UNIVERSITY OF CALIFORNIA

Los Angeles

Formal Synthesis and Data-Driven Verification of Cyber-Physical Systems

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical Engineering

by

Ayça Balkan

2017

ABSTRACT OF THE DISSERTATION

Formal Synthesis and Data-Driven Verification of

Cyber-Physical Systems

by

Ayça Balkan

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 2017

Professor Paulo Tabuada, Chair

At a conference[1] in March 2015, while advocating self-driving cars, Elon Musk, the chief executive officer of Tesla Motors, referred to (traditional) cars as "two-ton death machines". The biggest promise of autonomous systems is the elimination of, or compensation for, the human-error factor. However, making sure that such systems which are designed by humans themselves, will behave as we expect them to do, is an unarguably challenging task. According to the RAND Corporation's recent findings in [KP16], demonstrating the failure rate of an autonomous vehicle to a particular degree of precision would take about 8.8 billion miles of test driving, which corresponds to *400 years with a fleet of 100 autonomous vehicles.* This suggests that, as engineers, we need to develop alternative methods to excessive, brute force testing in order to fulfill the promise of bringing a safe product to the market. Moreover, in safety-critical applications such as self-driving cars, this promise should not only be filled, but it should be provably filled.

Most autonomous systems that are safety-critical also have a cyber-physical nature, meaning that they involve an interplay between cyber and physical components. In fact, the complex interactions between these components are what makes the design and verification of Cyber-Physical Systems (CPS) notoriously difficult. In this thesis, we look at the problem of correct design of CPS from two different angles. In the first chapter, we position ourselves within a correct-by-design paradigm, where the idea is not to spend a

[1] 2015 NVIDIA GPU Technology Conference, San Jose

heroic effort in the verification phase, by ensuring that the design process itself leads to a certificate of correctness. In the second chapter, we concentrate on generating correctness certificates for a completed design candidate. Here, with the challenges specific to the CPS in mind, we do not take a model based approach, but instead, a data driven one.

In the first chapter the problem setting is as follows: "Given a CPS and a desired specification in a formal language such as Linear Temporal Logic, can we synthesize provably-correct controller software that will enforce the specification?" This question has already been addressed in the literature by bringing tools from computer science into the realm of control-theory. However, for an arbitrary specification the resulting methods are computationally expensive to a point that synthesis becomes intractable for large-scale systems. In this thesis, we address this problem by studying a certain class of specifications, which we call *mode-target formulas.* This class is small enough to lead to tractable algorithms, yet rich enough to answer interesting control problems. It is inspired by numerous control applications where there are multiple modes and corresponding (possibly multiple) targets to be reached and attained, hence the name. In Chapter 1, we formally define the mode-target formulas and then propose a solution methodology to perform controller synthesis, when the specifications are given in a mode-target setting.

In the second part of the thesis, we make use of a more traditional design tool for control theorists: Lyapunov functions. These functions serve as certificates for stability and thus have a significant role in the verification of dynamical systems. Even though stability analysis using Lyapunov techniques is a well-studied problem, the majority of the current literature only addresses it when the dynamics of the system is known and can be expressed in the form of a differential or difference equation. This creates a gap between the theory and practice because, for most industrial scale systems, there is no such simple representation of the underlying dynamics. Instead, what is available to the engineers in most scenarios is the ability to perform simulations. With this observation in mind, in Chapter 2, we take the first step to close this gap. In particular, provided that the underlying dynamics can be modeled as a switched linear system, we propose a technique to analyze the stability of the system by observing the response of the system to uniformly

randomly sampled initial conditions.

The dissertation of Ayça Balkan is approved.

Todd Millstein

Vwani Roychowdhury

Lieven Vandenberghe

Paulo Tabuada, Committee Chair

University of California, Los Angeles

2017

Anne ve babama.

Table of Contents

List of Figures

List of Tables

Acknowledgments

Looking back at the last 6 years, I wouldn't be able to create this thesis without the intellectual and emotional help of many people. This section is for them.

I want to start with my brilliant, enthusiastic advisor and mentor Paulo Tabuada. Paulo will always be an inspiration for me with his curious, rigorous and aesthetically pleasing approach to scientific problems. I cannot thank him enough for his great intellectual input, patience, extremely detailed feedback and availability. He made sure that the only thing I needed to stress about was science and took care of the rest behind the curtains - what a great privilege that was! Thank you for everything, Paulo.

Next, I want to thank my committee members Lieven Vandenberghe, Vwani Roychowdhury and Todd Millstein for their valuable feedback and making this PhD possible. I also cannot emphasize enough how helpful Electrical Engineering Graduate Student Office has been throughout my whole PhD journey. Thank you Deeona, Mandy, Ryo for answering all my questions and helping me navigate in this overly complicated world of rules and procedures.

I am very grateful that I got to collaborate with the smartest people I know. I want to thank Moshe Vardi for a very inspiring and pleasant visit to Rice University, which led to the first chapter of this thesis. My collaborators at Toyota Technical Center showed me that scientific pursuit is still possible in the corporate world. I want to thank James Kapinski, Xiaoqing Jin and Jyotirmoy Deshmukh for that. A special thanks goes to Jyotirmoy Deshmukh. With no exaggeration, he is the most pleasant person that I have ever worked with! His intellectual and emotional guidance is what kept me going at hard times. It has been an absolute pleasure to know you, Jyo. This thesis wouldn't have been possible without the generous support and hard work of my labmates Omar Hussien, Joris Kenanian, Matthias Rungger, Yasser Shoukry and Sina Caliskan. I especially want to thank my coauthors Omar Hussien and Joris Kenanian for their open-handed help. I want to thank Raphael Jungers as well. I truly appreciate his contributions to the second chapter of this thesis and feel so lucky to get to work with such a bright and friendly human being.

The process of getting a PhD is not only filled with intellectual challenges but a ton of emotional ones as well. Hence, I want to thank my friends who were with me along the way. Without them, this thesis wouldn't exist, that is for sure. I want to thank Can Karakus for always being there

for me. Through Skype, through Google chat, through phone, through coffee breaks, through grand meals. We got through this together. He supported me in every possible way and I want to thank all the dices that were rolled in the universe for crossing our paths. Thank you Can. Another big thank you goes to Jonathan Cheng. I don't how what I would do without his wisdom. He always gave me great perspective and massive emotional support through infinite laughters. Thank you for being my pillar, Jon. I also want to thank Daniel O'Connor. He is the full-package. Not only he answered all my optimization related questions, but he was always there when I needed a friend. He inspired me more than he knows. I have never met someone with so much willingness to learn. Thank you, Daniel. I also owe a great amount of thanks to Natascha Chtena, Onur Guyaguler, Elif Semra Ceylan, Jed Lacktriz, Sezai Emre Tuna, Zach Schlossberg, Laura Almagor, Zach Buchman, Kostya Bakhurin, Yifan Sun, Shaunak Mishra and Caitlin Decker. You people are the reason why I was able to keep going at various stages in this process. Thank you. Last but not least, Minnos, the bipolar yet the best cat in the world. The smiles he put on my face really assisted me through all the bumps. Thanks my little cat friend.

Finally, my parents Ayse Balkan and Tuna Balkan. This thesis exists because of the sacrifices they made, the support they provided and the infinite love they poured on me through these years despite the continents between us. Thanks Balkans, this is for you.

Vita

2010	B.Sc. (Electrical and Electronics Engineering), Middle East Technical University, Ankara, Turkey.
2012	M.Sc. (Electrical Engineering), University of California, Los Angeles, CA
Winter 2012	Preliminary Exam Fellowship, Electrical Engineering, University of California, Los Angeles, CA
Summer 2015	Co-op, Model Based Development Group, Toyota Technical Center, Gardena, CA
Summer 2016	Software Modeling Intern, Medtronic Inc., Northridge, CA
October, 2016	Best Paper Awardee, ACM SIGBED International Conference on Embedded Software, EMSOFT 16'

Publications

Ayca Balkan, Paulo Tabuada, Jyotirmoy Deshmukh, Xiaoqing Jin, James Kapinski *"Underminer: A Framework for Identifying Non-converging Behaviors in Black Box System Models"* (submitted to) ACM Transactions on Embedded Computing Systems.

Jonathan DeCastro, Rudiger Ehlers, Matthias Rungger, **Ayca Balkan**, Hadas Kress-Gazit, *"Automated generation of dynamics-based runtime certificates for high-level control"* Journal of Discrete Event Dynamical Systems: Special Topical Issue on Formal Methods in Control, June 2017, Volume 27, Issue 2, pp 371-405.

Ayca Balkan, Paulo Tabuada, Jyotirmoy Deshmukh, Xiaoqing Jin, James Kapinski *"Underminer: A Framework for Identifying Non-converging Behaviors in Black Box System Models"* ACM SIGBED International Conference on Embedded Software, EMSOFT 16'.

Petter Nilsson, Omar Hussien, **Ayca Balkan**, Yuxiao Chen, Aaron Ames, Jessy Grizzle, Necmiye Ozay, Huei Peng, Paulo Tabuada *"Correct-By-Construction Adaptive Cruise Control: Two Approaches"* IEEE Transactions on Control Systems Technology, July 2016, Volume: 24, Issue: 4, pp 1294 - 1307.

Ayca Balkan, Paulo Tabuada, Moshe Vardi, *"Mode-Target Games: Reactive Synthesis for Control Applications."* (conditionally accepted for publication in) IEEE Transactions on Automatic Control.

Ayca Balkan, Jyotirmoy Deshmukh, James Kapinski, Paulo Tabuada, *"Simulation-Guided Contraction Analysis"* 1st Indian Control Conference, ICC 15'.

Petter Nilsson, Omar Hussien, Yuxiao Chen, **Ayca Balkan**, Matthias Rungger, Aaron Ames, Jessy Grizzle, Necmiye Ozay, Huei Peng, Paulo Tabuada *"Preliminary Results on Correct-by-Construction Control Software Synthesis for Adaptive Cruise Control"* 53rd IEEE Conference of Decision and Control, CDC '14.

Paulo Tabuada, **Ayca Balkan**, Sina Yamac Caliskan, Yasser Shoukry and Rupak Majumdar, *"Input-output robustness for discrete systems."* ACM SIGBED International Conference on Embedded Software, EMSOFT 12'.

Ayca Balkan, Min Gao, Lei He, Paulo Tabuada *"A Behavioral Algorithm for State of Charge Estimation"* 26th Electrical Vehicle Symposium, EVS 12'.

CHAPTER 1

Controller Synthesis for Mode-Target Games

The results in this chapter are developed under the correct-by-design philosophy for Cyber-Physical Systems (CPS) advocating control design methodologies that produce, not only the controller, but also a proof of its correctness. This design philosophy should be contrasted with the widely used design-and-verify approach under which a designer re-designs the controller to weed out the bugs that are found during multiple verification rounds. By placing greater emphasis and effort in the design phase it is possible to greatly reduce the verification efforts thereby reducing the design time and cost of complex CPS [Var08, Ses15, Sif15, AHV15].

The correct-by-design philosophy, however, is not without its own challenges and the purpose of this chapter is to address one of the most critical: computational complexity. If one takes Linear Temporal Logic (LTL) as the specification formalism, it is known that synthesizing a controller enforcing such specifications is doubly exponential in the length of the formula. This led several researchers to seek fragments of LTL that are small enough for the complexity of synthesis to be lower, yet large enough to be practically relevant [AMP98, AL04, BJP12, Ehl11, WTM13, KE12]. Among these, the one that had the biggest practical impact was the Generalized Reactivity (1) fragment, abbreviated as GR(1), for which the controller synthesis can be solved in polynomial time in the size of the transition system [BJP12]. Even though the GR(1) fragment was not originally intended for control applications, several researchers demonstrated its usefulness to synthesize correct-by-design controllers in practical scenarios [KF07, LOT13]. Later, extending the ideas in [BJP12], the Generalized Rabin (1) fragment was shown to be the largest class of LTL specifications for which the controller synthesis problem is still polynomial in the size of the transition system, unless P=NP [Ehl11].

In this chapter, inspired by control applications, we introduce a new fragment of LTL termed Mode-Target (MT). An MT formula describes a setting where there are modes and corresponding targets for each mode. When the system is in a certain mode, the specification requires the system to reach one of the possible targets for that mode and stay there as long as the mode does not change. If the mode changes, there is no obligation to reach or stay within the target region of the previous mode. We use MT formulas to define mode-target games, a subclass of LTL games. The winning condition of an MT game is an MT formula and, moreover, the game graph conforms to additional restrictions on the structure of the modes. We believe that modeling the desired behavior of control systems in this way, via modes and targets, is quite natural for designers. We support this claim in Section 2 by giving three concrete examples from different application domains that illustrate the usefulness of MT games. The first example is an adaptive cruise controller, whose specifications are outlined by the International Standardization Organization (ISO). The second example builds on [JDK14], where researchers from the Toyota Technical Center described the desired behavior for an air-fuel-ratio controller in signal temporal logic. The third example is the control of certain chemicals inside a nuclear power plant during shutdown and startup operations as outlined in [HRK09]. We show that the controller synthesis problem for all of these examples can be posed as finding a winning strategy for an MT game.

The contributions of this chapter can be summarized as follows:

- We propose MT as a practically useful LTL fragment from a modeling perspective. We provide three concrete control applications as an illustration of the large class of problems that can be naturally modeled as MT games.
- We introduce the notion of simple games that abstracts the key properties of GR(1) and MT games so as to prove the correctness and complexity of the proposed algorithms in a transparent manner. In doing so, we provide a new and simpler proof for the correctness and complexity estimates of the existing controller synthesis algorithms for GR(1) while highlighting the commonalities and differences between GR(1) and MT games. In particular, we show that MT games are also GR(1) games.

- We propose an algorithm to synthesize controllers enforcing MT specifications which requires $O(\sum_i t_i n^2)$ symbolic steps where n is the number of states in the game graph and t_i is the number of targets corresponding to mode i. In contrast, the complexity of the algorithm resulting from embedding MT games into GR(1) games and using existing synthesis algorithms for the GR(1) fragment is $O(\sum_i tn^2) = O(mtn^2)$ where m is the number of modes and t is the largest number of targets across all the modes. Although these two complexity upper bounds coincide when the number of targets for each mode is the same, we empirically show in Section 5 that the proposed synthesis algorithm still outperforms the synthesis algorithm obtained via the GR(1) embedding in this situation.

The rest of the chapter is organized as follows. In Section 1, we review the syntax and semantics of LTL and introduce LTL games. We formally define MT games in Section 2 and illustrate their usefulness via examples from control. In Section 3 we present an algorithm for solving MT games. We then show in Section 4 that every MT game can be formulated as a GR(1) game. This leads to an alternative solution for MT games via existing algorithms to solve GR(1) games. We experimentally compare the two algorithms for the solution of MT games in Section 5 and conclude this chapter with Section 6.

1 Preliminaries

We start by reviewing the syntax and semantics of Linear Temporal Logic (LTL) and corresponding games.

1.1 Linear Temporal Logic

Consider a set of atomic propositions P. LTL formulas are constructed according to the following grammar:

$$\varphi ::= p \in P \,|\, \neg\,\varphi \,|\, \varphi \,\vee\, \varphi \,|\, \bigcirc \varphi \,|\, \varphi \,\mathcal{U}\, \varphi.$$

We denote the set 2^P by Σ, where 2^P is the set of all subsets of P. An *infinite word* is an element of Σ^ω where Σ^ω denotes the set of all infinite strings or words obtained by concatenating elements or letters in Σ. Similarly, the set of all finite strings is denoted by Σ^*. We also regard elements $w \in \Sigma^\omega$ as maps $w : \mathbb{N} \to \Sigma$. Using this interpretation we denote $w(i)$ by w_i. In the context of LTL, the index i models time and w_i is interpreted as the set of atomic propositions that hold at time i.

The semantics of an LTL formula φ is described by a satisfaction relation $\models$ that defines when the string $w \in \Sigma^\omega$ satisfies the formula φ at time $i \in \mathbb{N}$, denoted by $w, i \models \varphi$:

- For $p \in P$, we have $w, i \models p$ iff $p \in w_i$,
- $w, i \models \neg\varphi$ iff $w, i \not\models \varphi$,
- $w, i \models \varphi \vee \psi$ iff $w, i \models \varphi$ or $w, i \models \psi$,
- $w, i \models \bigcirc\varphi$ iff $w, i+1 \models \varphi$,
- $w, i \models \varphi\,\mathcal{U}\,\psi$ iff there exists $k \geq i$ such that $w, k \models \psi$ and for all $i \leq j < k$, we have $w, j \models \varphi$.

We write $w \models \varphi$ when $w, 0 \models \varphi$. We use the short hand notation $\varphi \wedge \psi$, for $\neg(\neg\varphi \vee \neg\psi)$, and **True** for $\neg\varphi \vee \varphi$. We further abbreviate $\mathbf{True}\,\mathcal{U}\,\varphi$ as $\Diamond\varphi$ which means that φ *eventually* holds and $\neg\Diamond\neg\varphi$ by $\Box\varphi$, which says that φ *always* holds. We call the operators $\bigcirc$, $\mathcal{U}$, $\Box$, and $\Diamond$ *temporal operators*.

We write $W(\varphi)$ to denote the set of all infinite words which satisfy φ, i.e., $W(\varphi) := \{\sigma \in \Sigma^\omega | \sigma \models \varphi\}$. We say that ψ_1 and ψ_2 are *semantically equivalent*, and write $\psi_1 \equiv \psi_2$, if $W(\psi_1) = W(\psi_2)$.

1.2 Games

A two-player player game over a *game graph* is a tuple $G = (V, E, P, L)$ consisting of:

- A finite set V of states partitioned into V_0 and V_1, i.e., $V = V_0 \cup V_1$ and $V_0 \cap V_1 = \varnothing$;
- A *transition relation* $E \subseteq V \times V$;
- A finite set of atomic propositions P;
- A *labeling function* $L : V \to 2^P$ mapping every state in V to the set of atomic propositions that hold true on that state.

In this definition, V_0 and V_1 are the states from which only player 0 and player 1 can move, respectively. Thus, the state determines which player can move. We assume that for every state $v \in V$, there exists some $v' \in V$ such that $(v, v') \in E$. The function L can be naturally extended to infinite strings $r \in V^\omega$ by $L(r) = L(r_0)L(r_1)L(r_2)\ldots \in \Sigma^\omega$.

A *play* r in a game graph G is an infinite sequence of states $r = v_0 v_1 \ldots \in V^\omega$, such that for all $i \geq 0$, we have $(v_i, v_{i+1}) \in E$. A strategy for player 0 is a partial function $f : V^* \times V_0 \to V$ such that whenever $f(r, v)$ is defined $(v, f(r, v)) \in E$. We denote the set of all plays under strategy f starting from state v by $\Omega_{f,v}(G)$, and the set of all possible plays for a given game graph G by $\Omega(G)$. For a given LTL formula φ and a game graph $G = (V, E, P, L)$, we use $W_G(\varphi)$ as the short-hand notation for $W(\varphi) \cap L(\Omega(G))$.

For the purposes of this chapter, an *LTL game* is a pair (G, φ) consisting of a game graph G, and a *winning condition* φ which is an LTL formula. A play r in a game (G, φ) is winning for player 0 if $L(r) \in W(\varphi)$. A strategy f for player 0 is *winning* from state v, if all plays starting in v which follow f are winning for player 0. For a given game (G, φ), $[\![\varphi]\!]_G$ denotes the set of states from which player 0 has a winning strategy, this is the *winning set* of player 0. When it is clear from the context which game graph we are

referring to, we drop the subscript and just write $[\![\varphi]\!]$.

The sets from which player 0 can force a visit to a set of states V' is denoted by $\mathrm{Pre}(V')$, i.e.,

$$\mathrm{Pre}\,(V') = \{v \in V_0 \,|\, \exists_{v' \in V'}\ (v, v') \in E\} \cup \{v \in V_1 \,|\, \forall_{v' \in V}\ (v, v') \in E \Rightarrow v' \in V'\}$$

We recall the following fixed-point notation for a given monotone mapping $F : 2^V \to 2^V$:

$$\nu X F(X) = \cap_i X_i, \text{ where } X_0 = V, \text{ and, } X_{i+1} = F(X_i), \text{ and}$$
$$\mu X F(X) = \cup_i X_i, \text{ where } X_0 = \varnothing, \text{ and } X_{i+1} = F(X_i).$$

In other words, $\nu X F(X)$ and $\mu X F(X)$ are the greatest and least fixed-point of the mapping F, respectively.

In the rest of the chapter, we abuse notation and sometimes use a set of states $V' \subseteq V$ as an LTL formula. In this case V' is to be interpreted as an atomic proposition that holds only on the states in V'. Whenever, V' defines an atomic proposition not in P, we can always extend P to contain V'. However, for the sake of simplicity we will not explicitly do so.

We call φ a *positional formula* if it does not contain any temporal operators and a *reachability formula* if $\varphi = \Diamond p$ for some positional formula p. We say that φ is a *GR(1) formula* if it has the following form:

$$\varphi = \bigwedge_{i_1 \in I_1} \Box\Diamond a_{i_1} \implies \bigwedge_{i_2 \in I_2} \Box\Diamond g_{i_2}, \tag{1.1}$$

for some positional formulas a_{i_1}, g_{i_2} and finite sets I_1 and I_2. We call games with winning conditions given as a GR(1) formula *GR(1) games*. We refer the reader to [BJP12] for further details on GR(1) formulas.

2 Mode-Target Games

2.1 Motivation

As the automotive technology evolves, conventional cruise control (CCC) is being replaced by adaptive cruise control (ACC). ACC has two *modes* of operation: the speed mode and the time-gap mode. In the speed mode, ACC behaves exactly like CCC, i.e., it reaches a pre-set speed and maintains it. The time-gap mode is what differentiates ACC from CCC. In this mode, ACC keeps pace with the car in front, the *lead car*. This pace is characterized by the *headway*, the quantity that captures the time required by the ACC equipped vehicle to brake and avoid a collision when the lead car suddenly slows down. We consider the specifications for ACC set by the International Organization of Standardization (ISO) in [ISO10]. Following these specifications, the target region corresponding to the speed mode can be defined as $\{v : |v - v_{\text{des}}| \leq \epsilon_v\}$, where v, v_{des} and ϵ_v denote the velocity of the car, the desired velocity, and the allowable tolerance for the velocity respectively. Similarly, the target region of the time gap mode is formalized as $\{\tau : |\tau - \tau_{\text{des}}| \leq \epsilon_\tau\}$, where τ is the headway, τ_{des} is the desired headway, and ϵ_τ is the desired tolerance for the headway.[1] In each mode, the specification is to reach and stay in the desired target region as long as the current mode does not change. We can express this specification as the conjunction of individual specifications for the time-gap mode and the speed mode, i.e., $\varphi_{\text{timegap}} \wedge \varphi_{\text{speed}}$, where:

$$\varphi_{\text{timegap}} := (\Diamond\Box M_{\text{timegap}} \implies \Diamond\Box T_{\text{timegap}}), \tag{1.2}$$

$$\varphi_{\text{speed}} := (\Diamond\Box M_{\text{speed}} \implies \Diamond\Box T_{\text{speed}}). \tag{1.3}$$

Here, M_{timegap} and M_{speed} are the atomic propositions that hold whenever the corresponding modes are active. Similarly, T_{timegap} and T_{speed} are satisfied when $\tau \in \{\tau : |\tau - \tau_{\text{des}}| \leq \epsilon_\tau\}$ and $v \in \{v : |v - v_{\text{des}}| \leq \epsilon_v\}$, respectively.

[1] In addition to the mode-target behavior, [ISO10] requires the headway to be kept above a certain value regardless of the mode and at all times. However, this is a simple safety specification for which a controller can be synthesized separately and composed with the mode-target controller afterwards.

Table 1.1: The modes and the corresponding target A/F ratios as given in [JDK14]. In this table, λ_{ref}, λ_{ref}^{pwr} correspond to the optimal A/F ratios in normal and power-enrichment mode. The corresponding atomic propositions are written in parentheses.

Mode	**Target A/F Ratio**
Start-up ($M_{\text{start-up}}$)	$[0.9\lambda_{ref}, 1.1\lambda_{\text{ref}}]$ ($T_{\text{start-up}}$)
Normal (M_{normal})	$[0.98\lambda_{ref}, 1.02\lambda_{\text{ref}}]$ (T_{normal})
Power-Enrichment (M_{power})	$[0.8\lambda_{\text{ref}}^{\text{pwr}}, 1.2\lambda_{\text{ref}}^{\text{pwr}}]$ (T_{power})
Fault (M_{fault})	$[0.9\lambda_{\text{ref}}, 1.1\lambda_{ref}]$ (T_{fault})

Implication (1.2) only requires the time gap to be reached if the system enters and stays in the time gap mode forever. Hence, it seems that a controller may simply ignore the time gap mode if it knows that this mode will be eventually left. However, since we synthesize *causal* controllers, i.e., controllers that cannot foretell the future, any such controller will start driving the system to the time gap target once the system enters the time gap mode. Similarly, once the system leaves the time gap mode to enter the speed mode there is no need to reach the time gap mode anymore and the controller starts driving the system to the speed target. This is consistent with the ACC requirements in the ISO standard [12] that do not require a target to be reached once the corresponding mode is left.

We now consider an engine control example: the control of a combustion engine. As the researchers in the Toyota Technical Center argued in [JDK14], the specifications for the air-fuel (A/F) ratio controller of an internal combustion engine can be naturally expressed in terms of modes and corresponding targets. We now summarize these specifications given in [JDK14]. There are four different modes of operation: start-up mode, normal mode, power-enrichment mode, and fault mode. Only one of these modes is active at any given time. Furthermore, for each mode there is a required A/F ratio. The specification for the controller is to bring the A/F ratio to this target value and keep it there unless the mode changes. We compile the target A/F ratios corresponding to each mode in Table 1.1, where λ_{ref}, and $\lambda_{\text{ref}}^{\text{pwr}}$ are the optimal A/F ratios for normal and "full throttle" driving conditions respectively. Defining the atomic propositions for modes and targets according to Table 1.1, we get the following LTL formula that captures the desired

behavior: $\varphi_{\text{start-up}} \wedge \varphi_{\text{normal}} \wedge \varphi_{\text{power}} \wedge \varphi_{\text{fault}}$, where

$$\varphi_{\text{start-up}} := (\Diamond\Box M_{\text{start-up}} \implies \Diamond\Box T_{\text{start-up}}),$$
$$\varphi_{\text{normal}} := (\Diamond\Box M_{\text{normal}} \implies \Diamond\Box T_{\text{normal}}),$$
$$\varphi_{\text{power}} := (\Diamond\Box M_{\text{power}} \implies \Diamond\Box T_{\text{power}}),$$
$$\varphi_{\text{fault}} := (\Diamond\Box M_{\text{fault}} \implies \Diamond\Box T_{\text{fault}}).$$

The last example we present is the control of a pressurized water reactor[2] during shutdown and start-up stages. Even though the chemical processes that take place in nuclear power plants are well studied under normal conditions, they are still yet to be fully understood in the presence of transient behaviors, particularly during shutdown and start-up. Therefore, it is important to ensure correct operation during these critical phases. In [HRK09], the authors document the specifications set by Électricité de France (EdF) for both of these modes of operation. Here we present a simplified version of these specifications. According to [HRK09], there are two shutdown procedures that can be followed based on the current temperature and concentration of the materials in the plant: hot shutdown and cold shutdown. In the hot shutdown mode, there is a target hydrogen concentration that must be achieved. In the cold shutdown mode, the shutdown can be performed with or without oxygenation depending on factors such as financial cost, risk, and specifics of the power plant. For both of these modes the control objective is to attain and sustain a certain chemical content in the reactor. Table 1.2 summarizes these target chemical concentrations corresponding to each operation mode. Accordingly, in this case the LTL formula describing the desired behavior is $\varphi_{\text{start-up}} \wedge \varphi_{\text{cold}} \wedge \varphi_{\text{hot}}$, which is conjunction of the specifications for the start-up mode, the hot shutdown mode, and the cold shutdown

[2] A pressurized water reactor is a type of nuclear power plant that constitutes the majority of nuclear power plants in Western countries, including the US.

Table 1.2: The modes and the targeted concentration of chemicals in each mode as given in [HRK09]. In parentheses, we provide the notation for the atomic propositions corresponding to each mode and target.

Mode	**Target Chemical Content**
Start-up ($M_{\text{start-up}}$)	Sodium < 0.1 mg/kg Hydrazine > 0.1 mg/kg ($T_{\text{start-up}}$)
Hot shutdown (M_{hot})	$15\text{cm}^3/\text{kg} < \text{H}_2 < 50\text{cm}^3/\text{kg}$ (T_{hot})
Cold shutdown (M_{cold})	$\text{O}_2 > 1\,\text{mg/kg}$ ($T_{\text{cold, w / oxy}}$) $\text{H}_2 > 50\,\text{N}\,\text{cm}^3\text{kg}$ ($T_{\text{cold, w/o oxy}}$)

mode, where

$$\begin{aligned}
\varphi_{\text{start-up}} &:= (\Diamond\Box M_{\text{start-up}} \implies \Diamond\Box T_{\text{start-up}})\,,\\
\varphi_{\text{hot}} &:= (\Diamond\Box M_{\text{hot}} \implies \Diamond\Box T_{\text{hot}})\,,\\
\varphi_{\text{cold}} &:= \big(\Diamond\Box M_{\text{cold}} \implies \big(\Diamond\Box T_{\text{cold, w/ oxy}} \vee \Diamond\Box T_{\text{cold, w/o oxy}}\big)\big)\,.
\end{aligned}$$

2.2 Mode-Target Formulas and Games

The preceding examples illustrate the scenarios that we want to capture with a suitable LTL fragment. All of the control problems we just described share the following properties that define our setting:

(P1) There are modes and corresponding targets.

(P2) If the system enters a mode, it should reach one of the targets associated with that mode and remain there.

(P3) If the mode changes, there is no obligation to reach any of the targets of the previous mode anymore.

We also make the following observation regarding the dynamics of the modes:

(P4) There is at most one mode active at any given time.

With these properties in mind, we now formally define mode-target formulas and games. For a game to be a mode-target game, its winning condition must be given by a mode-

target formula and the corresponding game graph should have a specific structure capturing **(P1)**-**(P4)**.

Let T and M be finite sets of atomic propositions: $T = \cup_i T_i$ and $M = \{M_1, M_2, \ \dots M_m\}$, where $T_i = \{T_{i,1}, T_{i,2}, \dots, T_{i,t_i}\}$. Here, the M_i, $T_{i,j}$ represent the mode i, and j^{th} target of mode i respectively. We start with a game graph G labeled with modes and targets, i.e., $G = (V, E, M \cup T, L)$ where $L : V \to 2^{M \cup T}$. The winning condition for player 0 is given by a mode-target formula.

Definition 2.1 (Mode-Target Formula)**.** An LTL formula is a *mode-target formula* if it has the form

$$\varphi := \bigwedge_{i=1}^{m} \left(\Diamond\Box M_i \implies \bigvee_{j=1}^{t_i} \Diamond\Box T_{i,j} \right). \tag{1.4}$$

We can interpret φ as: *if the system eventually settles in M_i, then it should eventually settle in one of the modes in T_i.* This formula captures **(P2)** because it guarantees that the system will reach one of the target regions in T_i if the system stays in mode M_i from a certain time onwards. As we explained previously, the left-hand side of the implication in (1.4) ensures that if the mode changes, the system does not have to reach or stay in any of the corresponding targets of the previous mode, as asserted by **(P3)**. It is true that φ can also be satisfied by switching between modes infinitely often. However, as it is the case in the ACC, A/F ratio, and pressurized water reactor examples, the modes can be partially if not fully determined by an external signal that the controller cannot change. In these cases, by construction, the controller will make progress towards the target of the current mode since it cannot predict if the system will remain in the current mode or switch to a different mode. Also note that for the ACC and A/F ratio control examples each T_i is simply a singleton, since there is only one target region that can be reached for all modes. This is not the case, however, for the pressurized water reactor control example.

To address **(P4)** we make the following assumption on the modes:
(A) Modes are mutually exclusive, i.e., $M_i \in L(v) \implies M_j \notin L(v)$, $\forall j \neq i, \forall v \in V$.

Definition 2.2 (Mode-Target Games)**.** We call LTL games with winning condition given

by a mode-target formula and a labeling function L that satisfies **(A)**, *mode-target games*.

Note that, a mode-target game is a Streett game [Str81] with additional structure imposed by the assumption **(A)** on the labeling function.

3 Solving Mode-Target Games

3.1 Decomposition of the Winning Set

We start by introducing a few notions that are critical to understand the solution of MT games described in this section.

Let $S_1 \subseteq \Sigma^*$ and $S_2 \subseteq \Sigma^* \cup \Sigma^\omega$. We define the concatenation of these sets as

$$S_1 S_2 := \{\sigma \in \Sigma^* \cup \Sigma^\omega | \sigma = \sigma_1 \sigma_2,\, \sigma_1 \in S_1,\, \sigma_2 \in S_2\}.$$

A *property* Φ is a subset of Σ^ω. The set of suffixes of a property Φ is denoted by $\mathrm{Post}(\Phi)$, i.e., $\mathrm{Post}(\Phi) := \{\sigma' \in \Sigma^\omega | \sigma\sigma' \in \Phi, \text{ for some } \sigma \in \Sigma^*\}$. A property Φ is an *absolute liveness property* iff $\Sigma^*\Phi \subseteq \Phi$. We call φ an *absolute liveness formula* if $W(\varphi)$ is an absolute liveness property. A formula φ is an absolute liveness formula iff $\varphi \equiv \Diamond\varphi$ (see [Sis94]). It follows that any formula of the form $\Diamond\phi$, for some ϕ is an absolute liveness formula.

We now introduce a class of games that includes both GR(1) games and MT games. The definition of this class of games distills the properties that are essential for a simple and transparent derivation of its solution.

Definition 3.1. An LTL game (G, φ) is said to be *simple* if the winning condition defined by φ can be written as:

$$\varphi = \Box \bigwedge_{i \in I} \varphi_i, \qquad \varphi_i = \Diamond p_i \vee \psi_i, \tag{1.5}$$

where p_i is a positional formula and ψ_i is an absolute liveness formula that satisfies:

$$W_G(\psi_i) \subseteq W(\varphi). \tag{1.6}$$

Lemma 3.2. *Every GR(1) game is a simple game.*

Proof. Given a GR(1) formula φ, the following holds:

$$\varphi \equiv \bigvee_{i_1 \in I_1} \Diamond\Box\neg a_i \vee \bigwedge_{i_2 \in I_2} \Box\Diamond g_{i_2} \overset{(1)}{\equiv} \Box \bigwedge_{i_2 \in I_2} \left(\left(\bigvee_{i_1 \in I_1} \Diamond\Box\neg a_{i_1} \right) \vee \Diamond g_{i_2} \right),$$

where $\overset{(1)}{\equiv}$ follows from very similar arguments to those in the proof of Lemma 1.4 in Appendix A. Note that $\vee_{i_1 \in I_1} \Diamond\Box\neg a_{i_1}$ implies φ, i.e., $W\left(\vee_{i_1 \in I_1} \Diamond\Box\neg a_{i_1}\right) \subseteq W(\varphi)$. Therefore, $\varphi \equiv \Box \wedge_{i_2 \in I_2} \left(\Diamond g_{i_2} \vee \psi_{i_2}\right)$, where $\psi_{i_2} := \vee_{i_1 \in I_1} \Diamond\Box\neg a_{i_1}$, for all $i_2 \in I_2$, which completes the proof of the lemma. □

Lemma 3.3. *Every mode-target game is simple.*

Proof.

$$\begin{aligned} \varphi &= \bigwedge_{i=1}^{m} \left(\Diamond\Box M_i \implies \vee_{j=1}^{t_i} \Diamond\Box T_{i,j} \right) \overset{(1)}{\equiv} \bigwedge_{i=1}^{m} \left(\Box\Diamond\neg M_i \vee \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \right) \\ &\overset{(2)}{\equiv} \Box \bigwedge_{i=1}^{m} \left(\Diamond\neg M_i \vee \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \right), \end{aligned}$$

where $\overset{(1)}{\equiv}$ is due to Lemma 1.5 in Appendix A, while $\overset{(2)}{\equiv}$ follows from Lemma 1.4 in Appendix A and $\Box\varphi_1 \wedge \Box\varphi_2 \equiv \Box(\varphi_1 \wedge \varphi_2)$.

The last formula has the form given in the statement of the simple games, where p_i is $\neg M_i$ and ψ_i is $\vee_{j=1}^{t_i} \Diamond\Box \left(M_i \wedge T_{i,j}\right)$. Then, we are only left with showing that $W_G(\psi_i) \subseteq W_G(\varphi)$.

Recall that in MT games for all $v \in V$, if $M_i \in L(v)$ then $M_j \notin L(v)$ for all $j \neq i$. It follows that for any $r \in V^\omega$ we have: $L(r) \models \Diamond\Box M_i \implies L(r) \models \Diamond\Box\neg M_j$, for all $j \neq i$.

Moreover, note that $W(\Diamond\Box\neg M_j) \subseteq W(\Box\Diamond\neg M_j)$. Therefore, the following holds:

$$\begin{aligned} & W_G\left(\bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})\right) \\ & \subseteq W\left(\Box \bigwedge_{\ell\in I_{\setminus i}} \Diamond\neg M_\ell\right) \text{ where } I_{\setminus i} = \{1,2,\ldots,m\} \setminus \{i\} \\ & \subseteq W\left(\Box\left(\bigwedge_{\ell\in I_{\setminus i}} \Diamond\neg M_\ell \vee \bigvee_{j=1}^{t_\ell} \Diamond\Box(M_\ell \wedge T_{\ell,j})\right)\right). \end{aligned} \tag{1.7}$$

Also note that

$$\begin{aligned} & W_G\left(\bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})\right) \subseteq W_G\left(\Box\Diamond\neg M_i \vee \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})\right) \\ & = W_G\left(\Box\left(\Diamond\neg M_i \vee \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})\right)\right), \end{aligned} \tag{1.8}$$

where the last equality is due to Lemma 1.4 in Appendix A.

By combining the inclusions (1.8) and (1.7) we get

$$W_G\left(\bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})\right) \subseteq W_G\left(\Box\bigwedge_{i=1}^{m}\left(\Diamond\neg M_i \vee \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})\right)\right),$$

which completes the proof of the lemma. □

The winning condition for simple games can be written as a conjunction of formulas φ_i preceded by $\Box$ where each φ_i can be decomposed as a disjunction between a reachability formula and a formula ψ satisfying (1.6). We now show that it is easy to modify algorithms that synthesize winning strategies for reachability games to obtain an algorithm for a conjunction of reachability formulas preceded by $\Box$. The approach in this algorithm remains valid even when we disjoin these reachability formulas with absolute liveness formulas ψ_i's, in virtue of (1.6). The inclusion given in (1.6) ensures that a play in (G, ψ_i) that is winning for player 0, is also winning in (G, φ). Therefore, one can adopt a compositional approach to the solution of simple games. A small modification to an

algorithm that computes $[\![\varphi_i]\!]$ leads to an algorithm computing $[\![\Box \bigwedge_{i\in I} \varphi_i]\!]$. The next result makes these ideas precise.

Theorem 3.4. *The winning set for player* 0 *in a simple game* (G, φ) *is given by*

$$[\![\varphi]\!] = \nu Z \bigcap_{i\in I} [\![\psi_i \vee \Diamond(p_i \wedge \circ Z)]\!]\,. \tag{1.9}$$

Proof. See Section 2 in Appendix A. □

The proof of the first part of Theorem 3.4, follows the existing methods for constructing winning strategies for Generalized Büchi games [DJW97], in which the winning condition is given by

$$\bigwedge_{i\in I} \Box\Diamond B_i \equiv \Box \bigwedge_{i\in I} \Diamond B_i, \tag{1.10}$$

for some subset of states $B_i \subseteq V$. The winning condition we are interested in, given in (1.5), is slightly different from the one given in (1.10) due to the additional ψ_i term. However, inclusion (1.6) ensures that any play that is winning for (G, ψ_i) is also winning for (G, φ). Hence, by simply computing $\nu Z \bigcap_{i\in I} [\![\psi_i \vee \Diamond(p_i \wedge \circ Z)]\!]$ we can obtain a winning strategy for player 0 in a simple game. Moreover, this strategy can be seen as the composition of the strategies for games with the simpler winning condition $\psi_i \vee \Diamond(p_i \wedge \circ Z)$.

Theorem 3.4 shows how the structure of simple games makes it possible to combine the sets $[\![\psi_i \vee \Diamond p_i]\!]$ as in (1.9) to compute the final winning set. In particular, we conclude that modularity observed in the solution of GR(1) games is not due to the structure of GR(1) formulas but rather to the structure of simple game formulas. Hence, this structure can be leveraged beyond GR(1) games as we did for MT games. Note how Theorem 3.4 describes the solution to both GR(1) and MT games. For later reference we instantiate (1.9) for MT games:

$$[\![\varphi]\!] = \nu Z \bigcap_{i=1}^{m} \left[\!\!\left[\bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \vee \Diamond(\neg M_i \wedge \circ Z) \right]\!\!\right] \tag{1.11}$$

and explain in the next section how to compute the winning sets

$$\left[\!\!\left[\bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \vee \Diamond(\neg M_i \wedge \circ Z)\right]\!\!\right] \tag{1.12}$$

so as to make use of (1.11). Note that if we instead instantiate (1.9) for the GR(1) formula (1.1) we obtain

$$\nu Z \bigcap_{i_2 \in I_2} \left[\!\!\left[\bigvee_{i_1 \in I_1} \Diamond\Box\neg a_{i_1} \vee \Diamond(\neg g_{i_2} \wedge \circ Z)\right]\!\!\right]. \tag{1.13}$$

The structures of the fixed-point expressions given in (1.13) and (1.11) are very much alike, but not the same. While in GR(1) games for each $i_2 \in I_2$, i.e., for each guarantee, the same persistency property is required to be satisfied ($\vee_{i_i \in I_1} \Diamond\Box\neg a_{i_1}$), in the case of MT games, the persistency part of the specification depends on the current mode, i.e., the index i, as in (1.11) ($\vee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})$).

3.2 Computation of the Winning Set

In [KPP05], Kesten, Piterman and Pnueli presented a μ-calculus formula which characterizes $[\![\vee_{i \in I} \Diamond\Box p_i \vee \Diamond q]\!]$, where p_i and q are positional formulas. This μ-calculus formula yields the following fixed-point expression:

$$\mu Y \bigcup_{i \in I} (\nu X(\mathrm{Pre}(X) \cap [\![p_i]\!]) \cup [\![q]\!] \cup \mathrm{Pre}(Y)). \tag{1.14}$$

Using (1.14) it is easy to see that the winning set (1.11) is given by the following fixed-point:

$$[\![\varphi]\!] = \nu Z \left(\bigcap_{i=1}^{m} \mu Y \bigcup_{j=1}^{t_i} (\nu X(\mathrm{Pre}(X) \cap [\![M_i \wedge T_{i,j}]\!]) \cup ([\![\neg M_i]\!] \cap \mathrm{Pre}(Z)) \cup \mathrm{Pre}(Y)) \right). \tag{1.15}$$

We refer to the algorithm defined by the iterative computation of the preceding fixed-point as the **MT** algorithm. In the worst case, the **MT** Algorithm can take $O(\sum_i t_i n^2)$ iterations, where t_i is the number of targets dedicated to mode i and n is the number of vertices in the game graph G. We summarize this in the following theorem.

Theorem 3.5. *Mode-target games can be solved by the symbolic algorithm* ***MT*** *requiring* $O(\sum_{i=1}^{m} t_i\, n^2)$ Pre *computations.*

Proof. In [BCJ97] Browne et al. show that a fixed point expression with *alternation depth* k can be computed in $O(n^{\lfloor 1+k/2 \rfloor})$ iterations. Note that given a fixed-point expression the alternation depth is simply the number of alternating greatest and least fixed point operators.

The alternation depth of the fixed-point expression (1.15) is three. Moreover, the computation of the fixed-point involves sequentially evaluating t_i fixed-point expressions for each mode, which results in $O\left(\sum_{i=1}^{m} t_i n^2\right)$ Pre computations in the worst case. □

Theorem (3.5) only addresses the computation of the winning set for the controller. However, the fixed-point computation given in (1.15) is constructive in the sense that we can find a winning strategy by storing the intermediate sets that are computed during its evaluation. The precise construction and implementation of the winning strategy follows the same approach as in GR(1) games [BJP12]. However, for the sake of completeness we provide the details of the winning strategy synthesis in Appendix A, Section 3. Note that contrary to the winning strategy for GR(1) games, the winning strategy for MT games is memoryless since player 0 only needs to know what the current mode is.

4 Solving Mode-Target Games via GR(1) Games

In this section, we describe how to transform a given MT game into a GR(1) game, thereby obtaining another algorithm to solve MT games that is based on the existing synthesis algorithms for the GR(1) fragment. To simplify the notation in the next proposition we

introduce the atomic proposition $\bar{T}_{i,j}$ defined by:

$$\bar{T}_{i,j} = \begin{cases} T_{i,j}, & \text{if } j \leq t_i \\ \texttt{false} & \text{otherwise.} \end{cases}$$

Proposition 4.1. *Every MT game with game graph G is equivalent to the GR(1) game (G, φ), where*

$$\varphi = \left(\bigwedge_{j=1}^{\max_i t_i} \square\diamond \wedge_{i=1}^{m} (\neg M_i \vee \neg \bar{T}_{i,j}) \right) \Longrightarrow \left(\bigwedge_{i=1}^{m} \square\diamond \neg M_i \right), \tag{1.16}$$

Proof. See Appendix A, Section 4. □

The proof of (1.16) has two main steps. In the first step, we show that the MT game is equivalent to the GR(1) game (G, φ_1), where

$$\varphi_1 = \left(\bigvee_{i=1}^{m} \bigvee_{j=1}^{\max_i t_i} \diamond\square (M_i \wedge \bar{T}_{i,j}) \right) \vee \left(\bigwedge_{i=1}^{m} \square\diamond \neg M_i \right). \tag{1.17}$$

The equivalence of (G, φ) to the MT game relies on assumption **(A)**. Also note that the formula in (1.17) is satisfied either when the system settles down in a mode and in one of the corresponding targets or when it toggles between the modes indefinitely, which matches the initial motivation of the MT fragment. Since the formula given in (1.17) is a GR(1) formula with $\sum_i t_i$ assumptions and m guarantees, this part of the proof already leads to a synthesis algorithm for MT games. In the second part of the proof we show[3] how to construct a GR(1) game with fewer assumptions that is equivalent to (G, φ_1) and for which the statement of Proposition 4.1 holds. Again assumption **(A)** lies at the heart of the proof. This assumption restricts the modes to be mutually exclusive and therefore enforces additional structure on MT games, which lets us simplify the formula in (1.17).

The formula given in (1.16) is a GR(1) formula with $\max_i t_i$ assumptions, and m guarantees. Notice that this formula has at most the same number of assumptions as φ_1 since

[3]This part of the proof is based on a comment we received from an anonymous reviewer of the preliminary version of our results presented in [BVT16].

$m \max_i t_i \leq m \sum_i t_i$. Due to Proposition 4.1 we can now simply apply the algorithm given in [BJP12] to the game graph G with the winning condition (1.16) to solve the MT game. This algorithm is based on the computation of the following fixed-point:

$$\nu Z \left(\bigcap_{i=1}^{m} \mu Y \left(\bigcup_{j=1}^{\max_i t_i} \nu X \left(\mathrm{Pre}(X) \cap \cup_{\ell=1}^{m} [\![M_\ell \wedge \bar{T}_{\ell,j}]\!] \right) \cup \mathrm{Pre}(Y) \cup ([\![\neg M_i]\!] \cap \mathrm{Pre}(Z)) \right) \right). \tag{1.18}$$

We refer to the algorithm defined by the iterative computation of the preceding fixed-point as the **GR(1)-Emb** algorithm for *GR(1) Embedding*. In the worst case, the **GR(1)-Emb** algorithm can take $O(m \max_i t_i n^2)$ iterations, where m is the number of modes in the MT formula, t_i is the number of targets dedicated to mode i, and n is the number of vertices in the game graph G. This follows from the fact that solving GR(1) games according to the fixed-point computation in [BJP12] takes $O(n_a n_g n^2)$ symbolic steps where n_g is the number of guarantees and n_a is the number of assumptions. Then, the bound $O(m \max_i t_i n^2)$ follows from the fact that $n_a = \max t_i$ and $n_g = m$ as in (1.16).

The following result summarizes the discussion in this section.

Theorem 4.2. *Mode-target games can be solved by the symbolic algorithm* ***GR(1)-Emb*** *requiring* $O(m \max_i t_i n^2)$ Pre *computations.*

Proof. Similar to the proof of Theorem 3.5, this result follows from the fact that the given fixed-point expression is of alternation depth three. Moreover, in each iteration of the algorithm we sequentially compute $m \max_i t_i$ fixed-point expressions which results in $O(m \max_i t_i n^2)$ Pre computations in the worst case. □

Comparing the complexities of the **MT** and the **GR(1)-Emb** algorithms as given in Theorem 4.2 and Theorem 3.5, we get

$$O\left(\sum_{i=1}^{m} t_i \, n^2 \right) \leq O\left(m \max_i t_i \, n^2 \right). \tag{1.19}$$

Although the **GR(1)-Emb** and the **MT** algorithms compute the same winning set, the

MT algorithm has better worst case complexity than the **GR(1)-Emb** algorithm. Moreover, the equality in (1.19) holds iff $t_\ell = \max_i t_i$ for all $\ell \in \{1, 2, \dots m\}$, i.e., if the number of targets associated with each mode is equal. In this special case, assuming the number of targets for each mode to be t, the fixed-point that needs to be computed for the **GR(1)-Emb** algorithm is

$$\nu Z \left(\bigcap_{i=1}^{m} \mu Y \bigcup_{j=1}^{t} (\nu X (\mathrm{Pre}(X) \cap \cup_{\ell=1}^{m} [\![M_\ell \wedge T_{\ell,j}]\!]) \cup ([\![\neg M_i]\!] \cap \mathrm{Pre}(Z)) \cup \mathrm{Pre}(Y)) \right), \quad (1.20)$$

while for the **MT** algorithm the fixed-point computation given in (1.15) becomes

$$\nu Z \left(\bigcap_{i=1}^{m} \mu Y \bigcup_{j=1}^{t} (\nu X (\mathrm{Pre}(X) \cap [\![M_i \wedge T_{i,j}]\!]) \cup ([\![\neg M_i]\!] \cap \mathrm{Pre}(Z)) \cup \mathrm{Pre}(Y)) \right). \quad (1.21)$$

As can be seen from (1.20) and (1.21), even in this special case where the two different approaches have the same worst-case complexity, the computations performed by **GR(1)-Emb** and **MT** differ. While the fixed-point expression (1.21) has $\cup_{\ell=1}^{m} [\![M_\ell \wedge T_{\ell,j}]\!]$ for every mode index i, the fixed-point in (1.21) replaces this set with $[\![M_i \wedge T_{i,j}]\!]$ for each i. Since $[\![M_i \wedge T_i]\!] \subseteq \cup_{\ell=1}^{m} [\![M_\ell \wedge T_{\ell,j}]\!]$ for all i and j, due to the monotonicity of the given fixed-point operator, the **MT** algorithm performs no worse than the **GR(1)-Emb** in terms of number of iterations. Moreover, for a given i and j, in order to compute the fixed-point in the variable X, the algorithm **MT** only requires the storage of the set $[\![M_i \wedge T_{i,j}]\!]$ instead of $\cup_{\ell=1}^{m} [\![M_\ell \wedge T_{\ell,j}]\!]$. This suggests that the algorithm **MT** might also have better space complexity. To investigate these differences in practice, we provide in the next section an experimental comparison of two implementations for each of the two algorithms presented in this chapter: **GR(1)-Emb**, and **MT**.

5 Experimental Comparison

The winning set and a corresponding winning strategy can be computed by iterating the operators on the right hand sides of (1.15) and (1.18) until a fixed-point is reached. We can improve the time efficiency of a direct implementation of this iteration by using two

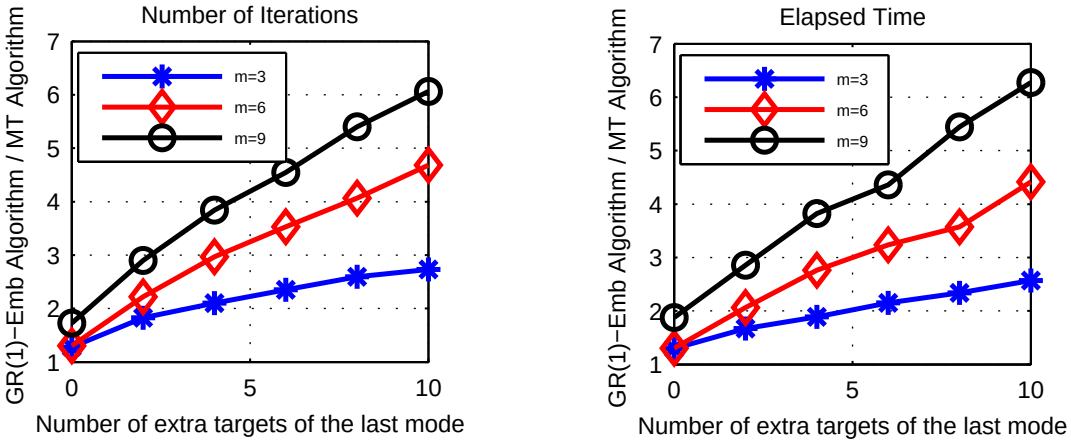
important ideas from the literature. In [EL86], the authors make the following observation: if one wants to compute the largest (smallest) fixed-point of an operator and one already knows a set that contains (is contained in) this fixed-point, then the largest (smallest) fixed-point computation can be started from this value instead of V ($\varnothing$). By using this idea, the authors showed that the complexity of their computation does not depend on the number of fixed-point operators but rather the number of such fixed-point alternations, i.e., alternation depth. Taking the same idea a step further, in [BCJ97], by exploiting monotonicity, the authors point state that one can use the intermediate values of the sets to initialize the fixed-point computations. This method also leads to improved time efficiency, but now with the cost of the requirement to store the value of intermediate sets that are not necessary for the computation of the final fixed-point. However, as mentioned in Section 3.2, the construction of the winning strategy depends upon these intermediate values. Therefore, in our experiments we use the method described in [BCJ97], since the extra memory allocation is partly unavoidable when the desired end product is a winning strategy, and not just the winning set.

In this section, we discuss the experimental time complexity of the algorithms **GR(1)-Emb** and **MT**. We present three sets of experiments. The first two are designed to compare the performance of the two algorithms in different scenarios, while the last one demonstrates a concrete application of the MT fragment in the design of the ACC example described in Section 5.3.

5.1 Random Linear Time-Invariant Systems with Multiple Targets

We start with the simplest class of dynamical systems: linear time-invariant systems. We demonstrate how the performance of the two algorithms differs as the theoretical worst-case gap between the **GR(1)-Emb** algorithm and **MT** algorithm deepens. To this end, we consider a scenario where all modes but one have a single associated target. For this remaining mode, starting from a single target we gradually increase the number of associated targets in order to accentuate the difference between the two sides of the inequality in (1.19). We provide the descriptions of all mode and target sets in [BVT].

In Fig. 1.1, we summarize our findings for the case when we have three, six and nine modes. We plot in Fig. 1.1a the ratio between the number of iterations it takes for the **GR(1)-Emb** algorithm versus the **MT** algorithm to compute the winning set. In Fig. 1.1b we compare the two algorithms in the same fashion, but now in terms of the elapsed time. Each data point represents the average value we obtained after computing the winning set on 20 random linear time-invariant systems. All systems have the form $\dot{x} = Ax + Bu$, where the entries of the matrices A and B are randomly chosen from the set $[-1, 1]$. The state space and the input space are the sets $[-6, 6] \times [-6, 6]$, and $[-4, 4]$, respectively. As can be seen from both figures, **MT** outperforms **GR(1)-Emb**, and the performance difference becomes progressively more prominent as the number of extra targets and modes increase.



(a) The ratio of number of iterations of **GR(1)-Emb** to **MT**.

(b) The ratio of elapsed time until convergence of **GR(1)-Emb** to **MT**.

Figure 1.1: Comparison of the algorithms **GR(1)-Emb** and **MT** when there are multiple targets corresponding to one of the modes.

5.2 Unicycle Cleaning Robot

We consider a scenario where a unicycle robot cleans the rooms on a hotel floor. The robot has to reach one of the rooms that is not clean and stay there, until an external signal indicates that the current room has been cleaned. We now explain how we model this scenario as an MT game. Assume that there are two rooms, defined by the atomic propositions T_1 and T_2. Each mode-target pair corresponds to a different subset of rooms that need to be cleaned. Specifically, M_1, M_2, and M_3 indicate that only the first room,

only the second room, and both of the rooms need to be cleaned, respectively. Accordingly, the MT formula corresponding to this scenario is:

$$\wedge_{i=1}^{2}(\Diamond\Box M_i \implies \Diamond\Box T_i) \wedge (\Diamond\Box M_3 \implies (\Diamond\Box T_1 \vee \Diamond\Box T_2)).$$

Note that, if there are k rooms, the number of modes is $2^k - 1$.

We first construct the game graph corresponding to the dynamics of the cleaning robot. The differential equations:

$$\dot{x} = v\cos(\theta),\ \dot{y} = v\sin(\theta),\ \dot{\theta} = \omega,$$

offer a simplified model for a 3-wheel robot equipped with differential drive. The pair $(x, y) \in \mathbb{R}^2$ denotes the position of the robot, $\theta \in [-\pi, \pi[$ denotes its orientation, and $(v, \omega) \in \mathbb{R}^2$ are the control inputs, linear velocity v and angular velocity ω. For this example we restrict the position (the location of the rooms) to the set $[1, 7.5] \times [1, 7.5]$, input to the set $[0, 0.5]\times[-0.5, 0.5]$ and create an abstraction[4] using the PESSOA [MDT10] tool. This abstraction is stored as an Ordered Binary Decision Diagram [Ake78] (OBDD) and constitutes the game graph describing the dynamics of the cleaning robot. It has 21141 vertices or states and 6 inputs that are available at each state.

We now describe the dynamics of the modes. When the robot is in room i that has not yet been cleaned, the mode can change to the mode where the room i does not need to be cleaned anymore. The nondeterminism in this change models an external signal indicating whether the cleaning in the current room has been completed or not. When all the rooms are cleaned, a nondeterministic mode transition can occur to any other mode to restart the process. In Fig. 1.2, we illustrate the dynamics of the modes when there are two rooms. As can be seen, there is a nondeterministic transition from M_3 to M_2 as the robot enters the room 1 (T_1). Similarly, if the system is in M_1 (only room 1 is not clean), when the robot reaches room 1, the system can take a nondeterministic transition to any of the other modes, i.e., we restart the cleaning process once all the rooms are cleaned.

[4]The parameters used for the abstraction were $\eta = 0.25$, $\mu = 0.5$, and $\tau = 0.5$. An explanation of the meaning

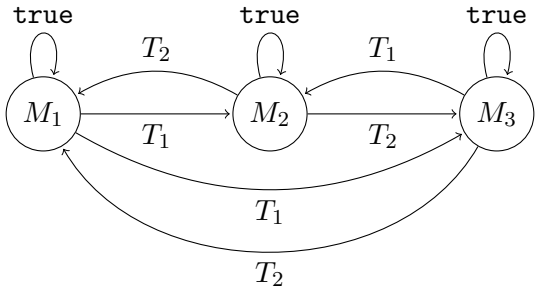


Figure 1.2: Mode dynamics for the cleaning robot, when there are two rooms (M_1: room 1 not clean, M_2: room 2 not clean, M_3 both rooms are not clean).

To obtain the final game graph describing the dynamics of both the modes and the cleaning robot, we compose the game graph describing the modes and the game graph describing the dynamics of the robot. Note that, the second player in this game arises due to the conservative nature of the abstraction, as explained in [Tab09], and the nondeterminism in the mode changes, both of which can be modeled as an adversarial disturbance.

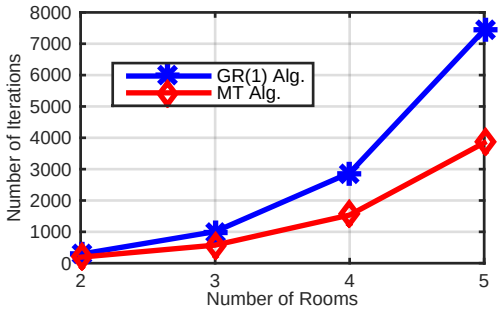
We compare the performance of the **GR(1)-Emb**, and the **MT** algorithms as we increase the number of rooms from 2 to 5. The rooms are boxes of various dimensions, whose descriptions can be found at [BVT].

Fig. 1.3 summarizes our findings. Fig. 1.3a and Fig. 1.3b illustrate that, as the number of rooms increases, the gap between the performance of the algorithm **MT** and the algorithm **GR(1)-Emb** increases significantly both in terms of number of iterations of the fixed-point algorithms as well as the computation time. Note that, when there are k rooms we have, $m \max_i t_i = (2^k - 1)k$, and $\sum_i t_i = \sum_{j=1}^{k} \binom{k}{j} k$. Therefore, the widening of the performance gap is expected, since as the number of rooms increases, so does the difference between the worst case time complexities of **GR(1)-Emb** and **MT**.

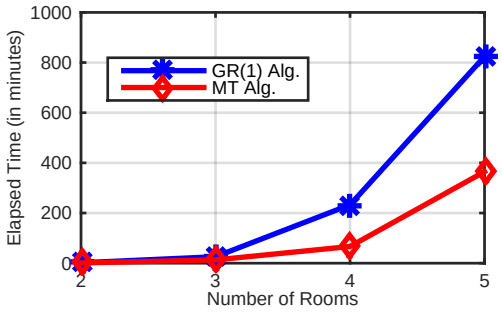
5.3 Adaptive Cruise Control (ACC)

The last example demonstrates the usefulness of the MT fragment by applying it on the ACC design problem that we detailed in Section 2. We model the dynamics of the ACC equipped vehicle by a hybrid system with two discrete states which specify whether there

of these parameters is given in [MDT10].



(a) The number of iterations until convergence for the algorithms **GR(1)-Emb** and **MT**.



(b) Elapsed time until convergence for the algorithms **GR(1)-Emb** and **MT**.

Figure 1.3: Comparison of the algorithms **GR(1)-Emb** and **MT** on the cleaning robot case study for varying number of rooms.

is a lead car or not. The continuous states describe the evolution of the velocity of the ACC equipped vehicle (v) as well as the velocity of the lead car (v_L), and the distance to the lead car (h) whenever there is one. The net action of braking and engine torque applied to the wheels (F_w) is viewed as the control input and is assumed to satisfy the bound $-0.3mg \leq F_w \leq 0.2mg$, where m is the mass of the ACC equipped vehicle and g is the gravitational constant. Via PESSOA, we constructed a discrete abstraction of this hybrid system, which together with the dynamics of the modes constitutes the game graph of the MT game. The abstraction contains over 1.5 million states. We refer the reader to [NHB16] for the details of the construction of this abstraction and a complete description of the corresponding hybrid model. The winning condition of the game is the conjunction of the safety specification φ_{safety} with the MT formula $\varphi_{\text{speed}} \wedge \varphi_{\text{timegap}}$, where

$$\begin{aligned} \varphi_{\text{safety}} &\equiv \Box[\tau \geq \tau_{\text{safe}}], \\ \varphi_{\text{speed}} &\equiv (\Diamond\Box M_{\text{speed}} \implies \Diamond\Box[v_{\text{des}} - \epsilon_v, v_{\text{des}} + \epsilon_v]), \\ \varphi_{\text{timegap}} &\equiv (\Diamond\Box M_{\text{timegap}} \implies \Diamond\Box[\tau_{\text{des}} - \epsilon_\tau, \tau_{\text{des}} + \epsilon_\tau]). \end{aligned} \tag{1.22}$$

The values of the parameters appearing in (1.22) are $\tau_{\text{safe}} = 1$ s, $v_{\text{des}} = 25$ m/s, $\epsilon_v = 1$, $\tau_{\text{des}} = 1.6$, and $\epsilon_\tau = 1$. Note that the additional safety formula, φ_{safety}, can be handled separately by first synthesizing a safety controller and then composing this controller with a controller synthesized solely for the MT formula, $\varphi_{\text{speed}} \wedge \varphi_{\text{timegap}}$.

In Figure 1.4, we present the winning set computed via the **MT** Algorithm. As can

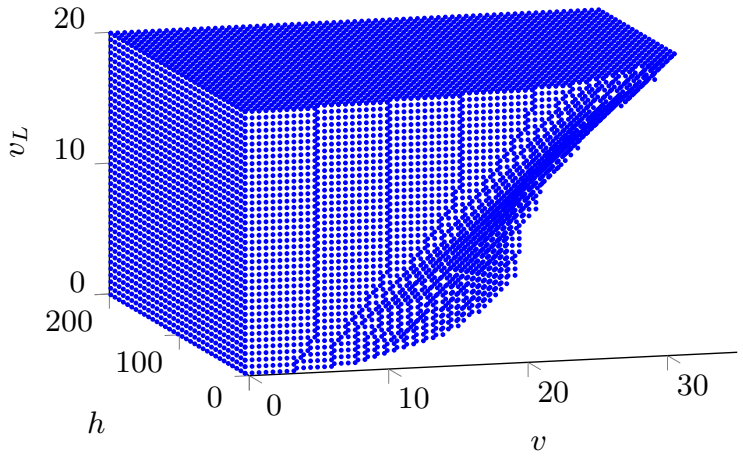


Figure 1.4: The winning set computed by the **MT** Algorithm.

be seen, the domain does not contain the points where h, the headway, is small and v, the velocity of the ACC vehicle, is high, since there is no sequence of control inputs to maintain a safe headway starting from these states. We simulated the MT controller on CARSIM, an industry standard car dynamics simulation package, for the following scenario: at time $t = 0$ s, a lead car is present driving below the desired speed $v_{\text{des}} = 25$ m/s of the ACC car, then leaves the lane at $t = 3$ s, allowing the ACC car to reach and attain its desired speed. At $t = 13$ s, a new lead car cuts in 30 m in front of the ACC car and starts decelerating. This means that the ACC car should slow down in order to increase the headway. Fig. 1.5 presents the behavior of the MT controller. Notably, all constraints, which are indicated by green lines, are satisfied throughout the simulations. For a detailed discussion on hardware implementation of the MT controller and further experimental results, we refer the reader to [NHB16].

The different experimental results suggests the following: **(1) MT** is consistently better than **GR(1)-Emb**. Even for the case when the theoretical worst case complexities of both algorithms are the same, **MT** outperforms **GR(1)-Emb**. However, the performance increase is not always considerable in this case; **(2)** as the gap between $\max_i t_i$ and $\sum_i t_i$ widens, so does the performance difference between **GR(1)-Emb** and **MT**, which is in accordance with the results in Section 3.2.

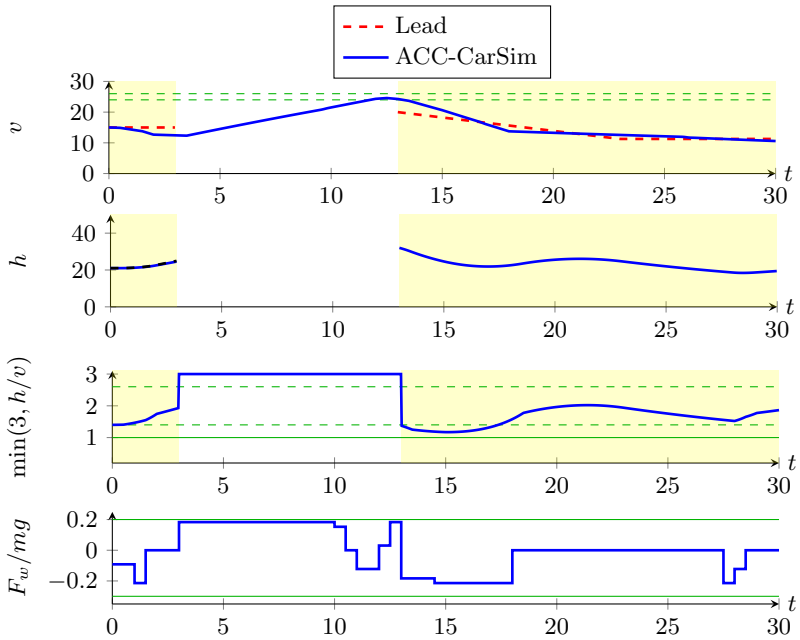


Figure 1.5: Simulation results in CarSim of the PESSOA controllers. The plots show, from top to bottom, velocities, headway, time headway, and applied control input. Grayed areas indicate that the system is in specification mode M_{timegap}. Dashed green lines indicate target sets, solid green indicate safety sets.

6 Conclusions

We introduced a new class of LTL games called mode-target games and argued that these games can be used to model a variety of control design problems encountered in practice. We provided two algorithms to solve MT games. The first algorithm is based on transforming MT games to simple games, a class of LTL games for which we provide a synthesis algorithm. This leads to an algorithm that solves MT games in a number of steps polynomial in the size of the game graph. We next provided a different algorithm, that relies on the fact that every MT game can be embedded into a GR(1) game. We also showed that the direct algorithm has better worst case complexity than the algorithm obtained via the GR(1) embedding. These observations were validated through multiple simulations. As future work, we plan on investigating whether additional structure arising in control problems can lead to further simplifications both in MT games as well as other LTL games.

CHAPTER 2

Data-Driven Stability Analysis of Black-box Switched Linear Systems

Today's complex cyber-physical systems are characterized by the interaction of a large number of heterogeneous components. Consequently, the models used to analyze these systems are equally complex and consist of heterogeneous sub-models relying on different assumptions and based on principles from different scientific disciplines. It is not uncommon to encounter a patchwork of differential equations, difference equations, hybrid automata, lookup tables, custom switching logic, low-level legacy code, etc. To further compound the difficulty in analyzing these systems, different components of a complex engineered system are typically designed by different suppliers. Although a high-level specification for these components may be known, detailed models are not available for intellectual property reasons. We are thus faced with a tremendous gap between the existing analysis techniques that rely on closed-form models and the models available in industry. It is, therefore, not surprising the emphasis that industry places on simulation since despite the complexity of models, it is always possible to simulate them. This raises the question of whether we can provide formal guarantees about certain properties of these complex systems based solely on the information obtained via their simulations. In this thesis, we focus on one of the most important of such properties in the context of control theory: stability.

More formally, we consider a dynamical system as in:

$$x_{k+1} = f(k, x_k), \tag{2.1}$$

where, $x_k \in X$ is the state and $k \in \mathbb{N}$ is the time index. In this thesis, we study switched

systems, but we believe that the presented results can be extended to more general classes of dynamical systems. We start with the following question to serve as a stepping stone: Given N pairs, (x_1, y_1), (x_2, y_2), ..., (x_N, y_N) belonging to the behavior of the system (2.1), (i.e., $y_k = f(k, x_k)$ for some k), what can we say about the stability of the system (2.1)? For the rest of the chapter, we use the term *black-box* to refer to models where we do not have access to the model, i.e., to f, yet we can indirectly learn information about f by observing pairs of points (x_k, y_k) as defined in (2.1).

A potential approach to this problem is to first identify the dynamics, i.e., the function f, and then apply existing techniques from the model-based stability analysis literature. However, unless f is a linear function, there are two main reasons behind our quest to directly work on system behaviors and bypass the identification phase: 1) Even when the function f is known, in general, stability analysis is a very difficult problem [BT99]; 2) Identification can potentially introduce approximation errors, and can be algorithmically hard as well. Again, this is the case for switched systems [Lau15]. A fortiori, the combination of these two steps in an efficient and robust way seems far from obvious.

In recent years, increasing number of researchers started addressing various verification and design problems in control of black-box systems [BFG16, BTD16, HM14, DMV13]. In particular, the initial idea behind this chapter was influenced by the recent efforts in [TPS08, KDS14], and [BL15] on using simulation traces to find Lyapunov functions for systems with known dynamics. In these works, the main idea is that a Lyapunov function candidate decreasing along several finite trajectories starting from different initial conditions should also decrease along every other trajectory. Then, once a Lyapunov function candidate is constructed, this intuition is put to test by verifying the candidate function either via off-the-shelf tools as in [TPS08] and [KDS14], or via sampling-based techniques as in [BL15]. This also relates to almost-Lyapunov functions introduced in [LLZ16], which presents a relaxed notion of stability proved via Lyapunov functions decreasing everywhere except on a small set. Note that, since we do not have access to the dynamics, these approaches cannot be directly applied to black-box systems. However, these ideas raise the following problem that we address in this chapter: By observing that

a candidate Lyapunov function decreases on a large number of observations, we empirically build a certain confidence that such candidate Lyapunov function is a bona-fide Lyapunov function. *Can we translate this confidence into a confidence in the stability of the underlying system?*

Note that, even in the case of a 2D linear system, the connection between these two beliefs is nontrivial. In fact, one can easily construct an example where a candidate Lyapunov function decreases everywhere on its levels sets, except for an arbitrarily small subset, yet, almost all trajectories diverge to infinity. For example, the system

$$x^+ = \begin{bmatrix} 0.14 & 0 \\ 0 & 1.35 \end{bmatrix} x,$$

admits a Lyapunov function candidate on the unit circle except on the two red areas shown in Fig. 2.1. Moreover, the size of this "violating set" can be made arbitrarily small by changing the magnitude of the unstable eigenvalue. Nevertheless, the only trajectories that do not diverge to infinity are those starting on the stable eigenspace, i.e., on the x axis, has zero measure.

In this work, we take the first steps to infer stability from observations of switched linear systems. In addition to the preceding example, there are other reasons to temper our expectations for proving stability from data: identifying and deciding stability of arbitrary switched linear systems is NP-hard [Jun09].

The stability of switched systems is closely related to the *joint spectral radius* (JSR) of the matrices modeling the dynamics in each mode. Deciding stability amounts to deciding whether the JSR is less than one [Jun09]. In this chapter, we present an algorithm to bound the JSR of a switched linear system from a finite number N of observations. This algorithm partly relies on tools from the random convex optimization literature (also known as chance-constrained optimization, see [Cal10, NS06, CG08]), and provides an upper bound on the JSR with a user-defined confidence level. As N increases, this bound gets tighter. Moreover, with a closed form expression, we characterize what is the exact trade-off between the tightness of this bound and the number of samples. In order to

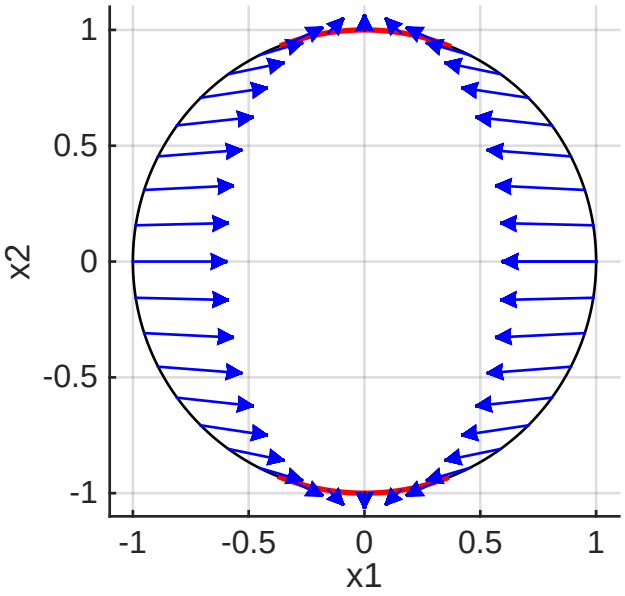


Figure 2.1: A simple dynamics and the level set of an "almost Lyapunov function". Even though this function decreases at almost all points in its level set, almost all trajectories diverge to infinity.

understand the quality of our upper bound, the algorithm also provides a deterministic lower bound. Finally, we provide an asymptotic guarantee on the gap between the upper and the lower bound, for large N.

The organization of this chapter is as follows: We start Section 1 by introducing the notations that will be used for the rest of this chapter. We then provide the necessary background in stability of switched systems and its relationship with JSR. In Section 2, we present a deterministic lower bound for the JSR. Section 3 presents the main contribution of this chapter of the thesis where we provide a probabilistic stability guarantee for a given switched system, based on finite observations. In Section 4, we experimentally demonstrate the performance of the presented techniques on randomly generated switched linear systems as well as on a concrete example from networked control domain.

1 Preliminaries

1.1 Notation

We consider the usual finite normed vector space $(\mathbb{R}^n, \ell_2)$, $n \in \mathbb{N}_{>0}$, with ℓ_2 the classical Euclidean norm. We denote the set of linear functions in $\mathbb{R}^n$ by $\mathcal{L}(\mathbb{R}^n)$, and the set of real symmetric matrices of size n by $\mathcal{S}^n$. A symmetric matrix P is called positive definite iff $x^T P x > 0$ for all $x \in \mathbb{R}^n \setminus \{0\}$ and it is called positive semi-definite iff $x^T P x \geq 0$ for all $x \in \mathbb{R}^n$. We denote the set of positive definite matrices is denoted by $\mathcal{S}^n_{++}$. We write $P \succ 0$ to state that P is positive definite, and $P \succeq 0$ to state that P is positive semi-definite. Given a set $X \subset \mathbb{R}^n$, and $r \in \mathbb{R}_{>0}$ we write $rX := \{x \in X : rx\}$ to denote the scaling of this set. We denote by $\mathbb{B}$ (respectively $\mathbb{S}$) the ball (respectively sphere) of unit radius centered at the origin. We denote the ellipsoid described by the matrix $P \in \mathcal{S}^n_{++}$ as E_P, i.e., $E_P := \{x \in \mathbb{R}^n : x^T P x = 1\}$. Finally, we denote the spherical projector on $\mathbb{S}$ by $\Pi_{\mathbb{S}} := x/\|x\|$.

For an ellipsoid centered at the origin, and for any of its subsets $\mathcal{A}$, the *sector* defined by $\mathcal{A}$ is the subset

$$\{t\mathcal{A},\ t \in [0,1]\} \subset\ \mathbb{R}^n.$$

We denote by $E_P^{\mathcal{A}}$ the sector induced by $\mathcal{A} \subset E_P$. In the particular case of the unit sphere, we instead write $\mathbb{S}^{\mathcal{A}}$. We can notice that $E_P^{E_P}$ is the volume in $\mathbb{R}^n$ defined by E_P: $E_P^{E_P} = \{x \in \mathbb{R}^n : x^T P x \leq 1\}$.

The spherical Borelian σ-algebra denoted by $\mathcal{B}_{\mathbb{S}}$ is defined by:

$$\mathcal{A} \in \mathcal{B}_{\mathbb{S}} \iff \mathbb{S}^{\mathcal{A}} \in \mathcal{B}_{\mathbb{R}^n}.$$

We provide $(\mathbb{S}, \mathcal{B}_{\mathbb{S}})$ with the classical unsigned and finite uniform spherical measure on $\mathbb{S}$, denoted by σ^{n-1}. It is associated to $\mathcal{B}_{\mathbb{S}}$, the spherical Borelian σ-algebra, and is derived from the Lebesgue measure λ.

$$\forall\ \mathcal{A} \in \mathcal{B}_{\mathbb{S}},\ \sigma(\mathcal{A}) = \frac{\lambda(\mathbb{S}^{\mathcal{A}})}{\lambda(\mathbb{B})}.$$

In other words, the spherical measure of a subset of the sphere is related to the Lebesgue measure of the sector of the unit ball it induces. Notice that $\sigma^{n-1}(\mathbb{S}) = 1$. Since $P \in \mathcal{S}^n_{++}$, it can be written in its Cholesky form

$$P = L^T L, \tag{2.2}$$

where L is an upper triangular matrix. Note that, L^{-1} maps the elements of $\mathbb{S}$ to E_P. Then, we define the measure on the ellipsoid σ_P on the σ-algebra $\mathcal{B}_{E_P} := L^{-1}\mathcal{B}_{\mathbb{S}}$, where $\forall\, \mathcal{A} \in \mathcal{B}_{E_P}$, $\sigma_P(\mathcal{A}) = \sigma^{n-1}(L\mathcal{A})$.

For $m \in \mathbb{N}_{>0}$, we denote by M the set $M = \{1, 2, \ldots, m\}$. Set M is provided with the classical σ-algebra associated to the finite sets: $\Sigma_M = \wp(M)$, where $\wp(M)$ is the set of subsets of M. We consider the uniform measure μ_M on (M, Σ_M).

We define $Z = \mathbb{S} \times M$ as the Cartesian product of the unit sphere and M. We denote the product σ-algebra $\mathcal{B}_{\mathbb{S}} \bigotimes \Sigma_M$ generated by $\mathcal{B}_{\mathbb{S}}$ and Σ_M: $\Sigma = \sigma(\pi_{\mathbb{S}}^{-1}(\mathcal{B}_{\mathbb{S}}), \pi_M^{-1}(\Sigma_M))$, where $\pi_{\mathbb{S}} : Z \to \mathbb{S}$ and $\pi_M : Z \to M$ are the standard projections. On this set, we define the product measure $\mu = \sigma^{n-1} \otimes \mu_M$. Note that, μ is a uniform measure on Z and $\mu(Z) = 1$.

1.2 Switched Linear Systems

A *switched dynamical system* is collection of dynamical systems with a switching mechanism specifying which dynamical system is active at each time instant. Such a dynamical system with a set of modes $\mathcal{M} = \{f_i, i \in M\}$ can be mathematically modeled by the discrete-time system:

$$x_{k+1} = f_{\tau(k)}(x_k), \tag{2.3}$$

where $x_k \in X \subseteq \mathbb{R}^n$, k is the time index, and $\tau(k)$ is the switching sequence $\tau : \mathbb{N} \to M$. Such multimodal systems appear frequently in practice. Some common scenarios include sudden parameter changes, subcomponent failures or changes in the environment. For example, as explained in [ZB98], the dynamics of the two-wheeled Hilare-type mobile robot changes depending on whether the wheels are rolling or sliding. Switched dynamical systems also emerge due to control techniques that include switching between different control

laws [ZB98, KDF03, Mor98b]. Such control techniques have applications in automotive industry, power converters, various mechanical systems as reported in [Mor98a, HZD04].

The particular realization of the switched system in (2.3), where all the subsystems are linear time-invariant is called a *switched linear system*. Given a set of modes $\mathcal{M} = \{A_i, i \in M\}$, such system is of the form:

$$x_{k+1} = f(k, x_k), \tag{2.4}$$

with $f(k, x_k) = A_{\tau(k)} x_k$ and switching sequence $\tau : \mathbb{N} \to M$. We next define homogeneity which is an important concept for some of the upcoming results in this chapter.

Definition 1.1. A function $g : \mathbb{R}^p \to \mathbb{R}^q$ is called *homogenous of degree* ℓ if it satisfies:

$$\forall\, \alpha > 0, \forall x \in \mathbb{R}^p,\ g(\alpha x) = \alpha^\ell g(x).$$

Property 1.1. Let $\xi(x, k, \tau)$ denote the state of the system (2.4) at time k starting from the initial condition x and with switching sequence τ. The dynamical system (2.4) is homogeneous: $\xi(\gamma x, k, \tau) = \gamma \xi(x, k, \tau)$.

It is well-known that the joint spectral radius of the set of matrices $\mathcal{M}$ closely relates to the stability of the system (2.4). Joint spectral radius is the maximum asymptotic growth rate of the norm of the state under the dynamics (2.4) over all possible initial conditions and sequences of matrices of $\mathcal{M}$.

Definition 1.2 (from [Jun09])**.** Given a finite set of matrices $\mathcal{M} \subset \mathbb{R}^{n \times n}$, its *joint spectral radius* (JSR) is given by

$$\rho(\mathcal{M}) = \lim_{k \to \infty} \max_{i_1, \ldots, i_k} \left\{ ||A_{i_1} \ldots A_{i_k}||^{1/k} : A_{i_j} \in \mathcal{M} \right\}.$$

Property 1.2 (Corollary 1.1, [Jun09]). Given a finite set of matrices $\mathcal{M}$, the corresponding switched dynamical system is stable if and only if $\rho(\mathcal{M}) < 1$.

Property 1.3 (Proposition 1.3, [Jun09]). Given a finite set of matrices $\mathcal{M}$, and any invertible matrix T,

$$\rho(\mathcal{M}) = \rho(T \mathcal{M} T^{-1}),$$

i.e., the JSR is invariant under similarity transformations (and is a fortiori a homogeneous function: $\forall \gamma > 0,\ \rho(\mathcal{M}/\gamma) = \mathcal{M}/\gamma$).

2 A Deterministic Lower Bound for the JSR

We start by computing a lower bound for ρ which is based on the following theorem from the switched linear systems literature.

Theorem 2.1. *[Jun09, Theorem 2.11] For any finite set of matrices such that $\rho(\mathcal{M}) < \frac{1}{\sqrt{n}}$, there exists a Common Quadratic Lyapunov Function (CQLF) for $\mathcal{M}$, that is, a $P \succ 0$ such that:*

$$\forall\ A \in \mathcal{M},\ A^T P A \preceq P.$$

CQLFs are useful because they can be computed (if they exist) with semidefinite programming (see [BV04]), and they constitute a stability guarantee for switched systems as we formalize next.

Theorem 2.2. *[Jun09, Prop. 2.8] Consider a finite set of matrices $\mathcal{M}$. If there exist a $\gamma \geq 0$ and $P \succ 0$ such that*

$$\forall\ A \in \mathcal{M}, A^T P A \preceq \gamma^2 P,$$

then $\rho(\mathcal{M}) \leq \gamma$.

Note that the smaller γ is, the tighter is the upper bound we get on $\rho(\mathcal{M})$. Therefore, we can consider, in particular, the optimal solution γ^* of the following optimization problem:

$$\begin{array}{ll} \min_{\gamma,P} & \gamma \\ \text{s.t.} & (Ax)^T P(Ax) \leq \gamma^2 x^T P x,\ \forall\ A \in \mathcal{M},\ \forall\, x \in \mathbb{R}^n, \\ & P \succ 0, \gamma \geq 0. \end{array} \tag{2.5}$$

Even though this upper bound is more difficult to obtain in a black-box setting where only a finite number of observations are available, in this section we leverage Theorem 2.1 in order to derive a straightforward lower bound.

The following theorem shows that the existence of a CQLF for the system in (2.4) can be checked by considering N pairs $(x_i, j_i) \in \mathbb{R}^n \times M$, where $i \in \{1, \dots N\}$. Recall that in our setting, we assume that we observe pairs of the form (x_k, x_{k+1}), but we do not observe the mode applied to the system during this time step.

Theorem 2.3. *For a given uniform sampling:*

$$\omega_N := \{(x_1, j_1), (x_2, j_2), \dots, (x_N, j_N)\} \subset \mathbb{R}^n \times M,$$

let $W_{\omega_N} = \{(x_1, y_1), \dots, (x_N, y_N)\}$ *be the corresponding available observations, which satisfy*

$$y_i = A_{j_i} x_i \ \forall (x_i, y_i) \in W_{\omega_N}.$$

Also let $\gamma^*(\omega_N)$ *be the optimal solution of the following optimization problem:*

$$\begin{aligned} min_P \quad & \gamma \\ s.t. \quad & (y_i)^T P (y_i) \leq \gamma^2 x_i^T P x_i, \ \forall (x_i, y_i) \in W_{\omega_N} \\ & P \succ 0, \gamma \geq 0. \end{aligned} \tag{2.6}$$

Then, we have:

$$\rho(\mathcal{M}) \geq \frac{\gamma^*(\omega_N)}{\sqrt{n}}.$$

Note that, (2.6) can be efficiently solved by semidefinite programming and bisection on the variable γ (see [BV04]).

Proof. Let $\epsilon > 0$. By definition of γ^*, there exists no matrix $P \in \mathcal{S}^n_{++}$ such that:

$$(Ax)^T P (Ax) \leq (\gamma^*(\omega_N) - \epsilon)^2 x^T P x, \quad \forall x \in \mathbb{R}^n, \forall\, A \in \mathcal{M}.$$

Since joint spectral radius is a homogenous function as described in Property 1.3, this means that there exists no CQLF for the scaled set of matrices $\frac{\mathcal{M}}{(\gamma^*(\omega_N) - \epsilon)}$. Then, using Theorem 2.1, we conclude:

$$\frac{\rho(\mathcal{M})}{\gamma^*(\omega_N)} \geq \frac{1}{\sqrt{n}}.$$

□

3 A Probabilistic Stability Guarantee

In this section, we show how to compute an upper bound on ρ, with a user-defined confidence $\beta \in (0,1)$. We do this by constructing a CQLF which is valid with probability at least β. Note that, the existence of a CQLF implies $\rho \leq 1$ due to Theorem 2.2.

Even though the solution of the optimization problem in (2.5) provides a CQLF, solving this problem as stated is not possible since it involves infinitely many constraints and we do not have access to the set of matrices $\mathcal{M}$. Nevertheless, we show that the solution of the optimization problem (2.6) allows us to not only compute a lower bound, but also a (probabilistic) upper bound on the JSR.

We now analyze the relationship between the solutions of the optimization problem (2.5) with infinitely many constraints and the following optimization problem with finitely many constraints:

$$\begin{aligned} \min_P \quad & \lambda_{\max}(P) \\ \text{s.t.} \quad & (A_j x)^T P (A_j x) \leq \left((1+\eta)\gamma^*(\omega_N)\right)^2 x^T P x, \qquad \forall (x,j) \in \omega_N \subset Z, \\ & P \succeq I, \end{aligned} \tag{2.7}$$

where $Z := \mathbb{S} \times M$, $\eta > 0$, and $\gamma^*(\omega_N)$ is the optimal solution to the optimization problem (2.6). Recall that ω_N is an N-uniform random sampling of the set Z. Note that, instead of the set $\mathbb{R}^n$ we sample on the unit sphere $\mathbb{S}$. This is due to homogeneity of the dynamics stated in Property 1.1, since it implies that it is sufficient to show the decrease of a CQLF on an arbitrary set enclosing the origin, e.g., $\mathbb{S}$.

For the rest of the discussion, we refer to the optimization problem (2.7) by $\mathrm{Opt}(\omega_N)$. We denote its optimal solution by $P(\omega_N)$. We drop the explicit dependence of P on ω_N when it is clear from the context. There are a few points that are worth noting about (2.7). Firstly, due to Property 1.1, we can replace the constraint $P \succ 0$ with the constraint $P \succeq I$. Moreover, for reasons that will become clear later in the discussion, we chose the

objective function as $\lambda_{\max}(P)$, instead of solving a feasibility problem in P. Lastly, the additional η factor is introduced to ensure strict feasibility of (2.7), which will be helpful in the following discussion. However, since our results are valid for arbitrarily small values of η, introduction of this factor will not hamper the practical accuracy of our technique.

The curious question whether the optimal solution of the sampled problem $\text{Opt}(\omega_N)$ is a feasible solution to (2.5) has been widely studied in the literature [Cal10]. It turns out that under certain technical assumptions, one can bound the proportion of the constraints of the original problem (2.5) that are violated by the optimal solution of (2.7), with some probability which is a function of the sample size N.

In the following theorem, we adapt a classical result from random convex optimization literature to our problem.

Theorem 3.1 (adapted from Theorem 3.3, [Cal10]). *Let d be the dimension of Opt(ω_N) and $N \geq d+1$. Consider the optimization problem Opt(ω_N) given in (2.7), where ω_N is a uniform random sampling of the set Z. Then, for all $\epsilon \in (0,1)$ the following holds:*

$$\mu^N \left\{\omega_N \in Z^N : \mu(V(\omega_N)) \leq \epsilon\right\} \geq 1 - \sum_{j=0}^{d} \binom{N}{j} \epsilon^j (1-\epsilon)^{N-j}, \tag{2.8}$$

where μ^N denotes the product probability measure on Z^N, and $V(\omega_N)$ is defined by

$$V(\omega_N) = \{z \in Z : (A_j z)^T P(\omega_N)(A_j z) > ((1+\eta)\gamma^*)^2 z^T P(\omega_N) z\},$$

i.e., it is the set of constraints of (2.5) that are violated by the optimal solution of Opt(ω_N).

Theorem 3.1 states that the optimal solution of the sampled problem $\text{Opt}(\omega_N)$ violates no more than an ϵ fraction of the constraints in the original optimization problem (2.5) with probability β, where β goes to 1 as N goes to infinity. This means that, the ellipsoid computed by $\text{Opt}(\omega_N)$ is "almost invariant" except a set of measure ϵ. The rest of this section builds on this, and it has three intermediate results leading us to our main theorem. In Proposition 3.2, we start by showing how to upper bound the measure of points that are violating the constraints on the set $\mathbb{S}$ in terms of the measure of the points that are

violating the constraints in the product space Z.

Proposition 3.2. *Let $\epsilon \in (0,1)$ and $\gamma \in \mathbb{R}_{>0}$. Consider the set of matrices $\mathcal{M}$ and $A_j \in \mathcal{M}$ satisfying:*

$$(A_j x)^T P (A_j x) \leq \gamma^2 x^T x, \quad \forall\, x \in Z \setminus V, \forall\, j \in M, \tag{2.9}$$

for some $V \subset Z$ where $\mu(V) \leq \epsilon$, then the following holds:

$$(A_j x)^T P (A_j x) \leq \gamma^2 x^T x, \quad \forall\, x \in \mathbb{S} \setminus \mathbb{S}', \forall\, j \in M, \tag{2.10}$$

for some $\mathbb{S}' \subset \mathbb{S}$ where $\sigma^{n-1}(\mathbb{S}') \leq m\epsilon$.

Proof. Let $V_{\mathbb{S}} = \pi_{\mathbb{S}}(V)$ and $V_M = \pi_M(V)$. We know that Σ_M is the disjoint union of its 2^m elements $\{\mathcal{M}_i, i \in \{1, 2, \dots 2^m\}\}$. Then V can be written as the disjoint union $V = \sqcup_{1 \leq i \leq 2^m} (\mathcal{S}_i, \mathcal{M}_i)$ where $\mathcal{S}_i \in \Sigma_{\mathbb{S}}$. We notice that $V_{\mathbb{S}} = \sqcup_{1 \leq i \leq 2^m} \mathcal{S}_i$, and

$$\sigma^{n-1}(V_{\mathbb{S}}) = \sum_{1 \leq i \leq 2^m} \sigma^{n-1}(\mathcal{S}_i).$$

We have

$$\begin{aligned} \mu(V) &= \mu(\sqcup_{1 \leq i \leq 2^m} (\mathcal{S}_i, \mathcal{M}_i)) = \sum_{1 \leq i \leq 2^m} \mu(\mathcal{S}_i, \mathcal{M}_i) \\ &= \sum_{1 \leq i \leq 2^m} \sigma^{n-1} \otimes \mu_M(\mathcal{S}_i, \mathcal{M}_i) \\ &= \sum_{1 \leq i \leq 2^m} \sigma^{n-1}(\mathcal{S}_i) \mu_M(\mathcal{M}_i). \end{aligned}$$

Note that we have $\min_{j \in M} \mu_M(\{j\}) = \frac{1}{m}$. Then since $\forall\, i$, $\mu_M(\mathcal{M}_i) \geq \min_{j \in M} \mu_M(\{j\}) = \frac{1}{m}$, we get:

$$\sigma^{n-1}(V_{\mathbb{S}}) \leq \frac{\mu(V)}{\frac{1}{m}} \leq m\epsilon. \tag{2.11}$$

This means that

$$(A_j x)^T P (A_j x) \leq \gamma^2 x^T P x, \ \forall\, x \in \mathbb{S} \setminus V_{\mathbb{S}}, \forall\, m \in M, \tag{2.12}$$

where $\sigma^{n-1}(V_{\mathbb{S}}) \leq m\epsilon$. $\square$

Remark 3.3. If the modes are not sampled uniformly random, then Proposition 3.2 and the subsequent results still hold by simply replacing the factor $\frac{1}{m}$ in (2.11) with $\min_{j\in M} \mu_M(\{j\})$, i.e., the probability of sampling the mode that is least likely to be active.

Having established an upper bound on the measure of the violating set on $\mathbb{S}$, by exploiting convexity-preserving structure of the dynamics, we next show in Lemma 3.4 how one can compute an upper bound on the JSR when this "almost invariant" set is the unit sphere, i.e., $\mathbb{S}$.

Lemma 3.4. *Let $\epsilon \in (0, \frac{1}{2})$ and $\gamma \in \mathbb{R}_{>0}$. Consider the set of matrices $\mathcal{M}$ and $A \in \mathcal{M}$ satisfying:*

$$(A_j x)^T (A_j x) \leq \gamma^2 x^T x, \quad \forall\, x \in \mathbb{S} \setminus \mathbb{S}', \forall\, j \in M, \tag{2.13}$$

where $\mathbb{S}' \subset \mathbb{S}$ and $\sigma^{n-1}(\mathbb{S}') \leq \epsilon$, then we have:

$$\rho(\mathcal{M}) \leq \frac{\gamma}{\alpha(\epsilon)}$$

where $\alpha(\epsilon)$ is given in (B.4).

Proof. Note that, (2.13) implies that: $A_j(\mathbb{S}\setminus\mathbb{S}') \subset \gamma\mathbb{B}$. Note that, the dynamics of switched linear systems is convexity-preserving, meaning that for any set of points $X \subset \mathbb{R}^n$ we have:

$$A_j \text{convhull } (X) \subset \text{convhull } (A_j(X)), \forall\, j \in M.$$

Therefore, we get:

$$A_j \text{convhull } (\mathbb{S} \setminus \mathbb{S}') \subset \text{convhull } (A_j(\mathbb{S} \setminus \mathbb{S}')) \subset \gamma\mathbb{B}, \forall\, j \in M.$$

Let $\alpha(\epsilon) := \inf_{X\in\mathcal{X}_\epsilon} \sup\{r : r\mathbb{B} \subset \text{convhull } (\mathbb{S} \setminus X)\}$, where $\mathcal{X}_\epsilon = \{X \subset \mathbb{S} : \sigma^{n-1}(X) \leq \epsilon\}$.

Then, by Lemma 3.1 in Appendix B, we have:

$$A_j(\alpha(\epsilon)\mathbb{B}) \subset A_j(\text{convhull } (\mathbb{S} \setminus \mathbb{S}')) \subset \gamma\mathbb{B}, \quad \forall\, j \in M,$$

where $\alpha(\epsilon)$ is given as in (B.4). Therefore, we get:

$$\alpha(\epsilon)A_j(\mathbb{B}) \subset \gamma\mathbb{B}, \quad \forall\, j \in M,$$

which implies that $\rho(\mathcal{M}) \leq \frac{\gamma}{\alpha(\epsilon)}$. □

We now know how to compute an upper bound on the JSR when the "almost invariant" ellipsoid is $\mathbb{S}$. Thanks to invariance of ρ under similarity transformations as stated in Property 1.3, if this is not the case, we can simply perform a change of coordinates mapping this ellipsoid to $\mathbb{S}$ and compute the JSR in the new coordinates system instead. To do this, in the next theorem, we bound the measure of violating constraints on $\mathbb{S}$ after the change of coordinates, in terms of the measure of the violated constraints on $\mathbb{S} \times M$ in the original coordinates.

Proposition 3.5. *Let $\gamma \in \mathbb{R}_{>0}$. Consider a set of matrices $A \in \mathcal{M}$, and a matrix $P \succ 0$ satisfying:*

$$(A_j x)^T P (A_j x) \leq \gamma^2 x^T P x, \ \forall\, (x, j) \in \mathbb{S} \setminus \mathbb{S}', \forall\, j \in M \tag{2.14}$$

for some $\mathbb{S}' \subset \mathbb{S}$ where $\sigma^{n-1}(\mathbb{S}') \leq \epsilon$. Then, by defining L as in (2.2) and $\bar{A}_j = L^{-1} A_j L$, the following also holds:

$$(\bar{A}_j x)^T (\bar{A}_j x) \leq \gamma^2 x^T x, \ \forall\, x \in \mathbb{S} \setminus \mathbb{S}', \forall\, j \in M,$$

for some $\mathbb{S}' \subset \mathbb{S}$ such that: $\sigma(\mathbb{S}') \leq \epsilon\kappa(P)$, where

$$\kappa(P) = \sqrt{\frac{\lambda_{\max}(P)^n}{\det(P)}}.$$

Proof. We perform the change of coordinates defined by $L^{-1} \in \mathcal{L}(\mathbb{R}^n)$ which maps $\mathbb{S}$ to

E_P, defined as in (2.2). We can then rewrite (2.14) in this new coordinates system as in:

$$(\bar{A}_j x)^T (\bar{A}_j x) \leq \gamma^2 x^T x, \ \forall\, x \in E_P \setminus L^{-1}(V_{\mathbb{S}}), \forall\, m \in M. \tag{2.15}$$

Due to the homogeneity of the dynamics described in Property 1.1, this implies:

$$(\bar{A}_j x)^T (\bar{A}_j x) \leq \gamma^2 x^T x, \ \forall\, x \in \mathbb{S} \setminus \Pi_{\mathbb{S}}(L^{-1}(V_{\mathbb{S}})), \forall\, m \in M. \tag{2.16}$$

We now show how to relate $\sigma^{n-1}(V_{\mathbb{S}})$ to $\sigma^{n-1}(\Pi_{\mathbb{S}}(L^{-1}(V_{\mathbb{S}})))$. Consider $\mathbb{S}^{V_{\mathbb{S}}}$, the sector of $\mathbb{B}$ defined by $V_{\mathbb{S}}$. We denote $C := L^{-1}(\mathbb{S}^{V_{\mathbb{S}}})$ and $V' := \Pi_{\mathbb{S}}(L^{-1}(V_{\mathbb{S}}))$. We have $\Pi_{\mathbb{S}}(C) = V'$ and $\mathbb{S}^{V'} \subset \frac{1}{\lambda_{\min}(L^{-1})} C$. This leads to:

$$\sigma^{n-1}(V') = \lambda(\mathbb{S}^{V'}) \leq \lambda\left(\frac{1}{\lambda_{\min}(L^{-1})} C\right).$$

Then, the following holds:

$$\begin{aligned}
\sigma^{n-1}(V') &\leq \frac{1}{\lambda_{\min}(L^{-1})^n} \lambda(C) \\
&\leq \frac{1}{\lambda_{\min}(L^{-1})^n} \lambda(L^{-1}(\mathbb{S}^{V_{\mathbb{S}}})) \\
&= \frac{|\det(L^{-1})|}{\lambda_{\min}(L^{-1})^n} \lambda(\mathbb{S}^{V_{\mathbb{S}}}), \qquad (2.17) \\
&= \sqrt{\frac{\lambda_{\max}(P)^n}{\det(P)}}\, \sigma^{n-1}(V_{\mathbb{S}}) \qquad (2.18)
\end{aligned}$$

where (2.17) follows from the fact that

$$\lambda(Q(X)) = |\det(Q)| \lambda(X),$$

for any set $X \subset \mathbb{R}^n$ and $Q \in \mathcal{L}(\mathbb{R}^n)$ (see e.g. [Rud87]). Putting together (2.16) and (2.18) we get the statement of the theorem where $\mathbb{S}' = \Pi_{\mathbb{S}}(L^{-1}(V_{\mathbb{S}}))$. □

We are now ready to prove our main theorem by putting together all the above pieces. For a given level of confidence β, we prove that the upper bound $\gamma^*(\omega_N)$, which is valid solely

on finitely many observations, is in fact a true upper bound, at the price of increasing it by the factor $\frac{1}{\delta(\beta,\omega_N)}$. Moreover, as expected, this factor gets smaller as we increase N and decrease β.

Theorem 3.6. *Consider an n-dimensional switched linear system as in* (2.4) *and a uniform random sampling* $\omega_N \subset Z$*, where* $N \geq \frac{n(n+1)}{2}+1$*. Let* $\gamma^*(\omega_N)$ *be the optimal solution to* (2.7)*. Then, for any given* $\beta \in (0,1)$ *and* $\eta > 0$*, we can compute* $\delta(\beta,\omega_N)$*, such that with probability at least* β *we have:*

$$\rho \leq \frac{\gamma^*(\omega_N)(1+\eta)}{\delta(\beta,\omega_N)},$$

where $\lim_{N\to\infty}\delta(\beta,\omega_N) = 1$ *with probability* 1*.*

Proof. By definition of $\gamma^*(\omega_N)$ we have:

$$(A_j x)^T P(A_j x) \leq (\gamma^*(1+\eta))^2 x^T P x, \quad \forall\,(x,j) \in \omega_N$$

for some $P \succ 0$. Then, by rewriting Theorem 3.1 we also have:

$$\mu^N\left\{\omega_N \in Z^N : \mu(V(\omega_N)) \leq \epsilon\right\} \geq 1 - I(1-\epsilon; N-d, d+1), \tag{2.19}$$

where $I(\ell; a, b)$ is the regularized incomplete beta function. Let

$$\epsilon(\beta, N) = 1 - I^{-1}(1-\beta; N-d, d+1).$$

Then, by Proposition 3.1 with probability at least β the following holds:

$$(A_j x)^T P(A_j x) \leq (\gamma^*(1+\eta))^2 x^T P x, \quad \forall (x,j) \in Z \setminus V.$$

By Proposition 3.2 and Proposition 3.5, this implies that with probability at least β the following also holds:

$$(\bar{A}_j x)^T (\bar{A}_j x) \leq \gamma^2 x^T x,\ \forall\, x \in \mathbb{S} \setminus \mathbb{S}', \forall\, j \in M,$$

for some $\mathbb{S}'$ where $\sigma^{n-1}(\mathbb{S}') \leq m\epsilon\kappa(P)$. Then, applying Lemma 3.4, we can compute

$$\delta(\beta, \omega_N) = \alpha(\epsilon'(\beta, \omega_N)),$$

where

$$\epsilon'(\beta, \omega_N) = \frac{1}{2} m\kappa(P(\omega_N))\epsilon(\beta, N) \tag{2.20}$$

such that with probability at least β we have:

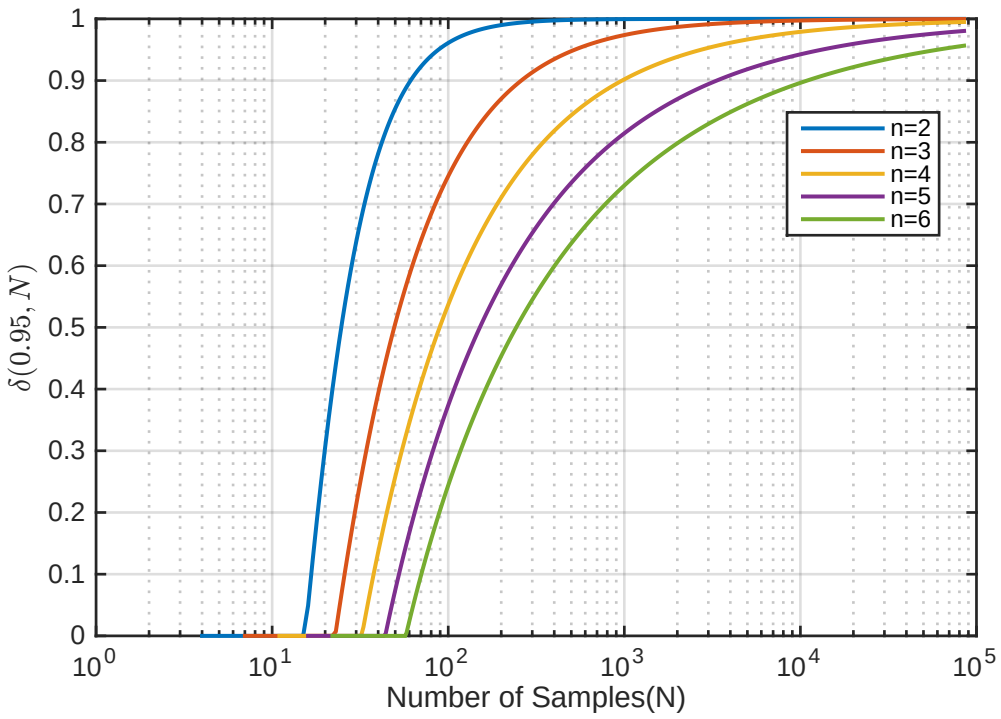
$$\bar{A}_j\mathbb{B} \subset \frac{\gamma^*(\omega_N)(1+\eta)}{\delta(\beta, \omega_N)}\mathbb{B}, \, \forall \, j \in M.$$

As stated in Property 1.3, since the JSR is invariant under similarity transformations, this means that with probability at least β:

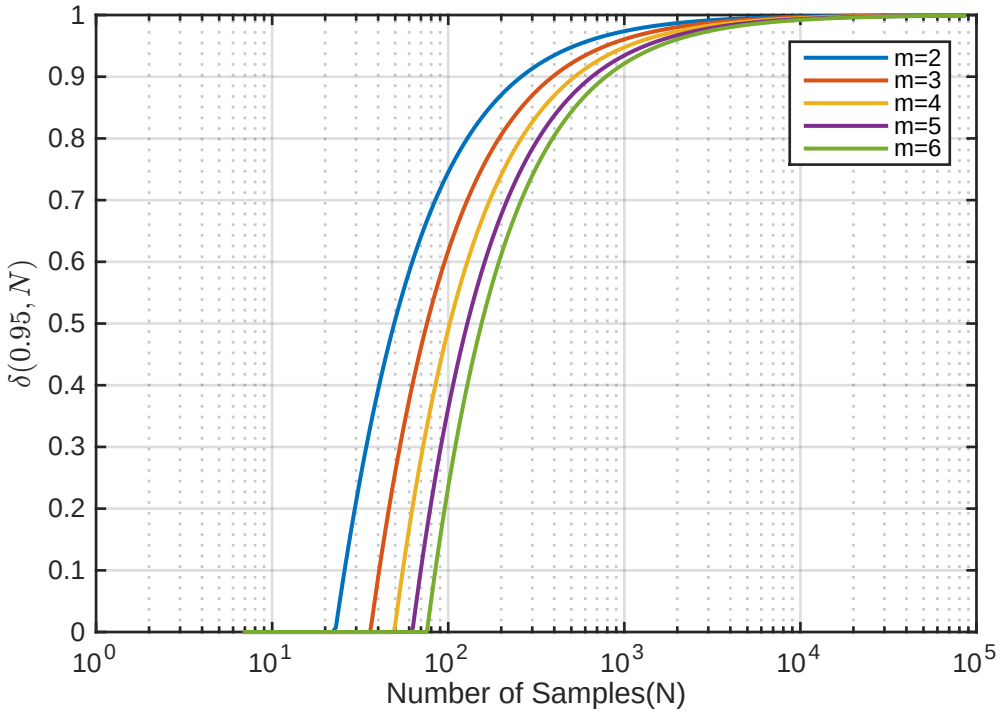
$$\rho \leq \frac{\gamma^*(\omega_N)(1+\eta)}{\delta(\beta, \omega_N)},$$

which completes the proof of the first part of the theorem. Note that, the ratio $\frac{1}{2}$ introduced in the expression of ϵ' is due to the homogeneity of the system described in Property 1.1, which implies that $x \in V_{\mathbb{S}} \iff -x \in V_{\mathbb{S}}$. We refer the interested reader to Appendix B, Section 4 for the second part of this proof, namely showing that $\lim_{N\to\infty} \delta(\beta, \omega_N) = 1$ with probability 1. □

In Fig. 2.2a and Fig. 2.2b, we examine the "best-case" convergence of $\delta(\beta, \omega_N)$, with respect to N when we fix $\beta = 0.95$. This corresponds to the scenario where $P = I$, i.e., the level sets of the computed CQLF are hyperspheres. Note that, for this case $\kappa(P(\omega_N)) = 1$, and hence δ is only a function of N and β when we fix n and m. Fig. 2.2a illustrates the evolution of $\delta(0.95, N)$ for varying n and $m = 2$. As can be seen, the convergence of δ slows down when we increase the state dimension n. This is intuitive because it means that for a given N, as we increase the state dimension, the price we have to pay in order to obtain an upper bound on the JSR using N observations is higher. We illustrate in Fig. 2.2b that the convergence of δ also becomes slower as we increase m for $n = 3$. Comparing Fig. 2.2a and Fig.2.2b, we see that the decrease in the convergence



(a) The convergence of $\delta(0.95, N)$ for $m = 2$, with increasing n.



(b) The convergence of $\delta(0.95, N)$ for $n = 3$, with increasing m.

speed due to the increase in number of modes is much less significant compared to that of due to the increase in the dimension.

4 Experimental Results

In this section, we provide an experimental evaluation of the bounds we provided in Theorem 2.3 and Theorem 3.6 on various discrete time switched linear systems. We also provide a concrete example from the networked control systems domain to illustrate the applicability of our technique. For all experiments, we set the confidence level to $\beta = 0.95$.

4.1 2-D Example

We first illustrate our technique on a two dimensional switched linear system with 4 modes, where $\mathcal{M}$ is given by:

$$\mathcal{M} = \left\{ \begin{bmatrix} -0.25 & -0.35 \\ 0.345 & 0.009 \end{bmatrix}, \begin{bmatrix} -0.25 & -0.35 \\ 0.345 & 0.001 \end{bmatrix} \begin{bmatrix} 0.702 & 0.43 \\ 0.477 & 0.011 \end{bmatrix}, \begin{bmatrix} 0.7016 & 0.43 \\ 0.4767 & 0.0106 \end{bmatrix} \right\}.$$

We compute a lower and an upper bound on the JSR for $N := 8 + 25k$, $k \in \{0, \ldots, 19\}$, according to Theorem 2.3 and Theorem 3.6, respectively. In Fig. 2.3, we illustrate the average performance of our technique over 10 different samplings. Fig. 2.3a shows the evolution of $\delta(\beta, \omega_N)$ as N increases. Note that, δ converges to 1 as expected. We plot the upper bound and lower bound for the JSR of the system in Fig. 2.3b. To demonstrate the performance of our technique, we also provide an estimate on the JSR computed by the JSR toolbox [VHJ14] with a precision of 0.0001, which turns out to be 0.9556. Note that, the plot for the upper bound starts from $N = 58$. This is due to the fact for $N = 8$, and $N = 33$, $\delta(\beta, \omega_N) = 0$, hence it is not possible to compute a nontrivial upper bound for these small values of N. As can be seen, the upper bound approaches a close vicinity of the real JSR with approximately 300 samples and after $N = 350$ samples, we can decide that the system is stable with probability 0.95. In addition, the gap between the upper and lower bound converges to a multiplicative factor of $\frac{1}{\sqrt{n}}$ as expected. This gap could

be improved by considering a more general class of common Lyapunov functions, such as those that can be described by sum-of-squares polynomials [PP02]. We leave this for future work.

4.2 4-D Example

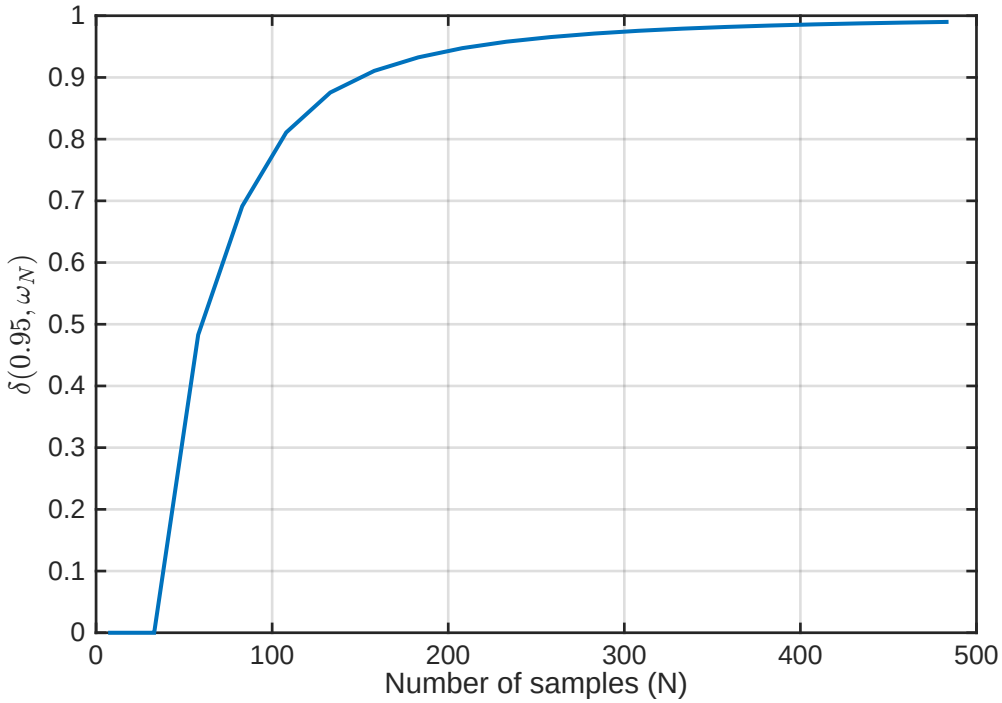
We now consider a four dimensional switched linear system with 6 modes, where $\mathcal{M}$ is defined by:

$$\mathcal{M} = \left\{ \begin{bmatrix} 0.05 & 0.24 & -0.21 & -0.31 \\ 0.1 & 0.26 & 0.25 & -0.31 \\ 0.07 & 0.23 & 0.02 & 0.12 \\ -0.29 & -0.21 & -0.18 & 0.15 \end{bmatrix}, \begin{bmatrix} 0.14 & -0.06 & -0.01 & 0.09 \\ -0.03 & 0.35 & 0.22 & 0.28 \\ 0.01 & 0.06 & 0.33 & 0.02 \\ 0.15 & 0.13 & 0 & 0.04 \end{bmatrix}, \begin{bmatrix} -0.15 & -0.01 & 0.35 & -0.46 \\ -0.31 & 0.09 & -0.42 & 0.06 \\ -0.01 & -0.32 & -0.05 & 0.16 \\ -0.34 & 0.05 & 0.08 & 0.17 \end{bmatrix} \right\}$$
$$\cup \left\{ \begin{bmatrix} 0.13 & 0.24 & -0.38 & -0.09 \\ 0.09 & -0.02 & -0.25 & 0.15 \\ -0.1 & -0.05 & 0.27 & -0.16 \\ 0.19 & 0.11 & 0.02 & -0.19 \end{bmatrix} \begin{bmatrix} -0.59 & 0.46 & 0.2 & 0.16 \\ 0.07 & -0.64 & 0.12 & 0.28 \\ 0.3 & 0.22 & -0.51 & -0.31 \\ 0.24 & 0.13 & -0.2 & 0.67 \end{bmatrix}, \begin{bmatrix} -0.57 & -0.01 & 0.03 & -0.2 \\ -0.14 & 0.03 & 0.02 & 0.49 \\ 0.13 & 0.06 & -0.38 & 0.05 \\ -0.03 & 0.47 & 0.12 & -0.32 \end{bmatrix} \right\}.$$

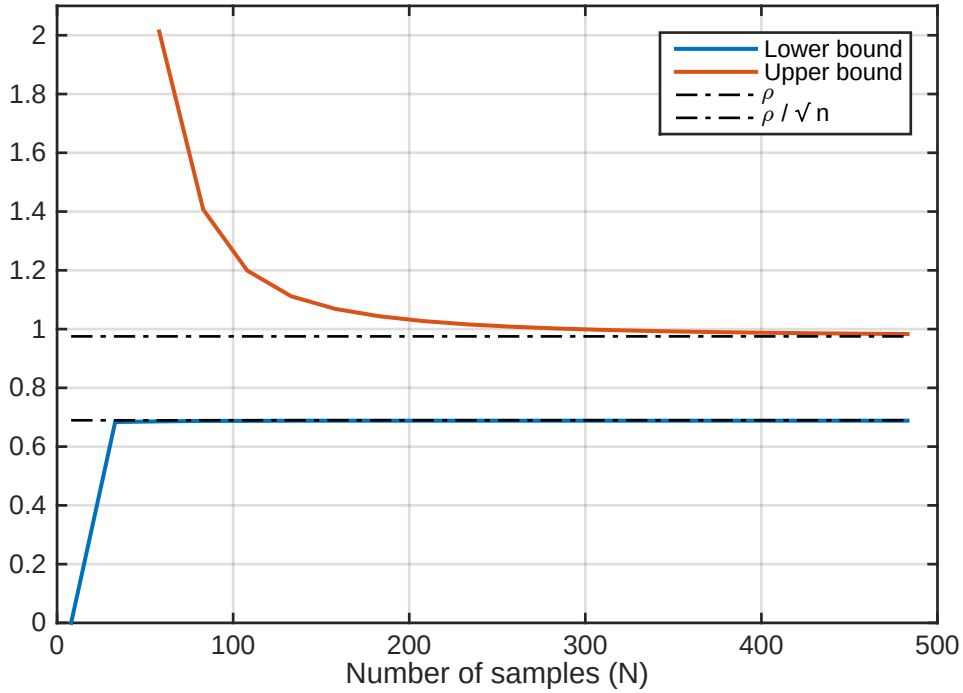
We again compute the lower and and the upper bound on the JSR for $N := 22 + 200k$, $k \in \{0, \ldots, 24\}$ given by Theorem 2.3 and Theorem 3.6, respectively. Fig. 2.4 illustrates the results. As expected, we can observe from Fig. 2.4a that even though the upper bound does converge to the real JSR (0.845), the convergence is much slower compared to the previous two dimensional example. Only after $N = 4750$, we can decide that the system is stable with probability $\beta = 0.95$. The slower convergence is also reflected in the evolution of δ as can be seen in Fig. 2.4b.

4.3 Average Behavior over Random Systems

We now apply our technique to randomly generated switched linear systems and investigate how its performance compares to the existing algorithms for switched systems with known dynamics, i.e., white-box systems. We perform three sets of experiments for different n, and m, and summarize our results in Fig. 2.5. For each set of experiments, we plot the average ratio of the upper bound given by Theorem 3.6 (black-box) to those computed by the JSR toolbox (white-box) [VHJ14], over 10 randomly generated switched systems.

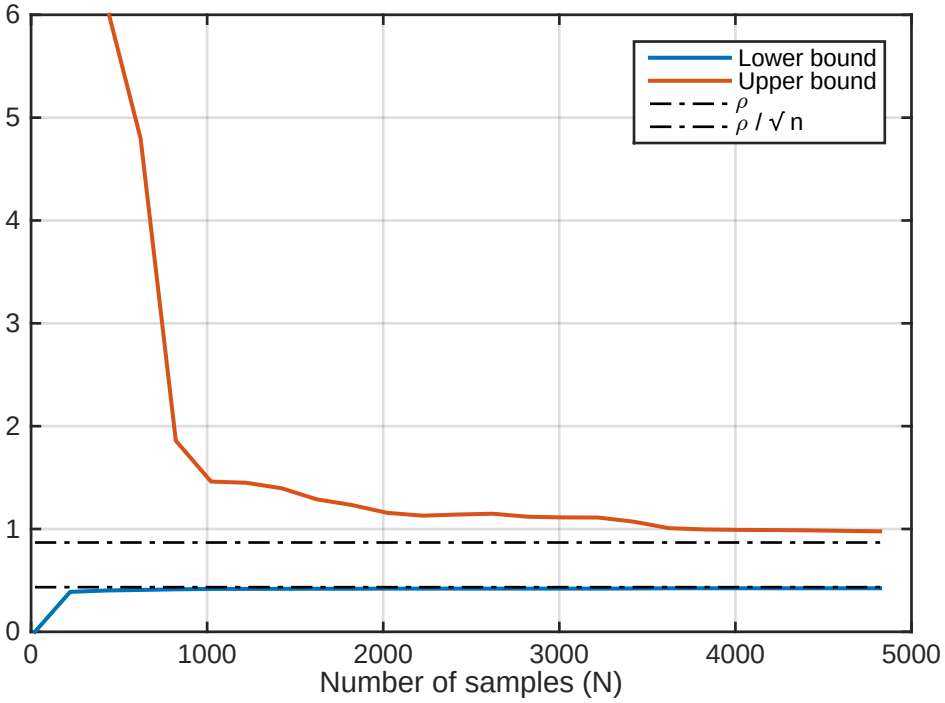


(a) Evolution of $\delta(\beta, \omega_N)$ with increasing N, for $\beta = 0.95$.

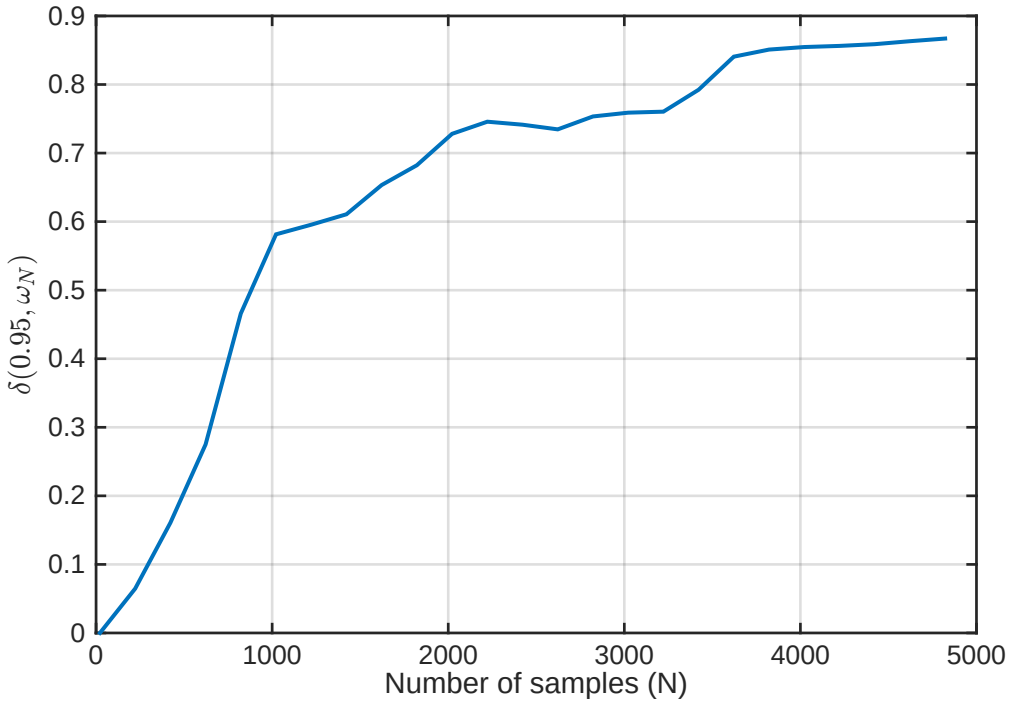


(b) Evolution of the upper and lower bounds on the JSR with increasing N, for $\beta = 0.95$.

Figure 2.3: The experimental evaluation of a randomly generated switched system with $n = 2$, $m = 4$.



(a) Evolution of the upper and lower bounds on the JSR with increasing N, for $\beta = 0.95$.



(b) Evolution of $\delta(\beta, \omega_N)$ with increasing N, for $\beta = 0.95$.

Figure 2.4: The upper and lower bounds on the JSR for a switched system with $n = 4$, $m = 6$.

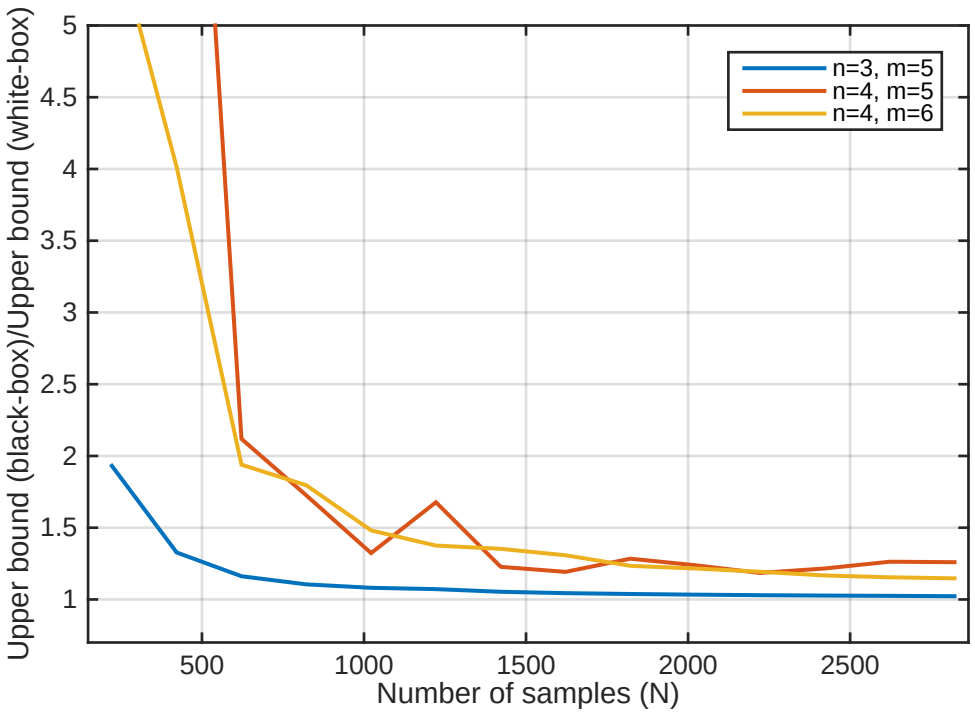


Figure 2.5: The comparison of the upper bounds computed in a black-box setting versus in a white-box setting using JSR toolbox [VHJ14].

As can be seen in Fig. 2.5, while increasing the number of modes does not significantly affect the quality of the bounds, the performance degrades considerably and consistently when the state dimension increases. Nevertheless, after 2500 samples, all upper bounds are within 25% of the upper bound on the JSR computed by the JSR toolbox.

Finally, we randomly generate $10,000$ test cases with switched linear systems of dimension between 2 and 7, number of modes between 2 and 5, and size of samples N between 30 and 800. We take $\beta = 0.95$ and check if the upper bound computed by our technique is greater than the actual JSR of the system. We get 9897 positive tests, out of $10,000$. The probability of success is 0.9897, which is significantly above the provided β. This is expected, since our techniques are based on worst-case analysis and thus fairly conservative.

4.4 Networked Control System

We now consider a linear time-invariant control system given as $x_{k+1} = Ax_k + Bu_k$, where we do not have access to its dynamics given by the matrices A and B. The control law is of the form $u_k = Kx_k$, where K is also unknown. The open-loop system is unstable with eigenvalues at $\{0.45, 1.1\}$. The controller stabilizes the system by bringing its eigenvalues to $\{0.8, -0.7\}$. The control input is transmitted over a wireless communication channel that is utilized by ℓ users, including the controller. The communication between the users and the recipients is performed based on the IEEE 802.15.4 MAC layer protocol [mac06], which is used in some of the proposed standards for control over wireless, e.g., WirelessHART [CNM10]. This MAC layer integrates both guaranteed slots and contention based slots. In this example, we consider a beacon-enabled mode of the MAC protocol. In this setup, a centralized control user periodically synchronizes and configures all the users. This period is named Beacon Interval. This interval is divided into two intervals: active and inactive period. The active period is divided into 6 slots. The first 2 slots correspond to the contention access period (CAP), and the next 4 slots correspond to the contention free period (CFP). In the CAP, the users can only send their message if the channel is "idle" with carrier-sense multiple access with collision avoidance (CSMA/CA). In the CFP however, each user has guaranteed time slots, during which there are no packet losses. In our example, the third and fourth slots are designated for the controller, while the fifth and sixth slots are allocated to the other users. Finally, during the inactive period, all users enter a low-power mode to save energy. We illustrate the overall structure of this communication protocol in Fig. 2.6. We now want to decide whether the resulting closed-loop networked control system is stable by simulating it starting from different initial conditions.

Note that, the closed-loop dynamics of the underlying system when the controller is active is $A_c = A + BK$. Then, we can model the overall networked control system by the linear switched system $x_{k+1} = \bar{A}x_k$, where $\bar{A} \in \mathcal{M}$ and

$$\mathcal{M} = \{A^2A_c^2A^4, A_cAA_c^2A^4, AA_c^3A^4, A_c^4A^4\}. \tag{2.21}$$

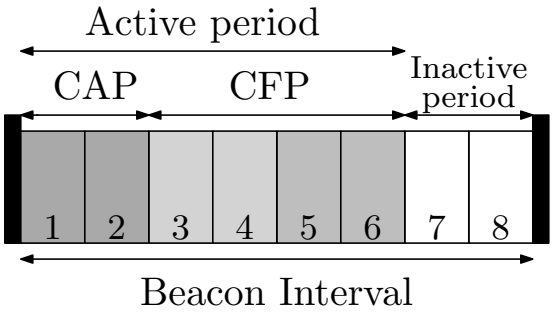


Figure 2.6: The time allocation structure of the modified IEEE 802.15.4 MAC layer.

Note that, in (2.21), each mode corresponds to a different utilization of the CFP by the users. For example, the mode defined by $A_c A A_c^2 A^4$ is active when the first slot in the CFP is assigned to the controller and the second slot is assigned to the other users. We assume that all of the users using the channel have an equal probability of being assigned to a time slot during the CFP. Therefore, the probability of each mode in $\mathcal{M}$ being active is $\left\{\frac{1}{(\ell-1)^2}, \frac{1}{\ell(\ell-1)}, \frac{1}{(\ell-1)\ell}, \frac{1}{\ell^2}\right\}$, i.e., the modes given in (2.21) will not be active with the same probability. Hence, we make use of Remark 3.3 and update our bounds accordingly. Fig. 2.7 demonstrates the resulting upper and lower bounds on the JSR. As can be seen, approximately after 500 samples, the upper bound on the JSR drops below 1, which lets us decide that the given closed-loop networked control system is stable, with probability 0.95.

5 Conclusions

In this chapter, we investigated the question of how one can conclude stability of a dynamical system when a model is not available and, instead, we have randomly generated state measurements. Our goal is to understand how the observation of well-behaved trajectories *intrinsically* implies stability of a system. It is not surprising that we need some standing assumptions on the system, in order to allow for any sort of nontrivial stability certificate solely from a finite number of observations.

The novelty of our contribution is twofold: First, we use as standing assumption that the unknown system can be described by a switching linear system. This assumption covers

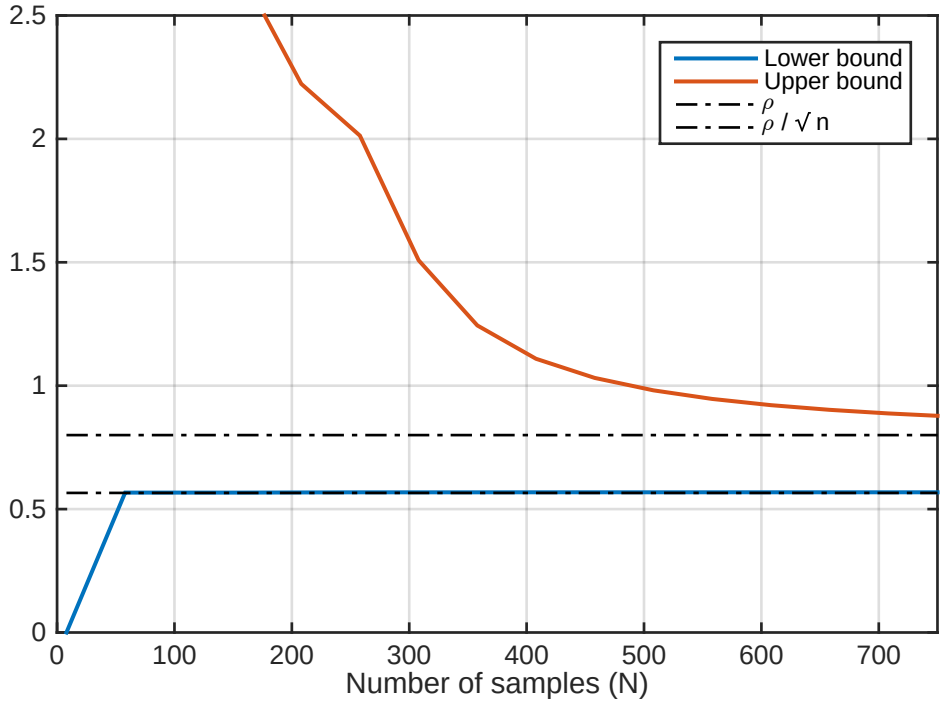


Figure 2.7: The evolution of the computed upper and lower bounds on the JSR with respect to the number of simulations collected from the networked control system.

a wide range of systems of interest, and to our knowledge no such "black-box" result has been available so far on switched systems. Second, we apply powerful techniques from chance constrained optimization. The application is not obvious, and relies on geometric properties of linear switched systems.

We believe that this guarantee is quite powerful, in view of the hardness of the general problem. In the future, we plan to investigate how to generalize our results to more complex or realistic systems. We are also improving the numerical properties of our technique by incorporating sum-of-squares optimization, and relaxing the sampling assumptions on the observations. We finally note that we can get asymptotically tighter bounds by considering longer trajectories as opposed to only observing a state and its successor.

CHAPTER 3

Conclusions and Future Work

In this thesis, we addressed two important problems in the context of cyber-physical systems: synthesis of correct-by-design controllers and stability verification of closed-loop system. In the first chapter of the thesis, we introduced mode-target specifications as a natural way of modeling the desired behavior of various dynamical systems from numerous application domains. Relying on the extensive literature in the abstraction based controller design for cyber-physical systems, we introduced an algorithm to synthesize controllers enforcing such mode-target specifications. The running time of this algorithm is polynomial in the number of states of the abstraction describing the dynamical system of interest. We showed the practicality of the algorithm on the design of an adaptive cruise controller system. At the core of this chapter was the observation that the specificity of control systems can be exploited when building upon ideas from the formal methods literature. This of course leads to the interesting question and important future direction: *What are other specific properties of control systems that we can exploit to develop more efficient control specific formal synthesis algorithms compared to the generic ones developed by the computer science community?*

In the second chapter, we steered away from the model based approach in the first chapter and studied the stability of switched linear systems based solely on their uniformly randomly sampled state trajectories. Given a user-defined confidence level and a switched linear system, we proposed an algorithm that informs the user about how stable the given system is. The algorithm closes the gap between the information obtained from finitely many state trajectories and the stability of the overall system. This is possible due to specific properties of switched linear systems that lets us infer the behavior of the trajectories starting between the sampled points. We hope that our technique serves as the

beginning of a worthwhile quest to discover similar properties arising in more general classes of dynamical systems.

APPENDIX A

Mode-Target Games

1 Preliminary Lemmas

A property Φ is a *stable property* iff $\text{Post}(\Phi) \subseteq \Phi$, i.e., if Φ is closed under suffixes. We call φ a *stable* formula if $W(\varphi)$ is a stable property. It is proved in [Sis94] that a formula φ is a stable formula iff $\Box\varphi \equiv \varphi$. Then it follows that any formula of the form $\Box\phi$, for some ϕ is a stable formula. Moreover, the conjunction of stable formulas is also a stable formula. Take two stable formulas φ_1 and φ_2; then $\varphi_1 \wedge \varphi_2 \equiv \Box\varphi_1 \wedge \Box\varphi_2 \equiv \Box(\varphi_1 \wedge \varphi_2)$, which is a stable formula. Also recall that a property Φ is an *absolute liveness property* iff $\Sigma^*\Phi \subseteq \Phi$. We call φ an *absolute liveness formula* if $W(\varphi)$ is an absolute liveness property.

Lemma 1.1. *Given the formulae φ_1 and φ_2, if we have $W_G(\varphi_1 \wedge \varphi_2) = \emptyset$, then the following holds:*

$$W_G(\neg\varphi_1 \vee \varphi_2) = W_G(\neg\varphi_1).$$

Proof.

$$\begin{aligned} & W_G(\neg\varphi_1 \vee \varphi_2) \\ = \;& W_G((\neg\varphi_1 \wedge \varphi_2) \vee (\neg\varphi_1 \wedge \neg\varphi_2) \vee (\varphi_1 \wedge \varphi_2)) \\ = \;& W_G((\neg\varphi_1 \wedge \varphi_2) \vee (\neg\varphi_1 \wedge \neg\varphi_2)) = W_G(\neg\varphi_1). \end{aligned}$$

□

Lemma 1.2. *Given the sets of LTL formulae $\cup_{i\in I}\{\varphi_i\}$, $\cup_{i\in I}\{\psi_i\}$, and a game graph G, if for all $i \in I$ we have $W_G\left(\varphi_i \wedge \bigvee_{j\in I\setminus\{i\}} \psi_j\right) = \emptyset$, then the following holds:*

$$W_G\left(\vee_{i\in I}\varphi_i \implies \vee_{i\in I}\psi_i\right) = W_G\left(\wedge_{i\in I}\left(\varphi_i \implies \psi_i\right)\right)$$

Proof. The following holds:

$$\begin{aligned}
& W_G\left(\vee_{i\in I}\varphi_i \implies \vee_{i\in I}\psi_i\right)\\
= \;& W_G\left((\wedge_{i\in I}\neg\varphi_i)\vee(\vee_{j\in I}\psi_j)\right)\\
= \;& W_G\left(\left(\bigwedge_{i\in I}\neg\varphi_i\vee\psi_i\right)\vee\left(\bigwedge_{i\in I}\neg\varphi_i\vee\bigvee_{j\in I\setminus\{i\}}\psi_j\right)\right)\\
= \;& W_G\left(\bigwedge_{i\in I}(\neg\varphi_i\vee\psi_i)\right)\cup W_G\left(\bigwedge_{i\in I}\left(\neg\varphi_i\vee\bigvee_{j\in I\setminus\{i\}}\psi_j\right)\right)\\
\overset{(1)}{=} \;& W_G\left(\bigwedge_{i\in I}(\neg\varphi_i\vee\psi_i)\right)\cup W_G\left(\bigwedge_{i\in I}\neg\varphi_i\right)\\
\overset{(2)}{=} \;& W_G\left(\wedge_{i\in I}(\neg\varphi_i\vee\psi_i)\right)=W_G\left(\wedge_{i\in I}\left(\varphi_i \implies \psi_i\right)\right)
\end{aligned}$$

where $\overset{(1)}{=}$ follows from the fact that

$$W_G\left(\varphi_i\wedge\bigvee_{j\in I\setminus\{i\}}\psi_j\right)=\emptyset,\forall\, i\in I,$$

and Lemma 1.1, while $\overset{(2)}{=}$ follows from the inclusion $W_G\left(\wedge_{i\in I}\neg\varphi_i\right)\subseteq W_G\left(\wedge_{i\in I}(\neg\varphi_i\vee\psi_i)\right)$. □

Lemma 1.3. *Given a stable formula φ, and a winning strategy f for player 0 in (G,φ), we have $[\![\varphi]\!]=V^*$, where $V^*:=\cup_{v\in[\![\varphi]\!]}\cup_{r\in\Omega_{f,v}(G)}\cup_{i\in\mathbb{N}}r_i$, i.e., the set of all states visited under the strategy f.*

Proof. We note that it suffices to show $V^*\subseteq[\![\varphi]\!]$. The other direction is immediate due to the definition of V^*. Note that since φ is a stable formula, it is closed under suffixes. This means that any strategy f that is winning for (G,φ) is winning for $(G,\Box[\![\varphi]\!])$ as well. Therefore, any play $r\in\cup_{v\in[\![\varphi]\!]}\cup_{r\in\Omega_{f,v}(G)}$ always stays inside the set $[\![\varphi]\!]$, hence $V^*\subseteq[\![\varphi]\!]$, and the result follows. □

Lemma 1.4. *Let p and q be positional formulas, then*

$$\Box(\Diamond p\vee\Diamond\Box q)\equiv\Box\Diamond p\vee\Diamond\Box q.$$

Proof.

$$\Box(\Diamond p \vee \Diamond\Box q) \overset{(1)}{\equiv} \Box(\Diamond(p \vee \Box q)) \equiv \Box\Diamond(p \vee \Box q)$$
$$\overset{(2)}{\equiv} \Box\Diamond p \vee \Box\Diamond\Box q \overset{(3)}{\equiv} \Box\Diamond p \vee \Diamond\Box q,$$

where $\overset{(1)}{\equiv}$ holds since $\Diamond\varphi_1 \vee \Diamond\varphi_2 \equiv \Diamond(\varphi_1 \vee \varphi_2)$,and $\overset{(2)}{\equiv}$ is true because $\Box\Diamond(\varphi_1 \vee \varphi_2) \equiv \Box\Diamond\varphi_1 \vee \Box\Diamond\varphi_2$. Finally, $\overset{(3)}{\equiv}$ follows from $\Box\Diamond\Box q \equiv \Diamond\Box q$.

□

Lemma 1.5. *Let p and q be positional formulas, then*

$$(\Diamond\Box p \implies \Diamond\Box q) \equiv (\Diamond\Box p \implies \Diamond\Box(p \wedge q)).$$

Proof.

$$(\Diamond\Box p \implies \Diamond\Box(p \wedge q)) \equiv (\Diamond\Box p \implies (\Diamond\Box p \wedge \Diamond\Box q)) \equiv (\neg(\Diamond\Box p) \vee (\Diamond\Box p \wedge \Diamond\Box q))$$
$$\overset{(1)}{\equiv} (\neg(\Diamond\Box p) \vee \Diamond\Box p) \wedge (\neg(\Diamond\Box p) \vee \Diamond\Box q) \equiv \mathbf{True} \wedge (\neg(\Diamond\Box p) \vee \Diamond\Box q) \equiv (\Diamond\Box p \implies \Diamond\Box q),$$

where $\overset{(1)}{\equiv}$ holds because $\vee$ distributes over $\wedge$. □

2 Proof of Theorem 3.4

Let $Z^* = \nu Z \bigcap_{i\in I} [\![\psi_i \vee \Diamond(p_i \wedge \circ Z)]\!]$. We start by proving $Z^* \subseteq [\![\Box \wedge_{i\in I} \varphi_i]\!]$. We make the following observation:

$$\left((\Sigma^* p_1)(\Sigma^* p_2)\dots(\Sigma^* p_{|I|})\right)^\omega = W\left(\Box \bigwedge_{i\in I} \Diamond p_i\right) \subseteq W\left(\Box \bigwedge_{i\in I} \Diamond p_i \vee \psi_i\right).$$

This suggests that a strategy that visits all p_i's in a circular fashion is winning for player 0. We pick the visiting order $p_1 p_2 \dots p_i \dots p_{|I|}$, since it is enough to find one winning strategy. Therefore, whenever a play visits a state that satisfies p_i player 0 should be able to switch to a strategy that is winning for the game with the winning condition $\Diamond p_{i+1(mod|I|)}$. Next, we explain that this is in fact possible on Z^*.

The game starts at a state in Z^*. Player 0 follows the strategy that is winning for the game $(G, \psi_i \vee \Diamond(p_i \wedge \circ Z^*))$, from Z^*. If the game reaches a state $v \in [\![p_i]\!]$, then player 0 forces a visit to Z^*. After that player 0 starts following a strategy that is winning for the game with the winning condition: $\psi_{i+1(mod|I|)} \vee \Diamond(p_{i+1(mod|I|)} \wedge \circ Z^*)$. This switching is possible since $Z^* \subseteq [\![\psi_i \vee \Diamond(p_i \wedge \circ Z^*)]\!]$, for all $i \in I$. The circular switching can be implemented using a counter, with $|I|$ states.

Due to the disjunction of the reachability part of the formula with ψ_i, it is true that a play that follows the above strategy can be winning for (G, ψ_i) for some $i \in I$, instead of $(G, \Diamond p_i)$ for some $i \in I$. However, since we assumed that for each $i \in I$, ψ_i is an absolute liveness formula, and $W_G(\psi_i) \subseteq W(\Box \wedge_{i \in I} \varphi_i)$, even in this case the play is winning for $\Box \bigwedge_{i \in I} \varphi_i$. Therefore, $Z^* \subseteq [\![\Box \wedge_{i \in I} \varphi_i]\!]$.

Now, we show that the other direction, i.e., $[\![\Box \wedge_{i \in I} \varphi_i]\!] \subseteq Z^*$. To show that $[\![\Box \wedge_{i \in I} \varphi_i]\!] \subseteq Z^*$, it is sufficient to show $[\![\Box \wedge_{i \in I} \varphi_i]\!] \subseteq F([\![\Box \wedge_{i \in I} \varphi_i]\!])$, where $F(Z) := \cap_{i \in I} [\![\psi_i \vee \Diamond(p_i \wedge \circ Z)]\!]$ (see e.g. [Tar55]). Since $\Box \wedge_{i \in I} \varphi_i$ is a stable formula, we can invoke Lemma 1.3, with $\varphi = \Box \wedge_{i \in I} \varphi_i$ and conclude that

$$[\![\Box \wedge_{i \in I} \varphi_i]\!] = V^*,$$

where $V^* = \cup_{v \in [\![\Box \wedge_{i \in I} \varphi_i]\!]} \cup_{r \in \Omega_{v,f}(G)} \cup_{i \in \mathbb{N}} r_i$. Then, we get:

$$[\![\Box \wedge_{i \in I} \varphi_i]\!] = V^* \overset{(1)}{\subseteq} \bigcap_{i \in I} [\![(\psi_i \vee \Diamond p_i) \wedge \Box V^*]\!] \subseteq \bigcap_{i \in I} [\![(\psi_i \vee \Diamond(p_i \wedge \circ V^*)]\!] = F(V^*),$$

where $\overset{(1)}{\subseteq}$ follows from the definition of V^*, since it includes all states visited under the winning strategy for player 0 in (G, φ). We just proved that $V^* \subseteq F(V^*)$. Note that, for any $S \subseteq V$ we have $S \subseteq F(S) \implies S \subseteq Z^*$ due to [Tar55]. This shows that $V^* = [\![\Box \wedge_{i \in I} \varphi_i]\!] \subseteq Z^*$, which completes the proof.

3 Strategy Synthesis

Recall that a strategy is a partial function $f : V^* \times V_0 \to V$ such that whenever $f(r, v)$ is defined, $(v, f(r, v)) \in E$. We next construct a strategy with finite memory that can also be described by a finite-state automaton $A_f = (S, s_0, \delta_M, \delta)$, where S is a finite set that represents the memory, $s_0 \in S$ is the initial memory content, $\delta : V_0 \times S \to V$ is the transition function which determines the next state to be visited based on the current memory content and the current state of the game, and $\delta_M : S \times V \to S$ is the transition function that describes how the memory content should be updated according to the current memory content and current state of the game. We denote the natural extension of δ_M from $S \times V$ to $S \times V^*$ by δ_M^*. Once we specify A_f we can construct f as

$$f(v_0 \dots v_{i-1}, v_i) := \delta(v_i, \delta_M^*(s_0, v_0 \dots v_{i-1})). \tag{A.1}$$

We now consider the problem of constructing A_f. The set of memory states S is given by $S = \{1, 2, \dots, m\}$ and the initial memory state is $s_0 = 1$. The memory state being i indicates that the player 0 should follow a strategy that is winning for the game with the winning condition $\vee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \vee \Diamond\neg M_i$. If the play satisfies $\Diamond\neg M_i$ then the memory state is incremented. Otherwise, the play satisfies $\vee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})$ and the memory state remains the same. Moreover, we decide (other choices lead to equally valid solutions) that the memory states should evolve according to $1, 2, \dots, m, 1, 2, \dots$ This leads us to define the transition function δ_M by

$$\delta_M(v, i) = \begin{cases} i + 1 (\mathrm{mod}\, m) & \text{if } v \in [\![\neg M_i]\!] \\ i & \text{if } v \notin [\![\neg M_i]\!] \end{cases}$$

The final step is the definition of δ. It will be based on a set of edges that can be computed from the intermediate results obtained when computing the fixed-point in (1.15).

We start with some additional notation. We use $Y_i^{*\ell}$ to denote the set computed at the

ℓ^{th} iteration of the following fixed-point computation over Y:

$$\mu Y \left(\bigcup_{j=1}^{t_\ell} \nu X(\mathrm{Pre}(X) \cap [\![M_i \wedge T_{i,j}]\!]) \cup ([\![\neg M_i]\!] \cap [\![\varphi]\!]) \cup \mathrm{Pre}(Y) \right).$$

Similarly, $X_{i,j}^{*\ell *}$ denotes

$$\nu X(\mathrm{Pre}(X) \cap [\![M_i \wedge T_{i,j}]\!]) \cup ([\![\neg M_i]\!] \cap [\![\varphi]\!]) \cup \mathrm{Pre}(Y_i^{*\ell})).$$

For each counter value $k \in \{1, 2, \ldots, m\}$, we define the set of edges $E_k := E_{k,1} \cup E_{k,2} \cup E_{k,3}$, where

$$\begin{aligned}
E_{k,1} &= \{(v, v') \in E \mid v \in [\![\neg M_k]\!], \wedge v' \in [\![\varphi]\!]\}, \\
E_{k,2} &= \bigcup_{\ell > 1} \{(v, v') \in E \mid v \in Y_k^{*\ell} \wedge v \notin Y_k^{*<\ell} \wedge v' \in Y_k^{*<\ell}\}, \\
E_{k,3} &= \bigcup_{j=1}^{t_k} \bigcup_{\ell} \{(v, v') \in E \mid v \in X_{k,j}^{*\ell *} \cap [\![M_k \wedge T_{k,j}]\!] \wedge v \notin X_{k,j}^{*<\ell *} \wedge v' \in X_{k,j}^{*\ell *}\},
\end{aligned}$$

where $Y_k^{*<\ell} = \bigcup_{0 \le i < \ell} Y_k^{*i}$ and $X_{k,j}^{*<\ell *} = \bigcup_{0 \le \ell < k} X_{k,j}^{*\ell *}$. The set $E_{k,1}$ contains the transitions that player 0 can use to force the game to move from a state in $[\![\neg M_k]\!]$ to a state in $[\![\varphi]\!]$. $E_{k,2}$ corresponds to the transitions, that player 0 can force the game to make progress towards a state in $[\![\neg M_k]\!]$ or a state that will not leave $[\![M_k \wedge T_{k,j}]\!]$ forever for some j. The edges in $E_{j,3}$ are the transitions, where the game is at a state in $[\![M_k \wedge T_{k,j}]\!]$, and player 0 can force the game to stay in $[\![M_k \wedge T_{k,j}]\!]$ but cannot force it to make progress towards a state in $[\![\neg M_k]\!]$. Note that player 0 still wins by always taking the transitions in $E_{j,3}$ since even if there is no progress towards $[\![\neg M_k]\!]$, the game stays in $[\![M_k \wedge T_{k,j}]\!]$ forever as well.

We make use of edges E_k to define δ when the memory state is at k as $\delta(v_0, k) = v'$, where $(v_0, v') \in E_k$. Now, we have all the ingredients to construct the winning strategy f according to (A.1).

4 Proof of Proposition 4.1

We prove this proposition in two main steps. In the first step, we show that every mode-target game can be transformed into an equivalent GR(1) game.

Let (G, φ) be a mode-target game. Then the following holds:

$$\varphi = \bigwedge_{i=1}^{m} \left(\Diamond\Box M_i \implies \bigvee_{j=1}^{t_i} \Diamond\Box T_{i,j} \right) \overset{(1)}{\equiv} \bigwedge_{i=1}^{m} \left(\Diamond\Box M_i \implies \bigvee_{j=1}^{t_i} \Diamond\Box (M_i \wedge T_{i,j}) \right) \tag{A.2}$$

Let $\varphi_i = \Diamond\Box(M_i \wedge T_i)$ and $\psi_i = \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j})$. Since the modes are mutually exclusive, i.e., $M_i \in L(v) \implies M_j \notin L(v)$, $\forall\, j \neq i$ and $\forall v \in V$, the following holds:

$$W_G \left(\varphi_i \wedge \bigvee_{i \in I \setminus \{j\}} \psi_j \right) = \emptyset, \forall i \in I. \tag{A.3}$$

Then due to Lemma 1.2 we get: $\varphi \equiv \left(\bigvee_{i=1}^{m} \Diamond\Box M_i \implies \bigvee_{i=1}^{m} \bigvee_{i=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \right)$. Next we show:

$$\begin{aligned} & W_G \left(\textstyle\bigvee_{i=1}^{m} \Diamond\Box M_i \implies \bigvee_{i=1}^{m} \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \right) \\ & = W_G \left(\textstyle\bigvee_{i=1}^{m} \Diamond\Box M_i \implies \bigvee_{j=1}^{\max_i t_i} \Diamond\Box \vee_{i=1}^{m} (M_i \wedge \bar{T}_{i,j}) \right) \end{aligned}$$

where $\bar{T}_{i,j} = T_{i,j}$ if $j \leq t_i$ and $\bar{T}_{i,j} = \texttt{false}$, otherwise.

The inclusion:

$$W_G(\varphi) \subseteq W_G \left(\bigvee_{i=1}^{m} \Diamond\Box M_i \implies \bigvee_{j=1}^{\max_i t_i} \Diamond\Box \vee_{i=1}^{m} (M_i \wedge \bar{T}_{i,j}) \right)$$

is immediate since $\bigvee_{i=1}^{m} \bigvee_{j=1}^{t_i} \Diamond\Box(M_i \wedge T_{i,j}) \equiv \bigvee_{j=1}^{\max_i t_i} \bigvee_{i=1}^{m} \Diamond\Box(M_i \wedge \bar{T}_{i,j})$ and $\bigvee_{j=1}^{\max_i t_i} \bigvee_{i=1}^{m} \Diamond\Box(M_i \wedge \bar{T}_{i,j})$ implies $\bigvee_{j=1}^{\max_i t_i} \Diamond\Box \vee_{i=1}^{m} (M_i \wedge \bar{T}_{i,j})$. To show the other direction, we start with the following observation. Suppose $r \in V^\omega$, and let I be a finite index set. Then the following semantic

relation holds:

$$L(r) \models \Box \bigvee_{i\in I} p_i \implies L(r) \models \bigvee_{i\in I} \Box p_i \vee \bigvee_{\substack{J\subseteq I,\\ |J|>1}} \bigwedge_{j\in J} \Box\Diamond p_j, \tag{A.4}$$

where each p_i is a positional formula. Note that this follows from the fact that any word satisfying $\Box \vee_{i\in I} p_i$ should either always stay in one of the p_i's forever, and hence satisfy $\vee_{i\in I}\Box p_i$ or shuffle between at least two different p_i's, i.e., satisfy $\bigvee_{\substack{J\subseteq I,\\ |J|>1}} \bigwedge_{j\in J} \Box\Diamond p_j$. Let $I := \{1, 2, \dots m\}$. We are now ready to show the other direction as follows:

$$\begin{aligned}
&W_G\left(\bigvee_{i=1}^{m} \Diamond\Box M_i \implies \bigvee_{j=1}^{\max_i t_i} \Diamond\Box \bigvee_{i=1}^{m} (M_i \wedge \bar{T}_{i,j})\right)\\
&\overset{(1)}{\subseteq} W_G\left(\bigvee_{j=1}^{\max_i t_i}\bigvee_{i=1}^{m} \Diamond\Box(M_i \wedge \bar{T}_{i,j}) \vee \bigvee_{j=1}^{\max_i t_i}\bigvee_{i=1}^{m}\bigvee_{\substack{J\subseteq I_m,\\ |J|>1}}\bigwedge_{s\in J} \Box\Diamond(M_s \wedge \bar{T}_{s,j}) \vee \bigwedge_{i=1}^{m} \Box\Diamond\neg M_i\right)\\
&\overset{(2)}{\subseteq} W_G\left(\bigvee_{i=1}^{m}\bigvee_{j=1}^{\max_i t_i} \Diamond\Box(M_i \wedge \bar{T}_{i,j}) \vee \bigwedge_{i=1}^{m} \Box\Diamond\neg M_i\right),
\end{aligned}$$

where $\overset{(1)}{\subseteq}$ follows from the inclusion given in (A.4), distributivity of $\Diamond$ with respect to $\vee$ and the syntactic equivalence

$$\Diamond \bigwedge_{s\in J} \Box\Diamond(M_s \wedge \bar{T}_{s,j}) \equiv \bigwedge_{s\in J} \Box\Diamond(M_{s,j} \wedge \bar{T}_{s,j}).$$

Due to the disjointness of modes we have $W_G(M_i) \subseteq W_G(\neg M_j)$, $\forall j \neq i$, and therefore $\overset{(2)}{\subseteq}$ follows from the fact that $\bigvee_{\substack{J\subseteq I_m,\\ |J|>1}}\bigwedge_{s\in J} \Box\Diamond(M_s \wedge \bar{T}_{s,j})$ implies $\wedge_{i=1}^{m}\Box\Diamond\neg M_i$. Therefore we have:

$$W_G(\varphi) = W_G\left(\bigvee_{i=1}^{m} \Diamond\Box M_i \implies \bigvee_{j=1}^{\max_i t_i} \Diamond\Box \vee_{i=1}^{m} (M_i \wedge \bar{T}_{i,j})\right). \tag{A.5}$$

This completes the proof since we can rewrite the formula on the right hand side of the equality (A.5) and get and get the equality in (1.16).

APPENDIX B

Stability Analysis of Switched-Linear Systems

1 Notation and Background

Before proceeding to the main lemmas we use to prove Lemma 3.1, we first introduce some necessary definitions and related background.

Let d be a distance on $\mathbb{R}^n$. The distance between a set $X \subset \mathbb{R}^n$ and a point $p \in \mathbb{R}^n$ is $d(X,p) := \inf_{x\in X} d(x,p)$. Note that the map $p \mapsto d(X,p)$ is continuous on $\mathbb{R}^n$. Given a set $X \subset \mathbb{R}^n$, ∂X denotes the boundary of set X.

Definition 1.1. We define the *spherical cap* on $\mathbb{S}$ for a given hyperplane $c^T x = k$ as:

$$\mathcal{C}_{c,k} := \{x \in \mathbb{S} : c^T x > k\}.$$

Remark 1.2. Consider the spherical caps $\mathcal{C}_{c,k_1}$ and $\mathcal{C}_{c,k_2}$ such that $k_1 > k_2$, then we have:

$$\sigma^{n-1}(\mathcal{C}_{c,k_1}) < \sigma^{n-1}(\mathcal{C}_{c,k_2}).$$

Remark 1.3. The distance between the point $x = 0$ and the hyperplane $c^T x = k$ is $\frac{|k|}{\|c\|}$.

We now define the function $\Delta : \wp(\mathbb{S}) \to [0,1]$ as:

$$\Delta(X) := \sup\{r : r\mathbb{B} \subset \text{convhull } (\mathbb{S} \setminus X)\}. \tag{B.1}$$

Note that, $\Delta(X)$ can be rewritten as:

$$\Delta(X) = d(\partial\text{convhull } (\mathbb{S} \setminus X), 0). \tag{B.2}$$

Lemma 1.4. *Consider the spherical cap $\mathcal{C}_{c,k}$. We have:*

$$\Delta(\mathcal{C}_{c,k}) = \min\left(1, \frac{|k|}{\|c\|}\right).$$

Proof. Note that:

$$\text{convhull } (\mathbb{S} \setminus \mathcal{C}_{c,k}) = \left\{x \in \mathbb{B} : c^T x \leq k\right\}.$$

Then the following equalities hold:

$$\begin{aligned}
\Delta(\mathcal{C}_{c,k}) &= d(\partial\text{convhull } (\mathbb{S} \setminus \mathcal{C}_{c,k}), 0) \\
&= \min(d(\partial\mathbb{B}, 0), d(\partial\{x : c^T x \leq k\}, 0)) \\
&= \min(d(\mathbb{S}, 0), d(\{x : c^T x = k\}, 0)) \\
&= \min\left(1, \frac{|k|}{\|c\|}\right).
\end{aligned}$$

□

Corollary 1.5. *Consider the spherical caps $\mathcal{C}_{c,k_1}$ and $\mathcal{C}_{c,k_2}$ such that $k_1 \leq k_2$. Then we have:*

$$\Delta(\mathcal{C}_{c,k_1}) \leq \Delta(\mathcal{C}_{c,k_2}).$$

2 Preliminary Results

Lemma 2.1. *For any set $X \subset \mathbb{S}$, there exist c and k such that $\mathcal{C}_{c,k}$ satisfies: $\mathcal{C}_{c,k} \subset X$, and $\Delta(\mathcal{C}_{c,k}) = \Delta(X)$.*

Proof. Let $\tilde{X} := \text{convhull } (S \setminus X)$. Since d is continuous and the set $\partial\tilde{X}$ is compact, there exists a point $x^* \in \partial\tilde{X}$, such that:

$$\Delta(X) = d(\partial\tilde{X}, 0) = \min_{x \in \partial\tilde{X}} d(x, 0) = d(x^*, 0).$$

Next, consider the hyperplane which is tangent to the ball $\Delta(X)\mathbb{B}$ at x^*, which we denote

by $\{x : c^T x = k\}$. Then we have:

$$\Delta(X) = d(x^*, 0) = d(\{x : c^T x = k\}, 0) = \min\left(1, \frac{|k|}{\|c\|}\right).$$

Now, consider the spherical cap defined by this tangent plane i.e., $\mathcal{C}_{c,k}$. Then, by Lemma 1.4 we have $\Delta(\mathcal{C}_{c,k}) = \min\left(1, \frac{|k|}{\|c\|}\right)$. Therefore, $\Delta(X) = \Delta(\mathcal{C}_{c,k})$.

We next show $\mathcal{C}_{c,k} \subset X$. We prove this by contradiction. Assume $x \in \mathcal{C}_{c,k}$ and $x \notin X$. Note that, if $x \notin X$, then $x \in \mathbb{S} \setminus X \subset$ convhull $(\mathbb{S} \setminus X)$. Since $x \in \mathcal{C}_{c,k}$, we have $c^T x > k$. But due to the fact that $x \in$ convhull $(\mathbb{S} \setminus X)$, we also have $c^T x \leq k$, which leads to a contradiction. Therefore, $\mathcal{C}_{c,k} \subset X$. □

Proposition 2.2. *Let $\mathcal{X}_\epsilon = \{X \subset \mathbb{S} : \sigma^{n-1}(X) \leq \epsilon\}$. Then, for any $\epsilon \in (0, 1)$, the function $\Delta(X)$ attains its minimum over $\mathcal{X}_\epsilon$ for some X which is a spherical cap.*

Proof. We prove this via contradiction. Assume that there exists no spherical cap in $\mathcal{X}_\epsilon$ such that $\Delta(X)$ attains its minimum. This means there exists an $X^* \in \mathcal{X}_\epsilon$, where X^* is not a spherical cap and $\arg\min_{X \in \mathcal{X}_\epsilon}(\Delta(X)) = X^*$. By Lemma 2.1, we can construct a spherical cap $\mathcal{C}_{c,k}$ such that $\mathcal{C}_{c,k} \subset X^*$ and $\mathcal{C}_{c,k} = \Delta(X^*)$. Note that, we further have $\mathcal{C}_{c,k} \subsetneq X^*$, since X^* is assumed not to be a spherical cap. This means that, there exists a spherical cap $\sigma^{n-1}(\mathcal{C}_{c,k})$ such that $\sigma^{n-1}(\mathcal{C}_{c,k}) < \epsilon$.

Then, the spherical cap $\mathcal{C}_{c,\tilde{k}}$ with $\sigma^{n-1}(\mathcal{C}_{c,\tilde{k}}) = \epsilon$, satisfies $\tilde{k} < k$ by Remark 1.2. This implies

$$\Delta(\mathcal{C}_{c,\tilde{k}}) < \Delta(\mathcal{C}_{c,k}) = \Delta(X^*)$$

by Corollary 1.5. Therefore, $\Delta(\mathcal{C}_{c,\tilde{k}}) < \Delta(X^*)$. This is a contradiction since we initially assumed that $\Delta(X)$ attains its minimum over $\mathcal{X}_\epsilon$ at X^*. □

3 Main Lemma

Lemma 3.1. *Let $\epsilon \in (0, \frac{1}{2})$ and $\alpha : (0,1) \to \mathbb{R}_{\geq 0}$ be defined by:*

$$\alpha(\epsilon) := \inf_{X \in \mathcal{X}_\epsilon} \sup\{r : r\mathbb{B} \subset \textit{convhull}\,(\mathbb{S} \setminus X)\}, \tag{B.3}$$

where $\mathcal{X}_\epsilon = \{X \subset \mathbb{S} : \sigma^{n-1}(X) \leq \epsilon\}$. Then, $\alpha(\epsilon)$ is given by the formula:

$$\alpha(\epsilon) = \sqrt{1 - I^{-1}\left(2\epsilon; \frac{n-1}{2}, \frac{1}{2}\right)}, \tag{B.4}$$

where I is the regularized incomplete beta function.

Proof. By Proposition 2.2 we know that:

$$\alpha(\epsilon) = \Delta(\mathcal{C}_{c,k}), \tag{B.5}$$

for some spherical cap $\mathcal{C}_{c,k} \subset \mathbb{S}$, where $\sigma^{n-1}(\mathcal{C}_{c,k}) = \epsilon$. It is known (see e.g. [Li11]) that the area of such $\mathcal{C}_{c,k}$, is given by the equation:

$$\sigma^{n-1}(\mathcal{C}_{c,k}) = \frac{I\left(1 - \Delta(\mathcal{C}_{c,k})^2; \frac{n-1}{2}, \frac{1}{2}\right)}{2}. \tag{B.6}$$

Since, $\sigma^{n-1}(\mathcal{C}_{c,k}) = \epsilon$, we get the following set of equations:

$$\begin{aligned} \epsilon &= \frac{I\left(1 - \Delta(\mathcal{C}_{c,k})^2; \frac{n-1}{2}, \frac{1}{2}\right)}{2} \\ 1 - \Delta(\mathcal{C}_{c,k})^2 &= I^{-1}\left(2\epsilon; \frac{n-1}{2}, \frac{1}{2}\right) \\ \Delta(\mathcal{C}_{c,k})^2 &= 1 - I^{-1}\left(2\epsilon; \frac{n-1}{2}, \frac{1}{2}\right). \end{aligned} \tag{B.7}$$

Then, the equalities (B.5) and (B.7) imply (B.3). □

4 Proof of $\lim_{N\to\infty}\delta(\beta,\omega_N)=1$

Recall that, $\delta(\beta,\omega_N)=\alpha(m\kappa(P(\omega_N))\epsilon(\beta,\omega_N))$. We first show that $\kappa(P(\omega_N))$ is uniformly bounded in N. The optimization problem $\text{Opt}(\omega_N)$ given in (2.7), with $\gamma^*(\omega_N)$ replaced by $\gamma^*(Z)(1+\frac{\eta}{2})$ is strictly feasible, and thus admits a finite optimal value K for some solution $P_{\eta/2}$. Note that, $\lim_{N\to\infty}\gamma^*(\omega_N)=\gamma^*(Z)$ with probability 1. Thus, for large enough N, $\gamma^*(\omega_N)(1+\eta)>\gamma^*(Z)(1+\frac{\eta}{2})$. This also means that, for large enough N, $\text{Opt}(\omega_N)$ admits $P_{\eta/2}$ as a feasible solution and thus the optimal value of $\text{Opt}(\omega_N)$ is bounded by K. In other words, $\lambda_{\max}(P(\omega_N))\leq K$. Moreover, since $\lambda_{\min}(P(\omega_N))\geq 1$, we also have $\det(P(\omega_N))\geq 1$, which means that

$$\kappa(P(\omega_N))=\sqrt{\frac{\lambda_{\max}(P(\omega_N))^n}{\det(P(\omega_N))}}\leq\sqrt{K^n}. \tag{B.8}$$

We next show that for a fixed $\beta\in(0,1)$ $\lim_{N\to\infty}\epsilon(\beta,N)=0$. Note that, $\epsilon(\beta,N)$ satisfies the following equation:

$$1-\beta=\sum_{j=0}^{d}\binom{N}{j}\epsilon^j(1-\epsilon)^{N-j}.$$

We can then upper bound the term $1-\beta$ as in:

$$1-\beta\leq(d+1)N^d(1-\epsilon)^{N-d}. \tag{B.9}$$

We prove $\lim_{N\to\infty}\epsilon(\beta,N)=0$ by contradiction. Assume that $\lim_{N\to\infty}\epsilon(\beta,N)\neq 0$. This means that, there exists some $c>0$ such that $\epsilon(\beta,N)>c$ infinitely often. Then, consider the subsequence N_k such that $\epsilon(\beta,N_k)>c$, $\forall\, k$. Then, by (B.9) we have:

$$1-\beta\leq(d+1)N_k^d(1-\epsilon)^{N_k-d}\leq(d+1)N_k^d(1-c)^{N_k-d}\,\forall\, k\in\mathbb{N}.$$

Note that $\lim_{k\to\infty}(d+1)N_k^d(1-c)^{N_k-d}=0$. Therefore, there exists a k' such that:

$$(d+1)N_{k'}^d(1-c)^{N_k'-d}<1-\beta,$$

which is a contradiction. Therefore, we must have $\lim_{N\to\infty} \epsilon(\beta, N) = 0$.

Putting this together with (B.8), we get:

$$\lim_{N\to\infty} m\kappa(P(\omega_N))\epsilon(\beta, \omega_N) = 0.$$

By the continuity of the function I^{-1} this also implies: $\lim_{N\to\infty} \alpha(m\kappa(P(\omega_N))\epsilon(\beta, \omega_N)) = 1$.

References

[AHV15] R. Alur, T. A. Henzinger, and M. Y. Vardi. "Theory in Practice for System Design and Verification." *ACM SIGLOG News*, **2**(1):46–51, January 2015.

[Ake78] S.B. Akers. "Binary Decision Diagrams." *Computers, IEEE Transactions on*, **C-27**(6):509–516, June 1978.

[AL04] R. Alur and S. La Torre. "Deterministic Generators and Games for LTL Fragments." *ACM Trans. Comput. Logic*, **5**(1):1–25, January 2004.

[AMP98] E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. "Controller Synthesis For Timed Automata.", 1998.

[BCJ97] A. Browne, E.M. Clarke, S. Jha, D.E. Long, and W. Marrero. "An improved algorithm for the evaluation of fixpoint expressions." *Theoretical Computer Science*, **178**(1-2):237 – 255, 1997.

[BFG16] F. Blanchini, G. Fenu, G. Giordano, and F. A. Pellegrino. "Model-Free Plant Tuning." *IEEE Transactions on Automatic Control*, **PP**(99):1–1, 2016.

[BJP12] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Saar. "Synthesis of Reactive(1) Designs." *J. Comput. Syst. Sci.*, **78**(3):911–938, May 2012.

[BL15] R. Bobiti and M. Lazar. "A Delta-sampling Verification Theorem for Discrete-time, Possibly Discontinuous Systems." In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, HSCC '15, pp. 140–148, New York, NY, USA, 2015. ACM.

[BT99] V. D. Blondel and J. N. Tsitsiklis. "Complexity of stability and controllability of elementary hybrid systems." *Automatica*, **35**(3):479 – 489, 1999.

[BTD16] A. Balkan, P. Tabuada, J. V. Deshmukh, X. Jin, and J. Kapinski. "Underminer: A framework for automatically identifying non-converging behaviors in black box system models." In *2016 International Conference on Embedded Software (EMSOFT)*, pp. 1–10, Oct 2016.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[BVT] A. Balkan, M. Vardi, and P. Tabuada. "Mode-Target Games: Reactive Synthesis for Control Applications." arXiv:1504.07702v4 [math.OC], 2016.

[BVT16] A. Balkan, M. Vardi, and P. Tabuada. "Controller Synthesis for Mode-Target Games." In *Proceedings 5th IFAC Conference on Analysis and Design of Hybrid Systems*, ADHS, 2016.

[Cal10] G. Calafiore. "Random Convex Programs." *SIAM Journal on Optimization*, **20**(6):3427–3464, 2010.

[CG08] M. C. Campi and S. Garatti. "The Exact Feasibility of Randomized Solutions of Uncertain Convex Programs." *SIAM Journal on Optimization*, **19**(3):1211–1230, 2008.

[CNM10] D. Chen, M. Nixon, and A. Mok. *WirelessHART: Real-Time Mesh Network for Industrial Automation*. Springer Publishing Company, Incorporated, 1st edition, 2010.

[DJW97] S. Dziembowski, M. Jurdzinski, and I. Walukiewicz. "How much memory is needed to win infinite games?" In *Logic in Computer Science, 1997. LICS '97. Proceedings., 12th Annual IEEE Symposium on*, pp. 99–110, Jun 1997.

[DMV13] P. S. Duggirala, S. Mitra, and M. Viswanathan. "Verification of Annotated Models from Executions." In *Proceedings of the Eleventh ACM International Conference on Embedded Software*, EMSOFT '13, pp. 26:1–26:10, Piscataway, NJ, USA, 2013. IEEE Press.

[Ehl11] R. Ehlers. "Generalized Rabin(1) Synthesis with Applications to Robust System Synthesis." In M. Bobaru, K. Havelund, G. J. Holzmann, and R. Joshi, editors, *NASA Formal Methods*, volume 6617 of *Lecture Notes in Computer Science*, pp. 101–115. Springer Berlin Heidelberg, 2011.

[EL86] E. A. Emerson and C. Lei. "Efficient Model Checking in Fragments of the Propositional Mu-Calculus (Extended Abstract)." In *Proceedings of the First Annual IEEE Symposium on Logic in Computer Science (LICS)*, pp. 267–278, June 1986.

[HM14] Z. Huang and S. Mitra. "Proofs from Simulations and Modular Annotations." In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pp. 183–192, New York, NY, USA, 2014. ACM.

[HRK09] H.Venz, W. Ruhle, and J. Kysela. "Start-up and Shutdown Practices in BWRs as well as in Primary and Secondary Circuits of PWRs, VVERs and CANDUs." Technical report, ANT International, 2009.

[HZD04] Z. Hu, B. Zhang, and W. Deng. "Output controllability of switched power converters as switched linear systems." In *The 4th International Power Electronics and Motion Control Conference, 2004. IPEMC 2004.*, volume 3, pp. 1665–1668 Vol.3, Aug 2004.

[ISO10] ISO 15622:2010 (E). "Intelligent transport systems – Adaptive Cruise Control systems – Performance requirements and test procedures." Technical report, International Organization for Standardization, 2010.

[JDK14] X. Jin, J. V. Deshmukh, J. Kapinski, K. Ueda, and K. Butts. "Powertrain Control Verification Benchmark." In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pp. 253–262, New York, NY, USA, 2014. ACM.

[Jun09] R. M. Jungers. "The joint spectral radius, Theory and applications." In *Lecture Notes in Control and Information Sciences*, volume 385. Springer-Verlag, Berlin, 2009.

[KDF03] N. S. Kumpati, A. O. Driollet, M. Feiler, and G. Koshy. "Adaptive control using multiple models, switching and tuning." *International Journal of Adaptive Control and Signal Processing*, **17**(2):87–102, 2003.

[KDS14] J. Kapinski, J. V. Deshmukh, S Sankaranarayanan, and N. Arechiga. "Simulation-guided Lyapunov Analysis for Hybrid Dynamical Systems." In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pp. 133–142, New York, NY, USA, 2014. ACM.

[KE12] J. Křetínský and J. Esparza. *Deterministic Automata for the (F,G)-Fragment of LTL*, pp. 7–22. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[KF07] H. Kress-Gazit and G. E. Fainekos. "Where's Waldo? sensor-based temporal logic motion planning." In *in IEEE International Conference on Robotics and Automation*, pp. 3116–3121, 2007.

[KP16] N. Kalra and S. M. Paddock. "Driving to Safety: How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability?" 2016.

[KPP05] Y. Kesten, N. Piterman, and A. Pnueli. "Bridging the gap between fair simulation and trace inclusion." *Information and Computation*, **200**(1):35 – 61, 2005.

[Lau15] F. Lauer. "On the complexity of switching linear regression." *arXiv preprint arXiv:1510.06920*, 2015.

[Li11] S. Li. "Concise Formulas for the Area and Volume of a Hyperspherical Cap." *Asian Journal of Mathematics & Statistics*, **4**:66–70, 2011.

[LLZ16] S. Liu, D. Liberzon, and V. Zharnitsky. "On almost Lyapunov functions for non-vanishing vector fields." In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 5557–5562, Dec 2016.

[LOT13] Jun L., N. Ozay, U. Topcu, and R.M. Murray. "Synthesis of Reactive Switching Protocols From Temporal Logic Specifications." *Automatic Control, IEEE Transactions on*, **58**(7):1771–1785, July 2013.

[mac06] "IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems- Local and Metropolitan Area Networks-Specific Requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)." Technical report, 2006.

[MDT10] M. Mazo, A. Davitian, and P. Tabuada. "PESSOA: A Tool for Embedded Controller Synthesis." In *Proceedings of the 22Nd International Conference on Computer Aided Verification*, CAV'10, pp. 566–569, Berlin, Heidelberg, 2010. Springer-Verlag.

[Mor98a] A. S. Morse. "Control Using Logic-Based Switching." In *Trends in Control: A European Perspective*, pp. 69–113. Springer-Verlag, 1998.

[Mor98b] A. S. Morse. "Supervisory Control of Families of Linear Set-Point Controllers - Part 1: Exact Matching." *IEEE Trans. Automat. Contr*, **41**:1413–1431, 1998.

[NHB16] P. Nilsson, O. Hussien, A. Balkan, Y. Chen, A. Ames, J. Grizzle, N. Ozay, H. Peng, and P. Tabuada. "Correct-By-Construction Adaptive Cruise Control: Two Approaches." *IEEE Transactions on Control Systems Technology*, 2016.

[NS06] A. Nemirovski and A. Shapiro. "Convex approximations of chance constrained programs." *SIAM Journal on Optimization*, **17**(4):969–996, 2006.

[PP02] A. Papachristodoulou and S. Prajna. "On the construction of Lyapunov functions using the sum of squares decomposition." In *Proceedings of 41st IEEE Conference on Decision and Control*, pp. 3482–3487, Las Vegas, Nevada USA, December 2002.

[Rud87] W. Rudin. *Real and complex analysis*. McGraw-Hill Book Co., New York, third edition, 1987.

[Ses15] S. A. Seshia. "New Frontiers in Formal Methods: Learning, Cyber-Physical Systems, Education, and Beyond." *CSI Journal of Computing*, **2**(4):R1:3–R1:13, June 2015.

[Sif15] J. Sifakis. "System Design Automation: Challenges and Limitations." *Proceedings of the IEEE*, **103**(11):2093–2103, 2015.

[Sis94] A.P. Sistla. "Safety, liveness and fairness in temporal logic." *Formal Aspects of Computing*, **6**(5):495–511, 1994.

[Str81] R. S. Streett. "Propositional Dynamic Logic of Looping and Converse." In *Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing*, STOC '81, pp. 375–383, New York, NY, USA, 1981. ACM.

[Tab09] P. Tabuada. *Verification and control of hybrid systems: a symbolic approach*. Springer, 2009.

[Tar55] A. Tarski. "A lattice-theoretical fixpoint theorem and its applications." *Pacific J. Math.*, **5**(2):285–309, 1955.

[TPS08] U. Topcu, A. Packard, and P. Seiler. "Local stability analysis using simulations and sum-of-squares programming." *Automatica*, **44**(10):2669 – 2675, 2008.

[Var08] M. Y. Vardi. *From Verification to Synthesis*, pp. 2–2. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[VHJ14] G. Vankeerberghen, J. Hendrickx, and R. Jungers. "JSR: A Toolbox to Compute the Joint Spectral Radius." In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pp. 151–156, New York, NY, USA, 2014. ACM.

[WTM13] E.M. Wolff, U. Topcu, and R.M. Murray. "Efficient reactive controller synthesis for a fragment of linear temporal logic." In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5033–5040, May 2013.

[ZB98] M. Zefran and J. W. Burdick. "Design of switching controllers for systems with changing dynamics." In *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)*, volume 2, pp. 2113–2118 vol.2, Dec 1998.