrix}$ for any B and any s with appropriate dimensions. Let us partition any state $x = (x^1, x^2)$ according to the dimensions of J_1 and J_2. Let $\epsilon > 0$ and $s = (s^1, s^2)$. By the assumption, there exist ϵ-perturbations $(d_0^2, d_1^2, \ldots, d_{N-1}^2)$ that bring s^2 to $\mathbf{0}$ under J_2. Let $d_n = (\mathbf{0}, d_n^2)$ for $0 \leq n < N$ be a sequence of ϵ-perturbations for the linear system with mapping J. We obtain the sequence $(x_0 = s, x_1, \ldots, x_N)$ with $x_N^2 = \mathbf{0}$: the ϵ-perturbations $d_0, \ldots, d_{N-1}$ have brought the second coordinate to $\mathbf{0}$. By the assumption, we also have $\mathbf{0} \in \widetilde{\mathcal{O}}_\epsilon(J_1, x_N^1)$, which gives ϵ-perturbations $(d_0^1, \ldots, d_M^1)$ that bring x_N^1 to $\mathbf{0}$ under J_1. Let us expand the sequence of perturbations for the linear system J with $d_{n+N} = (d_n^1, \mathbf{0})$ for $0 \leq n \leq M$. It is easy to see that $(d_0, \ldots, d_{N+M})$ bring the system from s to $\mathbf{0}$ due to the structure of J that is upper triangular.

A.4.5. Proof of Theorem 4.3.2

Recall that a set S is pseudo-reachable from s under A if for every $\epsilon > 0$, there exists a point $x_\epsilon \in S$ that is ϵ-pseudo-reachable from s under A. In this section, we show that pseudo-reachability in bounded semialgebraic sets is decidable.

We need the following lemma that shows that deciding pseudo-reachability in a given bounded set S reduces to checking whether $\overline{S} \cap \widetilde{\mathcal{O}}(J, s) = \emptyset$, allowing us to restrict our attention to compact sets and the existence of a pseudo-reachable point in a set as opposed to pseudo-reachability of the set as a whole.

Lemma A.4.1 *Let S be a bounded set. S is pseudo-reachable from s under A if and only if there exists $x \in \overline{S}$ that is pseudo-reachable from s under A.*

Proof *Suppose S is pseudo-reachable. Let $(\epsilon_i)_{i\in\mathbb{N}}$ be a sequence of positive numbers with $\lim_{\epsilon\to 0} = 0$, and $(x_i)_{i\in\mathbb{N}}$ be a sequence of elements of S such that x_i is ϵ_i-pseudo-reachable for all $i \geq 0$. By the Bolzano–Weierstrass theorem, boundedness of S implies that $(x_i)_{i\in\mathbb{N}}$ must have a limit point x in $\overline{S}$. To argue that x is pseudo-reachable, let $\epsilon > 0$. Since x is the limit point of $(x_i)_{i\in\mathbb{N}}$, there must exist an $\frac{\epsilon}{2}$-pseudo-orbit $(y_i)_{i\in\mathbb{N}}$ containing a point y_N such that $\|x - y_N\|_\infty < \frac{\epsilon}{2}$. Therefore, x is ϵ-pseudo-reachable from s via the sequence $s, y_1, \ldots, y_{N-1}, x$.*

Now suppose $x \in \overline{S}$ is pseudo-reachable. To argue that S is pseudo-reachable, let $\epsilon > 0$. Since $x \in \overline{S}$, there must exist a point $x' \in S$ such that $\|x' - x\|_\infty < \frac{\epsilon}{2}$. Since x is $\frac{\epsilon}{2}$-pseudo-reachable, x' must be ϵ-pseudo-reachable.

Now we are ready to prove the main theorem.

Theorem A.4.1 *Given a bounded semialgebraic set S, it is decidable whether S is pseudo-reachable from x_0 under A.*

Proof *It suffices to consider $A = J$ for J in real Jordan normal form (see Proposition 4.1.1) and S that is closed. Let $J_1, \ldots, J_t$ be all the blocks of J with spectral radius > 1, $J_{t+1}, \ldots, J_{t'}$ all the blocks of with spectral radius $= 1$ and $J_{t'+1}, \ldots, J_l$ all the blocks of with spectral radius < 1. Let M be an upper bound on the ℓ_2-norm of all vectors in S. We show how to decide whether there exists a point $x \in S$ that is also in $\widetilde{\mathcal{O}}(J, s)$.*

1. *Suppose J has a block J_i with an eigenvalue of modulus greater than 1 such that $s^i \neq \mathbf{0}$. Then $\widetilde{\mathcal{O}}(J, s) = \mathcal{O}(J, s)$ (Lemma 4.3.4). By Lemma A.4.3, there exists a computable N such that for all $n > N$, $\left\|J_i^n s^i\right\|_2 > M$ (observe that orbit itself is a pseudo-orbit), and therefore we only need to check whether any of the first N points in orbit of s under J belong to the set S.*

2. *Let $J_c = \operatorname{diag}(J_{t'+1}, \ldots, J_l)$. If for all $x \in S$, the projection x^c of x onto the coordinates governed by J_c is not $\mathbf{0}$, then using our characterization of the pseudo-orbit we can conclude that $x \in \widetilde{\mathcal{O}}(J, s)S$ only if $x^c \in \mathcal{O}(J_c, s^c)$. Now observe that because S is compact, it must be the case that $\inf_{x\in S} \|x^c\| > 0$ (since by assumption $\|x^c\|$ is never 0). Therefore, using Lemma A.4.3 we can compute a time bound N such that for $n > N$ and sufficiently small ϵ, $\|x_n^c\| < \inf_{x\in S} \|x^c\|$ and hence S can only be reached within the first N steps. It then remains to check whether any of $s, Js, \ldots, J^{N-1}s$ belongs to S.*

3. *Suppose 1 and 2 are not the case. Assuming S is not empty, it must then contain a point whose projection onto $J_c = \mathbf{0}$. In other words S must contain a point x whose projections onto blocks with spectral radius < 1 are all $\mathbf{0}$. From our characterization of the pseudo-orbit we can see that $x_i \in \Delta(J_i, s_i)$ for every $1 \leq i \leq m$ and hence $x \in \widetilde{\mathcal{O}}(J, s)$.*

A.4.6. Proof of Lemma 4.3.12

We now prove a generalization of Lemma 4.3.12. Let $\lambda_1, \ldots, \lambda_m$ be algebraic numbers of modulus 1 and let $p_1, \ldots, p_m$ be polynomials with algebraic coefficients. Let

n range over the natural numbers. We show how to effectively determine the value of $\liminf_{n\to\infty} |\sum_{j=1}^{m} p_j(1/n)\lambda_j^n|$. Moreover, if the value is strictly greater than 0, we show we can find an explicit bound Δ and $N \in \mathbb{N}$ such that for all $n > N$, we have $|\sum_{j=1}^{m} p_j(1/n)\lambda_j^n| > \Delta$. Lemma 4.3.12 follows as a special case.

We require some technical machinery from the theory of Diophantine approximations. We need the following theorem of Masser [124]. A proof can be found in [35] or [65].

Theorem A.4.2 ([124]) *Let $m \in \mathbb{N}$ be fixed and let $\lambda_1, \ldots, \lambda_m$ be complex algebraic numbers each of modulus 1. Consider the free Abelian group*

$$L = \{(v_1, \ldots, v_m) \in \mathbb{Z}^m : \lambda_1^{v_1}\lambda_2^{v_2}\ldots\lambda_m^{v_m} = 1\}.$$

L has a basis $\{\vec{\ell_1}, \ldots, \vec{\ell_p}\} \subseteq \mathbb{Z}^m$ (with $p \leq m$), where the entries of each of the $\vec{\ell_j}$ are all polynomially bounded in the total description length of $\lambda_1, \ldots, \lambda_m$. Moreover, such a basis can be can also computed in time polynomial in the total description length.

Let L be as described in Theorem A.4.2 above and suppose we have computed a basis $\{\vec{\ell_1}, \ldots, \vec{\ell_p}\} \subseteq \mathbb{Z}^m$. For each $j \in \{1, \ldots, p\}$, let $\vec{\ell_j} = (\ell_{j,1}, \ldots, \ell_{j,m})$. Now we define a set

$$T := \{(z_1, \ldots, z_m) \in \mathbb{C}^m : |z_1| = \cdots = |z_m| = 1 \text{ and } \\ \text{for each } j \in \{1, \ldots, p\},\ z_1^{\ell_{j,1}} \ldots z_m^{\ell_{j,m}} = 1\} \tag{A.7}$$

Notice that $|z| = 1 \iff \mathrm{Re}(z)^2 + \mathrm{Im}(z)^2 - 1 = 0$, and the $\ell_{j,k}$ are fixed integers, and thus the conditions above can be written as polynomials in the real and imaginary parts of z. Thus T is an algebraic set.

We now state a version of Kronecker's theorem on simultaneous Diophantine approximation. A derivation of this version of the theorem from the standard version ([72] Chap 23) can be found in [139].

Theorem A.4.3 (Kronecker's theorem, density version) *Let T be defined from $\lambda_1, \ldots, \lambda_m$ as in* (A.7)*. Then $\{(\lambda_1^n, \ldots, \lambda_m^n) : n \in \mathbb{N}\}$ is a dense subset of T.*

Theorem A.4.3 enables us to compute the lim inf by minimizing a function over a compact algebraic set:

Theorem A.4.4 *Let $\lambda_1, \ldots, \lambda_m$ be complex numbers of modulus 1. Let $p_1, \ldots, p_m$ be polynomials (with algebraic coefficients) with constant terms $c_1, \ldots, c_m$ respectively. Let $\mathbf{z} = (z_1, \ldots, z_m)$ and $\mathbf{c} = (c_1, \ldots, c_m)$. We have that*

$$\liminf_{n\to\infty} \left|\sum_{j=1}^{m} p_j(1/n)\lambda_j^n\right| = \liminf_{n\to\infty} \left|\sum_{j=1}^{m} c_j\lambda_j^n\right| = \inf_{\mathbf{z}\in T} |\mathbf{c}^\top \cdot \mathbf{z}| = \min_{\mathbf{z}\in T} |\mathbf{c}^\top \cdot \mathbf{z}|,$$

where T is the algebraic set computed in (A.7) *as the closure of $\{(\lambda_1^n, \ldots, \lambda_m^n) : n \in \mathbb{N}\}$.*

To prove the theorem, we need the following lemma that shows that we can replace the polynomials by their constant terms.

Lemma A.4.2 *Let $\lambda_1, \ldots, \lambda_m$ be complex numbers of modulus 1. Let $p_1, \ldots, p_m$ be polynomials (with algebraic coefficients) with constant terms $c_1, \ldots, c_m$ respectively. Then*

$$\liminf_{n\to\infty} \left| \sum_{j=1}^{m} p_j(1/n)\lambda_j^n \right| = \liminf_{n\to\infty} \left| \sum_{j=1}^{m} c_j \lambda_j^n \right|.$$

Proof *(of Theorem A.4.4). The first equality follows from Lemma A.4.2 and the second follows from Theorem A.4.3. The third equality holds because the function $\mathbf{z} \mapsto |\mathbf{c}^\top \cdot \mathbf{z}|$ is continuous and T is compact.*

Now, since T is an algebraic set, the minimum $\min_{\mathbf{z}\in T} |\mathbf{c}^\top \cdot \mathbf{z}|$ can be expressed in the theory of reals with addition and multiplication (omitting the encoding of absolute values):

$$\exists \mathbf{z} \in T.v = |\mathbf{c}^\top \cdot \mathbf{z}| \wedge \forall \mathbf{z}' \in T.v \leq |\mathbf{c}^\top \cdot \mathbf{z}'|$$

Therefore, by Tarski's theorem [147, 20, 44], we can characterize the unique v that attains the minimum.

Suppose the minimum v is some number $B > 0$. In this case, we require a bound $\Delta \in \mathbb{R}$ and $N \in \mathbb{N}$ such that $|\sum_{j=1}^{m} p_j(1/n)\lambda_j^n| > B$ for all $n > N$. By emulating the proof of Lemma A.4.2, we can find a bound N such that for all $n > N$, we have $|\sum_{j=1}^{m} p_j(1/n)\lambda_j^n| > B/2$. The required bounds are $\Delta = B/2$ and this N.

This concludes the proof of Lemma 4.3.12 and therefore also Theorem 4.3.1.

A.4.7. Proof of Lemma A.4.3

Lemma A.4.3 *1. Let $A \in \mathbb{R}^{m\times m}$ and $s \in \mathbb{R}^m$. If $\rho(A) < 1$, then for every $\delta > 0$ there exists an effectively computable $N \in \mathbb{N}$ and $\epsilon > 0$ such that after time N, all ϵ-pseudo-orbits are contained inside the ball $\mathcal{B}(\mathbf{0}, \delta)$.*

2. Suppose $J = diag(J_1, \ldots, J_z)$ is a Jordan normal form with one block J_i associated to eigenvalues outside the unit circle. for any $s = (s^1, \ldots, s^z)$ with $s^i \neq \mathbf{0}$ and for every $\delta > 0$, there exists an effectively computable $N \in \mathbb{N}$ and $\epsilon > 0$ such that after time N, all ϵ-pseudo-orbits of J are contained outside the ball $\mathcal{B}(\mathbf{0}, \delta)$.

Proof *Let $(x_n)_{n\in\mathbb{N}}$ denote an ϵ-pseudo-orbit starting from s with a sequence of disturbances $(d_n)_{n\in\mathbb{N}}$. Suppose $\rho(A) < 1$ and let $\gamma \in (\rho(A), 1)$. There is a constant $c > 0$ satisfying $\|A^n\|_2 \leq c\gamma^n$ for all n. Then we get*

$$\begin{aligned} \|x_n\|_2 &= \left\| A^n s + \sum_{k=0}^{n-1} A^k d_{n-k-1} \right\|_2 \leq \|A^n\|_2 \|s\|_2 + \sum_{k=0}^{n-1} \left\|A^k\right\|_2 \|d_{n-k-1}\|_2 \\ &\leq c\gamma^n \|s\|_2 + \sum_{k=0}^{n-1} m\,\epsilon\, c\gamma^k \leq c\gamma^n \|s\|_2 + \frac{m\,\epsilon\, c}{1-\gamma}. \end{aligned}$$

Taking $\epsilon = \delta(1-\gamma)/(2mc)$ and N with $\gamma^N \|s\|_2 \leq \delta/(2c)$ gives the intended result.

For the second part of the lemma, take an ϵ-pseudo-orbits of J as $(s = x_0, x_1, x_2, \ldots)$ with $x_{n+1} = Jx_n + d_n$. We have $\|x_n\|_2 \geq \left\|x_n^i\right\|_2$ where the states x_n are partitioned according to the structure of J and x_n^i is the one associated with J_i. Note that $(s^i = x_0^i, x_1^i, x_2^i, \ldots)$ satisfy $x_{n+1}^i = J_i x_n^i + d_n^i$. There is a constant $c > 0$ satisfying $\left\|J_i^{-n}\right\|_2 \leq c\gamma^n$ for all n with some $\gamma \in (\rho(J_i^{-1}), 1)$. We can write

$$
\begin{aligned}
& x_n^i = J_i^n s^i + \sum_{k=0}^{n-1} J_i^k d_{n-k-1}^i \quad \Rightarrow \quad s^i = J_i^{-n} x_n^i - \sum_{k=0}^{n-1} J_i^{k-n} d_{n-k-1}^i \\
& \Rightarrow \left\|s^i\right\|_2 \leq \left\|J_i^{-n}\right\|_2 \left\|x_n^i\right\|_2 + \sum_{k=0}^{n-1} \left\|J_i^{k-n}\right\|_2 \left\|d_{n-k-1}^i\right\|_2 \leq c\gamma^n \left\|x_n^i\right\|_2 + \frac{c\,m\,\epsilon}{1-\gamma} \\
& \Rightarrow \left\|x_n^i\right\|_2 \geq \frac{1}{c\gamma^n}\left[\left\|s^i\right\|_2 - \frac{c\,m\,\epsilon}{1-\gamma}\right].
\end{aligned}
$$

It is sufficient to take $\epsilon = \frac{(1-\gamma)}{2cm}\left\|s^i\right\|_2 > 0$ and N sufficiently large such that $2\delta c\gamma^N < \left\|s^i\right\|_2$. This forces x_n^i (thus also x_n) to move outside the ball $\mathcal{B}(\mathbf{0}, \delta)$ for all $n > N$.

A.4.8. Proof of Lemma A.4.2

We can write $p_j(1/n)$ as $c_j + \sum_{i=1}^{d_j} c_{(j,i)} \frac{1}{n^i}$, where c_j is the constant term, $c_{(j,i)}$ are the other coefficients, and d_j is the degree. Define $A_i = \sum_{i=1}^{d_j} |c_{(j,i)}|$ and observe that

$$
|p_j(1/n) - c_j| < |\sum_{i=1}^{d_j} c_{(j,i)} \frac{1}{n^i}| < \frac{\sum_{i=1}^{d_j} |c_{(j,i)}|}{n} = \frac{A_j}{n}
$$

Thus for any ϵ, setting $N_j(\epsilon) = \lceil A_j/\epsilon \rceil$ ensures that

$$
n > N_j(\epsilon) \implies |p_j(1/n) - c_j| < \epsilon.
$$

Define $N(\epsilon) = \max_{j \in \{1,\ldots,m\}} N_j(\epsilon/m)$.

Claim A.1 *Let S_n be defined as $|\sum_{j=1}^m c_j \lambda_j^n|$. For all $\epsilon > 0$,*

$$
S_n - \epsilon \leq |\sum_{j=1}^m p_j(1/n)\lambda_j^n| \leq S_n + \epsilon
$$

Taking the limit inferior of each term gives us the desired result.

Proof *(of Claim A.1) We write*

$$
|\sum_{j=1}^m p_j(1/n)\lambda_j^n| = |\sum_{j=1}^m (c_j + p_j(1/n) - c_j)\lambda_j^n|,
$$

which gives us

$$S_n - |\sum_{j=1}^{m}(p_j(1/n) - c_j)\lambda_j^n| \leq |\sum_{j=1}^{m} p_j(1/n)\lambda_j^n| \leq S_n + |\sum_{j=1}^{m}(p_j(1/n) - c_j)\lambda_j^n|$$

and thus

$$S_n - \sum_{j=1}^{m}|(p_j(1/n) - c_j)\lambda_j^n| \leq |\sum_{j=1}^{m} p_j(1/n)\lambda_j^n| \leq S_n + \sum_{j=1}^{m}|(p_j(1/n) - c_j)\lambda_j^n|,$$

by elementary properties of sums of absolute values. Observing that λ_js have absolute value 1, we can reduce the proposition above to

$$S_n - \sum_{j=1}^{m}|(p_j(1/n) - c_j)| \leq |\sum_{j=1}^{m} p_j(1/n)\lambda_j^n| \leq S_n + \sum_{j=1}^{m}|(p_j(1/n) - c_j)|.$$

Now setting $n > N(\epsilon) = \max_{j\in\{1,\ldots,m\}} N_j(\epsilon/m)\}$, we have $|(p_j(1/n) - c_j)| < \epsilon/m$ for all j, which gives us

$$S_n - \epsilon \leq \left|\sum_{j=1}^{m} p_j(1/n)\lambda_j^n\right| \leq S_n + \epsilon$$

A.4.9. Computing Real JNF in Polynomial Time

We discuss how to compute the the real Jordan normal form of A in polynomial time. First compute, in polynomial time, the (complex) Jordan normal form J' and matrices T, T^{-1} such that $A = TJ'T^{-1}$ using the algorithm from [34].

Computing J: Suppose, without loss of generality, that

$$J' = \text{diag}(J'_1, J'_2, \ldots, J'_{2k-1}, J'_{2k}, J'_{2k+1}, \ldots, J'_{2k+z})$$

where for $1 \leq j \leq k$, the Jordan blocks J_{2j-1} and J_{2j} have the same dimension and have conjugate eigenvalues $\lambda_j = a_j + b_j i$ and $\overline{\lambda} = a_j - b_j i$, respectively. The blocks $J'_{2k+1}, \ldots, J'_{2k+z}$, on the other hand, have real eigenvalues. J is obtained by replacing, for each $1 \leq j \leq k$, $\text{diag}(J'_{2j-1}, J'_{2j})$ with a real Jordan block of the same dimension with $\Lambda = \begin{bmatrix} a & -b \\ b & a \end{bmatrix}$ and keeping the blocks $J'_{2k+1}, \ldots, J'_{2k+z}$ unchanged.

Computing P: Let $\kappa(j)$ denote the multiplicity of the Jordan block J'_i for $1 \leq i \leq 2k+z$, and $v_1^1, \ldots, v_{\kappa(1)}^1, \ldots, v_1^{2k}, \ldots, v_{\kappa(2k)}^{2k}, \ldots, v_1^{2k+z}, \ldots, v_{\kappa(2k+z)}^{2k+z} \in \bar{\mathbb{Q}}^m$ be the columns of T. It will be the case that for all $1 \leq j \leq k$ and l, $v_l^{2j-1} = \overline{v_l^{2j}}$ in the sense that $v_l^{2j-1} = x_l^j + y_l^j i$ and $v_l^{2j} = x_l^j - y_l^j i$ for vectors $x_l^j, y_l^j \in \mathbb{R}^m$. Moreover, for $j > 2k$, $v_l^{2j} \in \mathbb{R}^m$. Finally, columns of P are obtained from columns of T as follows. For $1 \leq j \leq k$ and all l, replace v_l^{2j-1} with x_l^j and v_l^{2j} with y_l^j and keep v_l^{2k+z} for all l and $m > 0$ unchanged, in the same way the proof of existence of real Jordan normal form proceeds.

Computing P^{-1}: Summarizing the construction above, P is obtained from T by replacing columns $x + yi$ and $x - yi$, $x, y \in \mathbb{R}^m$ by x and y, respectively. Since $x = \frac{1}{2}(x+yi)+\frac{1}{2}(x-yi)$ and $y = -\frac{1}{2}i(x+yi)+\frac{1}{2}i(x-yi)$, this construction is linear and we can write $P = T \cdots A$ for some $A \in \mathbb{C}^{m\times m}$ with entries in $\{\frac{1}{2}, -\frac{1}{2}, \frac{1}{2}i, -\frac{1}{2}i, 1, 0\}$. Moreover, the linear transformation is clearly invertible: $x + yi = 1 \cdot x + iy$ and $x - yi = 1 \cdot x - (-i)y$, and hence $A^{-1} \in \mathbb{C}^{m\times m}$ with entries in $\{1, i, -i\}$. Finally, compute P^{-1} via $P = TA \implies P^{-1} = A^{-1}T^{-1}$, observing that we already know how to compute T^{-1} in polynomial time.

Curriculum Vitae

Research Interests

Abstraction-Based Controller Design, Learning in Control, Reinforcement Learning, Multi-Agent Systems, Robustness of Neural Networks.

Education

[2018–]	Doctoral Researcher, Max Planck Institute for Software Systems, Germany. Advisors: Rupak Majumdar.
[2011–2014]	M.Sc. in Electrical Engineering, University of Tehran, Iran.
[2006–2011]	B. Sc. in Electrical Engineering, University of Tehran, Iran.