BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

FORMAL METHODS FOR RESILIENT CONTROL

by

SADRA SADRADDINI

B.S., Sharif University of Technology, 2013
M.S., Boston University, 2017

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2018

© 2018 by
SADRA SADRADDINI
All rights reserved

Approved by

First Reader

Calin A. Belta, PhD
Professor of Mechanical Engineering
Professor of Systems Engineering
Professor of Electrical and Computer Engineering
Professor of Bioinformatics

Second Reader

Christos G. Cassandras, PhD
Professor of Electrical and Computer Engineering
Professor and Head of Systems Engineering

Third Reader

Sean B. Andersson, PhD
Associate Professor of Mechanical Engineering
Associate Professor of Systems Engineering

Fourth Reader

Wenchao Li, PhD
Assistant Professor of Electrical and Computer Engineering

Light, regardless how faint it is, brings illumination at last.

Qaranqush

Acknowledgments

This dissertation would not have been possible without help and support of many people. First and foremost, I want to thank my excellent advisor Calin Belta for his great support and supervision. I learned a countless number of lessons from him. I also acknowledge my dissertation committee members for their useful suggestions during both my proposal and final defenses.

I also want to thank my collaborators. First, I acknowledge Professor Vijay Gupta for hosting my visit to the University of Notre Dame. He influenced me through a lot of fascinating research ideas. Second, I mention Professor Murat Arcak of UC Berkeley for his project leadership and collaboration. I praise my co-authors, and amazing friends at the same time, Sivaranjani (Notre Dame), Eric (UC Berkeley), and Janos and Iman (Boston University). I learned a lot from you through innumerable hours of discussion and debate.

I acknowledge my longtime Iranian friends, many of whom also study abroad, Babak, Ali, Behnam, Arsalan, Reza, Iman, Nima, Shahrooz, and Arian, for their support and friendship, which made years of studying overseas possible. Finally, I want to thank my family members in Iran and the US. I gratefully acknowledge my wonderful parents, twin sisters, and grandmother. I also sanctify the memory of my dear grandfather and uncle, who passed away while I was very distant from them. I will never forget our last gathering at Tabriz Airport in August 2013. Finally, I praise Fahime for her unconditional love and support during the years I spent for this work.

Sadra Sadraddini
December 2017,
Boston, MA

FORMAL METHODS FOR RESILIENT CONTROL

SADRA SADRADDINI

Boston University, College of Engineering, 2018

Major Professor: Calin A. Belta, PhD
Professor of Mechanical Engineering
Professor of Systems Engineering
Professor of Electrical and Computer Engineering
Professor of Bioinformatics

ABSTRACT

Many systems operate in uncertain, possibly adversarial environments, and their successful operation is contingent upon satisfying specific requirements, optimal performance, and ability to recover from unexpected situations. Examples are prevalent in many engineering disciplines such as transportation, robotics, energy, and biological systems. This thesis studies designing correct, resilient, and optimal controllers for discrete-time complex systems from elaborate, possibly vague, specifications.

The first part of the contributions of this thesis is a framework for optimal control of non-deterministic hybrid systems from specifications described by signal temporal logic (STL), which can express a broad spectrum of interesting properties. The method is optimization-based and has several advantages over the existing techniques. When satisfying the specification is impossible, the degree of violation - characterized by STL quantitative semantics - is minimized. The computational limitations are discussed.

The focus of second part is on specific types of systems and specifications for which controllers are synthesized efficiently. A class of monotone systems is introduced for which formal synthesis is scalable and almost complete. It is shown that hybrid macroscopic traffic models fall into this class. Novel techniques in modular verification and synthesis are employed for distributed optimal control, and their usefulness is shown for large-scale traffic management. Apart from monotone systems, a method is introduced for robust constrained control of networked linear systems with communication constraints. Case studies on longitudinal control of vehicular platoons are presented.

The third part is about learning-based control with formal guarantees. Two approaches are studied. First, a formal perspective on adaptive control is provided in which the model is represented by a parametric transition system, and the specification is captured by an automaton. A correct-by-construction framework is developed such that the controller infers the actual parameters and plans accordingly for all possible future transitions and inferences. The second approach is based on hybrid model identification using input-output data. By assuming some limited knowledge of the range of system behaviors, theoretical performance guarantees are provided on implementing the controller designed for the identified model on the original unknown system.

Contents

1 Introduction 1

1.1 Motivation 1

1.2 The Objectives of This Dissertation 3

1.3 Related Work 5

1.4 Organization and Highlights of Results 10

2 Systems and Specifications 15

2.1 Specifications 15

2.1.1 Linear Temporal Logic 16

2.1.2 Metric Temporal logic 18

2.1.3 Signal Temporal logic 20

2.2 Systems 21

2.2.1 Hybrid Models 22

2.2.2 Equivalency of Hybrid Models 23

I Resilient Control of Hybrid Systems 25

3 Deterministic Hybrid Systems 26

3.1 Problem Formulation 26

3.2 Mixed-Integer Formulation 28

3.3 Optimal Control 32

3.3.1 Finite-Horizon Semantics 32

3.3.2 Infinite-Horizon Semantics 33

3.4 Initial Conditions 34
    3.4.1 Feasibility Check 34
    3.4.2 Quantifier Elimination 34
3.5 Examples 35

4 Non-Deterministic Hybrid Systems 40

4.1 Problem Formulation 41
4.2 Tube STL Control 42
4.3 Tube Design 45
    4.3.1 Set-Parameterization 46
    4.3.2 Optimization 47
4.4 Example 48

II Systems and Specifications with Specific Structures 51

5 Monotone Systems and Traffic Networks 52

5.1 Problem Statement and Approach 54
5.2 Finite Horizon Semantics 60
5.3 Infinite Horizon Semantics 64
    5.3.1 Global specifications: s-sequences and invariance 64
    5.3.2 Bounded-global specifications: ϕ-sequences 68
    5.3.3 Necessity of Open-loop Strategies 72
5.4 Model Predictive Control 76
5.5 Application to Traffic Management 80
    5.5.1 Model 80
    5.5.2 Monotonicity 84
    5.5.3 Case Study 85

6 Contract-based Design 92

6.1 Traffic Network Model 93

6.2 Problem Statement 97

6.3 Network Partitioning 101

6.4 Contract Synthesis 105

6.4.1 Assume-Guarantee Contracts 106

6.4.2 Monotonicity 107

6.4.3 Language Realization Set 108

6.4.4 Controlled Invariance 109

6.4.5 Contracts 112

6.5 Control Synthesis 113

6.5.1 Decentralized Model Predictive Control 114

6.5.2 Cooperative Model Predictive Control 116

6.6 Simulation Results 117

6.6.1 Macroscopic Simulation 117

6.6.2 Microscopic Simulation 119

6.7 Dynamic Contracts 120

6.7.1 Motivating Example 121

6.7.2 Interconnections and Contracts 122

6.7.3 Recursive Feasibility with Fixed Contracts 124

6.7.4 Dynamic Contracts 125

6.7.5 Optimal Contract Coordination 128

6.7.6 Contract Mining 129

6.7.7 example 129

7 Robust Distributed Set-Invariance 132

7.1 Problem Formulation 133

7.1.1 Problem Statement 136
7.2 Parameterized Set-Invariance 137
7.2.1 Convex Parameterization 137
7.2.2 Centralized Policy 140
7.2.3 Linear Delay Policy 140
7.3 Control with Structural Constraints 142
7.4 Structure Design 143
7.4.1 Binary encoding 144
7.4.2 Graph Optimization 145
7.5 Examples 146
7.5.1 Coupled Double Integrators 146
7.5.2 Platooning 148

III Learning-based Formal Synthesis 152

8 Formal Methods for Adaptive Control 153

8.0.1 Transition Systems 154
8.0.2 Quotient Transition System 155
8.0.3 LTL Control 156
8.1 Problem Formulation and approach 158
8.2 Parametric Transition System 160
8.3 Control Synthesis for Finite Systems 161
8.3.1 Parameter Estimation 161
8.3.2 Adaptive Transition System 162
8.3.3 Control Synthesis 164
8.4 Control Synthesis for Infinite Systems 166
8.5 Case Studies 167

9 Formal Model Identification **172**

9.1 Problem Statement 174

9.1.1 System and Assumptions 174

9.1.2 STL Specification 177

9.1.3 Problem Formulation 177

9.2 Approach 178

9.3 Formal Model Identification 179

9.3.1 PWA Systems 180

9.3.2 PWA Fitting 181

9.3.3 Adjusting Disturbances 182

9.3.4 Computational Aspects 184

9.4 Example 185

10 Conclusions **188**

10.1 Summary of the thesis 188

10.2 Future Directions 189

A Appendix **191**

A.1 Bounded-Global Formula 191

References **194**

Curriculum Vitae **206**

List of Tables

5.1 Parameters of the network in Fig. 5·5 86

5.2 Existence of ϕ-sequences for the network in Fig. 5·5 88

5.3 ϕ-sequence in the case study 89

6.1 Computation Time per Time Step and Accumulated Delay for Different Control Policies 119

6.2 Accumulated Delays for Different Control Methods 130

7.1 Margins of Correctness for Graphs in Fig. 7·1 147

7.2 Margins of Correctness for Predecessor Following 150

List of Figures

2·1 Example 1: DRA corresponding to $\varphi = \mathbf{GF}\pi_1 \wedge \mathbf{F}\pi_2$, where $F_1 = \{s_0\}$ (red), $I_1 = \{s_2\}$ (green). Runs that visit the green state infinitely many times and visit the red state finitely many times satisfy φ. . . . 18

3·1 A car is moving toward a traffic light. If encountered by yellow light, the choice has to be made about stopping before the traffic light or clear the intersection before the traffic light turns red. 35

3·2 Example 5: The feasibility envelope of the yellow light dilemma problem is the green shaded area shown for different values of T. The traffic light is located at $s = 0$ and the end of the intersection is at $s = 10$. . 37

3·3 Example 5 : Two sample trajectories: (red) car driving at constant velocity $v = 1.9$ does not consider the feasibility envelope and is not guaranteed to be able to react properly to the yellow light. (blue) car driving inside the feasibility envelope with maximum speed. The car slows down before reaching the intersection ($s = -8$) but, in case of not encountering yellow light, starts to speed up at $s = 6$. 38

3·4 Example 3.19: [Top left] The vector field and the rectangular regions [Top right] Control-effort-optimal satisfying trajectory [Bottom left] Maximally satisfying trajectory [Bottom right] Minimally violating trajectory with $U = [-2, 2]$. 39

4·1 Example 7: The smallest tube generated by three hyper-rectangles. . 49

4·2 Example 7: [Left]: Tube-based design with all trajectories guaranteed to have STL score greater than 0. [Right]: Design corresponding to the maximum worst-case STL score 1.36. 50

5·1 Example 8: [Left] The trajectory of the maximal system f^* which satisfies the specification. [Right] For the same controls, the trajectory of the original system f with w drawn from an uniform distribution over $L(w^*)$. 64

5·2 Example 9: [Left] The trajectory satisfying the conditions of s-sequences. [Right] The corresponding robust control invariant set inside $\mathcal{S}$. . . . 68

5·3 Example 10: [Left] The first 19 points of the trajectory of the maximal system f^* that satisfy the conditions in Theorem 7. A sample trajectory of f is also shown. [Right] The ω-limit set (red dots) of f^* is a 7-periodic orbit. 72

5·4 The fundamental diagram. The flow out of link l drops if the number of vehicles on the immediate downstream link l' is close to its capacity. The congestion is defined by this blocking behavior. 83

5·5 Traffic management case study: A network of freeways and urban roads. There are 14 intersections controlled by traffic lights and 4 ramp meters. 85

5·6 Traffic management case study: [Top Row] the trajectory of the maximal system obtained from applying the open-loop control policy (5.12) with initial condition x_0^ϕ [Middle Row] robust MPC generated trajectory with zero initial condition with disturbances chosen uniformly $L(w^*)$ [Bottom Row] trajectory generated from applying the open-loop control policy (5.12) with zero initial conditions and the same disturbances as in [Middle Row]. 91

6·1 Traffic Network Topology 95

6·2 Network Partitioned into 4 Subnetworks 104

6·3 Simulation Results 119

6·4 Macroscopic Simulation Trajectories 120

6·5 VISSIM Implementation 120

6·6 Example network $\mathcal{N}$ partitioned into four sub-networks $\mathcal{N}^i, i = 1, \cdots, 4$. Contract obligations from (6.7.2) are depicted on the right. Solid arrows indicate an obligation from one network to an adjacent subnetwork to limit incoming vehicular flow. Dashed arrows represent recursive feasibility obligations from a sub-network to its future self. . 121

6·7 Example 16: [Top] State over Time. [Bottom] The traffic lights history for the links in South-Eastern Intersection of $\mathcal{N}^1$. The lower color is for the light corresponding to the link in North-South direction and the upper one stands for the link in East-West direction. The pedestrian liveness requirement (both lights simultaneously getting red) is satisfied in each round of contract transitions (shown by thick vertical block lines).131

7·1 Circular graphs: [Left]: directed [Right]: undirected 147

7·2 Projection of the RCI set on the state-space of s_1 and a sample trajectory of 60 time steps. RCI sets correspond to [Left]: directed [Right]: undirected graphs in Fig. 7·1. 148

7·3 Optimal communication graphs for different values of K (complexity of the controller), η (maximum magnitude of the allowed disturbances) and ϵ (the degree of dynamical couplings). "CT" stands for computation time. 149

7·4 The costs of incumbent feasible solutions versus time. 150

7·5 Optimal communication graphs for platooning. The leader is considered as an adversary with bounded acceleration range. 151

8·1 Example 18: [Top] A PTS with two possible parameters θ_1, θ_2, and the corresponding transition systems [Bottom] The corresponding ATS . 163
8·2 Example 19: (Left): The Robot (shown in black) and its environment. (Middle): Snapshots of the executed Motion at time $k = 33$, and (Right) $k = 62$. The robot satisfies the specification. 168
8·3 Example 20: trajectory of the system versus time, which is always between -1 and 1. 170
8·4 Example 20: Snapshots of ϑ_k at various times, which are illustrated by the shaded regions. They always contain the ground truth parameter $\theta_1^* = 0.45$, $\theta_2^* = 1.11$, $\theta_3^* = -0.18$. 171

9·1 Example 21: The magnitude of non-determinism may depend on the distribution of data. 184
9·2 Example 22: (Left): Regions of interest. (Right): Computed polyhedral partition from (9.10). 186
9·3 Example 22 a sample closed-loop trajectory 187
9·4 Example 22 histogram of STL scores. 187

List of Abbreviations

ATS	Adaptive Transition System
LP	Linear Programming
LRS	Language Realization Set
LTL	Linear Temporal Logic
MILP	Mixed Integer Linear Programming
MIQP	Mixed Integer Quadratic Programming
MLD	Mixed Logical Dynamical
MPC	Model Predictive Control
MTL	Metric Temporal Logic
NNF	Negation Normal Form
PTS	Parametric Transition System
PWA	Piecewise Affine
QP	Quadratic Programming
RCI	Robust Control Invariant
STL	Signal Temporal Logic

Chapter 1

Introduction

1.1 Motivation

The goal of control theory is to develop algorithms to operate dynamical systems in desired manners. Most controllers are based on mathematical models, which are often not accurate since many details are ignored and left as uncertainties. In a good model, uncertainties are bounded within limited ranges. A central idea in control theory is to use persistent feedback to handle the mismatches between the model and the actual system, which are typically small since models predict the behaviors of actual systems reasonably well.

The theoretical foundations of feedback mechanisms and stabilization date back to 19th century. Since then, a lot of success have been achieved in automatic control, such as remarkable milestones in aerospace industry. Many paradigms in control theory - such as optimal control, robust control, and adaptive control - became mature by the late 20th century. Despite remarkable achievements, a new realm of challenges emerged in the late 20th century and has rapidly grown in the third millennium. The difficulty is dealing with systems that are increasingly complex, large in scale, and more interestingly - from my point of view - specified with objectives that are more complex than stability. The main reason that these challenges have emerged lately are recent technological advances in computation, sensing, and communication, which collectively have made automation possible in many complex systems that were mainly operated by humans before. Self-driving cars, for example, are now a possibility because of significant improvements in artificial intelligence and perception techniques. The objectives of self-driving cars are way more complicated than stability: they have to safely navigate through roads while avoiding obstacles and other vehicles (possibly driven by careless humans), respect traffic rules, and minimize energy consumption and travel times. On a broader scale, traffic management is a serious issue in many cities, where the infrastructure has to be utilized in the best possible way, but also a lot of complicated constraints exists because of traffic signals, pedestrians, and safety considerations. As many systems operate in uncertain environments, *resilience* is a necessity in the sense that the system has to achieve successful performance for all possible phenomena modeled as uncertainties, which are often adversarial. For instance, it is crucial to guarantee that no rear-end collisions occur in platoons of autonomous cars no matter how disturbances, like engine fluctuations and wind gusts, hit the system. It is not as relevant to ensure optimized H_2 or H_∞ performance in the Hardy space of system's frequency-domain representation, a well-studied problem in classical robust control.

The mentioned developments require rigorous mathematical techniques to be employed. For this reason, we have observed a growing trend of using *formal methods* [Baier and Katoen, 2008] in control theory. Formal methods were originally developed in the computer science community to reason about executions of software and digital circuits. These systems are often simple, but the specifications are complex, usually described by a version of temporal logics [Emerson, 1990], which is a natural framework to specify a broad range of behaviors such as safety, liveness, sequentiality, and their elaborate combinations. One strong argument for using formal methods in control engineering is providing formal certificates on system performances, which is often characterized by constraints. In fact, without using formal methods, one may have to test a system with infinite states (e.g., any system with continuous state space) infinitely many times to guarantee that a certain requirement is verified. Ideally, formal approaches in control theory are aimed to find tools and algorithms for controller design such that the following properties hold:

- **Correctness**: Given a system model with bounded uncertainties, an initial condition, and a specification over its trajectories, the specification is satisfied by all possible trajectories allowed within the uncertainty bounds.
- **Completeness:** No conservatism is introduced in the algorithm for controller design. If a controller exists, the algorithms should be able to find it. Also, the set of all admissible initial conditions from which correct controllers exist is relevant and has to be computed.
- **Optimality:** If more than one controller ensuring correctness exist (which is often the case), find the optimal one subject to a given cost criterion. A natural cost function is the degree of specification satisfaction - a notion that is formalized in Section 2.1.3. In particular, when correctness is not possible, finding a controller with the least amount of specification violation is an optimal control problem of interest.

1.2 The Objectives of This Dissertation

Achieving all the objectives listed above is formidable even for small systems and simple specifications. It is necessary to trade-off some of the objectives in the favor of others. A brief overview of the existing techniques on formal methods and control theory is provided in Section 1.3. In this thesis, no compromise is given on correctness. The controllers have to be correct-by-design - no heuristics is considered. If the developed algorithms are unable to find a controller guaranteeing correctness, the least violating ones are designed, with a certificate on the maximum amount of violation possible. The violation is characterized by introducing a quantitative notion of distance to satisfaction. Completeness is very important, but may be computationally intractable to achieve. The same rule applies to optimality. Suboptimal but computationally simple solutions are preferred.

Despite important advances in addressing the objectives mentioned earlier, the current techniques suffer from a number of serious drawbacks. The contributions of this thesis are formal methods solutions to the following problems:

- **Robust optimal control**: the literature on connection between formal methods and robust optimal control is slim. The existing methods are severely conservative, naive, and not resilient enough, especially for infinite-time specifications. The first part of this thesis is devoted to developing a framework for control of non-deterministic hybrid systems from temporal logic specifications.

- **Large-scale control:** a severe limitation in formal synthesis - and an argument often made against using formal methods in control theory - is scalability. Most existing techniques are not applicable to systems beyond, roughly, 5-6 dimensions. However, by taking advantage of the specific structure of the system and its specification, scalable methods are sought. Moreover, formal synthesis of distributed controllers - where a specific information flow structure is imposed on a network of controlled subsystems - is an open problem. The second part of this thesis concerns large-scale resilient synthesis for systems and specifications with special structures, with a strong emphasis on transportation applications.

- **Learning-based Control:** in many systems and specification, an accurate (simple) model is not available in practice. Thus, designing controllers with formal performance guarantees is challenging. A formal perspective on combining learning-based control and formal methods is lacking. In particular, in many safety-control systems, the specification should not be violated even during learning. Thus, approaches based on reinforcement learning are inappropriate. The third, and the final part of this thesis, provides a formal perspective on adaptive control and data-driven synthesis.

1.3 Related Work

Control technqiues in the spirit of formal methods date back to 1970s, when finite-time and infinite-time safety and reachability control of systems with set-valued disturbances were studied [Bertsekas and Rhodes, 1971, Bertsekas, 1972]. Reachability and safety are specifically important in safety-critical systems [Tomlin et al., 1998, Mitchell et al., 2005]. The robotics community, in particular, was attracted by the formalism offered in formal methods for specifying robotics missions, and correct-by-construction methods for motion planning [Bacchus and Kabanza, 2000, Fainekos et al., 2005].

Controlling systems described by differential or difference equations from temporal logic specifications caught the attention of control theorists in 2000s. In particular, significant attention was devoted to systems that involve both continuous and discrete behaviors (hybrid systems), where a lot of traditional control techniques fail. Followed by remarkable results on timed and hybrid automata in [Alur and Dill, 1994, Henzinger, 2000], and finite-state abstractions [Alur et al., 2000], automata-based control synthesis became a popular research direction. The idea was to represent continuous control systems by finite-state transition systems, where the connection between the two was formally established using simulation or bisimulation relations [Milner, 1989]. It was shown that all the possible executions of an infinite-state system can be contained in those of a finite system. In the case of bisimulation, these are equivalent, introducing no conservativeness. The specification is often captured by a finite-state automaton with a specific accepting condition. The synthesis problem, which yields the control strategy, becomes solving a Buchi or Rabin game [Grädel et al., 2002] on the product of the transition system and the specification automaton.

Control synthesis for linear and piecewise affine systems from linear temporal logic (LTL) specifications was studied in [Tabuada and Pappas, 2006, Kloetzer and Belta, 2008, Yordanov et al., 2012, Gol et al., 2014]. Approximate finite bisimulation quotients for nonlinear systems were investigated in [Pola et al., 2008, Pola and Tabuada, 2009, Zamani et al., 2012]. The main limitations of finite abstraction are the large computational burden of discretization in high dimensions and conservativeness when exact bisimulations are impossible or difficult to construct. More recently, the authors in [Coogan and Arcak, 2015] provided an efficient method to compute finite abstractions for mixed-monotone systems (a more general class than monotone systems). The authors in [Kim et al., 2017b] exploited monotonicity for compositional LTL control. While the approaches in [Coogan and Arcak, 2015, Kim et al., 2017b] are efficient for the computation of transitions, they still require state-space discretization, which is a severe limitation in high dimensions. Moreover, they are conservative since the finite abstractions are often not bisimilar with the original system.

Automata-based approaches provide Boolean answers to the existence of controllers. If an automata-based approach fails to find a control policy, there is no direct way to find a controller that is minimally violating. While some works have introduced violation metrics to automata-based synthesis [Tumova et al., 2013, Lahijanian et al., 2015], they are specifically tailored to applications in mobile robotics. Probabilistic methods, which are outside of the scope of this thesis, are an alternative way to introduce resilience into formal synthesis [Abate et al., 2008, Lahijanian et al., 2010, Fu and Topcu, 2015, Svoreňová et al., 2015, Sadigh and Kapoor, 2015, Mehr et al., 2017]. However, probabilistic characterization of the system dynamics has to be given in prior. Furthermore, specific difficulties arise in finite abstraction of continuous probabilistic systems (e.g., see [Abate et al., 2011]) and probabilistic control from infinite-time objectives [Chatterjee et al., 2016, Ehlers et al., 2016].

Receding-horizon implementations were used to combine optimal control and automata-based synthesis [Wongpiromsarn et al., 2010, Gol and Belta, 2014]. As an alternative approach, LTL optimization-based control of mixed-logical dynamical (MLD) systems [Bemporad and Morari, 1999] using mixed-integer programs was introduced in [Karaman et al., 2008, Wolff et al., 2014], and was recently extended to model predictive control (MPC) from signal temporal logic (STL) specifications in [Raman et al., 2014, Raman et al., 2015]. However, these approaches are unable to guarantee infinite-time properties like safety, and the results are fragile in the presence of disturbances. In order to recover from infeasibility in MPC optimization problems, specification modification [Ghosh et al., 2016] was proposed. However, without a formal perspective on resilience, it is not possible to provide a certificate on how bad a controller can perform. Moreover, the current MPC approaches are based on minimax optimization problems which are open-loop and severely conservative.

In some applications, the structural properties of the system and the specification can be exploited to consider alternative approaches to formal control synthesis. A class of these systems are monotone systems, in which the evolution of the state exhibits a type of *order preserving law*. Monotonicity is common in models of transportation, biological, and economic systems [May, 2007, Como et al., 2014, Coogan et al., 2016a, Kim et al., 2016]. Such systems are also *positive* in the sense that the state components are always non-negative. Control of positive systems have been widely studied in the literature [Haddad et al., 2010, Rantzer, 2011, De Leenheer and Aeyels, 2001]. Positive linear systems are always monotone [Rantzer, 2011]. Monotone dynamical systems have been extensively investigated in the mathematics literature [Hirsch, 1985, Hirsch, Morris W, Smith, 2005, Smith, 2008]. Angeli and Sontag [Angeli and Sontag, 2003] extended the notion of monotonicity to deterministic continuous-time control systems and provided results on interconnections of these systems. However, they assumed monotonicity with respect to both state and controls, which is very restrictive. Safety control of cooperative systems was investigated in [Hafner and Del Vecchio, 2011, Ghaemi and Del Vecchio, 2014, Meyer et al., 2016]. However, these work, like [Angeli and Sontag, 2003], assumed monotonicity with respect to the control inputs as well. Computational benefits gained from monotonicity for reachability analysis of hybrid systems were highlighted in [Ramdani et al., 2010].

An interesting large-scale control problem is traffic management. Due to computational complexity, traditional optimal control techniques such as Hamilton-Jacobi-Bellman (HJB) equations are not suitable for traffic control. Traffic-responsive strategies employ approximate dynamic programming methods such as model predictive control (MPC) for real-time optimization. Notable implementations are SCOOT [Hunt et al., 1981], OPAC [Gartner, 1983] and UTOPIA [Mauro and Di Taranto, 1990]. Other approaches include infinitesimal perturbation analysis [Geng and Cassandras, 2015, Fleck et al., 2016], stable MPC for freeway ramp metering [Koehler et al., 2016], control based on convex relaxations [Lovisari et al., 2014, Como et al., 2016], and disturbance localization [Sivaranjani et al., 2015, Sivaranjani et al., 2017]. However, real-time optimization is not possible beyond small-scale systems. Totally decentralized methods optimize the controls of each intersection individually but can cause gridlocks in the network [Gregoire et al., 2015]. Hierarchical distributed control architectures [Mirchandani and Head, 2001], while alleviating the real time computational complexity, are not able to formally guarantee global behaviors such as avoidance of traffic jams. We desire a method in which, while control decisions are optimized locally, desired specifications are *guaranteed* globally.

Formal methods for large-scale control is a growing research direction. The main idea is to divide a large system into smaller subsystems. Ideas based on compositional synthesis [Rungger and Zamani, 2015, Alur et al., 2016, Kim et al., 2015], small gain theorem [Dallal and Tabuada, 2015, Kim et al., 2017a], and assume-guarantee reasoning [Henzinger et al., 1998] were used to design controllers individually. Compositional methods, ideally and under some assumptions, apply to arbitrarily large systems. However, searching for contracts for interconnections has a large computational price and the existing methods are often excessively conservative as the search is performed over limited families of contracts [Kim et al., 2016]. Separable control invariant sets [Raković et al., 2010, Nilsson and Ozay, 2016] is an alternative used for decentralized infinite-time safety control. Controlling based on contracts leads to decentralized controllers, which do not take advantage of communication between subsystems. Numerous methods have been proposed to design static feedback gains that respect communication constraints or lead to sparse ones [Lin et al., 2013, Fardad and Jovanovic, 2014, Fattahi et al., 2015, Arastoo et al., 2016, Fazelnia et al., 2017, Fattahi et al., 2017]. However, these methods are unable to take state and input constraints into account while disturbances are also present. They do not allow correct-by-design constraint satisfaction, and one has to test the stabilizing controller to see whether they fulfill the constraints. This process can be expensive. The authors in [Summers and Lygeros, 2012, Conte et al., 2012, Wang and Ong, 2017] studied distributed MPC of linear systems subject to polyhedral and communication constraints, but disturbances were not included, which significantly eases computations. The authors in [Furieri and Kamgarpour, 2017] studied set-invariance for disturbed networks of linear systems with communication constraints, but finite-time guarantees were obtained.

Any guarantee in formal synthesis is valid as long as the model is valid. However, in many engineering applications, perfect models are not available or are too complex to use for synthesis purposes. One way to approach a model-free synthesis problem is to adopt learning-based control methods. For example, [Sadigh et al., 2014, Aksaray et al., 2016, Li et al., 2017] studied reinforcement learning from temporal logic specifications. However, a completely model-free approach does not provide any formal guarantee since it is always possible to observe a new behavior from the system that may cause the specification to be violated. To overcome this issue, some works proposed considering a (highly non-deterministic) model that contains all the possible behaviors of the yet-to-be-learned system. Thus, the state-space is safely explored - in the sense that a temporal logic specification is respected - while a possibly more accurate model and better performance may be obtained during learning [Aswani et al., 2013, Alshiekh et al., 2017]. In many applications, availability of a model is asking for too much information. For example, the work in [Aswani et al., 2013] assumes the prior system to be linear with all unknown non-linearities contained in a known polytope, which acts as a set-valued additive disturbance. In this setting, the values representing both the linear model and the disturbance polytope have to be known beforehand.

1.4 Organization and Highlights of Results

This dissertation is organized as follows.

Chapter 2: Preliminaries

First, the necessary background on temporal logics, its variants, and examples illustrating them are provided. Next, discrete-time hybrid systems and formalism of their various forms are given.

Chapter 3: Deterministic Hybrid Systems

A framework is developed for optimal control of deterministic mixed-logical dynamical (MLD) systems from signal temporal logic (STL) specifications. Both finite-time and infinite-time specifications are considered and their solution properties are discussed. The main contribution is introducing an efficient framework for encoding STL quantitative semantics into the optimal control problem. Therefore, trajectories with the highest degree of satisfaction can be computed.

Chapter 4: Non-Deterministic Hybrid Systems

Hybrid systems with additive polytopic disturbances are considered. These models are ubiquitous in modeling nonlinear systems with an arbitrary degree of precision. A novel method based on tube model predictive control (MPC) is introduced which provides firm lower-bounds on the degree of STL satisfaction - for both finite-time and infinite-time specifications.

Chapter 5: Monotone Systems and Traffic Management

A class of monotone hybrid systems is introduced in which the partial order is defined on the positive orthant, state components are always non-negative, a unique maximal disturbance exists, and specifications encourage smaller values for state components. It is shown that traffic models fulfill these assumptions. We prove open-loop control policies are (almost) necessary and sufficient to guarantee STL satisfaction. The computational benefits and usefulness of applying the theoretical results on mixed urban-freeway traffic management are illustrated.

Chapter 6: Contract-based Design

Control of Large-scale systems is considered. The focus is, again, on monotone systems and traffic management. We explain how to partition large networks into smaller ones, efficiently find contracts characterizing the behavior of interconnections, and synthesize controllers in a distributed way in an MPC fashion. Two approaches are taken. In first, contracts are found by decomposing robust control invariant sets in an augmented form of state-control-space. These contracts are fed to each subsystem's MPC constraints. Second, an approach based on contract mining and compositional synthesis is studied. A library of contracts is found, where a central supervisor assigns the optimal contracts to the subsystems. We show through simulations that dynamic contracts achieve better performance than fixed contracts.

Chapter 7: Distributed Robust Set-Invariance

Here we consider large networked linear systems with polytopic additive disturbances and specifications described as infinite-time polytopic invariance. This problem arises in many safety-critical applications where hard state and control constraints are important. We consider two problems. First, given a communication graph characterizing the structure of the information flow in the network, we find the optimal distributed control policy by solving a single linear program. Second, we find the sparsest communication graph required for the existence of a distributed invariance-inducing control policy using mixed-integer linear program. Illustrative examples, including one on vehicular platooning, are presented.

Chapter 8: Formal Methods for Adaptive Control

We consider a discrete-time system with constant but initially unknown parameters. The task is to control the system from linear temporal logic (LTL) specifications. We introduce the notions of non-deterministic parametric and adaptive transition systems and show how to use tools from automata-based synthesis to compute adaptive control strategies for finite systems. For infinite systems, we first compute abstractions in the form of parametric finite quotient transition systems and then apply the techniques for finite systems. Unlike traditional adaptive control methods, the approach is correct by design, does not require a reference model, and can deal with a much broader range of systems and specifications. Illustrative case studies show that provably correct adaptive control strategies significantly enhance resilience.

Chapter 9: Formal Model Identification

For many control systems, an accurate (simple) model is asking for too much information. Thus, designing controllers with formal performance guarantees is challenging. A framework is developed to use input-output data from an unknown system to synthesize controllers from signal temporal logic (STL) specifications. First, by imposing mild assumptions on system continuity, in a Lipschitz sense, we find a set-valued piecewise affine (PWA) model that contains all the possible behaviors of the concrete system. Next, we use tube MPC from chapter 3. Lower-bound certificates are provided on the degree of STL satisfaction of the closed-loop concrete system. Illustrative examples are presented.

Chapter 10: Concluding Remarks

This chapter summarizes the results of this thesis and outlines future research directions.

Preliminaries

Chapter 2

Systems and Specifications

In this chapter, we provide the necessary background on linear-time specifications, and models for discrete-time systems.

2.1 Specifications

The sets of real, non-negative real, natural numbers, and Boolean values are denoted by $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{N}$, and $\mathbb{B}$ respectively. The empty set is denoted by $\emptyset$. Given a set S, we use $|S|$, 2^S, $2^S_{-\emptyset}$ to denote its cardinality, power set, and power set excluding the empty set, respectively. All time intervals are interpreted in discrete-time: $[t_1, t_2] = \{t_1, \cdots, t_2\}$, $t, t_1, t_2 \in \mathbb{N}$, $t_1 \leq t_2$.

An alphabet $\mathcal{A}$ is a finite set of symbols $\mathcal{A} = \{a_1, a_2, \cdots, a_A\}$. A finite (infinite) word is a finite-length (infinite-length) string of symbols in $\mathcal{A}$. For example, $\sigma_1 = aba$ is a finite word, and $\sigma_2 = a(b)^\omega$ and $\sigma_3 = b(ab)^\omega$ are infinite words over $\mathcal{A} = \{a, b\}$, where ω stands for infinitely many repetitions. We use $\mathcal{A}^*$ and $\mathcal{A}^\omega$ to denote the set of all finite and infinite words that can be generated from $\mathcal{A}$, respectively. Given an infinite word $\sigma = a_0 a_1 a_2 \cdots$, we use the following notations to refer to specific parts of σ: $\sigma_t := a_t$, $\sigma[t_1 : t_2] := a_{t_1} a_{t_1+1} \cdots a_{t_2}$, and $(\sigma, t) := a_t a_{t_1+1} \cdots$ (also called a suffix).

2.1.1 Linear Temporal Logic

Linear temporal logic (LTL) was introduced in [Pnueli, 1977]. Its syntax is recursively defined over a set of atomic propositions Π as follows:

$$\varphi ::= \text{True} \mid \pi \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \, |\mathbf{X}\varphi| \, \varphi_1 \mathbf{U} \varphi_2, \tag{2.1}$$

where $\pi \in \Pi$ is an atomic proposition, φ_1, φ_2, and φ are LTL formulas, $\neg$ stands for negation, $\wedge$ for conjunction, and $\mathbf{X}$, $\mathbf{U}$, are temporal "next", and "until" operators, respectively. Additional Boolean and temporal operators can be derived follows:

- Boolean "disjunction": $\varphi_1 \vee \varphi_2 := \neg(\neg\varphi_1 \wedge \neg\varphi_2)$;
- temporal "eventually": $\mathbf{F}\varphi := \text{True}\mathbf{U}\varphi$;
- temporal "always": $\mathbf{G}\varphi := \neg(\mathbf{F}\neg\varphi)$.

LTL semantics is interpreted over suffixes of infinite words over 2^{Π}:

- $(\sigma, t) \models \pi \Leftrightarrow \pi \in \sigma_t$;
- $(\sigma, t) \models \neg\varphi \Leftrightarrow (\sigma, t) \not\models \varphi$;
- $(\sigma, t) \models \varphi_1 \wedge \varphi_2 \Leftrightarrow (\sigma, t) \models \varphi_1 \wedge (\sigma, t) \models \varphi_2$;
- $(\sigma, t) \models \mathbf{X}\varphi \Leftrightarrow (\sigma, t+1) \models \varphi$;
- $(\sigma, t) \models \varphi_1 \mathbf{U} \varphi_2 \Leftrightarrow \exists t' \geq t \text{ s.t. } (\sigma, t') \models \varphi_2, (\sigma, t'') \models \varphi_1 \forall t'' \in [t, t')$;

where $\models$: LTL $\times (2^{\Pi})^{\omega} \times \mathbb{N} \to \{\text{True}, \text{False}\}$ is the satisfaction function. We write $(\sigma, t) \models \varphi$ when $\models (\varphi, \sigma, t) = \text{True}$. For the derived operators, the semantics is

- $(\sigma, t) \models \varphi_1 \vee \varphi_2 \Leftrightarrow (\sigma, t) \models \varphi_1 \vee (\sigma, t) \models \varphi_2$;
- $(\sigma, t) \models \mathbf{F}\varphi \Leftrightarrow \exists t' > t \text{ s.t. } (\sigma, t) \models \varphi$;

- $(\sigma, t) \models \mathbf{G}\varphi \Leftrightarrow \forall t' > t$ s. t. $(\sigma, t) \models \varphi$.

Definition 1. The *language* of an LTL formula, denoted by $L(\varphi)$, is defined as:

$$L(\varphi) := \{\sigma \in (2^{\Pi})^{\omega} \big| (\sigma, 0) \models \varphi\}. \tag{2.2}$$

Definition 2. A Deterministic Rabin Automaton (DRA) is defined as the tuple $\mathcal{R} = (S, s^0, \mathcal{A}, \alpha, \Omega)$, where:

- S is a set of states;
- s^0 is the initial state;
- $\mathcal{A}$ is a finite set of inputs (alphabet);
- α is a transition function $\alpha : S \times \mathcal{A} \to S$;
- $\Omega = \{(F_1, I_1), \cdots, (F_r, I_r)\}$ is a finite set of pairs of sets of states, where $F_i, I_i \subset S, i = 1, \cdots, r$.

An infinite word $w \in \mathcal{A}^{\omega}$ determines a sequence of inputs for $\mathcal{R}$ that results in the *run* $\zeta(w) = s_0 s_1 \cdots$, where $s_{k+1} = \alpha(s_k, a_k)$, $s_0 = s^0$, and a_k is the k'th input appearing in w. We define $Inf(\zeta) = \{s | s \text{ appears infinitely often in } \zeta\}$. A run ζ is *accepted* by $\mathcal{R}$ if there exists $i \in \{1, \cdots, m\}$ such that $Inf(\zeta) \cap F_i = \emptyset$ and $Inf(\zeta) \cap I_i \neq \emptyset$. In other words, F_i is visited finitely many times and I_i is visited infinitely often for some i. The language of $\mathcal{R}$, denoted by $L(\mathcal{R})$, $L(\mathcal{R}) \subset \mathcal{A}^{\omega}$, is defined as the set of all elements in $\mathcal{A}^{\omega}$ that produce accepting runs.

It is known that given an LTL formula φ over Π, one can construct a DRA $\mathcal{R}_{\varphi}$ with input set $\mathcal{A} = 2^{\Pi}$ such that $L(\mathcal{R}_{\varphi}) = L(\varphi)$ [Grädel et al., 2002]. Therefore, verifying whether an infinite word satisfies an LTL formula becomes equivalent to checking the Rabin acceptance condition. There exists well-established algorithms and software for this procedure [Klein and Baier, 2006].

Example 1. Consider $\varphi = \mathbf{GF}\pi_1 \wedge \mathbf{F}\pi_2$, which is an LTL formula over $\Pi = \{\pi_1, \pi_2\}$, stating that "π_1 holds infinitely often, and π_2 eventually holds". The DRA $\mathcal{R}_{\varphi}$

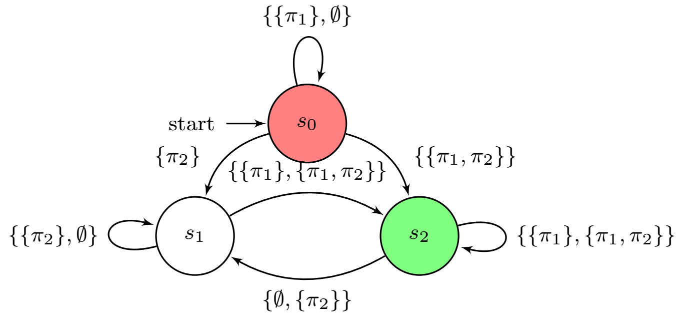

$\{\{\pi_1\},\emptyset\}$
start
s_0
$\{\pi_2\}$
$\{\{\pi_1\},\{\pi_1,\pi_2\}\}$
$\{\{\pi_1,\pi_2\}\}$
$\{\{\pi_2\},\emptyset\}$
s_1
s_2
$\{\{\pi_1\},\{\pi_1,\pi_2\}\}$
$\{\emptyset,\{\pi_2\}\}$

Figure 2·1: Example 1: DRA corresponding to $\varphi = \mathbf{GF}\pi_1 \wedge \mathbf{F}\pi_2$, where $F_1 = \{s_0\}$ (red), $I_1 = \{s_2\}$ (green). Runs that visit the green state infinitely many times and visit the red state finitely many times satisfy φ.

corresponding to this formula is illustrated in Figure 2·1. For example, we have $\{\pi_2\}\overline{\{\pi_1,\pi_2\}} \models \varphi$ (φ is satisfied), but $\overline{\{\pi_1\}} \not\models \varphi$ (φ is violated since π_2 never appears), and $\{\pi_1\}\overline{\emptyset\{\pi_2\}} \not\models \varphi$ (because π_1 does not hold infinitely often).

2.1.2 Metric Temporal logic

Metric temporal logic (MTL) was introduced in [Koymans, 1990]. Though it was originally proposed for continuous-time systems, its semantics still applies to discrete-time setting. The main ability in MTL is specifying time bounds for temporal operators, for which the semantics is naturally defined as:

- $(\sigma, t) \models \varphi_1 \mathbf{U}_{[t_1,t_2]}\varphi_2 \Leftrightarrow \exists t' \in [t_1, t_2]$ s. t. $(\sigma, t') \models \varphi_2 \ \wedge \ \forall t' \in [t_1, t), (\sigma, t') \models \varphi;$
- $(\sigma, t) \models \mathbf{F}_{[t_1,t_2]}\varphi \Leftrightarrow \exists t' \in [t_1, t_2]$ s. t. $(\sigma, t') \models \varphi;$
- $(\sigma, t) \models \mathbf{G}_{[t_1,t_2]}\varphi \Leftrightarrow \forall t' \in [t_1, t_2], (\sigma, t') \models \varphi,$

where $t, t_1, t_2 \in \mathbb{N}$, $t_1 \leq t_2$. MTL allows punctual operators (e.g., $[t_1, t_1]$ as interval) and also unbounded operators (e.g., $[t_1, \infty]$ as interval). Since the time is discrete, there is no theoretical distinction between LTL and MTL as any MTL formula can be converted into an LTL formula using LTL's "next" operator. However, representation of formulas will be very inefficient. For example, we have the following:

$$\mathbf{F}_{[2,5]}\varphi = \mathbf{XX}\varphi \vee \mathbf{XXX}\varphi \vee \mathbf{XXXX}\varphi \vee \mathbf{XXXXX}\varphi.$$

Definition 3. The bound of an MTL formula φ, denoted by b^φ, is defined as the time required to decide the satisfaction of φ, which is recursively computed as [Dokhanchi et al., 2014]:

- $b^\pi = 0$;
- $b^{\varphi_1 \wedge \varphi_2} = b^{\varphi_1 \vee \varphi_2} = \max(b^{\varphi_1}, b^{\varphi_2})$;
- $b^{\mathbf{F}_{[t_1,t_2]}\varphi} = b^{\mathbf{G}_{[t_1,t_2]}\varphi} = t_2 + b^\varphi$;
- $b^{\varphi_1 \mathbf{U}_{[t_1,t_2]} \varphi_2} = t_2 + \max(b^{\varphi_1}, b^{\varphi_2})$.

The satisfaction of φ by (σ, t) is decided only by $\sigma[t : t + b^\varphi]$, and the rest of the word is irrelevant. Therefore, instead of $(\sigma, t) \models \varphi$, we occasionally write $\sigma[t : t + b^\varphi] \models \varphi$ with the same meaning.

Definition 4. An MTL formula φ is *bounded* if $b^\varphi < \infty$.

Definition 5. An MTL formula is in negation normal form (NNF) if all negation operators appear immediately before atomic propositions.

It can be shown that any LTL/MTL formula can be brought into NNF (see, e.g., [Baier and Katoen, 2008]).

Definition 6 ([Ouaknine and Worrell, 2006]). A *safety MTL* formula is an MTL formula that when written in NNF, in which all "until" and "eventually" intervals are bounded.

Safety formulas are practically ubiquitous. Any MTL formula containing unbounded "eventually" or "until" operators can be approximated by an safety MTL formula by under-approximating the unbounded intervals by bounded intervals. However, bounded under-approximation is not sound for the unbounded "always" operator. A safety formula can be satisfied (respectively, violated) with infinite-length (respectively, finite-length) signals [Ouaknine and Worrell, 2006].

2.1.3 Signal Temporal logic

Signal Temporal Logic (STL) is a variant of MTL in which atomic propositions are replaced by predicates over real signals - making it more suitable for systems and control applications. An n-dimensional real signal is $\mathbf{s} = s_0 s_1 \cdots, \mathbf{s} \in (\mathbb{R}^n)^\omega$. A predicate over s is in the form $(p(s) \geq 0)$, where $p : \mathbb{R}^n \to \mathbb{R}$. We have

$$(\mathbf{s}, t) \models (p(s) \geq 0) \Leftrightarrow s_t \geq 0.$$

The rest of STL semantics is similar to those of MTL. STL was also originally introduced in [Maler and Nickovic, 2004] to reason about continuous-time real signals. However, similar to MTL, its semantics still applies to discrete-time. The main ingredient that makes STL popular is its quantitative semantics, defined as follows.

Definition 7. Given predicates on $\mathbb{R}^n$, the *STL score* is a function $\rho : (\mathbb{R}^n)^\omega \times \mathrm{STL} \times \mathbb{N} \to \mathbb{R}$, which is recursively defined as:

- $\rho(\mathbf{s}, f(s) \geq 0, t) = f(s_t)$;
- $\rho(\mathbf{s}, \varphi_1 \wedge \varphi_2, t) = \min(\rho(\mathbf{s}, \varphi_1, t), \rho(\mathbf{s}, \varphi_2, t)$;
- $\rho(\mathbf{s}, \varphi_1 \vee \varphi_2, t) = \max(\rho(\mathbf{s}, \varphi_1, t), \rho(\mathbf{s}, \varphi_2, t)$;
- $\rho(\mathbf{s}, \mathbf{F}_{[t_1,t_2]}\varphi, t) = \max_{k\in[t_1,t_2]} \rho(\mathbf{s}, \varphi, t)$;
- $\rho(\mathbf{s}, \mathbf{G}_{[t_1,t_2]}\varphi, t) = \min_{k\in[t_1,t_2]} \rho(\mathbf{s}, \varphi, t)$;
- $\rho(\mathbf{s}, \varphi_1 \mathbf{U}_{[t_1,t_2]}\varphi_2, t) = \max_{t'\in[t_1,t_2]} \min \left\{\rho(\mathbf{s}, \varphi, t'), \min_{t''\in[t_1,t)} \rho(\mathbf{s}, \varphi, t'')\right\}$.

As implied by Definition 7, STL score is a measure indicating how strongly a formula is satisfied by a signal. Positive (respectively, negative) robustness indicates satisfaction (respectively, violation) of the formula. STL score function has the following distance property [Maler and Nickovic, 2004]:

$$\big|\rho(\mathbf{s}, \varphi, k) - \rho(\mathbf{s}', \varphi, t)\big| \leq \sup_{k>t} \|s_k - s'_k\|_\infty. \tag{2.3}$$

The following definition is useful:

Definition 8. Given a STL formula φ with predicates on $\mathbb{R}^n$, the *language realization set*, denoted by $LRS(\varphi)$ is defined as:

$$LRS(\varphi) = \left\{\zeta \in \mathbb{R}^{n(b^\varphi+1)} | \zeta[0 : b^\varphi] \models \varphi\right\} \tag{2.4}$$

Example 2. Consider signal $\mathbf{r} \in \mathbb{R}^\omega$, where $r_k = k, k \in \mathbb{N}$, and $\pi = (r^2 \leq 10)$. We have $\rho(\mathbf{r}, \mathbf{G}_{[0,3]}\pi, 0) = \min(10-0, 10-1, 10-4, 10-9) = 1$ (satisfaction) and $\rho(\mathbf{r}, \mathbf{F}_{[4,6]}\pi, 0) = \max(10-16, 10-25, 10-36) = -6$ (violation).

2.2 Systems

Definition 9. A transition system is defined as a tuple $(X, U, \rightarrow)$, where

- X is the set of states;
- U is the set of controls;
- $\rightarrow \in X \times U \times X$ is the set of transitions.

A transition system is blocking if for some $(x, u) \in X \times U$, $\nexists x^+ \in X$ such that $(x, u, x^+) \in \rightarrow$. A transition system is *deterministic* if $(x, u, x^+), (x, u, y^+) \in \rightarrow$, then $x^+ = y^+$.

An alternative formalism of discrete-time systems is adopting the following form:

$$x_{t+1} = F(x_t, u_t, w_t), \tag{2.5}$$

where t, $t \in \mathbb{N}$ is time, and

- $x_t \in X$ is the state, $X \subseteq \mathbb{R}^{n_r} \times \mathbb{B}^{n_b}$;

- $u_t \in U$ is the control input, $U \subseteq \mathbb{R}^{m_r} \times \mathbb{B}^{m_b}$;
- $w_t \in W$ is the disturbance (environment input), $W \subseteq \mathbb{R}^{q_r} \times \mathbb{B}^{q_b}$.

Note that both X and U may include real and binary values. For instance, the set of controls in the traffic model developed in Sec. 5.5 includes binary values for decisions on traffic lights and real values for ramp meters. We usually assume that X, U are bounded, unless stated otherwise.

Definition 9 is more general than (2.5). We can embed (2.5) in a transition system $(X, U, \rightarrow)$, where $(x, u, x^+) \in \rightarrow$ if and only if $\exists w \in W \text{ s.t. } x^+ = F(x, u, w)$.

2.2.1 Hybrid Models

Definition 9 can be used to embed a broad range of systems. In particular, hybrid systems, where both discrete and continuous behaviors occur, are particularly interesting. Important models of discrete-time hybrid systems are explained as follows.

Piecewise Affine Systems

Definition 10. A piecewise affine (PWA) system is a transition system, as Definition 9, where $x, u, x^+ \in \rightarrow$ if and only if

$$x^+ \in \begin{cases} \{A_1 x + B_1 u + c_1\} \oplus W_1, & (x, u) \in \mathcal{T}_1, \\ \vdots & \vdots \\ \{A_{\mathcal{M}} x + B_{\mathcal{M}} u + c_{\mathcal{M}}\} \oplus W_{\mathcal{M}}, & (x, u) \in \mathcal{T}_{\mathcal{M}}, \end{cases} \tag{2.6}$$

where $\mathcal{T}_i$, $i = 1, \cdots, \mathcal{M}$, are interior-disjoint polyhedral sets such that $\bigcup_{i=1}^{\mathcal{M}} \mathcal{T}_i = X \times U$. The number of modes is $\mathcal{M}$.

We usually assume that the sets of additive disturbances W_i, $i = 1, \cdots, \mathcal{M}$, are polytopes. PWA systems can handle nonlinearities to an arbitrary degree of accuracy. The form in (2.6) allows for non-determinism, which enriches the range of behaviors that it can capture. A PWA system is deterministic if $W_i = 0$, $i = 1, \cdots, \mathcal{M}$.

Mixed-logical Dynamical Systems

Definition 11. An MLD system [Bemporad and Morari, 1999] is a system in form (2.5) which can be written as:

$$x_{t+1} = Ax_t + B_u u_t + B_w w_t + D_\delta \delta_t + D_r r_t, \tag{2.7a}$$

$$E_\delta \delta_t + E_r r_t \leq E_x x_t + E_u u_t + E_w w_t + e, \tag{2.7b}$$

where $\delta_t \in \{0,1\}^{n_\delta}$ and $r_t \in \mathbb{R}^{n_r}$ are auxiliary variables and A, B_u, B_w, D_δ, D_r, E_δ, E_r, E_x, E_u, E_w, e are appropriately defined constant matrices such that (2.7) is well-posed in the sense that given x_t, u_t, w_t, the feasible set for x_{t+1} is a single point equal to $F(x_t, u_t, w_t)$ in (2.5). The inequality is interpreted element-wise.

Introducing auxiliary variables and enforcing (2.7b) can capture nonlinear F [Bemporad and Morari, 1999]. The system equations are brought into mixed-integer linear constraints by transforming system (2.5) into its MLD form.

Max-Min-Plus-Scaling Systems

Definition 12. A max-min-plus-scaling (MMPS) system is a system in form (2.5) and is defined using the following syntax [De Schutter and Van den Boom, 2001]:

$$F ::= F_{\text{linear}} | \max(F_1, F_2) | \min(F_1, F_2) | F_1 + F_2 | \alpha F, \tag{2.8}$$

where F_{linear} stands for linear systems of the following form: $x^+ = Ax + B_u u + B_w w$, $\alpha \in \mathbb{R}$, and F_1, F_2 are MMPS systems.

Using min and max operators, MMPS systems can handle discontinuities. MMPS systems are common in modeling systems with saturation constraints, which are typical in capacitated flow networks.

2.2.2 Equivalency of Hybrid Models

It was shown in [Heemels et al., 2001] that (deterministic) MMPS, MLD, and PWA systems are, under mild assumptions, equivalent. It was also shown that linear complementary systems, and extended linear complementary systems, also belong to this equivalency class. We do not provide the complete proofs and the underlying procedures, as they are well-documented in [Heemels et al., 2001], but instead show simple examples. As it turns out, MLD form is the most preferred one as its optimal control is easily cast as a mixed-integer programming problem.

Example 3 (PWA to MLD). Consider the following discrete-time PWA system which is a switched system with two modes and bounded $X \subset \mathbb{R}$, $U = \{0, 1\}$:

$$x^+ = \begin{cases} a_1 x + c_1 & u = 0, \\ a_2 x + c_2 & u = 1, \end{cases} \tag{2.9}$$

where $a_i, c_i, i = 1, 2$, are scalars. Let $M > \sup_{x \in X} |x|$. Using big-M method, the system can translated into MLD form as follows:

$$\begin{array}{l} a_1 x + c_1 - Mu \leq r \\ a_1 x + c_1 + Mu \geq r \\ a_2 x + c_2 + Mu - M \leq r \\ a_2 x + c_2 - Mu + M \geq r \\ x^+ = r \end{array} \tag{2.10}$$

It is straightforward to verify that (2.9) and (2.10) are equivalent.

Example 4 (MMPS to PWA to MLD). Consider the following discrete-time MMPS system:

$$x^+ = a \min(x, x_0) + bu + c \tag{2.11}$$

where $X, U \subset \mathbb{R}$ are bounded and x_0 is a constant. Note that (2.11) is equivalent to

$$x^+ = \begin{cases} ax + bu + c & x \leq x_0, \\ ax_0 + bu + c & x \geq x_0. \end{cases} \tag{2.12}$$

Note that the system is ill-posed for $x = x_0$. Using big-M method, (2.12) is equivalent to the following form:

$$\begin{array}{l} ax + bu + c - M(1 - \delta) \leq r, \\ ax + bu + c + M(1 - \delta) \geq r, \\ ax_0 + bu + c - M\delta \leq r, \\ ax_0 + bu + c + M\delta \geq r, \\ x \geq x_0 - M + M\delta, \\ x \leq x_0 + M\delta, \\ x^+ = r, \\ \delta \in \{0, 1\}. \end{array} \tag{2.13}$$

Note that $z = 1$ and $z = 0$ imply $x \geq x_0$, and $x \leq x_0$, respectively. It is straightforward to verify that (2.11) and (2.13) are equivalent.

Part I

Resilient Control of Hybrid Systems

Chapter 3

Deterministic Hybrid Systems

In this chapter, we introduce a method for control of deterministic MLD systems from STL specifications. In comparison with the existing works [Karaman et al., 2008, Raman et al., 2014], a key feature of our method is encoding STL quantitative semantics using slack variables. Thus no additional binary variable is needed beyond those required for well-known Boolean encoding. The solution properties are discussed and numerical examples are presented.

3.1 Problem Formulation

We consider MLD systems without disturbances:

$$x_{t+1} = Ax_t + B_u u_t + B_\delta \delta_t + B_r r_t, \tag{3.1a}$$

$$E_\delta \delta_t + E_r r_t \preceq E_x x_t + E_u u_t + e, \tag{3.1b}$$

$$y_t = C_x x_t + C_u u_t + c, \tag{3.1c}$$

where $y_t \in \mathbb{R}^{n_y}$ is the system output, and C_x, C_u, and c are appropriately sized matrices. We consider a negation-free STL formula with n_y predicates, each in the following form:

$$\pi_i := (y_i \geq 0), \tag{3.2}$$

where y_i is the i'th component of y. Note that the form in (3.2) is not restrictive, as $y_i \leq 0$ can be converted into (3.2) by replacing C_x, C_u, and c by their negative values. Moreover, a negation appearing immediately before a predicate can be eliminated:

$$\neg(y_i \geq 0) \Rightarrow (y_i - \epsilon \leq 0), \tag{3.3}$$

where $\epsilon > 0$ is a number smaller than the machine precision. Our control synthesis algorithm is optimization based and most solvers can not handle strict inequalities. If the STL formula φ is bounded by b^{φ}, we are only interested in finding a finite-time trajectory:

$$\xi_{\text{finite}} := (x_0, u_0)(x_1, u_1) \cdots (x_{b^{\varphi}}, u_{b^{\varphi}}). \tag{3.4}$$

$$\zeta_{\text{finite}} := y_0 y_1 \cdots y_{b^{\varphi}}. \tag{3.5}$$

Problem 1 (Bounded STL Optimal Control)**.** Given an MLD system (3.1), an initial condition x_0^*, a bounded negation-free STL formula φ over predicates of the form (3.2), and a piecewise linear/quadratic cost function $J : \xi_{\text{finite}} \rightarrow \mathbb{R}$, find controls $u_t, t \in [0, b^{\varphi}]$, such that $\zeta_{\text{finite}} \models \varphi$.

Throughout this thesis, we assume full state knowledge. We also want to find the set of all possible initial conditions $X_0^{\max}$ such that $\zeta_{\text{finite}} \models \varphi$ if and only if $x_0 \in X_0^{\max}$. There are two ways to characterize $X_0^{\max}$. First, it can be represented as a union of polyhedra - each represented by a finite number linear equalities. This representation may be computationally prohibitive. An alternative approach is to design an algorithm that given x_0, it checks whether $x_0 \in X_0^{\max}$. This option is easier.

The second problem is dealing with infinite-time specifications. We consider specifications of the following form:

$$\phi = \varphi_b \wedge \mathbf{G}_{[\Delta,\infty)} \varphi_g, \tag{3.6}$$

where $\Delta \in \mathbb{N}$, and φ_b and φ_g are bounded negation-free STL formulas. We call the formulae in form of (3.6) as bounded-global STL. It can be shown that (see Appendix) almost all interesting STL formulae can be written as a finite number of disjunctions of bounded-global STL formulae. Without loss of generality, we assume $\Delta \geq b^{\varphi_b}$. Our approach to satisfy (3.6) is finding a trajectory that consists of a periodic suffix:

$$\xi_{\text{infinite}} := (x_0, u_0)(x_1, u_1) \cdots (x_{T_0}, u_{T_0}) \left((x_{T_0+1}, u_{T_0+1}) \cdots (x_{T_0+T}, u_{T_0+T})\right)^{\omega} \tag{3.7}$$

$$\zeta_{\text{infinite}} := y_0 y_1 \cdots y_{T_0} \left(y_{T_0+1} y_{T_0+2} \cdots y_{T_0+T}\right)^{\omega}, \tag{3.8}$$

where T_0 and T are the lengths of the prefix and period of suffix, respectively. These numbers are user-chosen parameters. Note that we have $x_{T_0+1} = F(x_{T_0+T}, u_{T_0+T})$.

Problem 2 (Bounded-Global STL Optimal Control)**.** Given an MLD system (3.1), an initial condition x_0^*, a bounded global STL formula ϕ in form of (3.6) over predicates of the form (3.2), and a piecewise linear/quadratic cost function $J : \xi_{\text{infinite}} \rightarrow \mathbb{R}$, find controls $u_t, t \in [0, T_0 + T]$ such that $\zeta_{\text{infinite}} \models \phi$.

Similar to Problem 1, we also want to find the set of all admissible initial conditions for Problem 2. If satisfaction is impossible for both Problem 1 and Problem 2 satisfaction is impossible, we want to minimize violation by maximizing $\rho(\zeta_{\text{finite}}, \varphi, 0)$ or $\rho(\zeta_{\text{infinite}}, \phi, 0)$, respectively.

3.2 Mixed-Integer Formulation

Our approach to both Problem 1 and Problem 2 is casting them as mixed-integer programming problems. System 3.1 is already a set of mixed-integer-linear constraints. We present a method to translate STL formulas into mixed-integer constraints. Converting logical properties into mixed-integer constraints is a common procedure which was employed for MLD systems in [Bemporad and Morari, 1999]. The authors in [Karaman et al., 2008] and [Raman et al., 2014] extended this technique to a framework for time bounded model checking of temporal logic formulas. A variation of this method is explained here.

For each predicate $\pi = (y \geq 0)$, as in (3.2), we define a binary variable $z_k^\pi \in \{0, 1\}$ such that 1 (respectively, 0) stands for true (respectively, false). The relation between z^π, robustness ρ, and x is encoded as:

$$y + M(1 - z^\pi) \geq \rho, \tag{3.9a}$$

$$y - M z^\pi \leq \rho. \tag{3.9b}$$

The constant M is a sufficiently large number such that for all times, $M \geq \max y_i, i = 1, \cdots, n_y$. In practice, M is chosen sufficiently large such that the constraint $y \leq M$ is never active. Note that the largest value of ρ for which $z^\pi = 1$ is y, which is equal to the robustness of π.

Now we encode the truth table relations. For instance, we desire to capture $1 \wedge 0 = 0$ and $1 \vee 0 = 1$ using mixed-integer linear equations. Disjunction and conjunction connectives are encoded as the following constraints:

$$z = \bigwedge_{i=1}^{n_z} z_i \;\Rightarrow\; z \leq z_i, i = 1, \cdots, n_z, \tag{3.10a}$$

$$z = \bigvee_{i=1}^{n_z} z_i \;\Rightarrow\; z \leq \sum_{i=1}^{n_z} z_i, \tag{3.10b}$$

where $z \in [0, 1]$ is declared as a continuous variables. However, it only can take binary values as evident from (3.10). Similarly, define $z_k^\varphi \in [0, 1]$ as the variable indicating whether $\mathbf{x}[k] \models \varphi$. An STL formula is recursively translated as:

$$\varphi = \bigwedge_{i=1}^{n_\varphi} \varphi_i \;\Rightarrow\; z_k^{\varphi} = \bigwedge_{i=1}^{n_\varphi} z_k^{\varphi_i}; \tag{3.11a}$$

$$\varphi = \bigvee_{i=1}^{n_\varphi} \varphi_i \;\Rightarrow\; z_k^{\varphi} = \bigvee_{i=1}^{n_\varphi} z_k^{\varphi_i}; \tag{3.11b}$$

$$\varphi = \mathbf{G}_I \psi \Rightarrow z_k^{\varphi} = \bigwedge_{k' \in I} z_{k'}^{\psi}; \tag{3.11c}$$

$$\varphi = \mathbf{F}_I \psi \Rightarrow z_k^{\varphi} = \bigvee_{k' \in I} z_{k'}^{\psi}; \tag{3.11d}$$

$$\varphi = \psi_1 \mathbf{U}_I \psi_2 \Rightarrow z_k^{\varphi} = \bigvee_{k' \in I} \left(z_{k'}^{\psi_2} \wedge \bigwedge_{k'' \in [k,k']} z_{k''}^{\psi_1} \right). \tag{3.11e}$$

Finally, we add the following constraints:

$$z_0^{\varphi} = 1, \;\; \rho \geq 0. \tag{3.12}$$

Theorem 1. The set of constraints in (3.9),(3.10),(3.11),(3.12) has the following properties:

i) we have $\zeta \models \varphi$ if the set of constraints is feasible;

ii) we have $\zeta \not\models \varphi$ if the set of constraints is infeasible;

iii) the largest ρ such that the set of constraints, while "$\rho \geq 0$" is removed from (3.12), is feasible is equal to $\rho(\zeta, \varphi, 0)$.

Proof. i) We provide the proof for (3.10), as the case for more complex STL formulas are followed in a recursive manner from (3.11). If $z = 1$, we have from (3.10a) that $z_i = 1, i = 1, \cdots, n_z$, which correctly encodes conjunctions. Similarly, $z = 1$ in

(3.10b) indicates that not all $z_i, i = 0, \cdots, n_z$ can be zero, or, $\exists i \in \{1, \cdots, n_z\}$ such that $z_i = 1$, which correctly encodes disjunctions. ii) Infeasibility can be recursively traced back into (3.10). For both (3.10a) and (3.10b), if $z = 1$ is infeasible, it indicates that $z_i = 0, i = 1, \cdots, n_z$. iii) We also prove this statement for (3.10) as it is the base of recursion for general STL formulas. Let $z_i = (y_i \geq \rho), i = 1, \cdots, n_y$. Consider (3.10a) and the following optimization problem:

$$\begin{aligned} \rho^{\max} = \operatorname{argmax} \quad & \rho, \\ \text{s.t.} \quad & y_i \geq \rho, i = 1, \cdots, n_z, \end{aligned}$$

where its solution is $\min\limits_{i=1,\cdots,n_z} (y_i)$, which is identical to the quantitative semantics for conjunction - see Definition 7. Similarly, consider (3.10b) and the following optimization problem:

$$\begin{aligned} \rho^{\max} = \operatorname{argmax} \quad & \rho, \\ \text{s.t.} \quad & \exists i \in \{1, \cdots, n_z\}, y_i \geq \rho, \end{aligned}$$

where the solution is $\max\limits_{i=1,\cdots,n_z} (y_i)$, which is identical to the quantitative semantics for disjunction. $\square$

Our integer formulation for Boolean connectives slightly differs from the formulation in [Karaman et al., 2008], [Raman et al., 2014], where lower bound constraints for the z's are required. For example, for translating $z = \bigwedge_{i=1}^{n_z} z_i$, it is required to add $z \geq \sum_{i=1}^{n_z} z_i - n_z + 1$ to impose a lower bound for z. However, these additional constraints become necessary only when the negation operator is present in the STL formula. Hence, they are removed in our formulation. This reduces the constraint redundancy and degeneracy of the problem. By doing so, we observed computation speed gains (up to reducing the computation time by 50%) in our case studies. Moreover, we encode quantitative semantics in a different way than [Raman et al., 2014], where a separate STL robustness-based encoding is developed which introduces additional integers.

In [Raman et al., 2014], min and max operators in Definition 7 were separately encoded using additional binary variables. We are almost always interested in either maximizing the STL score, or declaring a constraint that encourages larger STL score values. Due to property "iii" in Theorem 1, additional integers are not required to capture STL score hence our framework is computationally more efficient than the one in [Raman et al., 2014].

3.3 Optimal Control

3.3.1 Finite-Horizon Semantics

The solution to Problem 1 is

$$\begin{array}{rll} u_0^*, u_1^*, \cdots, u_{b^\varphi}^* = & \underset{u_0, u_1, \cdots, u_{b^\varphi}}{\arg\min} & J(\xi_{\text{finite}}) + \frac{1}{2}M(|\rho| - \rho) \\ & \text{subject to} & (3.1), x_0 = x_0^*, z_0^\varphi = 1. \end{array} \tag{3.13}$$

Theorem 2. The solution to (3.13) has the following properties:

1. A solution with $\rho \geq 0$ is obtained if and only if STL satisfaction is feasible.
2. If STL satisfaction is possible, $J(\xi_{\text{finite}})$ is minimized.
3. If STL satisfaction is impossible, $\rho(\zeta_{\text{finite}}, \varphi, 0)$ is maximized

Proof. 1) Since all the necessary constraints are included in (3.13) are included, the solution to (3.13) is sound and complete - no conservativeness is introduced. The proof follows from a similar argument in [Karaman et al., 2008]. 2) The term $\frac{1}{2}M(|\rho| - \rho)$, which is added to the cost function, is a convex term. By the virtue of Theorem 1, this leads to maximization of $\rho(\zeta_{\text{finite}}, \varphi, 0)$. If $\rho \geq 0$ is feasible, then $\frac{1}{2}M(|\rho| - \rho) = 0$ and the original cost J is minimized. 3) If $\rho \geq 0$ is infeasible, then $\frac{1}{2}M(|\rho| - \rho)$ effectively becomes $-M\rho$. Since M is a very large positive number, ρ is maximized. □

Therefore, (3.13) automatically decides to optimize the original cost in case STL satisfaction is possible, or minimize violation in case STL satisfaction is impossible. Since all constraints are mixed-integer linear, (3.13) is a MILP/MIQP problem, depending on whether J is piecewise affine or quadratic.

3.3.2 Infinite-Horizon Semantics

We propose the following solution to Problem 2 is

$$\begin{array}{rll} u_0^*, u_1^*, \cdots, u_T^* = & \underset{u_0, u_1, \cdots, u_T}{\arg\min} & J'(\xi'_{\text{finite}}) + \frac{1}{2}M(|\rho| - \rho) \\ & \text{subject to} & (3.1), x_0 = x_0^*, z_0^{\varphi} = 1. \\ & & x_{T_0+1} = F(x_{T_0+T}, u_{T_0+T}), \end{array} \tag{3.14}$$

where $\varphi = \varphi_b \wedge \mathbf{G}_{[\Delta,T)}\varphi_g$ and

$$\xi'_{\text{finite}} = (x_0, u_0)(x_1, u_1) \cdots (x_{T+b^{\varphi}}, u_{T+b^{\varphi}}),$$

and J' is defined in appropriate way - which is always possible as J' has more arguments than J. We prove that (3.14), if feasible, provides a suboptimal solution to Problem 2.

Theorem 3. If a solution to (3.14) with $\rho \geq 0$ exists, then $\zeta_{\text{infinite}} \models \phi$.

Proof. The proof is straightforward from the periodicity of the suffix. We only need to prove that $\zeta[t, t + b^{\varphi_g}] \models \varphi_g, \forall t > \Delta$. The case up to $t = T$ is already given. From periodicity we have $\zeta[t + k(T - T_0), t + b^{\varphi_g} + +k(T - T_0)] = \zeta[t, t + b^{\varphi_g}], \forall k \in \mathbb{N}$. Therefore, $\zeta[t, t+b^{\varphi_g}] = \zeta[T_0+\text{rem}(t-T_0, T-T_0), T_0+\text{rem}(t-T_0, T-T_0)+b^{\varphi_g}] \models \varphi$. □

Similar to the result in Theorem 2, STL score is maximized if (3.14) fails to find a trajectory satisfying ϕ. However, in contrast to the finite-horizon case, completeness property does not hold as feasibility depends on choices of T_0 and T. Intuitively, larger values of T_0 and T may result in feasibility.

3.4 Initial Conditions

Now we describe how to find the set of admissible initial conditions.

3.4.1 Feasibility Check

As mentioned earlier, a simple way to obtain a set of admissible initial conditions is sampling points from X, and checking whether $x_0 \in X_0^{\max}$ by feasibility of (3.13) or (3.14). In case of (3.13), no conservativeness is introduced hence $X_0^{\max}$ is constructed by exhaustive sampling. However, this approach is naive.

3.4.2 Quantifier Elimination

The alternative way to obtain $X_0^{\max}$ (in case of finite-horizon) is to eliminate quantifiers from (3.13). In other words, we project the intersection of language realization set and possible trajectories X. The details are outlined in [Sadraddini and Belta, 2016a], where Fourier-Motzkin elimination method is used. Alternative projection methods, such as approximate sampling based projection (e.g., see [Jones et al., 2004]) can also be used.

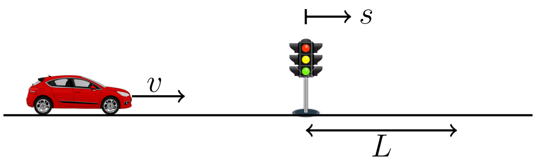


Figure 3·1: A car is moving toward a traffic light. If encountered by yellow light, the choice has to be made about stopping before the traffic light or clear the intersection before the traffic light turns red.

3.5 Examples

Example 5. We aim to find the feasibility envelope of a car passing an intersection (as shown in Fig. 3·1). We consider the following double integrator model for the car:

$$\begin{pmatrix} s_{t+1} \\ v_{t+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} s_t \\ v_t \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_t, \tag{3.15}$$

where state is $x = (s, v)^T$. The state space is given by $\mathcal{X} = \{0 \leq v \leq 2\}$ and the acceleration input u is bounded to $\mathcal{U} = \big\{u \big| -0.3 \leq u \leq 0.2\big\}$. We recall the MTL specification:

$$\text{yellow} \rightarrow \mathbf{F}_{[0,T]} \left(((x \leq 0) \wedge (v = 0)) \vee (x \geq 10)\right). \tag{3.16}$$

We construct $LRS(\varphi)$ in $2(T+1)$ dimensional space, and use Fourier-Motzkin elimination method to find $\mathcal{X}_0^{\max}$ in 2-dimensional space. The feasibility envelope for different values of T is shown in Fig. 3·2. For instance, for $T = 8$ it is observed that the feasibility envelope has two cavities. The physical interpretation of the lower cavity is straightforward but it is more subtle for the upper cavity, which explains that cars require to start to speed down within a certain distance from the intersection. On the other hand, if the light is still not turned yellow, cars can increase speed before reaching the intersection. This practice is not considered and recommended for human drivers [Liu et al., 1996], but is potentially applicable to autonomous driving.

For comparison, we have simulated two trajectories for the case $T = 8$. The first is a car driving with constant velocity $v = 1.9$ and the second is a car driving within the feasibility envelope but uses a one-step look ahead strategy to maximize its speed. The results are shown in Fig. 3·3. It is observed that the the state of the first car evolves out of the feasibility envelope at a time point. Therefore, if the light is turned yellow, the car is not able to stop before the traffic light or clear the intersection. However, the second car, although driving faster in average, is guaranteed to be able to properly respond to the yellow light.

Example 6. We consider a PWA system 2D with 4 modes:

$$A_1 = \begin{pmatrix} 1 & 1 \\ -0.7 & 1 \end{pmatrix}, B_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, c_1 = \begin{pmatrix} -5 \\ 0 \end{pmatrix} \tag{3.17a}$$

$$A_2 = \begin{pmatrix} 1.3 & 1.3 \\ 0 & 1.3 \end{pmatrix}, B_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, c_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{3.17b}$$

$$A_3 = \begin{pmatrix} 0.7 & 0.7 \\ -0.3 & 0.7 \end{pmatrix}, B_3 = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, c_3 = \begin{pmatrix} 5 \\ 0 \end{pmatrix} \tag{3.17c}$$

$$A_4 = \begin{pmatrix} 1.3 & 1 \\ 0.3 & 0.7 \end{pmatrix}, B_4 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, c_4 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{3.17d}$$

We consider the region $X = [-20, 20]$, and admissible set of controls $U = [-1010]$. We are interested in the following specification:

$$\varphi = \mathbf{F}_{[0,15]}\mathbf{G}_{[0,2]}\psi_1 \wedge \mathbf{F}_{[15,30]}\psi_2 \wedge \mathbf{G}_{[0,30]}\neg\psi_3, \tag{3.18}$$

where:

$$\psi_1 = (x_{[1]} \geq -15) \wedge (x_{[2]} \geq -15) \wedge (x_{[1]} \leq -10) \wedge (x_{[2]} \leq -10) \tag{3.19a}$$

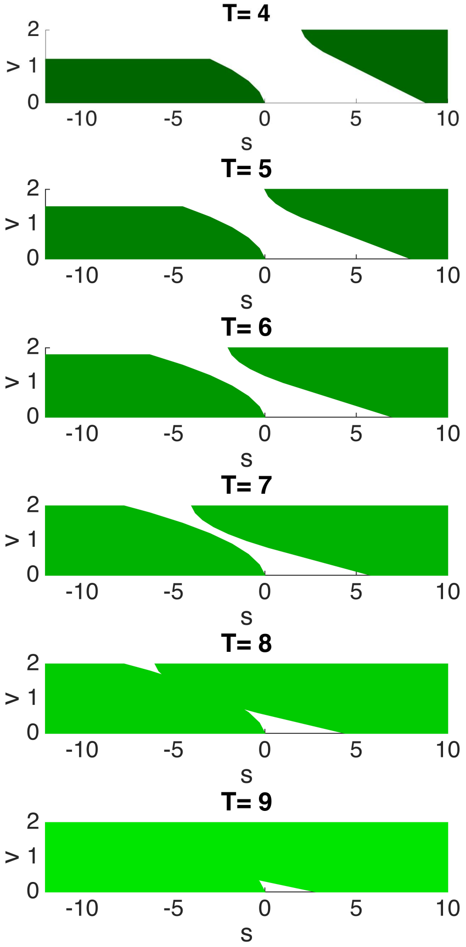


Figure 3·2: Example 5: The feasibility envelope of the yellow light dilemma problem is the green shaded area shown for different values of T. The traffic light is located at $s = 0$ and the end of the intersection is at $s = 10$.

$$\psi_2 = (x_{[1]} \geq 10) \wedge (x_{[2]} \geq 10) \wedge (x_{[1]} \leq 15) \wedge (x_{[2]} \leq 15) \tag{3.19b}$$

$$\psi_3 = (x_{[1]} \geq 0) \wedge (x_{[2]} \geq 2) \wedge (x_{[1]} \leq 15) \wedge (x_{[2]} \leq 8) \tag{3.19c}$$

Sub-specifications ψ_1, ψ_2 and ψ_3 correspond to rectangular regions in X. Figure 3·4 shows the workspace with the vector fields in each region. In plain English, (3.18) states that ψ_1 *is satisfied for 3 consecutive time steps between* $[0, 15]$*,* ψ_2 *is satisfied at least once in* $[15, 30]$*, and* ψ_3 *is never satisfied between* $[0, 30]$. The cost function is

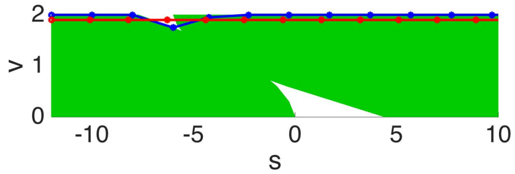


Figure 3·3: Example 5 : Two sample trajectories: (red) car driving at constant velocity $v = 1.9$ does not consider the feasibility envelope and is not guaranteed to be able to react properly to the yellow light. (blue) car driving inside the feasibility envelope with maximum speed. The car slows down before reaching the intersection ($s = -8$) but, in case of not encountering yellow light, starts to speed up at $s = 6$.

considered as L_2 norm of controls:

$$J = \sum_{t=0}^{30} u_t^2.$$

The initial condition is $x_0 = (-15, 10)^T$. Three trajectories are shown in Figure 3·4. The top left is the optimal one. It was computed using a MILP of 241 continuous, 572 binary variables and 2138 constraints in 150 seconds on a iMac quad-core 2.8 GHz Intel Core i5 machine. The obtained STL score is 0, which is the smallest possible. It is observed that the trajectory touches the boundaries of the rectangles, and further getting distance into ψ_1 and ψ_2 and away from ψ_3 increases control effort. The bottom left trajectory is the one with maximum possible STL score, which is equal to 2.5 and it was computed in 5 seconds. It is observed that the trajectory visits the centers of ψ_1 and ψ_2, and keeps a notable distance from ψ_3.

In order to obtain a trajectory with violation, we shrink U to $[-2, 2]$ to tighter the constraints. A maximally satisfying trajectory from $x_0 = (-10, 5)^T$ is shown in bottom right with $\rho(\zeta, \varphi, 0) = -3.9$. A feature of STL score is that predicates can be weighted. In order to obtain the latest trajectory, we multiplied the predicates of ψ_3 by 10 to further penalize violating trajectories entering ψ_3.

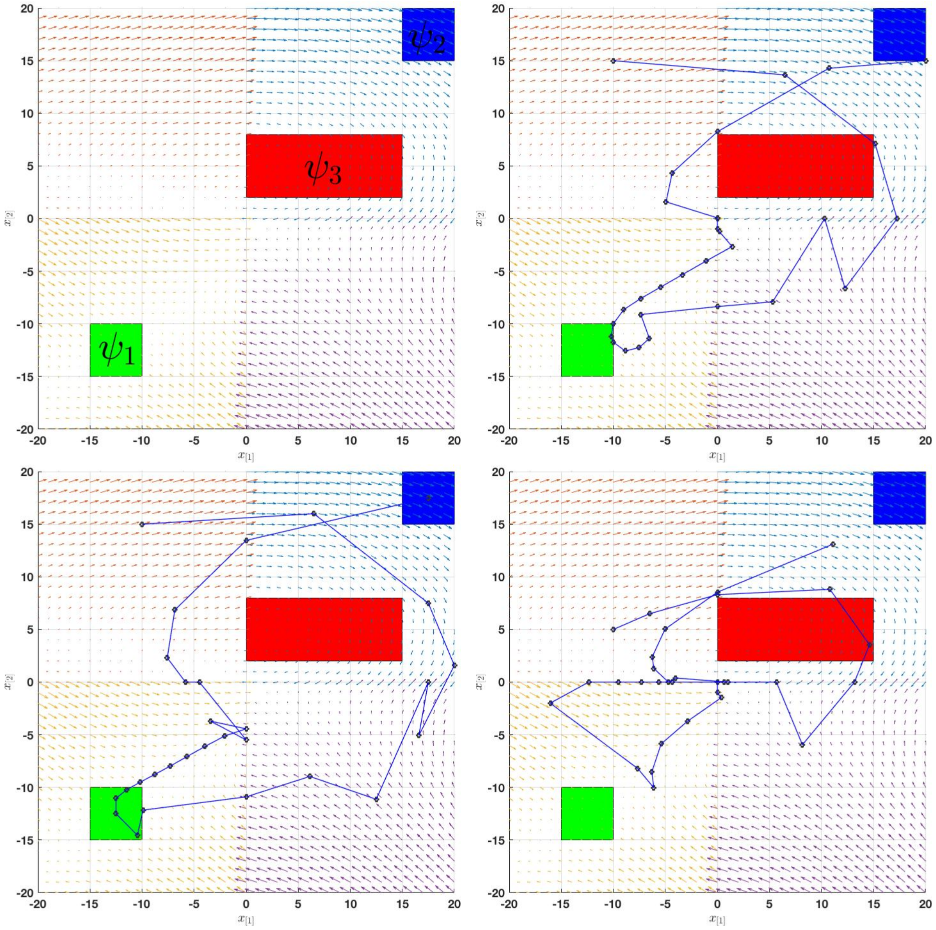


Figure 3·4: Example 3.19: [Top left] The vector field and the rectangular regions [Top right] Control-effort-optimal satisfying trajectory [Bottom left] Maximally satisfying trajectory [Bottom right] Minimally violating trajectory with $U = [-2, 2]$.

An example of an infinite-time specification for a PWA system is presented in Chapter 9.

Chapter 4

Non-Deterministic Hybrid Systems

In this chapter, the objectives are similar to Chapter 3, but uncertainties in form of additive disturbances are added to the system, which makes the problem much more difficult. The open-loop control sequences obtained in Chapter 3 are no longer the best solution as the trajectory will deviate from the planned one in a non-deterministic way. A feedback mechanism is necessary. The works in [Raman et al., 2015, Farahani et al., 2015, Sadraddini and Belta, 2015] considered receding horizon open-loop planning, where the controls at each time were chosen to be robust against all possible uncertainties within the horizon. This approach is severely conservative, as it is well-known that the set of open-loop robust feasible solutions is much smaller (it is often empty) than the set of feedback policies, which are given by dynamic programming. However, performing the Bellman iterations is intractable for PWA systems and STL specifications. A trade-off between complexity and completeness is necessary. We use the ideas of tube-MPC [Mayne et al., 2005], which was originally developed for constrained control of linear systems. The contribution of this chapter is tube-based STL control method for PWA systems. The idea is to break the controller design into two components: an open-loop policy for the system without disturbances (similar to Chapter 3), and an ancillary controller that is designed to handle disturbances in an optimal manner. The main challenge is dealing with mode transitions.

Notation

We retain the notation from previous chapters. Given two sets $A, B \subset \mathbb{R}^n$, we denote their Minkowski sum by $A \oplus B = \{a + b | a \in A, b \in B\}$ and Pontryagin difference by $A \ominus B = \{a \in A | a \oplus B \subseteq A\}$.

4.1 Problem Formulation

We consider a PWA system $F = (X, U, \rightarrow)$, $X \subset \mathbb{R}^n, U \subset \mathbb{R}^m$, as a transition system as Definition 9, such that $(x, u, x^+) \in \rightarrow$ if and only if:

$$x^+ \in \begin{cases} \{A_1 x + B_1 u + c_1\} \oplus W_1, & x \in X_1, \\ \vdots & \vdots \\ \{A_{\mathcal{M}} x + B_{\mathcal{M}} u + c_{\mathcal{M}}\} \oplus W_{\mathcal{M}}, & x \in X_{\mathcal{M}}, \end{cases} \tag{4.1a}$$

$$y_t = C_x x_t + C_u u_t + c, \tag{4.1b}$$

where $X_i, i = 1, \cdots, \mathcal{M}$, are polyhedral sets with disjoint interiors, $\bigcup_{i=1}^{\mathcal{M}}(X_i) = X$, $\mathcal{M}$ is the number of modes, and $W_i \in \mathbb{R}^n, i = 1, \cdots, \mathcal{M}$, are polytopic sets of additive disturbances. Each mode is an affine system with constants $A_i \in \mathbb{R}^{n \times n}, B_i \in \mathbb{R}^{n \times m}$ and $c_i \in \mathbb{R}^n$. The difference between (2.6) and (4.1) is that in latter, control space is not partitioned. This assumption disables us from considering systems where the switches are explicitly dependent on controls.

Definition 13. A control policy $\mu : X^* \times U^* \to U$ is a function that determines the control input at time t as a feedback of the history of the system:

$$u_t = \mu(x_0 x_1 \cdots x_t, u_0 u_1 \cdots u_{t-1}). \tag{4.2}$$

Definition 14. Given a control system $F = (X, U, \rightarrow)$, a control policy μ, a set of initial conditions $X_0 \subseteq X$, predicates as in (3.2),(4.1b), we define the *closed-loop language* as $\mathcal{L} = \mathcal{L}(F, X_0, \mu) \subset (\mathbb{R}^{n_y})^\omega$ such that $y_0 y_1 \cdots \in \mathcal{L}$ if and only if $x_0 \in X_0$ and

$$(x_k, \mu(x_0 x_1 \cdots x_t, u_0 u_1 \cdots u_{t-1}), x_{k+1}) \in \rightarrow, y_k = C_x x_t + C_u u_t + c, \forall k \in \mathbb{N}.$$

Problem 3. Given a PWA system (4.1), an initial condition x_0^*, a bounded STL formula φ over predicates of the form (3.2), (4.1b), find a control policy μ and a set of initial conditions X_0 such that $\zeta_{\text{infinity}} \models \varphi, \forall \zeta_{\text{infinity}} \in \mathcal{L}(F, X_0, \mu)$

We note that we provide the solution for bounded-global specifications, but the methods in this chapter are immediately applicable to bounded-global specifications as well. We will elaborate on this shortly. Similar to Chapter 3, when satisfaction is infeasible, we look for minimal violation. Problem 3 is difficult as one has to search over control policies and not control inputs. Even for for small problems, Problem 3 is often intractable. For example, if φ is a set-invariance specification, the complete solution to Problem 3 is equivalent to finding the maximal robust control invariant (RCI) set, which is undecidable, in general [Blanchini, 1999].

4.2 Tube STL Control

We propose μ to be of the following form:

$$\mu(x_0 x_1 \cdots x_t, u_0 u_1 \cdots u_{t-1}) = \mu^{\text{nom}}(x_0, t) + \mu^{\text{fb}}(x_t), \tag{4.3}$$

where $\mu^{\text{nom}} : X \times \mathbb{N} \rightarrow U$ is an open-loop control policy and $\mu^{fb} : \mathbb{R}^n \rightarrow U$ is a state-feedback control policy. Roughly, $\mu^{\text{nom}}(x_0)$ is computed offline and planned for nominal system, while μ^{fb} is the ancillary controller that corrects the deviations from the nominal trajectory in an online manner. We design μ^{fb} for all possible switchings of the system. This approach is conservative, but decouples the design of μ^{fb} and μ^{nom}. Note that $\mu^{\text{fb}}(x_t)$ has access to the knowledge of mode at time t. Thus, the mode is observable but its future is not modeled. The reason is that STL specifications are complicated and desirable trajectories often traverse through polyhedral regions $X_i, i = 1, \cdots, \mathcal{M}$, in a complicated manner.

Definition 15. Given a PWA system as in (4.1), the *nominal* PWA system is defined as function $f_{\text{nom}} : X \times U \to \mathbb{R}^n$, where:

$$f_{\text{nom}}(x, u) = A_i x + B_i u + c_i, x \in X_i. \tag{4.4}$$

Since it is deterministic, controlling the nominal PWA system from STL specifications can be accomplished by the techniques outlined in Chapter 3.

Definition 16. Given a PWA system as in Definition 4.1, the *switching-disturbance system* for mode $\sigma \in \{1, \cdots, \mathcal{M}\}$ is defined as as $\mathcal{F}_{\text{swd}} = ((\mathbb{R}^n \times \{1, \cdots, \mathcal{M}\}, \mathbb{U}, \rightarrow$ such that $((x, \sigma), u, x^+) \in\rightarrow$ if and only if:

$$x^+ \in \{A_\sigma x + B_\sigma u\} \oplus W_\sigma, \tag{4.5}$$

where $\sigma \in \{1, \cdots, \mathcal{M}\}$.

The switching-disturbance system is an aggregation of linear systems with polytopic disturbances. Its switching behavior is considered arbitrary. Following the earlier explanation, the mode is observable so it is considered as part of the state,

Definition 17. A *tube* for (4.5) is a robust control invariant (RCI) [Blanchini, 1999] set $T_X \subset \mathbb{R}^n$, associated with a control policy $\mu^{\text{tube}} : \mathbb{R}^n \times \{1, \cdots, \mathcal{M}\} \to \mathbb{R}^m$, such that the following relation holds:

$$\mathcal{L}(\mathcal{F}_{\text{swd}}, T_X, \mu^{\text{tube}}) \subseteq T^{\omega}, \tag{4.6}$$

i.e., the trajectory is always confined to T. Moreover, the set $T_U \subset \mathbb{R}^m$ is defined as:

$$T_u := \left\{ \mu^{\text{tube}}(x, \sigma) \middle| x \in T_X, \sigma \in \{1, \cdots, \mathcal{M}\} \right\}. \tag{4.7}$$

A tube is a RCI set for $\mathcal{F}_{\text{swd}}$. The *invariance-inducing* control policy can be rewritten as $\mu^{\text{tube}} : T_X \times \{1, \cdots, \mathcal{M}\} \to T_u$. The following theorem is the key result of this chapter.

Theorem 4. Consider a PWA system $\mathcal{F}$ as in (4.1), its nominal and switching-disturbance versions f_{nom} and $\mathcal{F}_{\text{swd}}$, initial condition $x_0 \in X$, and an STL formula ϕ with predicates in the form of (3.2),(3.1c). Let T and $\mu^{\text{tube}} : T \times \{1, \cdots, \mathcal{M}\} \to T_u$ be a tube and its invariance controller, respectively. Assume $x_0 \in X$. Let $\mu^{\text{nom}} : (X) \times \mathbb{N} \to (U \ominus T_U)$ be an open-loop control policy such that it results in the following nominal trajectory:

$$\xi^{\text{nom}} = (x_0^{\text{nom}}, u_0^{\text{nom}}), (x_1^{\text{nom}}, u_1^{\text{nom}}), \cdots,$$

$$\zeta^{\text{nom}} = y_0^{\text{nom}}, y_1^{\text{nom}}, \cdots,$$

and the following condition holds:

$$x_k^{\text{nom}} \in X_\sigma \ominus T_X, \forall k \in \mathbb{N} \tag{4.8}$$

for some $\sigma \in \{1, \cdots, \mathcal{M}\}$. Then, by replacing the following policy in (4.3)

$$\mu(x_t) = \mu^{\text{tube}}(x_t - x_t^{\text{nom}}, \Sigma(x_t)), \tag{4.9}$$

where $\Sigma : X \to \{1, \cdots, \mathcal{M}\}$ returns the current mode, then for all possible trajectories $\zeta \in \mathcal{L}(F, \mu, \{x_0^{\text{nom}}\} \oplus T)$, the following guarantee holds:

$$\rho(\zeta, \phi, 0) \geq \rho(\zeta^{\text{nom}}, \phi, 0) - \max_{x \in T_X, u \in T_U} \|C_x x + C_u u\|_\infty. \tag{4.10}$$

Proof. First, we show that the closed-loop trajectories remain within T-vicinity of the nominal trajectory. The proof is by induction. If $x_t \in \{x_t^{\text{nom}}\} \oplus T$ implies $x_{t+1} \in \{x_{t+1}^{\text{nom}}\} \oplus T_X$, which is immediately verified by the fact that T_X is a mode-agnostic RCI set: for all the points in $\{x_{t+1}^{\text{nom}}\} \oplus T_X$, for all modes, there exists a control input in T_u to keep the state within T_X. Note that all points in $x_t \in \{x_t^{\text{nom}}\} \oplus T$ belong to a single polyhedral region, as implied by (4.8). Next, any trajectory within T-vicinity of the nominal trajectory can decrease the STL score by at most $\max_{x \in T_X, u \in T_U} \|C_x x + C_u u\|_\infty$, which is the largest possible change in the value of a predicate of forms in (3.2),(3.1c). The rest of the proof holds by (2.3). □

4.3 Tube Design

We desire a tube with minimal $\max_{x \in T_X, u \in T_U} \|C_x x + C_u u\|_\infty$. Finding the maximal robust control invariant (RCI) set - a fixed-point - is a well-known undecidable problem [Blanchini, 1999]. Moreover, performing the fixed-point algorithm for PWA systems with additive disturbances is computationally challenging [Raković et al., 2004]. We desire finding the smallest RCI set. The authors in [Raković et al., 2007] formulated finding RCI sets for linear systems with polytopic disturbances and constraints as convex optimization problems - hence the cost function could be designed to promote smaller RCI sets. However, the method in [Raković et al., 2007] does not apply to switching systems. Here we propose a new method to compute optimized RCI sets for switching systems with additive disturbances.

4.3.1 Set-Parameterization

We parameterize the RCI sets by a user-specified number of hyper-rectangles. For each mode, there should exist a control input for each vertex of the hyper-rectangles such that for all allowable disturbances the vertex finds a successor in at least one of the hyper-rectangles - hence we enforce invariance by design. We show that the convex-hull of the hyper-rectangles is a RCI set. Define Γ, a positive integer, axis-aligned hyper-rectangles as

$$\mathcal{R}(p^\gamma, a^\gamma) := \{x \in \mathbb{R}^n | a^\gamma \leq x \leq p^\gamma + a^\gamma\},$$

where $p^\gamma \in \mathbb{R}^n, a^\gamma \in \mathbb{R}^n_+$. $\gamma = 1, \cdots, \Gamma$, are the lower-left corners and sides, respectively. The vertices of these hyper-rectangles are denoted by $q_k^\gamma, k = 1, \cdots, 2^n, \gamma = 1, \cdots, \Gamma$. Note that $\mathcal{R}(p^\gamma, a^\gamma) = \text{Convh}\left(\{q_k^\gamma\}_{k=0,\cdots,2^n-1}\right)$. Let $\theta := \{p^\gamma, a^\gamma\}_{\gamma=1,\cdots,\Gamma}$ be a set of $2n\Gamma$ decision variables. The vertex representation of disturbance sets are denoted by $W_\sigma = \text{Convh}(w_{\sigma,1}, \cdots, w_{\sigma,d_\sigma}), \sigma \in \{1, \cdots, \mathcal{M}\}$, where $d_\sigma \in \mathbb{N}_+$ is the number of vertices of W_σ. We define

$$\mathcal{T}(\theta) := \text{Convh}\left(\{q_k^\gamma\}_{\gamma=1,\cdots,\Gamma,k=1,\cdots,2^n}\right). \tag{4.11}$$

Lemma 1. If there exists $u_{k,\sigma}^{\gamma} \in \mathbb{R}^m, \gamma = 1, \cdots, \Gamma, k = 1, \cdots, 2^n, \sigma \in \{1, \cdots, \mathcal{M}\}$, such that

$$y_{k,\sigma,j}^{\gamma} \in \bigcup_{\gamma=1}^{\Gamma} \mathcal{R}(p^{\gamma}, a^{\gamma}), \tag{4.12}$$

where $y_{k,\sigma,j}^{\gamma} := A_{\sigma} p_k^{\gamma} + B_{\sigma} u_{k,\sigma}^{\gamma} + w_{\sigma,j}$. Then $\mathcal{T}(\theta)$ is a tube with

$$T_u(\theta) = \text{Convh}\{u_{k,\sigma}^{\gamma}\}_{k=1,\cdots,2^n,\gamma=1,\cdots,\Gamma,\sigma\in\{1,\cdots,\mathcal{M}\}}.$$

Proof. Consider any point $x \in \mathcal{T}(\theta)$. There exists $\lambda_k^{\gamma} \geq 0, \sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^{\gamma} = 1$, such that $x = \sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^{\gamma} q_k^{\gamma}$. Now introduce the control input $v_{\sigma} := \sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^{\gamma} u_{k,\sigma}^{\gamma}$. Note that $v_{\sigma} \in T_u$ since it is a convex combination of $2n\Gamma$ points in T_u. We have the following deductions:

$$\begin{array}{rl} y = & A_{\sigma} x + B_{\sigma} v_{\sigma} + w_{\sigma,j} \\ = & A_{\sigma} \sum_{\gamma=1}^{\Gamma} \sum_{k=0}^{2^n-1} \lambda_k^{\gamma} q_k^{\gamma} + B_{\sigma} \sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^{\gamma} u_{k,\sigma}^{\gamma} + w_{\sigma,j} \\ = & \sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^{\gamma} (A_{\sigma} q_k^{\gamma} + B_{\sigma} u_{k,\sigma}^{\gamma} + w_{\sigma,j}) \\ = & \sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^{\gamma} y_{k,\sigma,j}^{\gamma} \in \mathcal{T}(\theta) \\ \Rightarrow & (A_{\sigma} x + B_{\sigma} v_{\sigma}) \oplus \{w_{\sigma,j}\}_{j=1,\cdots,d_{\sigma}} \subseteq \mathcal{T}(\theta). \end{array}$$

Taking the convex hulls of both sides, we obtain $\{A_{\sigma} x + B_{\sigma} v_{\sigma}\} \oplus \mathbb{W}_{\sigma} \subseteq \mathcal{T}(\theta)$ and the proof is complete. $\square$

4.3.2 Optimization

The conditions in Lemma 1 are formulated as a set of constraints. Eq. (4.12) is equivalent to the following Boolean logic formula being true for $\gamma = 1, \cdots, \Gamma, k = 1, \cdots, 2^n, \sigma \in \{1, \cdots, \mathcal{M}\}, j = 1, \cdots, d_{\sigma}$:

$$\bigvee_{\beta=1}^{\Gamma} \left((p^{\beta} \leq y_{k,\sigma,j}^{\gamma}) \wedge (y_{k,\sigma,j}^{\gamma} \leq p^{\beta} + a^{\beta}) \right). \tag{4.13}$$

We encode (4.13) using binary decision variables and big-M method - technically similar to Sec. 9.3.2. Using basic convexity notions, we formulate the following MILP:

$$\begin{aligned} \theta^* = \arg\min_{\theta} \quad & \xi \\ s.t. \quad & \|q_k^\gamma\|_\infty \leq \xi, k = 1, \cdots, 2^n, \\ & (4.11), (4.12), (4.13), \gamma = 1, \cdots, \Gamma. \end{aligned} \tag{4.14}$$

The solution of (4.14) yields the smallest tube. A corresponding invariance-inducing control policy μ^{tube} can be designed from the proof of Lemma 1:

$$\begin{aligned} \mu^{\text{tube}}(x, \sigma) = \quad \arg\min_{u} \quad & J(u) \\ \text{subject to} \quad & x = \textstyle\sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^\gamma q_k^\gamma \\ & u = \textstyle\sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^\gamma u_{k,\sigma}^\gamma, \\ & \lambda_k^\gamma \geq 0, \textstyle\sum_{\gamma=1}^{\Gamma} \sum_{k=1}^{2^n} \lambda_k^\gamma = 1. \end{aligned} \tag{4.15}$$

where $J(u)$ is a user-defined convex linear/quadratic cost function. Eq. (4.15) is a linear/quadratic program with $2n\Gamma$ decision variables. Thus, μ^{tube} takes a PWA form. A proper choice for $J(u)$ is $\|A_\sigma x + B_\sigma u\|_\infty$, which penalizes the distance from the center of the tube.

4.4 Example

Example 7. Consider the system in Example 3.19. First, we compute the smallest tube. We let $r = 3$, and obtain a tube with $T_X \subset [-1.14\ 1.14]$ and $T_U \subset [-1.96\ 1.96]$. The tube is illustrated in Figure 4·1. The tube was constructed in 150 seconds. The largest STL violation permitted by the tube is 1.14. Next, we consider the initial condition $x_0 = (-15, 10)^T$ and design a nominal control policy with STL score greater than 1.14, hence all disturbed trajectories are guaranteed to satisfy φ. Since the maximum deterministic STL score is 2.5, we have a policy that the worst-case STL score for disturbed trajectories is $2.5 - 1.14 = 1.36$. Sample trajectories are

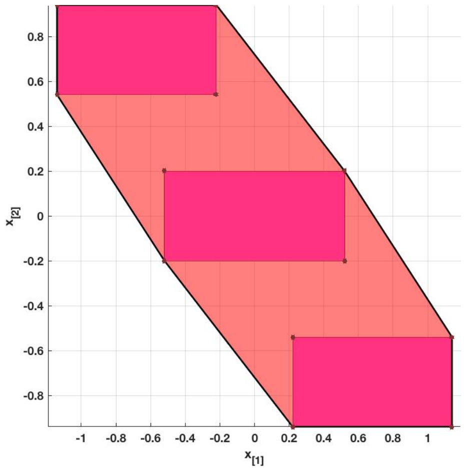


Figure 4·1: Example 7: The smallest tube generated by three hyper-rectangles.

shown in Figure 4·2.

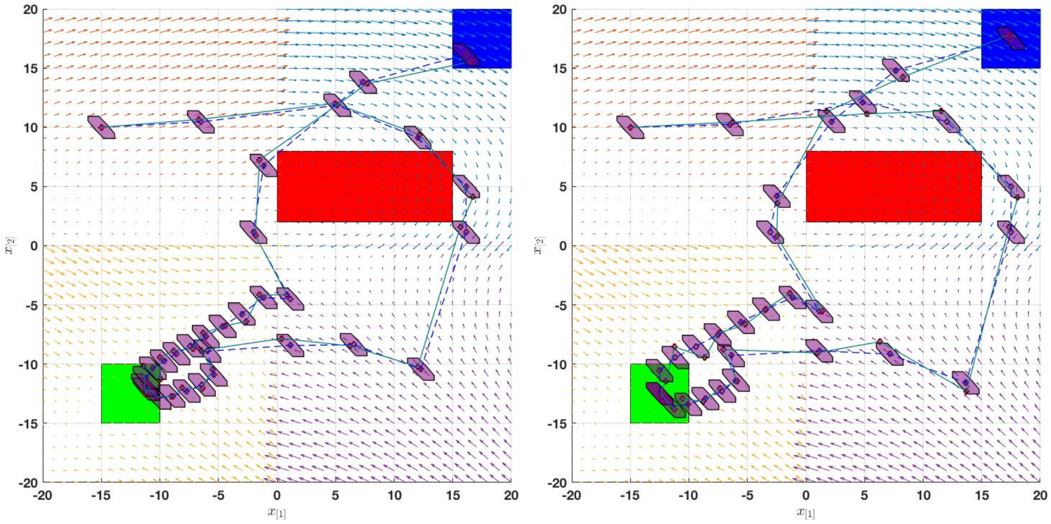


Figure 4·2: Example 7: [Left]: Tube-based design with all trajectories guaranteed to have STL score greater than 0. [Right]: Design corresponding to the maximum worst-case STL score 1.36.

Part II

Systems and Specifications with Specific Structures

Chapter 5

Monotone Systems and Traffic Networks

In this chapter, we study optimal STL control of discrete-time positive monotone systems (i.e., systems with state partial order on the positive orthant) with bounded disturbances. The STL specifications in this chapter are restricted to a particular form that favors smaller values for the state components. We assume that there exists a maximal disturbance element that characterizes a type of upper-bound for the evolution of the system. These assumptions are specifically motivated by the dynamics of traffic networks, where the disturbances represent the volume of exogenous vehicles entering the network and the maximal disturbance characterizes the rush hour exogenous flow. Our optimal control study is focused on STL formulae with infinite-time safety/persistence properties, which is relevant to optimal and correct traffic control in the sense that the vehicular flow is always free of congestion while the associated delay is minimized.

The key contributions of this chapter are as follows. First, for finite-time semantics, we prove that the existence of open-loop control policies is necessary and sufficient for maintaining STL correctness. However, we use MPC, which has a closed-loop nature, to achieve optimality. The main contribution of our STL MPC algorithm is guaranteed recursive feasibility. We show via a case study that our method is applicable to systems with relatively high dimensions.

This chapter is organized as follows. The problems are formulated in Sec. 8.1. The technical details for control synthesis from finite and infinite-time specifications are given in Sec. 8.3 and Sec. 8.4, respectively. The robust MPC framework is explained in Sec. 5.4. Finally, we introduce a traffic network model and explain its monotonicity properties in Sec. 5.5, where a case study is also presented.

Notation

For two integers a, b, we use $\mathrm{rem}(a, b)$ to denote the remainder of division of a by b (modulus). Given a set $\mathcal{S}$ and a positive integer K, we use the shorthand notation $\mathcal{S}^K$ for $\prod_{i=1}^{K} \mathcal{S}$. A signal is defined as an infinite sequence $\mathbf{s} = s_0 s_1 \cdots$, where $s_k \in \mathcal{S}$, $k \in \mathbb{N}$. Given $s_1, s_2, \cdots, s_K \in \mathcal{S}$, the repetitive infinite-sequence $s_1 s_2 \cdots s_K s_1 s_2 \cdots s_K \cdots$ is denoted by $(s_1 s_2 \cdots s_K)^\omega$. The set of all signals that can be generated from $\mathcal{S}$ is denoted by $\mathcal{S}^\omega$. We use $\mathbf{s}[k] = s_k s_{k+1} \cdots$ and $\mathbf{s}[k_1 : k_2] = s_{k_1} s_{k_1+1} \cdots s_{k_2}$, $k_1 < k_2$, to denote specific portions of $\mathbf{s}$. A *real* signal is $\mathbf{r} = r_0 r_1 r_2 \cdots$, where $r_k \in \mathbb{R}^n, \forall k \in \mathbb{N}$. A vector of all ones in $\mathbb{R}^n$ is denoted by 1_n. We use the notation $\mathbf{1}_n[0 : K] := 1_n \cdots 1_n$, where 1_n is repeated $K + 1$ times. The positive closed orthant of the n-dimensional Euclidian space is denoted by $\mathbb{R}^n_+ := \left\{x \in \mathbb{R}^n | x_{[i]} \geq 0, i = 1, \cdots, n\right\}$, where $x = (x_{[1]}, x_{[2]}, \cdots, x_{[n]})^T$. For $a, b \in \mathbb{R}^n$, the non-strict partial order relation $\preceq$ is defined as: $a \preceq b \Leftrightarrow b - a \in \mathbb{R}^n_+$.

Definition 18 ([Davey and Priestley, 2002]). A set $\mathcal{X} \subset \mathbb{R}^n_+$ is a *lower-set* if $\forall x \in \mathcal{X}, L(x) \subseteq \mathcal{X}$, where $L(x) := \left\{y \in \mathbb{R}^n_+ \middle| y \preceq x\right\}$.

It is straightforward to verify that if $\mathcal{X}_1, \mathcal{X}_2$ are lower-sets, then $\mathcal{X}_1 \cup \mathcal{X}_2$ and $\mathcal{X}_1 \cap \mathcal{X}_2$ are also lower-sets. We extend the usage of notation $\preceq$ to equal-length real signals. For two real signals $\mathbf{r}, \mathbf{r}$, we denote $\mathbf{r}'[t'_1 : t'_2] \preceq \mathbf{r}[t_1 : t_2]$, $t_2 - t_1 = t'_2 - t'_1$, if $r'_{t'_1+k} \preceq r_{t_1+k}, k = 0, 1, \cdots, t_2 - t_1$. Moreover, if $\mathbf{r}, \mathbf{r}' \in (\mathbb{R}^n_+)^\omega$, we are also allowed to write $\mathbf{r}'[t'_1 : t'_2] \in L(\mathbf{r}[t_1 : t_2])$.

5.1 Problem Statement and Approach

We consider discrete-time systems of the following form:

$$x_{t+1} = f(x_t, u_t, w_t), \tag{5.1}$$

where $x_t \in \mathcal{X}$ is the state, $\mathcal{X} \subset \mathbb{R}^n_+$, $u_t \in \mathcal{U}$ is the control input, $\mathcal{U} = \mathbb{R}^{m_r} \times \{0,1\}^{m_b}$, and $w_t \in \mathcal{W}$ is the disturbance (adversarial input) at time t, $t \in \mathbb{N}$, $\mathcal{W} = \mathbb{R}^{q_r} \times \{0,1\}^{q_b}$. The sets $\mathcal{U}$ and $\mathcal{W}$ may include real and binary values. For instance, the set of controls in the traffic model developed in Sec. 5.5 includes binary values for decisions on traffic lights and real values for ramp meters. These types of systems are positive as all state components are non-negative. We also assume that $\mathcal{X}$ is bounded.

Definition 19. System (5.1) is monotone (with partial order on $\mathbb{R}^n_+$) if for all $x, x' \in \mathcal{X}$, $x' \preceq x$, we have $f(x', u, w) \preceq f(x, u, w), \forall u \in \mathcal{U}, \forall w \in \mathcal{W}$.

The systems considered in this chapter are positive and monotone with partial order on $\mathbb{R}^n_+$. For the remainder of this chapter, we simply refer to systems in Definition 19 as monotone [1]. Although the results of this chapter are valid for any general $f : \mathcal{X} \times \mathcal{U} \times \mathcal{W} \to \mathcal{X}$, we focus on systems that can be written in the form of mixed-logical dynamical (MLD) systems.

Assumption 1. There exist $w^* \in \mathcal{W}$ such that

$$\forall x \in \mathcal{X}, \forall u \in \mathcal{U},\ f(x, u, w) \preceq f(x, u, w^*), \forall w \in \mathcal{W}. \tag{5.2}$$

---

[1] The term *cooperative* in dynamics systems theory is used specifically to refer to systems that are monotone with partial order defined on the positive orthant. We avoid using this term here as it might generate confusion with the similar terminology used for multi-agent control systems.

We denote $f(x,u,w^*)$ by $f^*(x,u)$ and refer to f^* as the *maximal system.* As it will be further explained in this chapter, the behavior of monotone system (5.1) is mainly characterized by its maximal f^*. Assumption 1 is restrictive but holds for many compartmental systems where the disturbances are additive and the components are independent. Therefore, the maximal system corresponds to the situation that every component takes its most extreme value. We also note that if Assumption 1 is removed, overestimating f by some f^* such that $f(x,u,w) \preceq f^*(x,u), \forall w \in \mathcal{W}$, is always possible for a bounded f. By overestimating f the control synthesis methods of this chapter remain correct, but become conservative.

We describe the desired system behavior using specifications written as STL formulas over a finite set of predicates. We assume that each predicate π is in the following form:

$$\pi = \left(a_\pi^T x \leq b_\pi\right), \tag{5.3}$$

where $a_\pi \in \mathbb{R}_+^n$, $b_\pi \in \mathbb{R}_+$. It is straightforward to verify that the closed half-space defined by (5.3) is a lower-set in $\mathbb{R}_+^n$. By restricting the predicates into the form (5.3), we ensure that a predicate remains true if the values of state components are decreased (Note that this is true for any lower set. We require linearity in order to decrease the computational complexity.). This restriction is motivated by monotonicity. For example, in a traffic network, the state is the vector representation of vehicular densities in different segments of the network. The satisfaction of a “sensible” traffic specification has to be preserved if the vehicular densities are not increased all over the network. Otherwise, the specification encourages large densities and congestion.

Definition 20. A control policy $\mu := \bigcup_{t\in\mathbb{N}} \mu_t$ is a set of functions $\mu_t : \mathcal{X}^{t+1} \to \mathcal{U}$, where

$$u_t = \mu_t(x_0, x_1, \cdots, x_t).$$

An *open-loop* control policy takes the simpler form $u_t = \mu_t(x_0)$, i.e., the decision on the sequence of control inputs is made using only the initial state x_0. On the other hand, in a (history dependent) *feedback* control policy, $u_t = \mu_t(x_0, x_1, \cdots, x_t)$, the controller implementation requires real-time access to the state and its history.

An infinite sequence of admissible disturbances is $\mathbf{w} = w_0 w_1 \cdots$, where $w_k \in \mathcal{W}$, $k \in \mathbb{N}$. Following the notation introduced in Sec. 5, the set of all infinite-length sequences of admissible disturbances is denoted by $\mathcal{W}^\omega$. Given an initial condition x_0, a control policy μ and $\mathbf{w} \in \mathcal{W}^\omega$, the *run* of the system is defined as the following signal:

$$\mathbf{x} = \mathbf{x}(x_0, \mu, \mathbf{w}) := x_0 x_1 x_2 \cdots ,$$

where $x_{t+1} = f(x_t, u_t, w_t), \forall t \in \mathbb{N}$. Now we formulate the problems studied in this chapter. In all problems, we assume a monotone system (5.1) is given, Assumption 1 holds, and all the predicates are in the form of (5.3).

Problem 4 (Bounded STL Control). Given a bounded STL formula φ, find a set of initial conditions $\mathcal{X}_0 \subset \mathcal{X}$ and a control policy μ such that

$$\mathbf{x}(x_0, \mu, \mathbf{w})[0] \models \varphi, \forall \mathbf{w} \in \mathcal{W}^\omega, \forall x_0 \in \mathcal{X}_0.$$

As mentioned in the previous section, the satisfaction of φ solely depends on $\mathbf{x}[0 : h^\varphi]$, where h^φ is obtained from Definition 3. The horizon h^φ can be viewed as the time when the specification ends. In many engineering applications, the system is required to uphold certain behaviors for all times. Therefore, guaranteeing infinite-time safety properties is important. We formulate *bounded-global* STL formulas in the form of

$$\varphi_b \wedge \mathbf{G}_{[\Delta,\infty]} \varphi_g, \tag{5.4}$$

where φ_b, φ_g are bounded STL formulas, $\mathbf{G}_{[\Delta,\infty]}$ stands for unbounded temporal "always", and $\Delta \geq h^{\varphi_b}$ is a positive integer. Formula (6.9) states that first, φ_b is satisfied by the signal from time 0 to Δ, and, afterwards, φ_g holds for all times.

Problem 5 (Bounded-global STL Control)*.* Given bounded STL formulas φ_b, φ_g, $\Delta \in [h^{\varphi_b}, \infty)$, find a set of initial conditions $\mathcal{X}_0 \subset \mathcal{X}$ and a control policy μ such that

$$\mathbf{x}(x_0, \mu, \mathbf{w})[0] \models \varphi_b \wedge \mathbf{G}_{[\Delta,\infty]}\varphi_g, \forall \mathbf{w} \in \mathcal{W}^\omega, \forall x_0 \in \mathcal{X}_0. \tag{5.5}$$

As a special case, we allow φ_b to be logical truth so Problem 5 reduces to *global STL* control problem of satisfying $\mathbf{G}_{[\Delta,\infty]}\varphi_g$. Note that if φ_g is replaced by logical truth, Problem 5 reduces to Problem 4. We have distinguished Problem 4 and Problem 5 as we use different approaches to solve them.

It can be shown that (see Appendix) a large subset of safety STL formulas can be written as $\bigvee_{i=1}^{n_\phi} \phi_i$, where each $\phi_i, i = 1, \cdots, n_\phi$, is a bounded-global formula. Therefore, the framework for solutions to Problem 5 can also be used for safety STL control as it leads to n_ϕ instances of Problem 5, where a solution to any of the instances is also a solution to the original safety STL control problem. The drawback to this approach is that n_ϕ can be very large.

Remark 1*.* We avoid separate problem formulations for STL formulas containing unbounded "eventually" or "until" operators as their unbounded intervals can be safely under-approximated by bounded intervals. However, bounded under-approximation is not sound for the unbounded "always" operator. A safety formula can be satisfied (respectively, violated) with infinite-length (respectively, finite-length) signals [Ouaknine and Worrell, 2006].

In the presence of disturbances, feedback controllers obviously outperform open-loop controllers. We show that the existence of open-loop control policies for guaranteeing the STL correctness of monotone systems in Problem 1 (respectively, Problem 2) is sufficient and (respectively, almost) necessary. The online knowledge of state is not necessary for STL correctness. But it can be exploited for planning controls optimally. While our framework can accommodate optimal control versions of Problem 4 and Problem 5, the focus of this chapter is on robust optimal control problem for global STL formulas - of form $\mathbf{G}_{[0,\infty)}\varphi$, where φ is a bounded formula. These type of problems are of practical interest for optimal traffic management (as discussed in Sec. 5.5).

We use a model predictive control (MPC) approach, which is a popular, powerful approach to optimal control of constrained systems. Given a planning horizon of length H [2], a sequence of control actions starting from time t is denoted by $u_t^H := u_{0|t}u_{1|t}\cdots u_{H-1|t}$. Given u_t^H and x_t, we denote the predicted H-step system response by $x_t^H(x_t, u_t^H, w_t^H) := x_{1|t}x_{2|t}\cdots x_{H|t}$, where $x_{k+1|t} = f(x_{k|t}, u_{k|t}, w_{k|t}), k = 0, 1, \cdots, H-1$, $x_{0|t} = x_t$, and $w_t^H := w_{0|t}w_{1|t}\cdots w_{H-1|t}$. At each time, u_t^H is found such that it optimizes a cost function $J\left(x_t^H, u_t^H\right)$, $J : \mathcal{X}^H \times \mathcal{U}^H \rightarrow \mathbb{R}$, subject to system constraints. When u_t^H is computed, only the first control action $u_{0|t}$ is applied to the system and given the next state, the optimization problem is resolved for u_{t+1}^H. Thus, the implementation is closed-loop.

Problem 6 (Robust STL MPC). Given a bounded STL formula φ, an initial condition x_0, a planning horizon H and a cost function $J\left(x_t^H, u_t^H\right)$, find a control policy such that $u_t = \mu(x_0, \cdots, x_t) = u_{0|t}^{\text{opt}}$, where $u_t^{H,\text{opt}} := u_{0|t}^{\text{opt}} \cdots u_{H-1|t}^{\text{opt}}$, and $u^{H,\text{opt}}$ is the

---

[2] The MPC horizon H should not be confused with the STL horizon h^{φ}.

following minimizer:

$$\begin{array}{rl} \arg\min_{u_t^H} & \max\limits_{w_t^H} J\left(x_t^H(x_t, u_t^H, w_t^H), u_t^H\right), \\ \text{s.t.} & \mathbf{x}(x_0, \mu, \mathbf{w})[0] \models \mathbf{G}_{[0,\infty]}\varphi, \forall \mathbf{w} \in \mathcal{W}^\omega, \\ & x_{k+1} = f(x_k, u_k, w_k), \forall k \in \mathbb{N}. \end{array} \tag{5.6}$$

The primary challenge of robust STL MPC is guaranteeing the satisfaction of the global STL formula while the controls are planned in a receding horizon manner (see the constraints in (5.6)). Our approach takes the advantage of the results from Problem 5 to design appropriate terminal sets for the MPC algorithm such that the generated runs are guaranteed to satisfy the global STL specification while the online control decisions are computed (sub)optimally. Due to the temporal logic constraints, our MPC setup differs from the conventional one. The details are explained in Sec. 5.4.

For computational purposes, we assume that J is a piecewise affine function of the state and controls. Moreover, the cost functions in our applications are non-decreasing with respect to the state in the sense that $x'_{k|t} \preceq x_{k|t}, k = 1, 2, \cdots, H \Rightarrow J(x_t^{H\prime}, u_t^H) \preceq J(x_t^H, u_t^H), \forall u_t^H \in \mathcal{U}^H$. As it will become clear later in the chapter, we will exploit this property to simplify the worst-case optimization problem in (5.6) to an optimization problem for the maximal system.

As mentioned earlier, a natural objective is maximizing STL robustness score. It follows from the linearity of the predicates in (5.3) and max and min operators in Definition 3 that STL robustness score is a piecewise affine function of finite-length signals. We can also consider optimizing a weighted combination of STL robustness score and a given cost function. We use this cost formulation for traffic application in Sec. 5.5.

5.2 Finite Horizon Semantics

In this section, we explain the solution to Problem 4. First, we exploit monotonicity to characterize the properties of the solutions. Next, we explain how to synthesize controls using a mixed integer linear programming (MILP) solver.

Lemma 2. Consider runs $\mathbf{x}$ and $\mathbf{x}'$ and an STL formula φ. If for some t, t', we have $\mathbf{x}'[t' : t' + h^\varphi] \preceq \mathbf{x}[t : t + h^\varphi]$, then $\mathbf{x}[t] \models \varphi$ implies $\mathbf{x}'[t'] \models \varphi$.

Proof. Since all predicates denote lower-sets in the form of (5.3), we have $x'_{t'} \preceq x_t \Rightarrow a_\pi^T x'_{t'} \leq a_\pi^T x_t$, $\mathbf{x}[t] \models \pi \Rightarrow \mathbf{x}'[t] \models \pi$. Thus, all predicates that were true by the valuations in $\mathbf{x}$ remain true for $\mathbf{x}'$. The negation-free semantics implies that without falsifying any predicate, a formula can not be falsified. Therefore, $\mathbf{x}[t] \models \varphi$ implies $\mathbf{x}'[t'] \models \varphi$ □

The *largest set of admissible initial conditions* is defined as:

$$\mathcal{X}_0^{\max} := \left\{ x_0 \in \mathcal{X} \middle| \exists \mu \text{ s.t. } \mathbf{x}(x_0, \mu, \mathbf{w}) \models \varphi, \forall \mathbf{w} \in \mathcal{W}^\omega \right\}.$$

The set $\mathcal{X}_0^{\max}$ is a union of polyhedra. Finding the half-space representation of all polyhedral sets in $\mathcal{X}_0^{\max}$ may not be possible for high dimensions. Therefore, we find a half-space representation for a subset of $\mathcal{X}_0^{\max}$. The following result states how to check whether $x_0 \in \mathcal{X}_0^{\max}$.

Theorem 5. We have $x_0 \in \mathcal{X}_0^{\max}$ if and only if there exists an open-loop control sequence

$$u_0^{ol,x_0} u_1^{ol,x_0} \cdots u_{h^\varphi-1}^{ol,x_0}$$

such that $\mathbf{x}^{ol,x_0}[0 : h^\varphi] \models \varphi$, where $\mathbf{x}^{ol,x_0}[0 : h^\varphi] = x_0^{ol,x_0} x_1^{ol} \cdots x_{h^\varphi}^{ol,x_0}$, and $x_{k+1}^{ol,x_0} = f^*(x_k^{ol,x_0}, u_k^{ol,x_0}), k = 0, \cdots, h^\varphi - 1, x_0^{ol,x_0} = x_0$.

Proof. (*Necessity*) Satisfaction of φ with $\mathbf{w} \in \mathcal{W}^{\omega}$ requires at least one satisfying run for the maximal system, hence a corresponding control sequence exists. Denote it by $u_0^{ol,x_0} u_1^{ol,x_0} \cdots, u_{h^{\varphi}-1}^{ol,x_0}$. (*Sufficiency*) Consider any run generated by the original system $x_{k+1} = f(x_k, u_k^{ol,x_0}, w_k)$. We prove that $x_k \preceq x_k^{ol,x_0}$ $, k = 0, 1, \cdots, h^{\varphi}$, by induction over k. The base case $x_0 \preceq x_0^{ol,x_0}$ is trivial ($x_0 = x_0^{ol,x_0}$). The inductive step is verified from monotonicity: $x_{k+1} = f(x_k, u_k^{ol,x_0}, w_k) \preceq f^*(x_0^k, u_k^k) = x_{k+1}^{ol,x_0}$. Therefore, $\mathbf{x}[0 : h^{\varphi}] \preceq \mathbf{x}^{ol,x_0}[0 : h^{\varphi}]$, $\forall \mathbf{w}[0 : h^{\varphi}-1] \in \mathcal{W}^{h^{\varphi}}$. It follows from Lemma 2 that $\mathbf{x}[0 : h^{\varphi}] \models \varphi, \forall \mathbf{w}[0 : h^{\varphi}-1] \in \mathcal{W}^{h^{\varphi}}$. □

Corollary 1. The set $\mathcal{X}_0^{\max}$ is a lower-set.

Proof. Consider any $x'_0 \in L(x_0), x_0 \in \mathcal{X}_0^{\max}$. Let $x'_{k+1} = f(x'_k, u_k^{ol,x_0}, w_k), k = 0, 1, \cdots, h^{\varphi} - 1$. It follows from monotonicity that $x'_k \preceq x_k^{ol,x_0}, k = 0, 1, \cdots, h^{\varphi}$, $\forall \mathbf{w}[0 : h^{\varphi}-1] \in \mathcal{W}^{h^{\varphi}}$. By the virtue of Lemma 2, $\mathbf{x}'[0 : h^{\varphi}] \preceq \mathbf{x}^{x_0,ol}[0 : h^{\varphi}]$. Therefore, we have $\forall x_0 \in \mathcal{X}_0^{\max}, x'_0 \in L(x_0) \Rightarrow x'_0 \in \mathcal{X}_0^{\max}$, which indicates $\mathcal{X}_0^{\max}$ is a lower-set. □

Corollary 2. If $x_0 \in \mathcal{X}_0^{\max}$ and μ^{ol} is the following open-loop control policy

$$\mu_t^{ol}(x_0) = u_t^{ol,x_0}, t = 0, 1, \cdots, h^{\varphi} - 1,$$

then $\mathbf{x}(x'_0, \mu, \mathbf{w})[0 : h^{\varphi}] \models \varphi, \forall \mathbf{w} \in \mathcal{W}^{h^{\varphi}}, \forall x'_0 \in L(x_0)$.

Proof. Follows from the proof of Corollary 1. □

The system equations are brought into mixed-integer linear constraints by transforming system (5.1) into its MLD form. As mentioned in Chapter 2, any PWA system can be transformed into an MLD.

Finally, the set of constraints in Theorem 5 can be cast as:

$$\begin{cases} x_0^{ol,x_0} = x_0, & \text{Initial condition;} \\ x_{k+1}^{ol,x_0} = f^*(x_k^{ol,x_0}, u_k^{ol,x_0}), & \text{System constraints;} \\ z_k^{\pi} = (a_\pi^T x_k^{ol,x_0} \leq b_\pi), & \text{Predicates;} \\ z_0^{\varphi} = 1, \rho \geq 0, & \text{STL satisfaction.} \end{cases} \tag{5.7}$$

Checking the satisfaction of the set of constraints in (5.7) can be formulated as a MILP feasibility problem. For a fixed initial condition x_0, the feasibility of the MILP indicates whether $x_0 \in \mathcal{X}_o^{\max}$. An explicit representation of $\mathcal{X}_o^{\max}$ requires variable elimination from (5.7), which is computationally intractable for a large MILP. Alternatively, we can set x_0 as a free variable while maximizing a cost function (e.g. norm of x_0) such that a large $L(x_0)$ is obtained. Another natural candidate is maximizing $\rho(\mathbf{x}^{ol,x_0}, \varphi, 0)$. It is worth to note that by finding a set of distinct initial conditions and taking the union of all $L(x_0)$, we are able to find a representation for an under-approximation of $\mathcal{X}_o^{\max}$.

MILPs are NP-complete, but powerful off-the-shelf solvers exist. The complexity of solving (5.7) grows exponentially with respect to the number of binary variables and polynomially with respect to the number of continuos variables. The number of binary variables in our framework is $\mathcal{O}\left(h^{\varphi}(n_\pi + m_b + q_b + n_\delta)\right)$ - n_π is the number of predicates - and the number of continuous variables is $\mathcal{O}\left(h^{\varphi}(n + m_r + q_r + n_r)\right)$. In other words, the exponential growth builds upon the intricacy of the specification and the number of modes demonstrated by the hybrid nature of the system. However, the complexity is polynomial with respect to the dimension of the state.

Example 8. Consider the following switched system:

$$x^+ = e^{A_u \tau} x + A_u^{-1}(I - e^{-A_u \tau}) w,$$

where $x = (x_{[1]}, x_{[2]})^T \in \mathbb{R}^2_+$, $u \in \mathcal{U}$ is the control input (switch), $\mathcal{U} = \{1, 2\}$, and

$$A_1 = \begin{pmatrix} 1 & 1 \\ 1 & -5 \end{pmatrix}, A_2 = \begin{pmatrix} -8 & 1 \\ 1 & 2 \end{pmatrix}.$$

The (additive) disturbance w is bounded to $L(w^*)$, where $w^* = (1.5, 1)^T$ and $\tau = 0.1$. This system is the discrete-time version of $\dot{x} = A_u x + w$ with sample time τ. Both matrices are Metzler (all off-diagonal terms are non-negative hence all the elements of its exponential are positive) and non-Hurwitz hence constant input results in unbounded trajectories. The system is desired to satisfy the following STL formula:

$$\varphi = \bigvee_{T=0}^{10} \left(\mathbf{F}_{[0,T]} p_1 \wedge \mathbf{F}_{\{T\}} p_2 \right),$$

where $p_1 = \big((x_{[1]} \leq 1) \wedge (x_{[2]} \leq 5)\big)$ and $p_2 = \big((x_{[1]} \leq 5) \wedge (x_{[2]} \leq 1)\big)$. In plain English, φ states that "within 10 time units, the trajectory visits the box characterized by p_1 first and then the box corresponding to p_2" (see Fig. 5·1). We transformed this system into its MLD form. We formulated the constraints in (5.7) as a MILP and set the cost function to maximize $\|x_0\|_\infty$ and used the Gurobi [3] MILP solver. The solution was obtained in less than 0.05 seconds on a 3GHz Dual Core MacBook Pro. We obtained $x_0 = (2.82\ 2.82)^T$ and the following open-loop control sequence: 1 2 1 2 2 1 1 1 1 1. By applying this control sequence, we sampled a trajectory of the original system f with values of w drawn from a uniform distribution over $L(w^*)$. Both the trajectories of f and f^* satisfy the specification. The results are shown in Fig. 5·1.

---

[3] `www.gurobi.com`

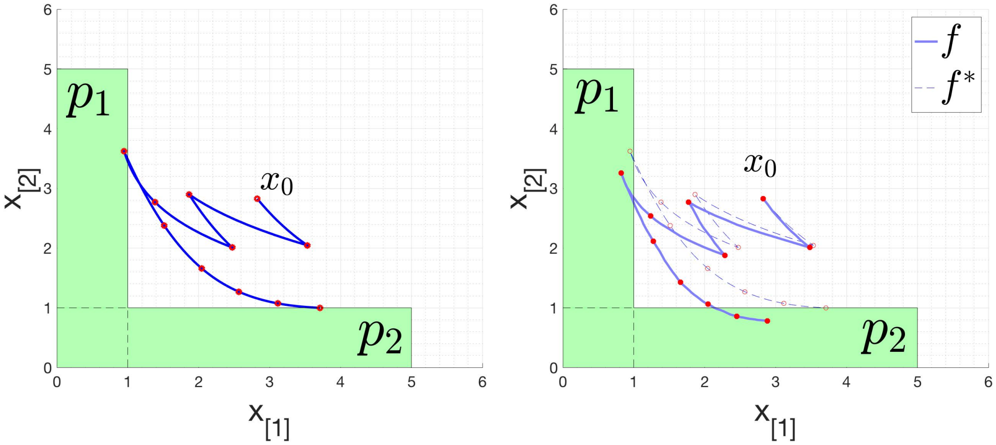


Figure 5·1: Example 8: [Left] The trajectory of the maximal system f^* which satisfies the specification. [Right] For the same controls, the trajectory of the original system f with w drawn from an uniform distribution over $L(w^*)$.

5.3 Infinite Horizon Semantics

In this section, we provide a solution to Problem 5. We show that the infinite-time property in (6.9) can be guaranteed using repetitive control sequences. First, we consider global specifications and extend the results from our previous work [Sadraddini and Belta, 2016b] in Sec. 5.3.1. Next, we show how to find controls for bounded-global STL formulas (Problem 5) in Sec. 5.3.2. Solution completeness is discussed in Sec. 5.3.3.

5.3.1 Global specifications: s-sequences and invariance

Consider the global specification $\mathbf{G}_{[0,\infty]}\varphi$, where φ is a bounded formula. We introduce some additional notation.

Definition 21. Given a bounded STL formula φ over predicates in the form (5.3), the *language realization set* (LRS) [Sadraddini and Belta, 2016a] is:

$$\mathcal{L}^{\varphi} := \left\{ x_0 x_1 \cdots x_{h^{\varphi}} \in \mathcal{X}^{h^{\varphi}} \middle| x_0 x_1 \cdots x_{h^{\varphi}} \models \varphi \right\}. \tag{5.8}$$

Proposition 1. The set $\mathcal{L}^{\varphi}$ is a lower-set.

Proof. For all $x_0 x_1 \cdots x_{h^{\varphi}} \in \mathcal{L}^{\varphi}$ and $x'_0 x'_1 \cdots x'_{h^{\varphi}} \preceq x_0 x_1 \cdots x_{h^{\varphi}}$, it follows from Lemma 2 that $x'_0 x'_1 \cdots x'_{h^{\varphi}} \models \varphi$. Thus, $x'_0 x'_1 \cdots x'_{h^{\varphi}} \in LRS(\varphi)$ hence $LRS(\varphi)$ is a lower set. ☐

It follows from the semantics of global operator that $\mathbf{x} \models \mathbf{G}_{[0,\infty]}\varphi$ is equivalent to $\mathbf{x}[t : t + h^{\varphi}] \in \mathcal{L}^{\varphi}, \forall t \in \mathbb{N}$.

Definition 22. A set $\Omega_{\mathcal{L}^{\varphi}} \subseteq \mathcal{L}^{\varphi}$ is a *robust control invariant* (RCI) set if:

$$\begin{array}{l} \forall\ x_0 x_1 \cdots x_{h^{\varphi}} \in \Omega_{\mathcal{L}^{\varphi}}, \exists u \in \mathcal{U}, \text{ s.t.} \\ x_1 x_2 \cdots x_{h^{\varphi}} f(x_{h^{\varphi}}, u, w) \ \in \Omega_{\mathcal{L}^{\varphi}}, \forall w \in \mathcal{W}. \end{array} \tag{5.9}$$

Satisfaction of $\mathbf{G}_{[0,\infty]}\varphi$ is accomplished by finding a RCI set in $\mathcal{L}^{\varphi}$. Note that unlike traditional definitions of RCI sets (e.g., [Blanchini, 1999]), where the set is defined in the state-space $\mathcal{X}$, our RCI set is defined in an augmented form of the state-space $\mathcal{X}^{h^{\varphi}}$. The language realization set can also be interpreted as the "safe" set in $(h^{\varphi} + 1)$-length trajectory space. The *maximal* RCI set inside $\mathcal{L}^{\varphi}$ provides a complete solution to the set-invariance problem. The computation of maximal RCI set requires implementing an iterative fixed-point algorithm which is computationally intensive for MLD systems and non-convex sets (see [Kerrigan, 2000, Raković et al., 2004] for discussion). We use monotonicity to provide an alternative approach. The following result is a more general version of the one in [Sadraddini and Belta, 2016b].

Theorem 6. Given a bounded formula φ, if there exists $\mathbf{x}^s[0 : h^{\varphi}] \in \mathcal{L}^{\varphi}$, and a sequence of controls: $u^s_0, \cdots, u^s_{T-1}$ - where T is a positive integer determining the length of the sequence - such that:

1. $\mathbf{x}^s[k : k + h^\varphi] \in \mathcal{L}^\varphi, k = 0, 1, \cdots, T$, where $x^s_{h^\varphi+k+1} = f^*(x^s_{h^\varphi+k}, u^s_k)$,
2. $\mathbf{x}^s[T : T + h^\varphi] \preceq \mathbf{x}^s[0 : h^\varphi]$,

then the following set is a RCI set in $\mathcal{L}^\varphi$:

$$\Omega_{\mathcal{L}^\varphi} := \bigcup_{k=0}^{T-1} L(\mathbf{x}^s[k : k + h^\varphi]). \tag{5.10}$$

Proof. For any $x'_0 x'_1 \cdots x'_{h^\varphi} \in \Omega^\varphi$, there exists $i \in \{0, 1 \cdots, T-1\}$ such that

$$x'_0 x'_1 \cdots x'_{h^\varphi} \in L(\mathbf{x}^s[i : i + h^\varphi]).$$

On one hand, we have $x^s_{i+1} \cdots x^s_{i+h^\varphi} f^*(x^s_{i+h^\varphi}, u^s_i) \in \Omega_{\mathcal{L}^\varphi}$. On the other hand, we have $x'_1 \preceq x^s_{i+1}, \cdots, x'_{h^\varphi} \preceq x^s_{i+h^\varphi}$. By applying u^s_i, monotonicity implies

$$\begin{aligned} & f(x'_{h^\varphi}, u^s_i, w) \preceq f^*(x^s_{i+h^\varphi}, u^s_i) = x^s_{i+1+h^\varphi}, \forall w \in \mathcal{W} \\ \Rightarrow\ & x'_1 x'_2 \cdots x'_{h^\varphi} f(x'_{h^\varphi}, u^s_i, w) \in \\ & L\left(x^s_{i+1} x^s_{i+2} \cdots x^s_{i+1+h^\varphi}\right)\right), \forall w \in \mathcal{W}. \end{aligned}$$

And the proof is complete from the fact that $x^s_{i+1} \cdots x^s_{i+1+h^\varphi} \in \Omega_{\mathcal{L}^\varphi}$ for all $i \in \{0, 1 \cdots, T-1\}$. □

Corollary 3. Let the conditions in Theorem 6 hold and $\mathbf{x}[t_0 : t_0 + h^\varphi] \in L(\mathbf{x}^s[0 : h^\varphi])$ for some $t_0 \in \mathbb{N}$. Consider the following control sequence starting from time $t_0 + h^\varphi$:

$$\mathbf{u}^s := \left(u^s_0 u^s_1 \cdots u^s_{T-1}\right)^\omega, \tag{5.11}$$

i.e., $u^s_t = u^s_{\mathrm{rem}(t-t_0-h^\varphi, T)}, t \geq t_0 + h^\varphi$. Let $x_{k+1} = f^*(x_k, u_k), k = t_0 + h^\varphi, \cdots$. Then we have $\mathbf{x}[t : t + h^{\varphi}] \in \Omega_{\mathcal{L}^{\varphi}}, \forall t \geq t_0$.

Proof. We prove by induction that $\mathbf{x}[t : t + h^{\varphi}] \in L(\mathbf{x}^s[\text{rem}(t - t_0, T) : \text{rem}(t - t_0) + h^{\varphi}]), \forall t \geq t_0$. The base case for $t = t_0$ is true. In order to prove the inductive step $\mathbf{x}[t + 1 : t + 1 + h^{\varphi}] \in L(\mathbf{x}^s[\text{rem}(t + 1 - t_0, T) : \text{rem}(t + 1 - t_0, T) + h^{\varphi}])$, we need to prove that $x_{t+k+1} \preceq x^s_{\text{rem}(t+1-t_0,T)+k}, k = 0, \cdots, h^{\varphi}$, for which we need to only prove the case for $k = h^{\varphi}$ as previous inequalities are already assumed by induction. We show $x_{t+h^{\varphi}+1} \preceq x^s_{\text{rem}(t+1-t_0,T)+h^{\varphi}}$ through monotonicity and the induction assumption that $x_{t+h^{\varphi}} \preceq x^s_{\text{rem}(t-t_0,T)+h^{\varphi}}$:

$$\begin{aligned} x_{t+h^{\varphi}+1} &= f^*(x_{t+h^{\varphi}}, u^s_{\text{rem}(t-t_0,T)}) \\ &\preceq f^*(x^s_{\text{rem}(t-t_0,T)+h^{\varphi}}, u^s_{\text{rem}(t-t_0,T)}) \\ &= x^s_{\text{rem}(t-t_0,T)+1+h^{\varphi}} \preceq x^s_{\text{rem}(t+1-t_0,T)+h^{\varphi}}. \end{aligned}$$

Note that $x^s_{T+h^{\varphi}} \preceq x^s_{h^{\varphi}}$. The "$\preceq$" in the last line can be replaced by "$=$" when $\text{rem}(t - t_0, T) + 1 \neq T$. □

We refer to the repetitive sequence of controls in (5.11) as an *s-sequence*. An s-sequence is an invariance inducing open-loop control policy. Once the latest $h^{\varphi} + 1$-length of system state are brought into $\Omega_{\mathcal{L}^{\varphi}}$, an s-sequence keeps the $h^{\varphi} + 1$-length trajectory of the system in $\Omega_{\mathcal{L}^{\varphi}}$ for all subsequent times.

The computation of an s-sequence requires solving an MILP for $\mathbf{x}^s[h^{\varphi} : T] \models \mathbf{G}_{[0,T]}\varphi$ (an instance of Problem 4) with an additional set of constraints in $\mathbf{x}^s[0 : h^{\varphi}] \models \varphi$ (again, an instance of Problem 4, but without the dynamical constraints. In other words, $\mathbf{x}^s[0 : h^{\varphi}]$ does not need to be a trajectory of the maximal system), and $\mathbf{x}^s[T : T + h^{\varphi}] \preceq \mathbf{x}^s[0 : h^{\varphi}]$ (linear constraints). We are usually interested in the shortest s-sequence since its computation requires the smallest MILP. Algorithmically, we start from $T = 1$ and implement $T \leftarrow T + 1$ until the MILP formulating the

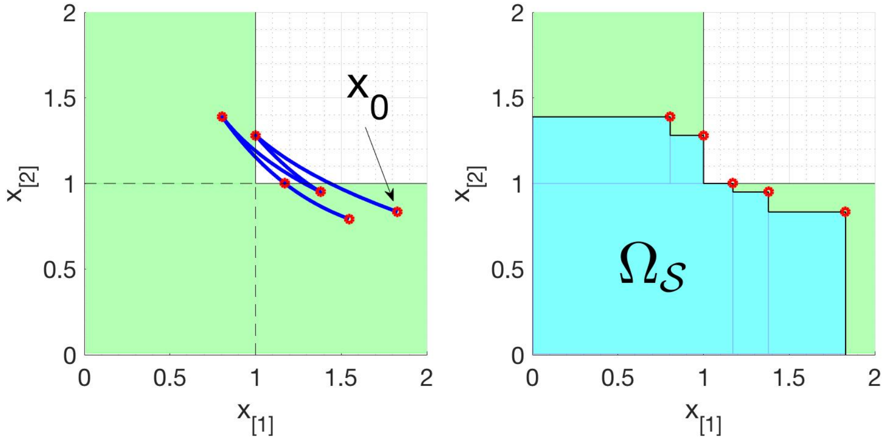


Figure 5·2: Example 9: [Left] The trajectory satisfying the conditions of s-sequences. [Right] The corresponding robust control invariant set inside $\mathcal{S}$.

conditions in Theorem 6 becomes feasible and an s-sequence is found. As it will be implied from results in Sec. 5.3.3, existence of an s-sequence is almost necessary for existence of a RCI set.

Example 9. Consider the system in Example 8. We wish to keep the trajectory in the set characterized by $p_1 \vee p_2$, i.e., $\mathcal{S} = L\left((1,5)^T\right) \cup L\left((5,1)^T\right)$. Note that this set is non-convex. We set the cost function to maximize $\|x_0\|_1$. The shortest s-sequence has $T = 5$ and is: $(2\ 1\ 2\ 1\ 1)^\omega$. The resulting trajectory satisfying the definition of s-sequence is shown in Fig. 5·2 (a). The corresponding robust control invariant set is shown in Fig. 5·2. (b) (cyan region), which is characterized by the $x_0^s, x_1^s, \cdots, x_4^s$ (red dots) that lie inside $\mathcal{S}$ (green region). Note that the $[0,2] \times [0,2]$ portion of the coordinates in Fig. 5·1 is shown here for a clearer representation of the details.

5.3.2 Bounded-global specifications: ϕ-sequences

Now we consider general bounded-global formulas - as in Problem 5 - and generalize the paradigm used for s-sequences. We provide the key result of this section.

Theorem 7. Given a bounded-global STL formula $\phi = \varphi_b \wedge \mathbf{G}_{[\Delta,\infty]}\varphi_g$, an initial condition x_0, a control sequence $u_0^\phi \cdots u_{\Delta+T+h^{\varphi_g}-1}^\phi$, where T is a positive integer, and a non-negative integer $T_0 < T$, let the following conditions hold:

1. $\mathbf{x}^\phi[0 : \Delta + T + h^{\varphi_g}] \models \varphi_b \wedge \mathbf{G}_{[\Delta,\Delta+T]}\varphi_g$,
2. $\mathbf{x}^\phi[\Delta + T : \Delta + T + h^{\varphi_g}] \preceq \mathbf{x}^\phi[\Delta + T_0 : \Delta + T_0 + h^{\varphi_g}]$;

where $x_{k+1}^\phi = f^*(x_k^\phi, u_k^\phi), k \in [0, \Delta + T + h_g^\varphi - 1]$, $x_0^\phi = x_0$. Let μ^{ol} be the open-loop control policy corresponding to the following control sequence:

$$\mathbf{u}^\phi := u_0^\phi \cdots u_{\Delta+T_0+h^{\varphi_g}-1}^\phi \left(u_{\Delta+T_0+h^{\varphi_g}}^\phi \cdots u_{\Delta+T+h^{\varphi_g}-1}^\phi\right)^\omega, \tag{5.12}$$

Then $\mathbf{x}(x_0', \mu^{ol}, \mathbf{w}) \models \phi, \forall \mathbf{w} \in \mathcal{W}^\omega, \forall x_0' \in L(x_0)$. Moreover, the following set is a RCI set in $\mathcal{L}^{\varphi_g}$:

$$\Omega_{\mathcal{L}^{\varphi_g}} := \bigcup_{i=0}^{T-T_0-1} L(\mathbf{x}^\phi[\Delta + T_0 + i : \Delta + T_0 + h^{\varphi_g} + i]). \tag{5.13}$$

Proof. We need to prove that $\mathbf{x}(x_0^\phi, \mu^{ol}, \mathbf{w}^*) \models \phi$, where $\mathbf{w}^* = (w^*)^\omega$. The fact that $\mathbf{x}(x_0', \mu^{ol}, \mathbf{w})[0] \models \phi, \forall x_0' \in L(x_0), \forall \mathbf{w} \in \mathcal{W}^\omega$, follows from monotonicity and Lemma 2. The fact that $\Omega_{\mathcal{L}^{\varphi_g}}$ is a RCI set follows from Theorem 6 as (5.10) is obtained from replacing $\Delta = T_0 = 0$ in (5.13). It follows that $\left(u_{\Delta+T_0+h^{\varphi_g}}^\phi \cdots u_{\Delta+T+h^{\varphi_g}-1}^\phi\right)^\omega$ is an s-sequence. For all $t \geq \Delta + T + h^{\varphi_g}$, let

$$x_{t+1}^\phi = f^*(x_t^\phi, u_{\Delta+T_0+h^{\varphi_g}+\mathrm{rem}(t-\Delta-T_0-h^{\varphi_g}, T-T_0)}^\phi). \tag{5.14}$$

Using Corollary 3, we have $\mathbf{x}^\phi[k + \Delta + T_0 : k + \Delta + T_0 + h^{\varphi_g}] \in \mathcal{L}^{\varphi_g}, \forall k \in \mathbb{N}$, and the proof is complete.

□

We refer to the sequence of controls in (5.12) as a ϕ*-sequence*. The computation of a ϕ-sequence requires solving an MILP for $\mathbf{x}^{\phi}[0:\Delta+T+h^{\varphi_g}] \models \varphi_b \wedge \mathbf{G}_{[\Delta,\Delta+T]}\varphi_g$ (an instance of Problem 4) with an additional set of constraints in $\mathbf{x}^{\phi}[\Delta{+}T:\Delta{+}T{+}h^{\varphi_g}] \preceq \mathbf{x}^{\phi}[\Delta+T_0:\Delta+T_0+h^{\varphi_g}]$ (linear constraints). Thus, similar to s-sequecnes, the computation of a ϕ-sequence is based on feasibility checking of a MILP. We have two parameters T and $T_0 < T$ to search over. We start from $T = 1$ and implement $T \leftarrow T+1$, while checking for all $T_0 < T$, until the corresponding MILP gets feasible. In Sec. 5.3.3, we discuss the necessity of existence of a feasible solution for some T_0, T.

Another interpretation of a ϕ-sequence is a sequence that consists of a initialization segment of length $\Delta+h^{\varphi_g}$ to bring the latest h^{φ_g} states of the system into $\Omega_{\mathcal{L}^{\varphi_g}} \subseteq \mathcal{L}^{\varphi_g}$ and a repetitive segment of length T to stay in $\Omega_{\mathcal{L}^{\varphi_g}}$. The repetitive segment is an s-sequence. Since control inputs eventually becoming periodic, the long-term behavior is expected to demonstrate periodicity, which leads to the following result based on Theorem 7.

Corollary 4. The ω-limit set of the run given by (5.14) is non-empty and corresponds to the following periodical orbit:

$$\left(x_0^{\phi,\infty} x_1^{\phi,\infty} \cdots x_{T-T_0-1}^{\phi,\infty}\right)^{\omega}, \tag{5.15}$$

where $x_k^{\phi,\infty} := \lim_{c\to\infty} x^{\phi}_{k+\Delta+T_0+c(T-T_0)}, k = 0, \cdots, T-T_0-1$.

Proof. We show that $x_t^{\phi} \preceq x^{\phi}_{t+T-T_0}, \forall t \geq \Delta+T_0$. Similar to the proof of Corollary 3, we use induction. The base case for $t = \Delta+T_0$ is already in the second condition in

Theorem 7. The inductive step is proven as follows:

$$x^{\phi}_{t+1+T-T_0} = f^*(x^{\phi}_{t+T-T_0}, u_{T-T_0+t}) \preceq f^*(x^{\phi}_t, u_t) = x^{\phi}_{t+1},$$

where from (5.12) we have

$$u_{t+T-T_0} = u_t = u^{\phi}_{t+\Delta+T_0+h^{\varphi_g}+\text{rem}(t-\Delta-T_0-h^{\varphi_g},T-T_0)}.$$

Thus, each component of the sequence $x^{\phi}_{\Delta+T_0+k}x^{\phi}_{\Delta+T+k}x^{\phi}_{\Delta+2T-T_0+k}\cdots$, $k = 0, \cdots, T-T_0$, is monotonically decreasing. *Monotone convergence theorem* [Sutherland, 1975] explains that a lower-bounded monotonically decreasing sequence converges (in this case, all values are lower-bounded by zero). Thus, $\lim_{c\to\infty} x^{\phi}_{\Delta+T_0+k+c(T-T_0)}, k = 0, \cdots, T-T_0$, exists and the proof is complete. □

Example 10. Consider the system in Example 8. We wish to satisfy

$$\phi = \mathbf{F}_{[0,5]}p_1 \ \wedge \ \mathbf{G}_{[5,\infty)} \left(\mathbf{F}_{[0,6]}p_1 \ \wedge \ \mathbf{F}_{[0,6]}p_2\right).$$

The specification is in form in (5.5) with $h^{\varphi_b} = \Delta = 5, h^{\varphi}_g = 6$. This specification requires that p_1 is visited at least once until $t = 5$ and, afterwards, p_1 and p_2 are persistently visited while the maximum time between two subsequent visits is not greater than 6. We find a ϕ-sequence solving a MILP for $T = 7, T_0 = 0$, while maximizing $\|x_0\|_1$. The obtained ϕ-sequence is $\mathbf{u}^{\phi} = 2\,2\,2\,2\,1\,2\,1\,1\,1\,1\,2\,(2\,2\,1\,1\,1\,1\,2)^{\omega}$ for $x_0 = (12.4, 0)^T$. The first $\Delta + T + h^{\varphi_g} + 1 = 5 + 7 + 6 + 1 = 19$ time points of the trajectory of the maximal system f^* satisfying the conditions in Theorem 7 is shown in Fig. 5·3 [Left]. A sample trajectory of f with values of $\mathbf{w}$ chosen uniformly from $\mathcal{W}$ is also shown. Both trajectories satisfy ϕ. The limit-set of f^*, which is a

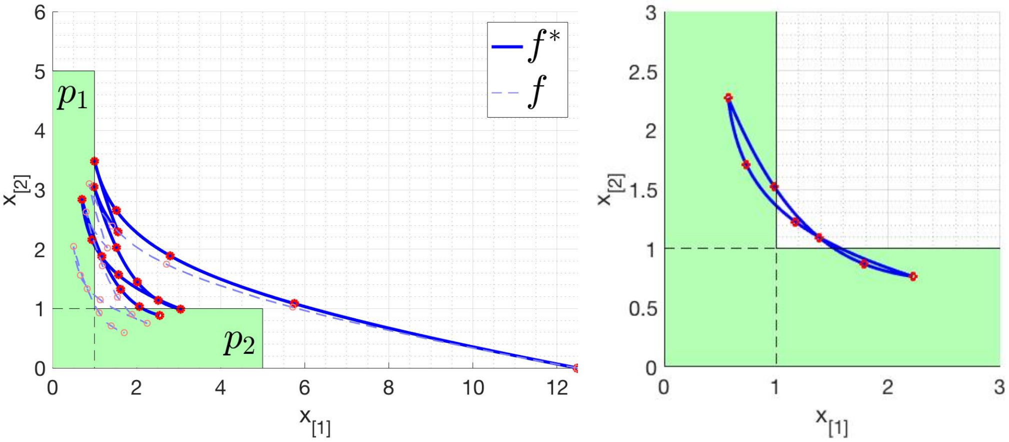


Figure 5·3: Example 10: [Left] The first 19 points of the trajectory of the maximal system f^* that satisfy the conditions in Theorem 7. A sample trajectory of f is also shown. [Right] The ω-limit set (red dots) of f^* is a 7-periodic orbit.

7-periodical orbit, is shown in Fig. 5·3 [Right].

5.3.3 Necessity of Open-loop Strategies

We showed that if there exists an initial condition and a finite length control sequence such that the statements in Theorem 6 hold, an open-loop control sequence is sufficient for satisfying of a bounded-global formula, as was formulated in Problem 5. In this section, we address the necessity conditions.

We show that the existence of open-loop control strategies for satisfying a bounded-global specifications is *almost necessary* in the sense that if a ϕ-sequence is not found using Theorem 7 for large values of T, then it is almost certain that no correct control policy (including feedback policies) exists, or, if exists any, it is *fragile* in the sense that a slight increase in the effect of the disturbances makes the policy invalid. We characterize the necessity conditions based on hypothetical perturbations in the disturbance set.

Theorem 8. Suppose system (5.1) is strongly monotone with respect to the maximal disturbance in the sense that for all $\epsilon > 0$, there exists a perturbed disturbance set $\mathcal{W}_p$ with maximal disturbance w_p^* such that

$$\forall x \in \mathcal{X}, \forall u \in \mathcal{U}, f(x,u,w^*) + 1_n\epsilon \preceq f(x,u,w_p^*). \tag{5.16}$$

Consider the bounded-global formula $\phi = \varphi_b \wedge \mathbf{G}_{[\Delta,\infty]}\varphi$. Given $\epsilon > 0$, the disturbance set is altered to $\mathcal{W}_p$ such that (5.16) holds. If there exists a control policy μ and an initial condition x_0 such that $\mathbf{x}(x_0, \mu, \mathbf{w}_p) \models \phi, \forall \mathbf{w}_p \in \mathcal{W}_p^{\omega}$, then there exists at least one open-loop control policy μ^{ol} in the form of a ϕ-sequence in (5.12) for the original system such that

$$T \leq {}^{A}\!/_{\epsilon^{n(h^{\varphi_g}+1)}}, \tag{5.17}$$

where A is a constant depending on $\mathcal{L}^{\varphi_g}$.

Proof. Given a bounded set $\mathcal{C} \subset \mathbb{R}^{n(h^{\varphi}+1)}$, we define the diameter $d(\mathcal{C}) := \inf\{d | s_1 \preceq s_2 + d1_{n(h^{\varphi}+1)}, \forall s_1, s_2 \in \mathcal{C}\}$ (e.g., the diameter of an axis-aligned hyper-box is equal to the length of its largest side). Consider a partition of $\mathcal{L}^{\varphi}$ by a finite number of cells, where the diameter of each cell is less than ϵ. The maximum number of cells required for such a partition is ${}^{A}\!/_{\epsilon^{n(h^{\varphi_g}+1)}}$, where A is a constant dependent on the shape and volume of $\mathcal{L}^{\varphi_g}$. A conservative upper bound on A can be given as follows. Define $a^* \in \mathbb{R}_+$ as

$$\arg\min_{a/\epsilon \in \mathbb{N}} \Big\{ \mathbf{x}[0:h^{\varphi_g}] \preceq a\mathbf{1}_n[0:h^{\varphi_g}], \forall \mathbf{x}[0:h^{\varphi_g}] \in \mathcal{L}^{\varphi_g} \Big\}.$$

Since $\mathcal{L}^{\varphi_g}$ is bounded and closed, a^* exists. We have $\mathcal{L}^{\varphi_g} \subseteq L(a^* 1_{n(h^{\varphi_g}+1)})$. Let A be $a^{*n(h^{\varphi_g}+1)}$ - the volume of $L(a^* 1_{n(h^{\varphi_g}+1)})$, which is a hyper-box. Partition $L(a^* 1_{n(h^{\varphi_g}+1)})$ into $N := {}^{A}\!/_{\epsilon^{n(h^{\varphi_g}+1)}}$ number of equally sized cubic cells with side length of ϵ. Such a partition also partitions $\mathcal{L}^{\varphi_g}$ to at most N number of cells where the diameter of each cell is not greater than ϵ.

Since there exists μ such that $\mathbf{x}(x_0, \mu, \mathbf{w}_p) \models \phi, \forall \mathbf{w}_p \in \mathcal{W}_p^\omega$, then there exist at least one run satisfying ϕ for system $x_{k+1} = f(x_k, u_k, w_p^*)$. Let $x_0, \cdots, x_{\Delta+h^{\varphi_g}+N}$ be the first $\Delta + h^{\varphi_g} + N + 1$ time points of a such a run. We have $\mathbf{x}[k : k + h^{\varphi_g}] \in \mathcal{L}^{\varphi_g}, k = \Delta, \cdots, \Delta + N$. Consider the sequence $\mathbf{x}[\Delta : \Delta + h^{\varphi_g}]\mathbf{x}[\Delta+1 : \Delta+1+h^{\varphi_g}] \cdots \mathbf{x}[\Delta+N : \Delta + N + h^{\varphi_g}]$. Consider a partition of $\mathcal{L}^{\varphi_g}$ with cells that for all cells the diameter is less than ϵ. By the virtue of *pigeonhole principle*, there exists a cell such that contains at least two time points $\mathbf{x}[k_1 : k_1 + h^{\varphi_g}]$ and $\mathbf{x}[k_2 : k_2 + h^{\varphi_g}]$, $\Delta \leq k_1 \leq k_2 \leq \Delta + N$. From the assumption on the diameter of the cells we have

$$\mathbf{x}[k_2 : k_2 + h^{\varphi_g}] \preceq \mathbf{x}[k_1 : k_1 + h^{\varphi_g}] + \epsilon \mathbf{1}_n[0 : h^{\varphi_g}]. \tag{5.18}$$

Now consider system $x'_{k+1} = f(x'_k, u_k, w^*)$ - the original maximal system - with $x'_{k_1+h^{\varphi_g}} = x_{k_1+h^{\varphi_g}}$. We prove that

$$x'_k + 1_n \epsilon \leq x_k, \forall k > k_1 + h^{\varphi_g}. \tag{5.19}$$

We use induction. The base case for $k = k_1 + h^{\varphi_g} + 1$ is verified using (5.16):

$$\begin{aligned} x'_{k_1+1+h^{\varphi_g}} + 1_n \epsilon = & \; f(x'_{k_1+h^{\varphi_g}}, u_{k_1+h^{\varphi_g}}, w^*) + 1_n \epsilon \\ \leq & \; f(x'_{k_1+h^{\varphi_g}}, u_{k_1+h^{\varphi_g}}, w_p^*) \\ = & \; x_{k_1+1+h^{\varphi_g}}. \end{aligned}$$

The inductive step is verified using monotonicity and (5.16):

$$\begin{aligned} x'_{k+1+h^{\varphi_g}} + 1_n \epsilon = & \; f(x'_{k+h^{\varphi_g}}, u_{k+h^{\varphi_g}}, w^*) + 1_n \epsilon \\ \leq & \; f(x'_{k+h^{\varphi_g}}, u_{k+h^{\varphi_g}}, w_p^*) \\ \leq & \; f(x_{k+h^{\varphi_g}}, u_{k+h^{\varphi_g}}, w_p^*) = x_{k+1+h^{\varphi_g}}. \end{aligned}$$

It immediately follows from (5.19) that

$$\mathbf{x}'[k_2 : k_2 + h^{\varphi_g}] + \epsilon\mathbf{1}_n[0 : h^{\varphi_g}] \leq \mathbf{x}[k_2 : k_2 + h^{\varphi_g}]. \tag{5.20}$$

Since the lefthand of (5.18) is the righthand of (5.20), we have:

$$\mathbf{x}'[k_2 : k_2 + h^{\varphi_g}] \leq \mathbf{x}[k_1 : k_1 + h^{\varphi_g}]. \tag{5.21}$$

This is reminiscent of the conditions in Theorem 6. Now by defining $x'_k := x_k, k = k_1, \cdots, k_1 + h^{\varphi_g} - 1$, we conclude that

$$\Omega'_{\mathcal{L}^{\varphi_g}} := \bigcup_{k=k_1}^{k_2-1} L(\mathbf{x}'[k : k + h^{\varphi}])$$

is a RCI set for system with adversarial disturbance set $\mathcal{W}$ and $(u_{k_1} \cdots u_{k_2+h_g^{\varphi}-1})^{\omega}$ is an s-sequence.

Now, once again, consider the original system $x'_{k+1} = f(x'_k, u_k, w^*)$ with $x'_0 = x_0$. Monotonicity implies $\mathbf{x}'[0 : k_1 + h^{\varphi_g}] \leq \mathbf{x}[0 : k_1 + h^{\varphi_g}]$. Thus, by applying $u_0, \cdots, u_{k_1+h^{\varphi_g}-1}$ and using Lemma 2, we have $\mathbf{x}'[0 : k_1 + h^{\varphi_g}] \models \varphi_b \wedge \mathbf{G}_{[\Delta,k_1]}\varphi_g$. Corollary 3 implies $x'[k_1 + h^{\varphi_g}] \models \mathbf{G}_{[0,\infty)}\varphi_g$ if $(u_{k_1}, \cdots, u_{k_2-1})^{\omega}$ is applied starting from time k_1. Finally, monotonicity and Lemma 2 immediately indicate that $\mathbf{x}(x''_0, \mu^{ol}, \mathbf{w}) \models \phi, \forall x''_0 \in L(x_0), \forall \mathbf{w} \in \mathcal{W}^{\omega}$, where μ^{ol} is the following open-loop control strategy producing the following control sequence:

$$u_0 \cdots u_{k_1-1}(u_{k_1} \cdots u_{k_2+h_g^{\varphi}-1})^{\omega},$$

which is in form of (5.12) with $k_1 = \Delta + T_0 + h^{\varphi_g}$ and $k_2 = \Delta + T$. Since $k_2 \leq \Delta + N$, we also have $T \leq N, N = {}^{A}/_{\epsilon^{n(h^{\varphi_g}+1)}}$, and the proof is complete.

$\square$

Corollary 5. Suppose that for all $T \leq T^{\max}, T_0 < T$, there does not exist an initial condition and a control sequence such that the conditions in Theorem 7 hold. Then there does not exist any solution to Problem 5 given that the maximal disturbance is w_p^* such that (5.16) holds with $\epsilon > \sqrt[n(h^{\varphi}g+1)]{T^{\max}}$.

The relation between the fragility in Theorem 8 and the length of the ϕ-sequence suggests that by performing the search for longer ϕ-sequences (which are computationally more difficult), the bound for fragility becomes smaller, implying that a correct control policy (if exists) is close to the limits (i.e., robustness score is close to zero, or the constraints are barely satisfied in the case with maximal disturbance). In practice, the bounds in Theorem 8 are very conservative and one may desire to find tighter bounds for specific applications.

Example 11. Consider Example 9. Suppose that there does not exist an s-sequence of length smaller than 144 with maximal disturbance w^*. The constant A (area in this 2D case, see proof of Theorem 8) of region corresponding to $p_1 \vee p_2$ is 9. Therefore, $\mathcal{S}$ can be partitioned into 144 equally sized square cells with side length 0.25. Note that we have $\epsilon^2 \geq 9/T$. Since the disturbances are additive, it follows that if $A_u^{-1}(I - e^{-A_u t})(w_p^* - w^*) > (0.25, 0.25)^T, u = 1, 2$, then there does *not* exist any control strategy μ and $x_0 \in \mathbb{R}_+^n$ such that $\mathbf{x}(x_0, \mu, \mathbf{w}_p) \models \mathbf{G}_{[0,\infty]}(x \in \mathcal{S}), \forall \mathbf{w}_p \in \mathcal{W}_p^{\omega}$.

5.4 Model Predictive Control

In this section, we provide a solution to Problem 6. We assume full knowledge of the history of state. As mentioned in Sec. 8.1, the cost function J is assumed to be non-decreasing with respect to the state values hence the system constraints are replaced with those of the maximal system. First, we explain the MPC setup for global STL formulas. Next, we prove that the proposed framework is guaranteed to generate runs that satisfy the global STL specification (6.9).

Let $t \geq h^\varphi - 1$. The case of $t < h^\varphi - 1$ is explained later. Given planning horizon H, the states that are predictable at time t using controls in u_t^H are $x_{1|t}, x_{2|t}, \cdots, x_{H|t}$. Given predictions $x_{1|t}, x_{2|t}, \cdots, x_{H|t}$, we need to enforce $\mathbf{x}[t-h^\varphi+1, t+H] \models \mathbf{G}_{[0,H-1]}\varphi$ at time t. Notice that

$$\mathbf{x}_{|t}[t-h^\varphi+1, t+H] := x_{t-h^\varphi+1} \cdots x_t x_{1|t} \cdots x_{H|t}, \tag{5.22}$$

i.e., the first h^φ time points are actual values, the rest are predictions. Also, note that the values in $\mathbf{x}[\tau : \tau + h^\varphi]$ are independent of the values in x_t^H for $\tau \leq t - h^\varphi$ and are not fully available for $\tau > t + H - h^\varphi$. Thus, $[t - h^\varphi + 1, t + H - h^\varphi]$ is the time window for imposing constraints at time t [Sadraddini and Belta, 2015].

The MPC optimization problem is initially written as (we do not use it for control synthesis as explained shortly):

$$\begin{array}{rl} \text{minimize} & J\left(x_t^H, u_t^H\right), \\ \text{s.t.} & x_{k+1|t} = f^*(x_{k|t}, u_{k|t}), k = 0, \cdots, H-1, \\ & \mathbf{x}_{|t}[t-h^\varphi+1, t+H] \models \mathbf{G}_{[0,H-1]}\varphi. \end{array} \tag{5.23}$$

The set of constraints in (5.23) requires the knowledge of $x_{t-h^\varphi+1} x_{t-h^\varphi+2} \cdots x_t$. Thus, the proposed control policy requires a finite memory for the history of last h^φ states. As it will be shown in Proposition 2, persistent feasibility of the constraints in (5.23) leads to fulfilling $\mathbf{G}_{[0,\infty]}\varphi$. However, persistent feasibility of the MPC setup in (5.23) is not guaranteed. We address this issue for the remainder of this section.

Definition 23. An MPC strategy is *recursively feasible* if, for all $t \in \mathbb{N}$, the control at time t is selected such that the MPC optimization problem at $t+1$ becomes feasible.

Our goal is to modify (5.23) such that it becomes recursively feasible. It is known that adding a (the maximal) RCI set acting as a terminal constraint is sufficient (and necessary) to guarantee recursive feasibility [Kerrigan and Maciejowski, 2001]. We add the terminal constraint $\mathbf{x}[t+H-h^{\varphi}:t{+}H] \in \Omega_{\mathcal{L}^{\varphi}}$ to (5.23) to obtain:

$$\begin{array}{rl} u_t^{H,opt} = & \underset{u_t^H \in \mathcal{U}^H}{\arg\min}\ J\left(x_t^H, u_t^H\right), \\ \text{s.t.} & x_{k+1|t} = f^*(x_{k|t}, u_{k|t}), k = 0, \cdots, H-1, \\ & \mathbf{x}_{|t}[t-h^{\varphi}+1, t+H] \models \mathbf{G}_{[0,H-1]}\varphi, \\ & \mathbf{x}_{|t}[t+H-h^{\varphi}:t+H] \in \Omega_{\mathcal{L}^{\varphi}}. \end{array} \tag{5.24}$$

Proposition 2. Let $\mu_t(x_0, \cdots, x_t) = \mu_t(x_{t-h^{\varphi}+1}, \cdots, x_t) = u_{0|t}^{H,opt}$, where $u^{H,opt} = u_{0|t}^{H,opt} \cdots u_{H-1|t}^{H,opt}$ is given by (5.24). If the optimization problem (5.24) is feasible for all $t \geq h^{\varphi}-1$, then $\mathbf{x}(x_0, \mu, \mathbf{w})[0] \models \mathbf{G}_{[0,\infty]}\varphi, \forall \mathbf{w} \in \mathcal{W}^{\omega}$.

Proof. We show that $\mathbf{x}(x_0, \mu, \mathbf{w})[0:k{+}h^{\varphi}] \models \mathbf{G}_{[0,k]}\varphi, \forall \mathbf{w} \in \mathcal{W}^*, \forall k \in \mathbb{N}$, using induction over k. Consider (5.24) for $t = k+h^{\varphi}-1$ for any $k \in \mathbb{N}$. The second constraint in (5.24) requires $\mathbf{x}_{|t}[k, k+h^{\varphi}] \models \varphi$, or equivalently, $x_k \cdots x_{k+h^{\varphi}-1} x_{1|k+h^{\varphi}-1} \models \varphi$. By applying $u_{0|t}^{opt}$, monotonicity implies $x_{k+h^{\varphi}} = f(x_{k+h^{\varphi}-1}, u_{0|t}^{opt}, w) \preceq x_{1|k+h^{\varphi}-1} = f^*(x_{k+h^{\varphi}-1}, u_{0|t}^{opt}), \forall w \in \mathcal{W}$. From Lemma 2 we have $\mathbf{x}[k:k+h^{\varphi}] \models \varphi$. Thus, we have shown $\mathbf{x}[k:k+h^{\varphi}] \models \varphi, \forall k \in \mathbb{N}$, and the proof is complete. □

Proposition 3. The MPC strategy corresponding to (5.24) is recursively feasible.

Proof. Suppose $u_t^H = u_{0|t} \cdots u_{H-1|t}$ and $x_t^H = x_{t+1|t} \cdots, x_{t+H-1|t}$ is a feasible solution for (5.24) at time t. Since $\Omega_{\mathcal{L}^{\varphi}}$ is a RCI set, there exist $u^r \in \mathcal{U}$ such that $\mathbf{x}_{|t}[t+H+1-h^{\varphi}:t+H+1] = x_{H-h^{\varphi}+1|t} x_{H-h^{\varphi}+2|t} \cdots x_{H|t} f(x_{H|t}, u^r, w) \in \Omega_{\mathcal{L}^{\varphi}}, \forall w \in \mathcal{W}$. Suppose $u_{0|t}$ is applied to the system. We have $x_{t+1} = f(x_t, u_{0|t}, w) \preceq f^*(x_t, u_{0|t}) = x_{1|t}, \forall w \in \mathcal{W}$.

Now, we prove that the optimization problem at time $t+1$ is feasible by showing that at least one feasible solution. Let $u_{t+1}^{H} = u_{1|t}u_{2|t}\cdots, u_{H|t}u^{r}$. We already showed that $x_{t+1} = x_{0|t+1} \preceq x_{1|t}$. By induction and using monotonicity, it follows that $x_{k|t+1} \preceq x_{k+1|t}, k =, 1, \cdots, H-2$. Thus, we have $x_{t-h^{\varphi}+2}\cdots x_{t+1}x_{1|t+1}\cdots x_{H-1|t+1} \preceq x_{t-h^{\varphi}+2}\cdots x_{1|t}x_{2|t}\cdots x_{H|t}$, which using Lemma 2 establishes that

$$x_{t-h^{\varphi}+2}\cdots x_{t+1}x_{1|t+1}\cdots x_{H-1|t+1} \models \mathbf{G}_{[0,H-1]}\varphi.$$

In order to complete the proof, it remains to show that

$$\mathbf{x}[t+H+1-h^{\varphi} : t+H+1] = x_{H+1-h^{\varphi}|t}\cdots x_{H|t+1} \models \varphi.$$

This follows from invariance. Note that $x_{H|t+1} = f^{*}(x_{H|t}, u^{r})$. Therefore, we have $x_{H+1-h^{\varphi}|t}\cdots\cdots x_{H|t+1} \in \Omega_{\mathcal{L}^{\varphi}}$, and since $\Omega_{\mathcal{L}^{\varphi}} \in \mathcal{L}^{\varphi}$, we have $x_{H+1-h^{\varphi}|t}\cdots\cdots x_{H|t+1} \models \varphi$, and the proof is complete. $\square$

The MPC optimization problem is also converted into a MILP problem. It is computationally easier to solve the optimization problem in (5.24) by solving T MILPs:

$$\begin{array}{rl} u_{t}^{opt,H} = & \underset{u_{t}^{H}\in\mathcal{U}^{H}, i=0,\cdots,T-1}{\arg\min} \quad J\left(x_{t}^{H}, u_{t}^{H}\right), \\ \text{s.t.} & x_{k+1|t} = f^{*}(x_{k|t}, u_{k|t}), k = 0, \cdots, H-1, \\ & \mathbf{x}_{|t}[t-h^{\varphi}+1, t+H] \models \mathbf{G}_{[0,H-1]}\varphi, \\ & \mathbf{x}_{|t}[t+H-h^{\varphi} : t+H] \in L(\mathbf{x}^{\varphi,x_0}[i : i+h^{\varphi}]). \end{array} \tag{5.25}$$

Note that all MILPs can be aggregated into a single large MILP in the expense of additional constraints for capturing non-convexities of the terminal condition.

Finally, consider $t < h^{\varphi}$. In this case, we require $H \geq h^{\varphi}$ and replace the interval $[t-h^{\varphi}+1, t+H-h^{\varphi}]$ with $[0, t+H-h^{\varphi}]$ for $t < h^{\varphi}$ in (5.25). For applications where initialization is not important in long-term (like traffic management), a simpler approach is to initialize the MPC from $t = h^{\varphi} - 1$ and assume all previous state values are zero (hence all the past predicates are evaluated as true).

Remark 2. In our previous work on STL MPC of linear systems [Sadraddini and Belta, 2015], we did not establish recursive feasibility. In order to recover from possible infeasibility issues, we proposed maximizing the STL robustness score (a negative value) whenever the MPC optimization problem became infeasible. Although recursive feasibility is guaranteed here, un-modeled disturbances and initial conditions outside $\mathcal{X}_0^{\max}$ can lead to infeasibility. The formalism in [Sadraddini and Belta, 2015] can be used to recover from infeasibility with minimal violation of the specification.

5.5 Application to Traffic Management

In this section, we explain how to apply our methods to traffic management. First, the model that we use for traffic networks is explained, which is similar to the one in [Coogan et al., 2016b] but freeways are also modeled. Next, the monotonicity properties of the model are discussed. We show that there exists a congestion-free set in the state-space in which the traffic dynamics is monotone. Finally, a case study on a mixed urban and freeway network is presented.

5.5.1 Model

The topology of the network is described by a directed graph $(\mathcal{V}, \mathcal{L})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{L}$ is the set of edges. Each $l \in \mathcal{L}$ represents a one-way traffic link from tail node $\tau(l) \in \mathcal{V} \cup \emptyset$ to head node $\eta(l) \in \mathcal{V}$, where $\tau(l) = \emptyset$ stands for links originating from outside of the network. We distinguish between three types of links based on their control actuations: 1) $\mathcal{L}_r$: road links actuated by traffic lights, 2) $\mathcal{L}_o$: freeway on-ramps actuated by ramp meters, 3) $\mathcal{L}_f$: freeway segments which are not directly controlled. Freeway off-ramps are treated the same way as the roads. Uncontrolled roads are also treated the same as freeways. We have $\mathcal{L}_r \cup \mathcal{L}_o \cup \mathcal{L}_f = \mathcal{L}$.

Remark 3. Some works, e.g. [Como et al., 2014], consider control over freeway links by varying speed limits, which adds to the control power but requires the existence of such a control architecture within the infrastructure. We do not consider this type of control actuation in this chapter but it can easily be incorporated into our model by modeling freeways links the same way as on-ramps, where the speed limit becomes analogous to the ramp meter input.

The number of vehicles on link l at time t is represented by $x_{[l],t} \in [0, c_l]$, which is assumed to be a continuous variable, and c_l is the capacity of l. In other words, vehicular movements are treated as fluid-like flow in our model. The number of vehicles that are able to flow out of l in one time step, if link l is actuated, is:

$$q_{[l],t} := \min \left\{ x_{[l],t}, \bar{q}_l, \min_{\{l' | \tau(l') = \eta(l)\}} \frac{\alpha_{l:l'}}{\beta_{l:l'}} (c_{l'} - x_{[l'],t}) \right\}, \tag{5.26}$$

where $\bar{q}_l$ is the maximum outflow of link l in one time step, which is physically related to the speed of the vehicles. The last argument in the minimizer determines the minimum supply available in the downstream links of l, where $\alpha_{l:l'} \in [0, 1]$ is the capacity ratio of link l' available to vehicles arriving from link l (typically portion of the lanes), $\beta_{l:l'} \in [0, 1]$ is the ratio of the vehicles in l that flow into l' (turning ratio). For simplicity, we assume capacity ratios and turning ratios are constants. System state is represented by $x \in \mathbb{R}^n_+ : \{x_{[l]}\}_{l \in \mathcal{L}}$, where n is the number of the links in the network. The state space is $\mathcal{X} := \prod_{l \in \mathcal{L}} [0, c_l]$.

A schematic diagram illustrating the behavior of $q_{[l]}$ with respect to the state variables $x_{[l]}, x_{[l']}$- which is known as the *fundamental diagram* in the traffic literature

[Geroliminis and Daganzo, 2008] - is shown in Fig. 5·4. The link flow drops if one (or more) of its downstream links do not have enough capacity to accommodate the incoming flow. In this case (when the last argument in (5.26) is the minimizer), we say the traffic flow is *congested*. Otherwise, the traffic flow is *free*. This motivates the following definition:

Definition 24. The *congestion-free set*, denoted by Π, is defined as the following region in the state space:

$$\begin{aligned}\Pi := \Big\{ \quad x \in \mathcal{X} \Big| & \min\{x_{[l]}, \bar{q}_l\} \leq \tfrac{\alpha_{l:l'}}{\beta_{l:l'}}(c_{l'} - x_{[l']}), \\ & \forall l, l' \in \mathcal{L}, \tau(l') = \eta(l) \Big\}.\end{aligned} \tag{5.27}$$

Proposition 4. The congestion-free set is a lower-set.

Proof. Consider $x \in \Pi$ and any $x' \in L(x)$. For all $l, l' \in \mathcal{L}, \tau(l') = \eta(l)$, we have $\min\{x'_l, \bar{q}_l\} \leq \min\{x_{[l]}, \bar{q}_l\}$ and $(c_{l'} - x_{[l]'}) \leq (c_{l'} - x'_{l'})$. Therefore, $\min\{x'_l, \bar{q}_l\} \leq \frac{\alpha_{l:l'}}{\beta_{l:l'}}(c_{l'} - x'_{l'})$. Thus $x' \in \Pi$, which indicates Π is a lower-set. $\square$

Note that Π is, in general, non-convex. The predicate $(x \in \Pi)$ can be written as a Boolean logic formula over predicates in the form of (5.3) as:

$$\begin{aligned}\bigwedge_{l, l' \in \mathcal{L}, \tau(l') = \eta(l)} & \bigg(\Big((x_{[l]} \leq \bar{q}_l) \wedge (x_{[l]} + \frac{\alpha_{l:l'}}{\beta_{l:l'}} x_{[l]'} \leq \frac{\alpha_{l:l'}}{\beta_{l:l'}} c_{l'}) \Big) \\ & \vee (q_{[l]} + \tfrac{\alpha_{l:l'}}{\beta_{l:l'}} x_{[l]'} \leq \tfrac{\alpha_{l:l'}}{\beta_{l:l'}} c_{l'}) \bigg).\end{aligned} \tag{5.28}$$

Notice how the minimizer in (5.27) is translated to a disjunction in (5.28).

Now we explain the controls. The actuated flow of link l at time t is denoted by

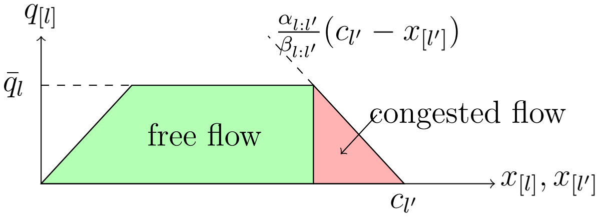


Figure 5·4: The fundamental diagram. The flow out of link l drops if the number of vehicles on the immediate downstream link l' is close to its capacity. The congestion is defined by this blocking behavior.

$\vec{q}_{[l],t}$, where we have the following relations:

$$\vec{q}_{[l],t} = \begin{cases} s_{[l],t} q_{[l],t}, & l \in \mathcal{L}_r, \\ \min\{q_{[l],t}, r_{[l],t}\}, & l \in \mathcal{L}_o, \\ q_{[l],t}, & l \in \mathcal{L}_f, \end{cases} \tag{5.29}$$

where $s_{[l],t} \in \{0,1\}$ is the traffic light for link l, where 1 (respectively, 0) stands for green (respectively, red) light, and $r_{[l],t} \in \mathbb{R}_+$ is the ramp meter input for on-ramp l at time t. Ramp meter input limits the number of vehicles that are allowed to enter the freeway in one time step. In order to disallow simultaneous green lights for links l, l' (which are typically pair of links pointing toward a common intersection in perpendicular directions), we add the additional constraints $s_{[l],t} + s_{[l]',t} \leq 1$. In simple gridded networks, as in our case study network illustrated in Fig. 5·5, it is more convenient to define phases for actuation in north-south or east-west directions that are unambiguously mapped to traffic lights for each individual link. The evolution of the network is given by:

$$x_{[l],t+1} = x_{[l],t} - \vec{q}_{[l],t} + w_{[l],t} + \sum_{l', \eta(l') = \tau(l)} \beta_{l':l} \vec{q}_{[l'],t}, \tag{5.30}$$

where $w_{[l],t}$ is the number of exogenous vehicles entering link l at time t, which is viewed as the adversarial input. The evolution relation above can be compacted into the form (5.1):

$$x_{t+1} = f_{\text{traffic}}(x_t, u_t, w_t), \tag{5.31}$$

where u_t and w_t are the vector representations for control inputs (combination of traffic lights and ramp meters) and disturbances inputs, respectively. Note that f_{traffic} represents a hybrid system which each mode is affine. The mode is determined by the control inputs and state (which determines the minimizer arguments). Some works consider nonlinear representations for the fundamental digram (Fig. 5·4), but they still can be approximated using piecewise affine functions.

5.5.2 Monotonicity

Theorem 9. System (5.31) is monotone in Π.

Proof. Consider $x', x \in \Pi, x \preceq x'$. We show that $f_{\text{traffic}}(x, u, w) \preceq f_{\text{traffic}}(x', u, w), \forall w \in \mathcal{W}, \forall u \in \mathcal{U}$. Observe in (6.1) that we only need to verify is proving that $x_{[l]} - \vec{q}_{[l]}$ is a non-decreasing function of $x_{[l]}$ as all other terms are additive and non-decreasing with respect to x. Since $x, x' \in \Pi$, the last argument in (5.26) is never the minimizer. Thus, for all $l \in \mathcal{L}$, we have $x_{[l]} - \vec{q}_{[l]} \in \{0, x_{[l]} - r_{[l]}, x_{[l]} - c_l, x_{[l]}\}$, depending on the mode of the system and actuations, which all are non-decreasing functions of $x_{[l]}$. Thus, f_{traffic} is monotone in Π. $\square$

The primary objective in our traffic management approach is finding control policies such that the state is restricted to Π, which not only eliminates congestion, but also ensures that the system is monotone hence the methods of this chapter become applicable. It is worth to note that the traffic system becomes non-monotone when flow is congested in diverging junctions, as shown in [Coogan and Arcak, 2014]. This phenomena is attributed to the first-in-first-out (FIFO) nature of the model.

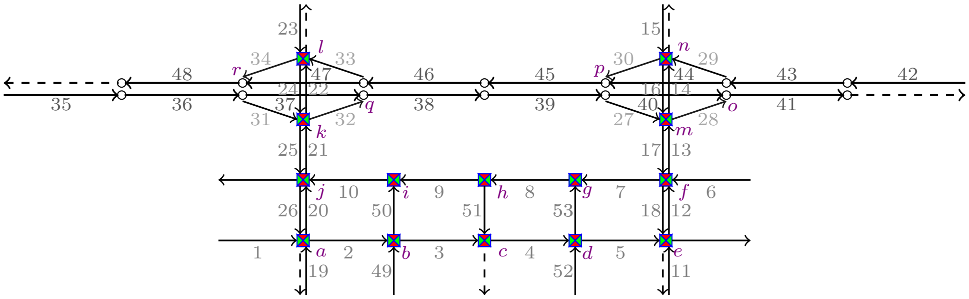


Figure 5·5: Traffic management case study: A network of freeways and urban roads. There are 14 intersections controlled by traffic lights and 4 ramp meters.

By assuming fully non-FIFO models, system becomes monotone in the whole state space. For a more thorough discussion on physical aspects of monotonicity in traffic networks, see [Coogan et al., 2016a].

The maximal system in (5.31) corresponds to the scenario where each w_l is equal to its maximum allowed value w_l^*.

5.5.3 Case Study

Network

Consider the network in Fig. 5·5, which consists of urban roads (links 1-26, 27,29,31,33 and 49-53), freeway segments (links 35-48) and freeway on-ramps (links 28,30,32,34). The layout of the network illustrates a freeway passing by an urban area, which is common in many realistic traffic layouts. There are 14 intersections (nodes a-n) controlled by traffic lights. Each intersection has two modes of actuation: north-south (NS) and east-west (EW). There are four entries to the freeway (nodes o-r) that are regulated by ramp meters. We have $n = 53$ and $\mathcal{U} = \mathbb{R}_+^4 \times \{0, 1\}^{14}$. Vehicles arrive from links 1,6,11,15,19,23,35,42,49 and 52. The parameters of the network are shown in Table 5.1.

Table 5.1: Parameters of the network in Fig. 5·5

links	parameters
$1-26, 49-53$	$\bar{q}_l = 15, c_l = 40$
$27-34$	$\bar{q}_l = 15, c_l = 30$
$35-48$	$\bar{q}_l = 40, c_l = 60$
Turning ratios	value
$\beta_{2:50}, \beta_{4:53}, \beta_{8:51}, \beta_{12:7}, \beta_{13:28}, \beta_{15:30}, \beta_{16:28},$ $\beta_{21:32}, \beta_{24:32}, \beta_{26:2}, \beta_{36:31}, \beta_{36:33}, \beta_{39:27}, \beta_{43:29}$	0.2
$\beta_{5:12}, \beta_{6:13}, \beta_{6:18}, \beta_{10:21}, \beta_{10:26}$	0.3
$\beta_{1:20}, \beta_{6:7}$	0.4
$\beta_{1:2}, \beta_{11:12}, \beta_{14:30}, \beta_{17:7}, \beta_{17:18}, \beta_{19:2}, \beta_{19:20},$ $\beta_{22:34}, \beta_{23:24}, \beta_{23:34}, \beta_{27:14}, \beta_{27:17}, \beta_{29:16}, \beta_{31:22},$ $\beta_{31:25}, \beta_{33:24}, \beta_{49:3}, \beta_{49:50}, \beta_{51:4}, \beta_{52:5}, \beta_{52:53},$	0.5
$\beta_{2:3}, \beta_{3:4}, \beta_{4:5}, \beta_{8:9}, \beta_{12:13}, \beta_{13:14}, \beta_{15:16}, \beta_{16:17}, \beta_{20:21},$ $\beta_{21:22}, \beta_{24:25}, \beta_{25:26}, \beta_{36:37}, \beta_{39:40}, \beta_{43:44}, \beta_{46:47},$	0.8
Capacity ratios	value
$\alpha_{19:2}, \alpha_{26:2}, \alpha_{17:7}, \alpha_{12:7}, \alpha_{13:28}, \alpha_{16:28}$ $\alpha_{14:30}, \alpha_{15:30}, \alpha_{21:32}, \alpha_{24:32}, \alpha_{22:34}, \alpha_{23:34}$	0.5
Disturbances (arrival rates)	
$w_1^* = w_6^* = 4.5, w_{11}^* = w_{15}^* = w_{19}^* = 5, w_{23}^* = 6$ $w_{35}^* = w_{42}^* = 20, w_{49}^* = w_{52}^* = 2$	

specification

As mentioned earlier, the primary objective is keeping the state in the congestion-free set. In addition, since the demand for the north-south side roads (links 49-53) is smaller than the traffic in the east-west roads, we add a timed liveness requirement for the traffic flow on links 49-53:

$$\psi = \bigwedge_{l=49,50,\cdots,53} (x_{[l]} \geq 5) \Rightarrow \mathbf{F}_{[0,3]}(x_{[l]} \leq 5),$$

which states that "if the number of vehicles on any of the north-south side roads exceeds 5, their flow is eventually actuated within three time units ahead". The global specification is given as:

$$\phi = \mathbf{G}_{[0,\infty]} \left((x \in \Pi) \wedge \psi \right). \tag{5.32}$$

Note that $h^{\varphi} = 3$, $\varphi = (x \in \Pi) \wedge \psi$.

Open-loop Control Policy

We use Theorem 7. The shortest ϕ-sequence that we found for this problem has $T = 5, T_0 = 0$. The corresponding MILP had 2357 variables (of which 1061 were binary) and 4037 constraints [4], which is solved using the Gurobi MILP solver in less than 6 seconds on a dual core 3.0 GHz MacBook Pro. The cost is set to zero in order to just check for feasibility. Even though finding an optimal solution and checking for feasibility of a MILP have the same theoretical complexity, the latter is executed much faster in practice.

For instance, finding a ϕ-sequence, while minimizing $\sum_{k=0}^{7} \|x_k^{\phi}\|_1$ takes 1238 seconds. Note that it is virtually intractable to attack a problem of this size (53 dimensional state) using any method that involves state-space discretization, such as the method in [Coogan and Arcak, 2015] (e.g., if each state-component is partitioned into 2 intervals, the finite-state problem size will be 2^{53}).

Monotonicity implies that any demand set $\mathcal{W}$ for which there exists a solution to Problem 5 is a lower-set. The set corresponding to the values at the bottom of Table 5.1 is one of them. Table 5.2 shows results existence results for some other demand scenarios. Computation times for solving a MILP do not demonstrate a generic behavior. For the rest of this section, the numerical examples are reported for the values in Table 5.1.

The control values in the ϕ-sequence are shown in Table 5.3. As stated in Theorem 7, starting from an initial condition in $L(x_0)$, applying the open-loop control policy (5.12) guarantees satisfaction of the specification. In other words, after applying the initialization segment, the repetitive controls in Table 5.3 become a fixed timetable for the inputs of the traffic lights and the ramp meters. Starting from x_0,

---

[4] The scripts for this case study are available in `http://blogs.bu.edu/sadra/format-monotone`

Table 5.2: Existence of ϕ-sequences for the network in Fig. 5·5

Demand Changes from Table 5.1	T	Existence	Comp. Time (s)
-	5	yes	6
-	6	no	4
-	7	no	10
-	8	no	75
-	9	no	11
-	10	yes	36
$w_1^* = w_6^* = 3, w_{11}^* = w_{15}^* = w_{19}^* = 6$	5	yes	5
$w_1^* = w_6^* = 4, w_{11}^* = w_{15}^* = w_{19}^* = 6$	5	no	0.5
$w_1^* = w_6^* = 1.5, w_{49}^* = w_{52}^* = 3.5$	5	yes	16
$w_1^* = w_6^* = 7.5, w_{11}^* = w_{15}^* = w_{19}^* = w_{23}^* = 2$	6	yes	9
$w_1^* = w_6^* = 9, w_{11}^* = w_{15}^* = w_{19}^* = w_{23}^* = 1$	5	yes	4
$w_1^* = w_6^* = 10, w_{11}^* = w_{15}^* = w_{19}^* = w_{23}^* = 0$	30	no	3.5
$w_{15}^* = w_{23}^* = 8, w_{35}^* = w_{42}^* = 10$	6	yes	23
$w_{15}^* = w_{23}^* = 0, w_{35}^* = w_{42}^* = 30$	5	yes	4

which is a 53-dimensional vector, we apply (5.12) using the values in Table 5.3. The trajectory of the maximal system is shown in Fig. 5·6 [Top]. The traffic signals are coordinated such that the traffic flows free of congestion. The black dashed lines represent the capacity of the links, and the dashed line in the fourth figure (from the left) represents the threshold for the liveness sub-specification (ψ). It is observed that all the state values for side road links (49-53) persistently fall below the threshold. The robustness values for ($x \in \Pi$) and ψ are shown in the fifth figure. As mentioned earlier, robustness corresponds to the minimum volume of vehicles that the system is away from congestion, or violating the specification. The robustness values are always positive, indicating satisfaction.

As stated in Theorem 4, the trajectory of the maximal system converges to a periodic orbit. It is worth to not that the number of vehicles on freeway links is significantly smaller than its capacity, which is attributed to the fact that the number designated for $\bar{q}$ (related to the maximum speed) of freeway links is relatively large (30, as opposed to 15 for roads). Therefore, freeway links are utilized in a way that there is enough space for high speed non-congested flow.

Table 5.3: ϕ-sequence in the case study

-	Initialization			Repetitive Controls				
node	u_0^ϕ	u_1^ϕ	u_2^ϕ	u_3^ϕ	u_4^ϕ	u_5^ϕ	u_6^ϕ	u_7^ϕ
a	EW	NS	NS	NS	EW	EW	NS	NS
b	NS	EW	EW	EW	NS	NS	EW	EW
c	EW	NS	NS	EW	EW	EW	NS	NS
d	EW	NS	EW	NS	EW	EW	NS	EW
e	EW	EW	NS	NS	NS	EW	EW	NS
f	NS	EW	NS	EW	NS	NS	EW	NS
g	NS	EW	NS	EW	EW	NS	EW	NS
h	EW	NS	EW	EW	EW	EW	NS	EW
i	EW	NS	EW	EW	NS	EW	NS	EW
j	EW	EW	NS	NS	EW	EW	EW	NS
k	EW	EW	NS	NS	NS	EW	EW	NS
l	NS	EW	EW	NS	NS	NS	EW	EW
m	EW	NS	NS	EW	NS	EW	NS	NS
n	NS	NS	EW	NS	EW	NS	NS	EW
o	0.0	0.0	0.0	12.8	0.0	0.0	0.0	0.0
p	4.0	14.0	0.0	9.5	0.0	4.0	11.5	0.0
q	0.0	0.0	10.0	0.0	2.5	0.0	0.0	10.0
r	5.5	0.0	4.0	14.0	11.5	5.5	0.0	4.0

Robust MPC

Here it is assumed that the controller has full state knowledge. We apply the techniques developed in Sec. 5.4. Using the result from the previous section, the set $\Omega_{\mathcal{L}^\varphi}$ is constructed in $\mathbb{R}_+^{212}$ $(= \mathbb{R}^{n(h^\varphi+1)}, n = 53, h^\varphi = 3)$. The cost criteria that we use in this case study is the total delay induced in the network over the planning horizon H. A vehicle is delayed by one time unit if it can not flow out of a link in one time step, which may be because of the actuation (e.g., red light) or waiting for the flow of other vehicles in the same link (i.e., we have $x_{[l]} \geq c_l$). We are also interested in maximizing the STL robustness score. The cost function is:

$$\begin{aligned} J_{\text{traffic}}(x^H, u^H) &:= -\zeta\ \rho(\mathbf{x}, \mathbf{G}_{[0,H-1]}\varphi, t - h^\varphi + 1) \\ &+ \sum_{k=0}^{H-1} \gamma^k \sum_{l \in \mathcal{L}} (x_{[l],t+k} - \vec{q}_{[l],t+k}), \end{aligned} \tag{5.33}$$

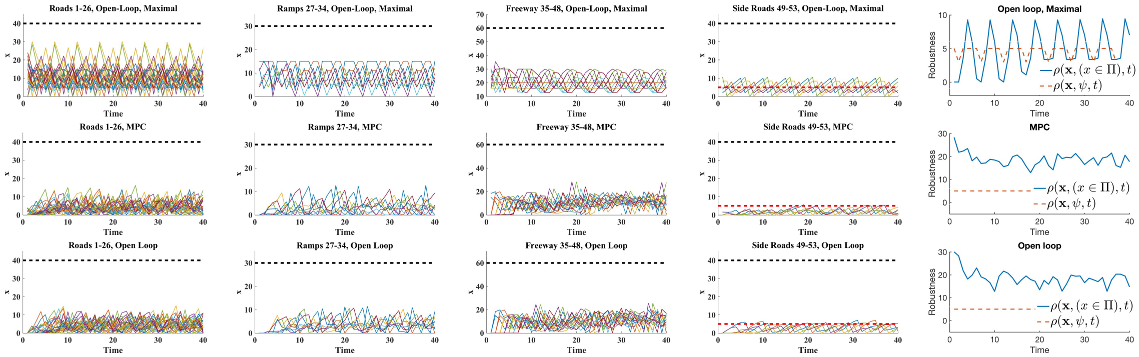
where $\vec{q}_{[l]}$, given by (5.29), is the amount of vehicles that flow out of link l, γ is the discount factor for delays predicted in further future, and ζ is a positive weight for robustness. Notice the connection between the time window of STL robustness score in (5.33) and MPC constraint enforcement in (5.24). It follows from Theorem 9 and STL quantitative semantics Definition 7 that the cost function above is non-decreasing with respect to the state in Π. Therefore, in order to minimize the worst case cost, the maximal system is considered in the MPC optimization problem.

Starting from zero initial conditions, we implement the MPC algorithm (5.25) with $H = 3$ for 40 time steps. We set $\zeta = 1000$, $\gamma = 0.5$ in (5.33). The disturbances at each time step were randomly drawn from $L(w^*)$ using a uniform distribution. The maximum computation time for each MPC step time step was less than 0.8 seconds (less than 0.5 seconds on average). The resulting trajectory is shown in Fig. 5·6 [Middle]. For the same sequence of disturbances, the trajectory resulted from applying the open-loop control policy (5.12) (using the values in Table 5.3) is shown in Fig. 5·6 [Bottom]. Both trajectories satisfy the specification. However, robust MPC has obviously better performance when costs are considered. The total delay accumulated over 40 time steps is:

$$J_{40} = \sum_{\tau=0}^{40} \sum_{l \in \mathcal{L}} (x_{[l],\tau} - \vec{q}_{[l],\tau}).$$

The cost above obtained from applying robust MPC was $J_{40} = 1843$, while the one for the open-loop control policy was $J_{40} = 2299$, which demonstrates the usefulness of the state knowledge in planning controls in a more optimal way. An optimal tuning of parameters η and γ requires an experimental study which is out of scope of this chapter. We only remark that we usually obtained larger delays with non-zero η, which shows that including STL robustness score in the MPC cost function may be useful even though the ultimate goal is minimizing the total delay.

It is worth to note that we also tried implementing the MPC algorithm (for the



Roads 1-26, Open-Loop, Maximal

Ramps 27-34, Open-Loop, Maximal

Freeway 35-48, Open-Loop, Maximal

Side Roads 49-53, Open-Loop, Maximal

Open loop, Maximal

Roads 1-26, MPC

Ramps 27-34, MPC

Freeway 35-48, MPC

Side Roads 49-53, MPC

MPC

Roads 1-26, Open Loop

Ramps 27-34, Open Loop

Freeway 35-48, Open Loop

Side Roads 49-53, Open Loop

Open loop

Figure 5·6: Traffic management case study: [Top Row] the trajectory of the maximal system obtained from applying the open-loop control policy (5.12) with initial condition x_0^ϕ [Middle Row] robust MPC generated trajectory with zero initial condition with disturbances chosen uniformly $L(w^*)$ [Bottom Row] trajectory generated from applying the open-loop control policy (5.12) with zero initial conditions and the same disturbances as in [Middle Row].

case $\mathbf{w} = (w^*)^\omega$, or the maximal system) without the terminal constraints, as in (5.23). The MPC got infeasible at $t = 8$. The violating constraints were those in $x \in \Pi$. This observation indicates that the myopic behavior of MPC in (5.23), when no additional constraints are considered, can lead to congestion in the network.

Chapter 6

Contract-based Design

In this chapter, we follow a divide and conquer approach to control synthesis for traffic networks. We partition the network into smaller subnetworks and synthesize controls locally for each subnetwork. For the dynamical interconnections between the subnetworks, we take an assume-guarantee approach [Henzinger et al., 1998]. Each subnetwork assumes the interconnection effects from its neighbors satisfy a set of contracts, while the controllers of neighboring subnetworks promise to maintain those contracts. Therefore, local controls can be planned optimally in a decentralized manner but with a global coordination induced by the contracts and the global network specification, which is ensured to be satisfied.

The main contributions of the chapter are as follows. First, we provide a method to synthesize assume-guarantee contracts by feasibility checking of a mixed-integer linear programming (MILP) problem, which is computed offline and it can be applied to relatively large networks. Second, we find local controls optimally using a robust MPC approach, which has feasibility guarantees for both the local constraints and contracts hence the overall global specification is ensured. We present a case study on an urban traffic network and provide preliminary results on applying our methods to microscopic traffic models.

This chapter is organized as follows. We formulate the problem in Sec. 8.1. In Sec.

6.3, we explain how to partition a network considering constraints of the system and specification. The technical details on computation of assume-guarantee contracts and control synthesis are explained in Sec. 6.4 and Sec. 6.5, respectively. The case study is presented in Sec. 8.5.

6.1 Traffic Network Model

We introduce some new notation. A link l is defined as a one-way traffic road segment with the following attributes:

- $c_l \in \mathbb{R}_+$ is the capacity of l;
- $x_{l,t} \in [0, c_l]$ is the volume of vehicles on l at time t, $t \in \mathbb{N}$;
- $q_l \in \mathbb{R}_+$ is the maximum outflow (maximum volume of vehicles that can flow out of l in one time step);
- $u_{l,t} \in \mathcal{U}_l$ is the control input of l at time t, where $\mathcal{U}_l$ depends on the type of l:
  - if l is controlled with traffic lights: $\mathcal{U}_l = \{0, 1\}$, where 1 (0) corresponds to green (red) light;
  - if l is controlled with ramp meter/speed limit: $\mathcal{U}_l = [0, 1]$, where u_l is the ratio of the maximum outflow q_l that is allowed to flow out of the link in one time step;
  - if l is uncontrolled: $\mathcal{U}_l = \{1\}$.

Additional constraints on control inputs (e.g., sequentiality constraints) are expressed using MTL formulas and are considered as a part of the problem formulated in Sec. 8.1.

Definition 25. A traffic network is defined as a tuple $\mathcal{N} = (\mathcal{L}, \delta, \alpha, \beta, \mathcal{A}, w)$, where:

- $\mathcal{L}$ is the set of links in the network;
- $\delta : \mathcal{L} \to 2^{\mathcal{L}}$, where $\delta(l)$ is the set of downstream links of l (downstream function; defines network topology);
- $\alpha : \mathcal{L} \times \mathcal{L} \to [0, 1]$, where $\alpha(l, l'), l' \in \delta(l)$, is the ratio of vacancies available in l' dedicated to l, which is assumed to be constant (capacity ratios);
- $\beta : \mathcal{L} \times \mathcal{L} \to [0, 1]$, where $\beta(l, l'), l' \in \delta(l)$, is the ratio of volume flowing from l to l', which is assumed to be constant (turning ratios);
- $\mathcal{A} \subset \mathcal{L} \times \mathcal{L}$ is the set of *antagonistic pairs*. Two links form an antagonistic pair if their traffic lights are not allowed to be green simultaneously [1] (defines traffic phases);
- $w : \mathcal{L} \times \mathbb{N} \to \mathbb{R}_+$, where $w(l, t)$ is the *exogenous demand* (volume of vehicles entering from outside of the network) towards l at time t.

Example 12. Consider the network in Fig. 6·1 with 84 links. This network represents two urban areas on north and south sides connected by three bridges in between (the network topology is inspired by the Boston-Cambridge area). Each link is shown as a directed edge between nodes (intersections) shown as squares. For all links we have $c_l = 40, q_l = 15$. The long bridges in the middle are divided into two separate links, and links 35, 41, 47, 53, 59, 65 are uncontrolled (i.e., there are no traffic lights in the middle of the bridges). We have $l' \in \delta(l)$ if the head of edge representing l is followed by the tail of edge representing l'. For example, we have $\delta(1) = \{2, 34, 44\}$, $\delta(14) = \{15, 75\}$, $\delta(53) = \{54\}$, $\delta(76) = \emptyset$, etc. Antagonistic pairs are determined by

---

[1] We assume that all antagonistic pairs are controlled by traffic lights.

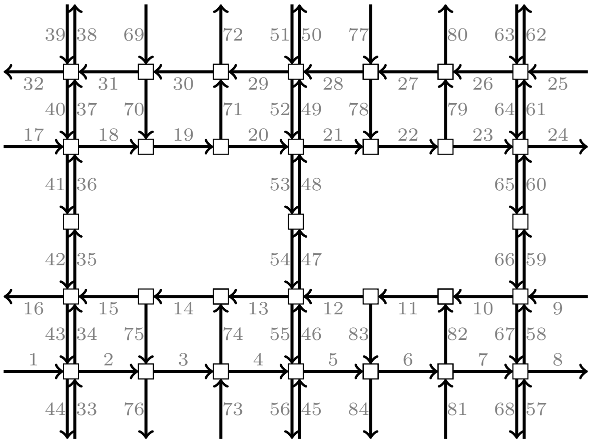


Figure 6·1: Traffic Network Topology

trivial traffic conventions. For instance, $\{12, 54\} \in \mathcal{A}$, as the pair head into a common intersection in perpendicular directions. The values for $w(l,t)$ vary between 0 and 8 vehicles per time step, depending on the location of the link. The detailed valuations for w, and for other network components from Definition 25 including α and β, are not provided here but are available in `sites.bu.edu/hyness/format-distributed`.

We define the set of outgoing links of a network by $\mathcal{L}^{\text{out}} := \{l \in \mathcal{L} | \delta(l) = \emptyset\}$ and the set of internal links as $\mathcal{L}^{\text{int}} := \mathcal{L} \setminus \mathcal{L}^{\text{out}}$. When describing the dynamics of a traffic network $\mathcal{N}$, we are only interested in the evolution of the internal links. The outflow of an internal link l at time t is defined as:

$$f_{l,t} := \min \left\{ x_{l,t}, u_{l,t} q_l, \min_{l' \in \delta(l)} \frac{\alpha(l,l')}{\beta(l,l')} (c_{l'} - x_{l',t}) \right\}. \tag{6.1}$$

Note that the outflow is zero if $u_{l,t} = 0$ (red light). The last argument of the minimizer is determined by the minimum available vacancy in the downstream links of l. Physically, the outflow model above is governed by the first-in-first-out (FIFO) rule [Coogan et al., 2016a]. As a consequence of this rule, lack of enough vacancy in a link blocks the flow of its upstream links to all other surrounding links. For example, in

Fig. 6·1, if link 4 does not have enough vacancy for accommodating flow from link 3, the flow from 3 to 74 is also blocked.

For all antagonistic pairs $(l, l') \in \mathcal{A}$, we have $u_{l,t} + u_{l',t} \leq 1, \forall t \in \mathbb{N}$. The evolution of an internal link l is given by:

$$x_{l,t+1} = \min\big\{x_{l,t+1} - f_{l,t} + \sum_{l', l \in \delta(l')} \beta(l', l) f_{l',t} + w(l, t),\ c_l\big\}. \tag{6.2}$$

The volume of vehicles that leave the network through an outgoing link at time t is:

$$y_{l,t} := \sum_{l', l \in \delta(l')} \beta(l', l) f_{l',t},\ l \in \mathcal{L}^{\text{out}}. \tag{6.3}$$

The network dynamics is represented in a compact form as the following discrete-time system:

$$x_{t+1} = F(x_t, u_t, w_t), \tag{6.4}$$

$$y_t = G(x_t, u_t), \tag{6.5}$$

where

- $x_t = \{x_{l,t}\}_{l \in \mathcal{L}^{\text{int}}}$, is the *state* at time t. We have $x_t \in \mathcal{X}, \forall t \in \mathbb{N}$, where $\mathcal{X} = \prod_{l \in \mathcal{L}^{\text{int}}} [0, c_l]$.
- $u_t = \{u_{l,t}\}_{l \in \mathcal{L}^{\text{int}}}$ is the *control input* at time t. We have $u_t \in \mathcal{U}, \forall t \in \mathbb{N}$, where $\mathcal{U} \subseteq \prod_{l \in \mathcal{L}^{\text{int}}} \mathcal{U}_l$.
- $w_t = \{w(l, t)\}_{l \in \mathcal{L}^{\text{int}}}$ is the *additive disturbance* at time t. We have $w_t \in \mathcal{W}, \forall t \in \mathbb{N}$, where $\mathcal{W} \subset \mathbb{R}_+^{|\mathcal{L}^{\text{int}}|}$ is the set of admissible disturbances.
- $y_t = \{y_{l,t}\}_{l \in \mathcal{L}^{\text{out}}}$ is the *output* at time t. We have $y_t \subset \mathbb{R}_+^{|\mathcal{L}^{\text{out}}|}, \forall t \in \mathbb{N}$.

Note that both $F : \mathcal{X} \times \mathcal{U} \times \mathcal{W} \to \mathcal{X}$ and $G : \mathcal{X} \times \mathcal{U} \to \mathbb{R}_+^{|\mathcal{L}^{\text{out}}|}$ are piecewise affine and positive.

6.2 Problem Statement

Consider a traffic network $\mathcal{N}$.

Assumption 2. The set of admissible additive disturbances is in the form $\mathcal{W} = L(w^{\max})$, $w^{\max} \in \mathbb{R}_+^{|\mathcal{L}^{\text{int}}|}$.

The assumption above is reasonable when the sources of the exogenous demands are independent of each other. Thus we have $w(l,t) \in [0, w_{l,t}^{\max}], \forall l \in \mathcal{L}^{\text{int}}, \forall t \in \mathbb{N}$.

Definition 26. A *control policy* $\mu = \{\mu_t \big| t \in \mathbb{N}\}$ is a set of relations that map the (partial) history of state and controls into an admissible control action:

$$u_t = \mu_t(x_0, \cdots, x_t, u_0, \cdots, u_{t-1}),$$

where $\mu_t : \mathcal{X}^{t+1} \times \mathcal{U}^t \to \mathcal{U}$.

We assume full and exact state knowledge. We later explain how to relax this assumption in Sec. 6.5. Given an initial condition x_0, a control policy μ and a sequence of additive disturbances $\mathbf{w} = w_0, w_1, \cdots$, the *system run* is defined as:

$$\zeta(x_0, \mu, \mathbf{w}) := (x_0, u_0), (x_1, u_1), \cdots . \tag{6.6}$$

Similarly, the *output run* is defined as $\xi(x_0, \mu, \mathbf{w}) := y_0, y_1, \cdots$.

Definition 27. The congestion-free set of a network is defined as the set $\Pi \subset \mathcal{X} \times \mathcal{U}$, where:

$$\Pi := \left\{(x,u)\middle|\, \min\{u_l q_l, x_l\} \leq \min_{l' \in \delta(l)} \left\{\tfrac{\alpha(l,l')}{\beta(l,l')}(c_{l'} - x_{l'})\right\} \wedge \right.$$
$$\left. x_l - f_l + \sum_{l', l \in \delta(l')} \beta(l', l) f_{l'} + w_l^{\max} \leq c_l, \forall l \in \mathcal{L}^{\text{int}} \right\}.$$

If the system is in the congestion-free set, then the two following properties hold. First, the last argument from the minimization in (6.1) is never the minimizer. Thus, the flow of a link is never obstructed due to the lack of enough vacancy in its downstream links. Second, c_l in (6.2) is also never the minimizer. Therefore, the total exogenous demand from outside of the network is accommodated in the network hence there is no flow obstruction from outside of the network as well. Our primary interest is finding a control policy such that the evolution of the network is always restricted to the congestion-free set.

Furthermore, we are also interested in various other objectives described using MTL. We allow for two types of atomic propositions. First are linear predicates over state, which are of the following form:

$$p_x = (a_1 x_{l_1} + a_2 x_{l_2} + \cdots + a_{n_p} x_{l_{n_p}} \leq b), \tag{6.7}$$

where $l_1, l_2, \cdots, l_{n_p}$ are the links whose vehicular volumes appear in p_x and $b, a_i \in \mathbb{R}_+, i = 1, \cdots, n_p$. Since all values are positive, the half space induced by the linear predicate above is a lower-set in $\mathcal{X}$ with partial order relation $\preceq_+$. Therefore, state predicates are not falsified when vehicular volumes are decreased. It is also assumed that no negation operator applies to state predicates. The second type of propositions are predicates over controls, which are in the following form:

$$p_u = (u_l \sim b_u), \tag{6.8}$$

where $\sim \in \{\leq, \geq, =\}, b_u \in [0,1]$. Using MTL temporal operators and Boolean connectives, we can describe a wide variety of temporal properties for traffic networks.

Example 13. Consider the network in Fig. 6·1 and the following MTL specifications:

- $\varphi_1 = \mathbf{F}_{[0,6)}\left((u_{12}=0) \wedge (u_{46}=0) \wedge (u_{54}=0)\right)$; which reads: "within 6 time units, all the traffic lights of links heading toward intersection at middle southern area turn red (hence pedestrians can cross the intersection in diagonal directions)".
- $\varphi_2 = \neg\big((u_{28}=0) \ \wedge \ \mathbf{F}_{\{1\}}(u_{28}=1) \wedge \mathbf{F}_{\{2\}}(u_{28}=0)\big)$; which states: "the traffic light of link 28 can not be green for just one time step."
- $\varphi_3 = (x_{59} + x_{60} + x_{65} + x_{66} \leq 100)$; which states: "the total volume of the vehicles on the eastern bridge is less than 100."
- $\varphi_4 = (x_{73} \leq 5) \vee \mathbf{F}_{[0,4)}(u_{73}=1)$; which translates to: "if the volume of vehicles on link 73 exceeds 5, its traffic light eventually turns green within 4 time units."

Given a bounded MTL formula φ, we consider specifications of the following form:

$$\Phi^{global} := \mathbf{G}_{[0,\infty)}\Phi, \tag{6.9}$$

where $\Phi = ((x,u) \in \Pi) \wedge \varphi$, and Π is the congestion-free set from Definition 27. The operator $\mathbf{G}_{[0,\infty)}$ (unbounded always) ensures that the congestion-free property and the requirements in φ hold for all times. Note that $h(\Phi) = h(\varphi)$. It can be shown that the proposition $((x,u) \in \Pi)$ can be transformed into a Boolean formula over state and control predicates by translating the minimizers into mixed-logical equations. We omit the explanation here as a similar procedure can be found in [Heemels et al., 2001]. Given Φ and its atomic propositions in the form of (6.7), (6.8), the infinite word generated by run (6.6) is denoted by $\sigma(\zeta(\mu, x_0, \mathbf{w}))$.

Control policies guaranteeing satisfaction of Φ^{global} are often not unique. Thus, we are interested in choosing one optimally with respect to a cost function. The cost criterion that we consider in this chapter is the total amount of delay induced in the network, which is defined as:

$$J(x_0, \mu, \mathbf{w}) = \sum_{t=0}^{\infty} \sum_{l \in \mathcal{L}^{\text{int}}} \gamma^t (x_{l,t} - f_{l,t}), \tag{6.10}$$

where $\gamma \in (0,1)$ is the discount factor that is introduced to make the infinite horizon cost properly defined. Observe that $x_{l,t} - f_{l,t}$ is the volume of vehicles on l at time t that are unable to travel in the network for one time step. It is straightforward to show that $J \leq \frac{1}{1-\gamma} \sum\limits_{l \in \mathcal{L}^{\text{int}}} c_l$.

Problem 7. Given a traffic network $\mathcal{N}$ with dynamics (8.1), a time bounded MTL specification φ with atomic propositions in the form of (6.7), (6.8), and the cost function J as in (6.10), find a control policy μ and a set of initial conditions $\mathcal{X}_0 \subset \mathcal{X}$ such that

$$\sigma_0(\zeta(x_0, \mu, \mathbf{w})) \models \Phi^{global}, \forall x_0 \in \mathcal{X}, \forall \mathbf{w}.$$

Furthermore, given $x_0 \in \mathcal{X}_0$, choose the optimal control policy such that the worst-case cost is minimized:

$$\begin{array}{ll} \text{minimize} & \max\limits_{\mathbf{w}} J((x_0, \mu, \mathbf{w})) \\ \text{subject to} & \sigma_0(\zeta(x_0, \mu, \mathbf{w})) \models \Phi^{global}, \forall \mathbf{w}. \end{array} \tag{6.11}$$

Our approach to Problem 7 involves some approximations. First, we only find a subset of all admissible initial conditions. We discuss the completeness of our method in Sec. 6.4. Second, we approximate the infinite horizon constrained optimal control in Problem 7 as a receding horizon optimal control problem, i.e. we use a model predictive control (MPC) scheme. As mentioned earlier, there are two primary challenges to this approach. First, we need to guarantee that control synthesis in a receding horizon manner ensures global specification (6.9), which has infinite time semantics. This issue is covered in Sec. 6.5. Second, constrained optimization is computationally expensive beyond small networks hence controls can not be found in a centralized manner in real time. To overcome this issue, we partition the network into smaller subnetworks. The dynamics of subnetworks become interconnected since vehicles that leave a subnetwork arrive in other subnetworks. As mentioned earlier, we follow an assume-guarantee approach to design contracts for the interconnections of the subnetworks. The details are explained in Sec. 6.4. Each subnetwork's MPC incorporates the relevant contracts, as explained in Sec. 6.5.

6.3 Network Partitioning

In this section, we explain how to partition a network into smaller subnetworks. We write φ in conjunction normal form (CNF) [2]:

$$\varphi = \bigwedge_{k=1}^{n_\varphi} \varphi_k^{conj}, \tag{6.12}$$

where each φ_k^{conj} can not be written as a conjunction of multiple MTL formulas. We define

$$\text{Links}(\varphi) = \{l \in \mathcal{L} | x_l \text{ or } u_l \text{ appears in } \varphi\}.$$

Assumption 3. Formula φ is *sparse* in the sense that for all $k, k' \in \{1, \cdots, n_\varphi\}, k \neq k'$, we have $\text{Links}(\varphi_k^{conj}) \subseteq \text{Links}(\varphi_{k'}^{conj})$ or $\text{Links}(\varphi_k^{conj}) \cap \text{Links}(\varphi_{k'}^{conj}) = \emptyset$.

The assumption above is reasonable in traffic networks since we are usually inter-

---

[2] Every MTL formula can be written in CNF.

ested in requirements at specific locations. While our method can handle non-sparse specifications by introducing some conservativeness (when designing contracts in Sec. 6.4), we do not discuss it in this chapter.

Given a network $\mathcal{N}$ and an integer N, we partition $\mathcal{N}$ into $\mathcal{N}^1, \mathcal{N}^2, \cdots, \mathcal{N}^N$. We take the following considerations into account when a network is partitioned. First, the set of subnetwork internal links has to be disjoint: $\bigcup_{i=1}^{N} \mathcal{L}^{\text{int},i} = \mathcal{L}^{\text{int}}$, $\mathcal{L}^{\text{int},i} \cap \mathcal{L}^{\text{int},j} = \emptyset, \forall i \neq j$, where $\mathcal{L}^{\text{int},i}$ is the set of internal links of subnetwork $\mathcal{N}^i$. Second, we desire that the size of a subnetwork (defined by the number of its internal links) does not exceed a predetermined bound K, hence the size of its MPC optimization problem (which is directly related to the number of internal links) is confined. Therefore, we have $|\mathcal{L}^{\text{int},i}| \leq K, i = 1, \cdots, N$. Third, MTL formula φ has to be translated into a conjunction of "local" MTL formulas for each subnetwork. Using the CNF of φ, we require that $\exists i \in \{1, \cdots, N\}$ such that $\text{Links}(\varphi_k^{conj}) \subseteq \mathcal{L}^{\text{int},i}$, $k = 1, \cdots, n_\varphi$. Fourth, for all $l, l' \in \mathcal{A}$, we have $\exists i \in \{1, \cdots, N\}$ such that $l, l' \in \mathcal{L}^{\text{int},i}$. Therefore, the links of each antagonistic pair are assigned to a single subnetwork. This is important since subnetworks are controlled in a decentralized way and the controls for antagonistic pairs are directly coupled. Fifth, for all $l \in \mathcal{L}^i$, we have $\delta^i(l) = \delta(l) \cap \mathcal{L}^i$, where δ^i defines the downstream function of $\mathcal{N}^i$. Finally, the partitioning should lead to sparsity in the sense that the interconnections between subnetworks are minimal. As it will be explained later, while not affecting the satisfaction of the global specification (6.9), interconnections impose constraints that may introduce conservativeness into the planning of controls. Within all possible partitionings, want to choose the one for which the total number of interconnections is minimized:

$$\text{minimize } \frac{1}{2} \sum_{i}^{N} \sum_{j, i \neq j}^{N} |\mathcal{L}^i \cap \mathcal{L}^j|.$$

The presence of constraints related to the specification makes our network partitioning problem different from the traditional graph partitioning problems in the literature [Shi and Malik, 2000]. For each internal link l of $\mathcal{N}$, we define N binary variables $b_l^i \in \{0,1\}, i = 1, \cdots, N$, where $b_l^i = 1$ indicates l is assigned to the internal links of network $\mathcal{N}^i$. We note that links in $\mathcal{L}^{\text{out}}$ are excluded from the assignment process. We formulate the requirements that were explained above as the following integer constraints:

$$\begin{cases} \sum_{i=1}^{N} b_l^i = 1, \forall l \in \mathcal{L}^{\text{int}}, \\ \sum_{l \in \mathcal{L}} b_l^i \leq K, \\ b_l^i = b_{l'}^i, \forall (l, l') \in \mathcal{A}, \\ b_l^i = b_{l'}^i, \forall l, l' \in \text{Links}(\varphi_k^{conj}), k = 1, \cdots, n_\varphi, \end{cases} \tag{6.13}$$

where $i = 1, \cdots, N$. The first states that the sets of internal links are disjoint, the second ensures that the size of each subnetwork is bounded by K, the third reads that antagonistic links are assigned to the same subnetwork, and fourth declares that all the links in non-conjunctive sub-specifications are assigned to the same subnetwork.

It is easy to show that the total number of interconnections is equal to the summation of the differences in binary assignments of links that are related by δ:

$$\sum_{i}^{N} \sum_{j, i \neq j}^{N} |\mathcal{L}^i \cap \mathcal{L}^j| = \sum_{l \in \mathcal{L}, l' \in \delta(l)} \sum_{i}^{N} \sum_{j, i \neq j}^{N} |b_l^i - b_{l'}^j|. \tag{6.14}$$

The decisions for $\{b_l^i\}_{i=1,\cdots,N, l \in \mathcal{L}^{\text{int}}}$ are found by solving the following integer programming problem:

$$\begin{array}{ll} \text{minimize} & \sum_{l \in \mathcal{L}, l' \in \delta(l)} \sum_{i}^{N} \sum_{j, i \neq j}^{N} |b_l^i - b_{l'}^j| \\ \text{subject to} & (6.13). \end{array} \tag{6.15}$$

The complexity of integer programming problems grow exponentially with the number

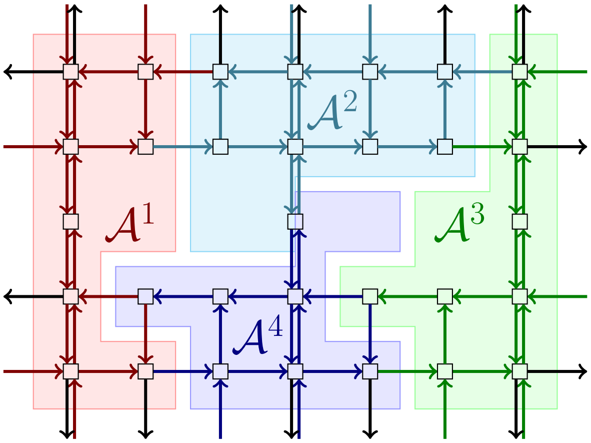


Figure 6·2: Network Partitioned into 4 Subnetworks

of variables. The choices for N, K are determined by the user. Usually, K is related to the computation power available for solving the MPC optimization problems in real time. We choose N as the minimum integer such that a feasible solution to (6.15) exists.

Example 14. The partitioning for the network in Fig. 6·1 with $N = 4, K = 20$, is illustrated in Fig. 6·2. Solving (6.15) with 524 binary variables took 0.65 seconds on a dual core 3GHz Macbook Pro using Gurobi [3]. The internal links of each subnetwork are shown with the same color.

Throughout this chapter, we use "local" and "global" to refer to the attributes of a subnetwork and the original network, respectively. Once a network is partitioned, α^i, β^i, $\mathcal{A}^i$, $i = 1, \cdots, N$, are constructed in the obvious way such that the values in α, β and the pairs in $\mathcal{A}$ are inherited. Note that an interconnection is essentially a local outgoing link for one subnetwork and a local internal link for the other. We define the local version of Φ for subnetwork $\mathcal{N}^i$ as:

$$\Phi^i := ((x_i, u_i) \in \pi^i) \wedge \bigwedge_{\mathrm{Links}(\varphi_k^{conj}) \subseteq \mathcal{L}^{\mathrm{int},i}} \varphi_k^{conj}. \tag{6.16}$$

---

[3] `www.gurobi.com`

Note that the control of each link is assigned to a single subsystem. The control policy, the set of initial conditions and the run of subsystem i are denoted by μ^i, $\mathcal{X}_0^i$, and ζ^i, respectively.

For each $\mathcal{N}^i$, $i = 1, \cdots, N$, we define the sets of downstream and upstream subnetworks as $Down(\mathcal{N}^i) := \{\mathcal{N}^j | \exists l \in \mathcal{L}^{\text{int},i}, \delta(l) \in \mathcal{L}^{\text{int},j}\}$, $Up(\mathcal{N}^i) := \{\mathcal{N}^j | \exists l \in \mathcal{L}^{\text{int},j}, \delta(l) \in \mathcal{L}^{\text{int},i}\}$, respectively. Since most interconnections are two-way, often two subnetworks are both upstream and downstream of each other. Vehicles leaving a subnetwork can enter to one of its downstream subnetworks or leave the entire network. In other words, some components of the output of a subnetwork become additive disturbances for its downstream subnetwork. For all interconnections we have:

$$w^i(l,t) = w(l,t) + \sum_{\mathcal{N}^j \in Up(\mathcal{N}^i), l \in \mathcal{L}^{j,\text{out}}} y^j_{l,t}. \tag{6.17}$$

We write the system equations for each subnetwork as $x^i_{t+1} = F^i(x^i_t, u^i_t, w^i_t)$, $y^i_t = G^i(x^i_t, u^i_t)$.

6.4 Contract Synthesis

In this section, we explain how the assume-guarantee contracts between subnetworks in a partition are found and used for decentralized control synthesis, while satisfying the global specification (6.9). We use the fact that traffic dynamics are *monotone* in the congestion-free set and the satisfaction of (6.9) can be converted into a set-invariance problem in the trajectory space of length $h(\Phi)$.

6.4.1 Assume-Guarantee Contracts

For notation convenience, we denote the components of output of $\mathcal{N}^i$ that affect $\mathcal{N}^j$ as $y_t^{i\to j} := \{y_{l,t}^i\}_{l\in\mathcal{L}^{\text{out},i},\mathcal{L}^{\text{int},j}}$. We also define $\xi^{i\to j} := y_0^{i\to j}, y_1^{i\to j}, \cdots$.

Definition 28. An *assume-guarantee contract* (AGC) $\psi^{i\to j}$ is an MTL formula, with atomic propositions as predicates over $y_t^{i\to j}$, such that:

- subnetwork $\mathcal{N}^j$ *assumes* that $\xi^{i\to j}$ (which acts on as disturbance) satisfies $\mathbf{G}_{[0,\infty)}\psi^{i\to j}$;
- subnetwork $\mathcal{N}^i$ *guarantees* that $\xi^{i\to j}$ (which is its output) satisfies $\mathbf{G}_{[0,\infty)}\psi^{i\to j}$.

The key idea in contract-based control design is that once AGCs are found such that local control polices exist to ensure them, then the local controllers can operate in a decentralized manner. We also require the following assumption for synchronization of time between subnetworks:

Assumption 4. All subnetworks have access to a global clock.

This assumption will be further discussed in Sec. 6.4.5. Given subnetworks $\mathcal{N}^1, \cdots, \mathcal{N}^N$, we design all AGCs such that

$$\bigwedge_{i=1}^{N}(\mathbf{G}_{[0,\infty)}(\Phi^i \wedge \bigwedge_{\mathcal{N}^j\in Down(\mathcal{N}^i)} \psi^{i\to j})) \Rightarrow \Phi^{global}. \tag{6.18}$$

Moreover, there should exist local control policies μ^i and a non-empty set of initial conditions $\mathcal{X}_0^i$, $i = 1, \cdots, N$, such that both assumptions and guarantees are met. That is to say for all allowable $\mathbf{w}^i$ that satisfy $\bigwedge_{\mathcal{N}^j\in Up(\mathcal{N}^i)} \psi^{j\to i}$ (assumptions), we have

$$\sigma_0^i(\zeta^i(x_0^i, \mu^i, \mathbf{w}^i)) \models \mathbf{G}_{[0,\infty)}(\Phi^i \wedge \bigwedge_{\mathcal{N}^j\in Down(\mathcal{N}^i)} \psi^{i\to j}), \tag{6.19}$$

for all $x_0^i \in \mathcal{X}_0^i$ (guarantees). Note that word σ^i is constructed from the local propositions including those of the local contracts.

As one can observe, the design of AGCs falls into circular reasoning for subnetworks that have two-way interconnections. Moreover, AGCs are often not unique. In this chapter, we formulate the problem of contract synthesis as a single constraint satisfaction problem, which leads to feasibility check for a MILP problem. Although the complexity grows exponentially (in the worst case) with respect to network size, we show that the computation time is small for fairly large networks (e.g., the network in Fig. 6·1). Note that contracts are computed offline.

6.4.2 Monotonicity

Proposition 5. System (8.1) is monotone with respect to the additive disturbances in the sense that $\forall x \in \mathcal{X}, \forall u \in \mathcal{U}$, we have $F(x,u,w) \preceq_+ F(x,u,w^{\max}), \forall w \in \mathcal{W}$.

Proposition 6. System (8.1),(6.5) is *monotone with respect to the state* in the congestion-free set, i.e. for all $(x,u) \in \Pi$, $(x',u) \in \Pi$ such that $x \preceq_+ x'$, we have $F(x,u,w) \preceq_+ F(x',u,w), \forall w \in \mathcal{W}$, and $G(x,u) \preceq_+ G(x',u)$.

Proof. First, it is easy to show that system (8.1) is continuous. We need to prove that all the components of the Jacobian $\frac{\partial F}{\partial x}$ are non-negative, whether they are given by the left are right derivative in case of discontinuities. We check the property for (6.2). Observe that $\frac{\partial x_{l,t+1}}{\partial x_{l,t}} \in \{0,1\}$, depending on the active minimizer from (6.1). The only case that $\frac{\partial x_{l,t+1}}{\partial x_{l',t}} < 0$ is when $\exists k, l' \in \delta(k), l \in \delta(k)$ such that $f_{k,l'} = \frac{\alpha(k,l')}{\beta(k,l')}(c_{l'} - x_{l',t})$. However, by the definition of the congestion free set, the last argument in (6.1) is never the minimizer. Thus, the system is monotone. □

It is worth to note that monotonicity property is not valid under FIFO rule for congested flow in diverging intersections [Coogan et al., 2016a]. Monotonicity enables us to evaluate the worst-case trajectories by setting the disturbances to $w^{\max}$, a technique that we use frequently in this chapter.

6.4.3 Language Realization Set

We define the extended state $s_t, t \in \mathbb{N}$ such that

$$s_t := \big((x_t, u_t), (x_{t+1}, u_{t+1}), \cdots, (x_{t+h(\Phi)-1}, u_{t+h(\Phi)-1})\big),$$

where $s_t \in \mathcal{S}$, $\mathcal{S} = \prod_{\tau=0}^{h(\Phi)}(\mathcal{X} \times \mathcal{U})$. For notation convenience, we define $X_{\mathcal{S}}$ and $U_{\mathcal{S}}$ such that $x_t = X_{\mathcal{S}}(s_t)$ and $u_t = U_{\mathcal{S}}(s_t)$. Using s_t and atomic propositions in Φ, one can obtain the observations in $\sigma_{[t,t+h(\Phi))}$. Therefore, we can check whether $\sigma_{[t,t+h(\Phi))}(s_t) \models \Phi$.

Definition 29. The *language realization set* (LRS) of an MTL formula Φ is defined as:

$$\mathrm{LRS}(\Phi) := \big\{s_0 \in \mathcal{S} | \sigma_{[0:h(\Phi))}(s_0) \models \Phi\big\}. \tag{6.20}$$

In other words, $\mathrm{LRS}(\Phi) \subseteq \mathcal{S}$ is the set of all $h(\Phi)$-length runs that generate suffixes satisfying Φ.

Proposition 7. The word generated by $\zeta(x_0, \mu, w)$ satisfies $\mathbf{G}_{[0,\infty)}\Phi$ if and only if $s_t \in \mathrm{LRS}(\Phi), \forall t \in \mathbb{N}$.

The relation $\preceq_{\mathcal{S}}$ is defined such that $s \preceq_{\mathcal{S}} s'$ indicates 1) $x_k \preceq_{+} x'_k$, and 2) $u_k = u'_k$, $k = 0, 1, \cdots, h(\Phi) - 1$.

Proposition 8. The relation $\preceq_{\mathcal{S}}$ is a partial order.

Proof. The proof follows from verifying that $\preceq_{\mathcal{S}}$ satisfies antisymmetry, reflexivity and transitivity. $\square$

For the remainder of this section, we use partial order $\preceq_{\mathcal{S}}$.

Proposition 9. LRS(Φ) $\subset \mathcal{S}$ is a lower-set.

Proof. (sketch) Consider any $s \in$ LRS(Φ) and s' such that $s' \preceq_{\mathcal{S}} s$. We need to show that $s' \in$ LRS(Φ). The decrease in the state components does not falsify any predicate over the state. In addition, the controls are unchanged hence predicates over control are not falsified either. Therefore $s' \in$ LRS(Φ) and LRS(Φ) is a lower-set. □

6.4.4 Controlled Invariance

The evolution of the extended state s_t is written as:

$$s_{t+1} = F_{\mathcal{S}}(s_t, v_t, d_t), \tag{6.21}$$

where $v_t = u_{t+h(\Phi)}$ is the extended control, $d_t = w_{t+h(\Phi)}$ is the extended disturbance, and $F_{\mathcal{S}}(s_t, v_t, d_t)$ is:

$$\big((x_1, u_1), \cdots, (x_{(h(\Phi)}, u_{(h(\Phi)}), (F(x_{(h(\Phi)}, v_t, d_t)), v_t)\big)\,.$$

Proposition 10. System (6.21) is monotone with respect to the extended state in LRS(Φ).

In order to satisfy $\mathbf{G}_{[0,\infty)}\Phi$, we need to compute a robust control invariant set (RCIS) [Kerrigan, 2000] that lies entirely in LRS(Φ). Computation of an RCIS inside a non-convex set for a hybrid system is a computationally difficult problem. We exploit monotonicity to propose an alternative computational approach. The following theorem is an extension of the one in our previous work [Sadraddini and Belta, 2016b]:

Theorem 10. Consider a sequence of extended states s_0^*, s_1^*,$\cdots$, s_{T-1}^*, and a sequence of extended controls v_0^*, v_1^*, $\cdots$, v_{T-1}^*, such that

1. $s_{k+1}^* = F_{\mathcal{S}}(s_k, v_k, d_k^{\max})$;
2. $s_k^* \in \mathrm{LRS}(\Phi)$, $k = 0, 1, \cdots, T-1$;
3. $s_T \preceq s_0^*$.

Then $\Omega := \bigcup_{k=0}^{T-1} L(s_k^*)$ is an RCIS inside LRS(Φ).

We refer to the obtained extended control sequence v_0^*, v_1^*, $\cdots$, v_{T-1}^* as s-sequence, where T is its length. The main feature of the theorem above is that we can compute an RCIS without using the traditional fixed-point algorithm [Kerrigan, 2000], which is computationally intractable for hybrid systems and does not guarantee termination in finite steps.

Piecewise affine systems can be transformed into a set of mixed-integer linear equations [Heemels et al., 2001]. Temporal logic constraints characterizing LSR(Φ) can also be translated into mixed-integer constraints [Raman et al., 2014]. Finally, the terminal condition $s_T^* \preceq s_0^*$ is a linear constraint. The details of the procedures are not explained in this chapter as they are well documented in the mentioned works. The conditions in Theorem 10 becomes equivalent to finding a feasible solution to a MILP problem. Note that even though the computational complexity of solving a MILP is NP-hard in general, finding a feasible solution is significantly faster than finding an optimal solution.

In order to compute an RCIS, we start from $T = 1$ and implement $T \leftarrow T + 1$ until the MILP for Theorem 10 becomes feasible. In [Sadraddini and Belta, 2016b], we only considered invariance of a set in the state space and we showed that computing RCISs using a simplified version of Theorem 10 is almost complete. That is to say, as T

becomes larger, a feasible solution should exist if there exists any non-empty RCIS. However, the same result does not hold here as we are also considering propositions over controls.

Example 15. We formulated the conditions in Theorem 10 for the network in Fig. 6·1. We have $\varphi = \bigwedge_{i=1}^{4} \varphi_i$, where φ_i's are given in Example 13. Note that $h(\varphi) = 6$. The smallest T for which a feasible solution exists is $T = 6$. The corresponding MILP has 5981 variables (1236 binary) and 2902 constraints. It takes 8.6 seconds on a dual core 3.0 GHz MacBook Pro to find a feasible solution and hence an RCIS in LRS(Φ), which lies in $\mathbb{R}_+^{420} \times \{0,1\}^{384}$.

Proposition 11. A control policy satisfying Φ^{global} starting from x_0 exists if $x_0 \in \mathcal{X}_0^{\Omega}$, where

$$\mathcal{X}_0^{\Omega} := \left\{ x_0 \in \mathcal{X} \middle| \exists i \in \{0, \cdots, T-1\}, x_0 \leq X_{\mathcal{S}}(s_i^*) \right\}. \tag{6.22}$$

The set of admissible initial conditions are characterized by the computed RCIS. In order to find the set of all admissible initial conditions, one has to compute the maximal RCIS, which is unfortunately not possible beyond few dimensions.

Proposition 12. [Sadraddini and Belta, 2016b] If $v_0^*, v_1^*, \cdots, v_{T-1}^*$ is a s-sequence corresponding to $s_0^*, s_1^*, \cdots, s_{T-1}^*$, then for all initial conditions in $L(X_{\mathcal{S}}(s_0^*))$, applying the following open-loop control sequence:

$$u^{\text{open-loop}} := \overline{v_0^*, v_1^*, \cdots, v_{T-1}^*} \tag{6.23}$$

guarantees satisfaction of Φ^{global} for all allowable $\mathbf{w}$.

Therefore, an open-loop solution to Problem 7 is obtained but without any optimality considerations.

6.4.5 Contracts

Now we explain how to extract contracts from a feasible solution for the constraints in Theorem 10. We denote s, s^* and $\mathcal{S}$ corresponding to subnetwork $\mathcal{N}^i$ by s^i, $s^{*,i}$ and $\mathcal{S}^i$, respectively.

Definition 30. The *global clock* is defined as $\mathbb{G} : \mathbb{N} \to \{0, 1, \cdots, T-1\}$, where $\mathbb{G}_t$ is its value at time t.

As mentioned earlier, we assume that all the local controllers have the knowledge of the global clock. The "natural" evolution of the global clock is such that if $\mathbb{G}_t = \tau$, then $\mathbb{G}_{t+1} = (\tau + 1) \bmod T$. However, in case local controllers share information, we allow the value of the global clock to be determined by the local controllers. The details are explained in Sec. 6.5.2.

Proposition 13. Suppose the global clock value at time t is τ and the extended state is s_t, where $s_t \preceq_{\mathcal{S}} s^*_\tau$. Then we have $y^i_t \preceq_+ y^{*,i}_\tau$, where $y^{*,i}_\tau = G^i(x^{*,i}_\tau, u^{*,i}_\tau)$, $x^{*,i}_\tau = X_{\mathcal{S}}(s^*_\tau), u^{*,i}_\tau = U_{\mathcal{S}}(s^*_\tau)$.

Therefore, we have

$$w^{i,\max}(l,\tau) = w^{max}(l,\tau) + \sum_{\mathcal{N}^j \in Up(\mathcal{N}^i), l \in \mathcal{L}^{j,\text{out}}} y^{*,j}_{l,\tau}. \tag{6.24}$$

Proposition 14. We have $s_t \preceq_{\mathcal{S}} s^*_\tau$ if and only if $s^i_\tau \preceq_{\mathcal{S}^i} s^{*,i}_\tau$, $i = 1, \cdots, N$.

Considering these, the contracts are given as:

$$\psi^{i\to j} = \bigwedge_{\tau=0}^{T-1} \left((\tau = \mathbb{G}_t) \Rightarrow (y^{i\to j}_t \leq y^{*,i\to j}_\tau) \right). \tag{6.25}$$

This contract synthesis process ensures that there exists at least one local control policy for each subnetwork such that the contracts are maintained alongside with the local specifications. One such local policy is the open-loop control sequence in (6.23), which can be implemented in a decentralized way. Often other local control policies also exist from which an optimal one can be selected considering a cost function, as discussed in the next section.

6.5 Control Synthesis

In this section, we explain how to find controls optimally. The cost function in (6.10) can be written as:

$$J := \sum_{i=1}^{N} J^i,\ \ J^i(x_0^i, \mu^i, \mathbf{w}^i) = \sum_{t=0}^{\infty} \sum_{l \in \mathcal{L}^{i,\text{out}}} \gamma^t (x_{l,t}^i - f_{l,t}^i). \tag{6.26}$$

It is worth to note that when the system is in the congestion-free set, $f_{l,t}$ only depends on the state and control of link l. Thus the decomposition of the cost as (6.26) is valid. We wish to find an optimal control policy μ^i for each subnetwork $\mathcal{N}^i$ such that its local specification and contracts to downstream neighbors are satisfied:

$$\begin{array}{ll} \text{minimize} & \max_{\mathbf{w}^i} J^i((x_0^i, \mu^i, \mathbf{w}^i)) \\ \text{subject to} & \sigma_0^i(\zeta^i(x_0^i, \mu^i, \mathbf{w}^i)) \models \big(\mathbf{G}_{[0,\infty)}\Phi^i \wedge \\ & \bigwedge_{j \in Down(\mathcal{N}^i)} \psi^{i \to j}\big),\ \forall \mathbf{w}^i. \end{array}$$

We find controls optimally using a decentralized MPC approach. In this setting, local controllers do not exchange any information and the pre-designed contracts are the only global provision. Next, we explain how to extend the decentralizing framework into a simple *cooperative MPC* algorithm where the local controllers determine the value of the global clock by exchanging some information.

6.5.1 Decentralized Model Predictive Control

Given an MPC prediction horizon H, we denote $u_t^{i,H} := u^i_{0|t}, \cdots, u^i_{H|t}$. The length H is determined by the user but as explained later, we recommend $H > h(\Phi)$. Given $w_t^{i,H} = w^i_{0|t}, \cdots, w^i_{H-1|t}$, the H-length prediction of the system state and output are denoted by $x_t^{i,H} = x^i_{0|t}, \cdots, x^i_{H|t}$ and $y_t^{i,H} = y^i_{0|t}, \cdots, y^i_{H|t}$, respectively. At each time, we optimize $u_t^{i,H}$, implement $u^i_{0|t}$ and solve the optimization problem at next time. The global clock at time t is supposed to be a known value. The MPC optimization problem at time t is given as follows:

$$
\begin{array}{rll}
u_t^{i,H} = & \text{argmin} & \displaystyle\sum_{k=0}^{H} \sum_{l \in \mathcal{L}^{i,\text{out}}} \gamma^k (x^i_{l,k|t} - f^i_{l,k|t}) \\
& \text{subject to} & s^i_{k-h(\Phi)|t} \in \text{LSR}(\Phi^i) \\
& & \xi^{i\to j}_{k|t} \models \mathbf{G}_{[0,H]} \psi^{i\to j}, j \in \mathit{Down}(\mathcal{N}^i), \\
& & s^i_{H-h(\Phi)|t} \in L(s^*_{\tau+H-h(\Phi)}), \\
& & x^i_{k+1|t} = F(x^i_{k|t}, u^i_{k|t}, w^{i,\max}_{k|t}), \\
& & \mathbb{G}_{t+k} = (\tau + k) \bmod T, k = 0, \cdots, H.
\end{array}
\tag{6.27}
$$

There are five lines of constraints that are explained as follows. The first is indicating that the all the H-length predictions of the extended states are in the language realization set, hence the specification is not violated in finite time. Due to the MTL temporal operators, the time window of constraints is shifted by $h(\Phi)$ [Sadraddini and Belta, 2015]. It is worth to note that we require the knowledge of the recent $h(\Phi)$-length history of the controls and states. For time $t = 0$, we assume that all the previous propositions from this history are true [Sadraddini and Belta, 2015]. The second line stands for the constraints of the contracts for downstream subnetworks. The third line states that the last predicted extended state has to lie inside the projection of RCIS Ω on $\mathcal{S}^i$, which provides a sufficient condition for establishing recursive feasibility (see Theorem 11 below). Since the partial ordering $\preceq_{\mathcal{S}^i}$ has equality constraints on controls, the length of the MPC horizon with free decision variables is $H - h(\Phi)$. Therefore, we need $H > h(\Phi)$. Otherwise, there exists only one feasible solution to (6.27). The fourth line is stating that the predictions are computed using the largest values of disturbances, which due to monotonicity, corresponds to the worst-case scenario. Thus, the MPC algorithm is *robust* in the sense that all constraints are satisfied for all allowable disturbances. The fifth line stands for the natural evolution of the global clock and the range of indices of predictions.

Theorem 11. The MPC optimization problem (6.27) is recursively feasible in the sense that if it is feasible at time t and a valid control decision is implemented, then it is guaranteed to be feasible at time $t + 1$.

Proposition 15. The global specification (6.9) is satisfied if (6.27) is recursively feasible for all subnetworks.

It should be noted that the solutions obtained from MPC are optimal only over the finite prediction horizon. Therefore, solutions can be suboptimal compared to the global optimum in (6.11). We also note that contracts can introduce conservativeness since subnetworks assume maximum allowable disturbances from the upstream neighbors. To mitigate this conservativeness, we desire that the number of subnetworks - and interconnections - be as small as possible. A very long prediction horizon can also cause conservativeness since the worst-case values at far future times are considered in the optimization problem. This issue can be alleviated by using an appropriate discount factor in the cost function. In the case when the state knowledge is noisy, the values for states in (6.27) have to be replaced by their upper-bound estimates. Due to monotonicity, this ensures that the correctness of the specification and the contracts are maintained for all possible state values. However, one may get infeasibility for (6.27) because of unrealistic state measurements, even though feasibility is guaranteed for the true state values. In this case, one has to relax the constraints. A less conservative treatment of noisy data may require a probabilistic framework, which is out of the scope of this chapter.

6.5.2 Cooperative Model Predictive Control

Here we assume that the local controllers are able to communicate with each other. In order to improve optimality, we introduce a simple modification for the decentralized MPC algorithm as follows. At each time, the controllers implement (6.27) for all allowable values of the global clock. Therefore, we define $J^{opt,i} = \{J^{opt,i}_\tau\}_{\tau \in \{0,\cdots,T-1\}}$, $J^{opt,i} \in \mathbb{R}^T_+$, where $J^{opt,i}_\tau$ is the optimal cost of MPC for subnetwork $\mathcal{N}^i$, $i = 1, \cdots, N$, obtained from setting the clock variable $\mathbb{G}_t$ to τ. If an MPC optimization problem is infeasible, we set its cost to ∞. Next, we choose the *best* global clock value as follows:

$$\tau^* = \min_{\tau \in \{0,\cdots,T\}} \sum_{i=1}^{N} J^{opt,i}_\tau. \tag{6.28}$$

Note that recursive feasibility guarantees that for at least one clock variable the sum of costs is finite. Distributed computation of (6.28) can be accomplished using a distributed average consensus algorithm [Cortés, 2006], assuming that the controllers of the subnetworks communicate on a connected graph. Then, each controller implements the controls which correspond to the minimum cost in the average $\frac{1}{N}\sum_{i=1}^{N} J^{opt,i}$. We note that our cooperative MPC technique is still preliminary and there are many open directions to improve this approach.

6.6 Simulation Results

In this section, we present numerical results of applying our methods to the network shown in Fig. 6·1. Motivated by time scales of real traffic networks, we consider a time step of 20 seconds in all simulations.

6.6.1 Macroscopic Simulation

We used the model in (8.1), which is "macroscopic" in the sense that it describes the aggregated vehicular dynamics instead of modeling each vehicle. The specification is given as in Example 15. For all MPC algorithms, we use $H = 11$ and $\gamma = 0.5$. We simulate the system for 45 time steps. The total delay accumulated over the network during this time frame is $\sum_{t=0}^{45} \sum_{l \in \mathcal{L}^{\text{int}}} (x_{l,t} - f_{l,t})$. All the computation times are given for implementations on a dual-core 3.0GHz Macbook Pro. The software for implementations are available for download in `sites.bu.edu/hyness/format-distributed`. In the following implementations, the values for the exogenous demand for the network are drawn from a uniform distribution over $L(w^{\max})$, with the exception of links from subnetwork $\mathcal{N}^1$, where we set their exogenous demands to their largest admissible values.

Open-loop (OL). We implemented the control sequence from (6.23). This control policy is not traffic-responsive but ensures the satisfaction of the global specification (6.9). The total accumulated delay was 4786 [vehicles×time-step]. The implementation does not require any online computation effort or measurement of the state.

Centralized MPC (CeMPC). We implemented the MPC algorithm (6.27) for the complete network $\mathcal{N}$. We do not need the contract constraints as the network is undivided in this setup. Therefore, there is no conservativeness induced by the contracts. The accumulated delay was 2878, which indicates about 40% decrease compared to the OL policy. However, the computation time for each time step is very large (see Table 6.1), which indicates that CeMPC is not suitable for real time traffic management.

Decentralized MPC (DeMPC). Here we implemented the MPC algorithm (6.27) for each subnetwork individually. The accumulated delay was 3216, which is a bit larger than the one for the CeMPC but still significantly smaller than the one for the OL policy. The computation times were less than a second (see Table 6.1). Therefore, DeMPC is appropriate for real time traffic management.

Cooperative MPC (CoMPC). Here we implemented the method from Sec. 6.5.2. During the simulation, the natural evolution of the global clock was overridden for 7 times, mainly in order to prioritize the heavy traffic in subnetwork $\mathcal{N}^1$. The accumulated delay was 3112, which is slightly smaller than the one for DeMPC. The computation times are longer due to solving multiple MILPs ($T = 6$ in this case), but the computations can be performed in parallel.

It is observed that the specification is satisfied by each implementation (which is also implied by the the fact that all the MPC problems were feasible). The trajectories always remain in the congestion-free set and all the sub-specifications in Example 13 are always met. For instance, the traffic lights corresponding to the sub-specifications φ_1, φ_2, and the vehicular volumes over time are shown in Fig. 6·4 (using DeMPC with all exogenous demands set to their maximum). It is also observed that the number of vehicles on the eastern bridge never exceeds 100 (sub-specification φ_3). For the only case when the volume on link 73 exceeded 5, its traffic light turned green immediately (sub-specification φ_4).

Table 6.1: Computation Time per Time Step and Accumulated Delay for Different Control Policies

	OL	CeMPC	DeMPC	CoMPC
Max. Comp. Time (s)	-	1762	0.93	7.09
Avg. Comp. Time (s)	-	86.7	0.17	1.21
Accumulated Delay	4786	2878	3216	3112

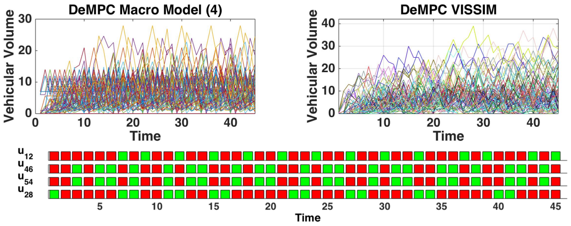


Figure 6·3: Simulation Results

6.6.2 Microscopic Simulation

Here we show some preliminary results on implementing our methods on microscopic traffic simulators. We used the PTV VISSIM[4] microscopic traffic simulator, which is a widely used, highly realistic simulator that incorporates many aspects of traffic such as driver models, vehicle classes, priorities, conflicts, etc. A screenshot of the VISSIM implementation of the traffic network in Fig. 6·1 is shown in Fig. 6·5. The simulator was used in a closed-loop setup: the controller has access to the traffic volumes and the traffic light settings via a MATLAB interface. The length of each link is set to 300 meters. The maximal speed of the vehicles was set to 15 m/s with an acceleration profile of a normal passenger car. The sample time is 20 seconds. We simulate the system for 15 minutes (equivalent to 45 time steps). We observe that a naive (centralized) MPC controller with no feasibility guarantees leads to congestion in the network after about 25 time steps. However, using the methods in this chapter we were able to to avoid congestion and satisfy all the specifications. The results are shown in Fig. 6·4. The video of our VISSIM implementation is included in `sites.bu.edu/hyness/format-distributed`.

---

[4] `http://vision-traffic.ptvgroup.com/products/ptv-vissim`

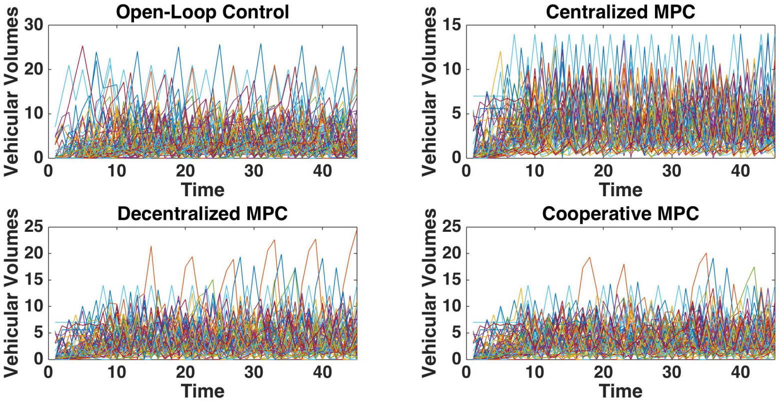


Figure 6·4: Macroscopic Simulation Trajectories

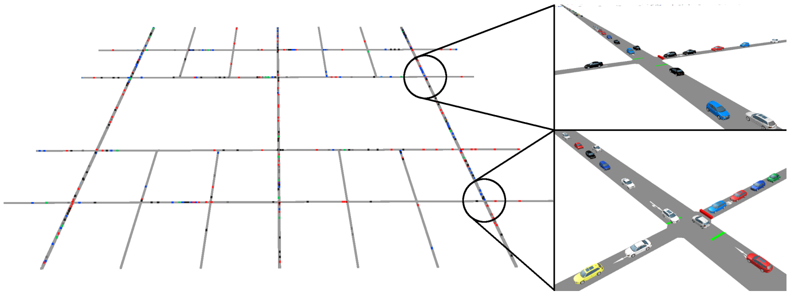


Figure 6·5: VISSIM Implementation

The macroscopic model (8.1) used for control synthesis does not necessarily capture all the behaviors in the VISSIM model. Therefore, we are unable to formally guarantee that the specification is always satisfied by the VISSIM model. However, in our simulations we observed that the VISSIM simulations always satisfy the specification using the controls we found for (8.1). We leave further investigation of the relation between macroscopic and microscopic models from a formal methods perspective to our future work.

6.7 Dynamic Contracts

We can find a library of contracts instead of using a fixed table of contracts, and switch between in reaction to real-time conditions them to obtain a better optimum.

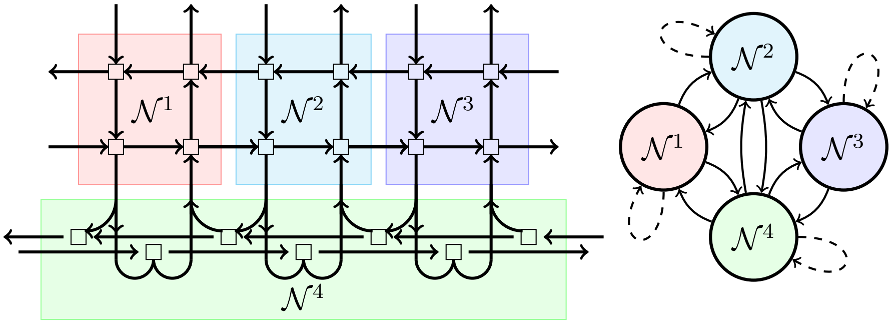


Figure 6·6: Example network $\mathcal{N}$ partitioned into four sub-networks $\mathcal{N}^i, i = 1, \cdots, 4$. Contract obligations from (6.7.2) are depicted on the right. Solid arrows indicate an obligation from one network to an adjacent sub-network to limit incoming vehicular flow. Dashed arrows represent recursive feasibility obligations from a sub-network to its future self.

We follow a compositional approach. The details are in [Kim et al., 2017c]. Each individual sub-network must first be "mined" for contracts over a range of demand severity levels and different local scenarios. The mined contracts are merged into a finite directed graph that serves as a high-level coordinator. Each node corresponds with the contract constraints imposed on each individual sub-network. Unsafe regions of the graph are scenarios where sub-networks' promises to each other are inconsistent. The coordinator graph contains edges only when a hand-off between contracts maintains recursive feasibility. A running example of a mixed freeway-urban network is used throughout this chapter. Simulations show a reduction in overall delay with dynamic contracts and that recursive feasibility is maintained when contract transitions occur.

6.7.1 Motivating Example

Consider a network $\mathcal{N}$ depicted in (6·6) which consists of sub-networks $\mathcal{N}^i, i = 1, \cdots, 4$. Sub-network $\mathcal{N}^4$ represents a high capacity freeway while the others are urban areas. All sub-networks are interconnected via on(off)-ramps or urban roads. At each urban intersection, 50% of the vehicles proceed straight, 20% turn left, and 30% turns into an un-modeled external environment (e.g., parking). We have $q^l = 15, c^l = 40$ for urban roads, $q^l = 60, c^l = 25$ for freeways and $q^l = 30, c^l = 15$ for ramps. For all entry links to the network, let $d^l[t] \in [0, \frac{1}{4}q^l], t \in \mathbb{N}$ [5].

We aim for multiple control objectives. First, the network must remain in the congestion-free region ψ at all times. Second, at each urban intersection traffic lights signalizing vertical and horizontal flows become simultaneously red infinitely often, allowing pedestrians to pass through the intersection in any direction and making the congestion-free specification harder to accomplish.

$$\phi = \mathbf{G}_{[0,\infty)}\Big(\bigwedge_{(l,k)\in\mathcal{A}} \psi \wedge \mathbf{F}_{[0,\infty)}(u_l = 0 \wedge u_k = 0)\Big). \tag{6.29}$$

6.7.2 Interconnections and Contracts

The distributed MPC scheme contains a circular dependency because each sub-network is unaware of neighboring networks' planned actions. Each sub-network needs to promise neighboring sub-networks that they will satisfy each other's assumptions, but the feasibility of such a promise depends on the actions of one's neighbors. Assume-guarantee contracts are MPC constraints that break this dependency.

Definition 31 (Assume-Guarantee Contract). Network $\mathcal{N}^i$'s assume-guarantee contract $\mathcal{C}^i$ along time interval $[kT, (k+1)T]$ consists of

- Assumption $\phi_a^i(x_*^i[kT], d_*^i[kT, (k+1)T])$ on the incoming demand and vehicles initially in the network:

---

[5] The full, detailed, parameter valuations of the network, code and simulation results of this chapter are publicly available in `http://blogs.bu.edu/sadra/research/dynamic-contracts`.

$$\bigwedge_{l\in\mathcal{L}^i} \big(x^l[kT] \le x_*^l[kT]\big) \tag{6.30}$$

$$\wedge \bigwedge_{t=kT}^{(k+1)T-1} \Big(\bigwedge_{l\in\mathcal{L}^i_{\text{in}}} \big(d^l[t] \le d_*^l[t]\big)\Big) \tag{6.31}$$

- Guarantee $\phi_g^i(x_*^i[kT+1,(k+1)T], y_*^i[kT+1,(k+1)T])$ on the terminal state and output trajectory:

$$\bigwedge_{l\in\mathcal{L}^i} \big(x^l[(k+1)T] \le x_*^l[(k+1)T]\big) \tag{6.32}$$

$$\wedge \bigwedge_{t=kT}^{(k+1)T} \Big(\bigwedge_{l\in\mathcal{L}^i_{\text{out}}} \big(y^l[t] \le y_*^l[t]\big)\Big) \tag{6.33}$$

The contract $\mathcal{C}^i$ is characterized by a set of parameters: the initial state $x_*[kT]$, external demand $d_*[kT,(k+1)T)$, terminal state $x_*[(k+1)T]$, and output trajectory $y_*[kT,(k+1)T])$ over output links. Sub-network $\mathcal{N}^i$'s assumption component states conditions over local and incoming links $\mathcal{L}^i$ and $\mathcal{L}^i_{\text{in}}$. The output guarantee is viewed as a signal $y_*[kT+1,(k+1)T])$ that upper bounds output trajectories of $y[kT+1,(k+1)T])$ on links in $\mathcal{L}^i_{\text{out}}$.

Definition 32 (Contract Satisfaction). A sub-network satisfies an assume-guarantee contract at time kT if for all $x[kT]$ and $d[kT,(k+1)T)$ satisfying (6.30) and (6.31) respectively, a control sequence $u[kT,(k+1)T)$ exists such that $\mathcal{N}^i$ remains in the local freeflow region ψ^i, and both the guarantee ϕ_g^i and MTL requirement ϕ^i are satisfied at kT.

Contract satisfaction for *all* assumption satisfying scenarios $x[kT]$ and $d[kT,(k+1)T)$ is difficult to certify for general non-linear dynamics. However, within the congestion free region ψ the dynamics exhibit a monotonicity property, where a partial ordering with respect to state trajectories is preserved, and the initial state $x_*[kT]$ and demand $d^i_*[kT,(k+1)T)$ jointly yield the most adversarial environment. An environment that satisfies ϕ^i_a cannot violate the guarantee if $x_*[kT]$ and $d^i_*[kT,(k+1)T)$ doesn't violate the assumption [Kim et al., 2016]. Thus, synthesizing a satisfying control sequence $u^i_*[kT,(k+1)T)$ for environmental scenario $x_*[kT]$ and $d_*[kT,(k+1)T)$ such that the system satisfies the guarantees and remains congestion free also ensures that the control sequence will be satisfactory under more benign scenarios. A set of contracts is consistent if all sub-network assumptions are implied by the guarantees.

Definition 33 (Assume-Guarantee Parameter Consistency). A set of contracts are *consistent* if if for all $\mathcal{N}^i$, input links $l \in \mathcal{L}^i_{\text{in}}$ and times $t \in [kT,(k+1)T)$

$$\sum_{k \in \mathcal{L}^l_{\text{up}}} \beta(l,k) y^k_*[t] \leq d^l_*[t]. \tag{6.34}$$

6.7.3 Recursive Feasibility with Fixed Contracts

Recursive feasibility can be viewed as a network making a promise to its future self that all future constraints will remain feasible. Recursive feasibility has two components, corresponding to the specification ϕ^i_g constraint and the contract constraint (6.33). Feasibility of the ϕ^i_g constraint at the kT-th time step has already been established via the initial state condition (6.30).

Let contract $\mathcal{C}^i$ is be periodically every T steps with fixed parameters. Consider two different MPC executions at times kT and $(k+1)T$. The guarantee constraint ϕ^i_g along the interval $[(k+1)T,(k+2)T)$ is feasible when (6.30) is satisfied at time $(k+1)T$. The MPC algorithm executing at time kT imposes the terminal state guarantee $x[(k+1)T] \leq x_*[(k+1)T]$, which implies that the initial state assumption at $(k+1)T$ with identical contract $\mathcal{C}^i$ is satisfied if:

$$\bigwedge_{l \in \mathcal{L}^i} x^l_*[(k+1)T] \leq x^l_*[(k+1)T]. \tag{6.35}$$

If $\mathcal{C}^i$ satisfies (6.35) then it is said to be a recursively feasible contract. The final MPC problem for each sub-network consists of the following constraints:

Problem 8 (Distributed MPC). Under the assumption that input demand satisfies (6.31), Each sub-network $\mathcal{N}^i$ computes a local control sequence $u[kT,(k+1)T)$:

$$\begin{array}{rl} \underset{u[kT,(k+1)T)}{\operatorname{argmin}} & \displaystyle\sum_{t=kT}^{(k+1)T} \sum_{l \in \mathcal{L}} \left(x^l[t] - f^l[t]\right) \\ \text{s.t.} & (x[kT,(k+1)T], u[kT,(k+1)T)) \models \phi^i \\ & \text{Guarantee (6.33) to adjacent networks} \\ & \text{Terminal State (6.32), Dynamics constraints} \end{array}$$

Assumption ϕ^i_a is encoded in the dynamics.

Proposition 16 (Infinite Horizon Spec. Satisfaction). If each network $\mathcal{N}^i$ satisfies its assume-guarantee contract, each assume-guarantee contract $\mathcal{C}^i$ is recursively feasible, and the global initial state $x[0]$ satisfies each initial state (6.30) assumption, then the distributed MPC algorithm satisfies the global specification.

6.7.4 Dynamic Contracts

(31) introduced contracts that are uniquely parametrized by $x_*[kT]$, $d_*[kT,(k+1)T)$, $x_*[(k+1)T]$, and $y_*[kT,(k+1)T)$, which do not change over many MPC horizons. Fixed contract parameters may lead to conservative guarantees if the network experiences less demand than expected and $d^l[kT,(k+1)T) \ll d^l_*[kT,(k+1)T)$ elementwise in (6.31). because conservative assumptions prevent aggressive responses to benign real-time conditions.

In general, a sub-network $\mathcal{N}^i$ can satisfy a collection of m^i assume-guarantee contracts,

$$\mathcal{P}^i = \{\mathcal{C}^i(p^i_1), \dots, \mathcal{C}^i(p^i_{m_i})\}. \tag{6.36}$$

each associated with different parameters (attributes)

$$\begin{gathered} p^i[kT,(k+1)T] = \Big(x^i_*[kT], d^i_*[kT,(k+1)T), \\ x^i_*[(k+1)T], y^i_*[kT,(k+1)T) u^i_*[kT,(k+1)T), J^i_*\Big) \end{gathered}$$

where p^i is used for notational compactness in (6.36). (6.7.6) provides a method to generate such a collection. Optimal control sequence $u^i_*[kT,(k+1)T)$ and an induced delay J^i_* are also computed and stored during the contract generation process.

Preserving formal guarantees restricts how contract parameters may change at runtime. First, contracts parameters must always be consistent in the sense of (33). Second, the notion of recursive feasibility needs to be modified to accommodate a changing set of requirements.

Concretely, a contract coordinator is a transition system with state space $\mathcal{P} = \prod_{i=1}^N \mathcal{P}_i = \prod_{i=1}^N \{\mathcal{C}^i(p^i_1), \dots, \mathcal{C}^i(p^i_{m_i})\}$ designed to ensure that these two properties are satisfied. Every coordinator transition corresponds to a potential change in contract parameters for each network and may execute every T time steps. After a transition, each sub-network is notified of the contract it must satisfy.

Consistent Contract Parameters

A coordinator state $p \in \mathcal{P}$ is a tuple $(p^1_{k_1}, \dots, p^N_{k_N})$ where each network $\mathcal{N}^i$ picks a single contract $\mathcal{C}(p^i_{k_i})$ it would like to satisfy. Not all elements of this space satisfy the contract consistency requirement in (33). For instance in (6·6) satisfaction of $\mathcal{N}^1$'s assumption is determined by $\mathcal{N}^2$ and $\mathcal{N}^4$'s guarantees. If contract parameters are such that $\mathcal{N}^2, \mathcal{N}^4$'s guarantees do not jointly imply assumption $\mathcal{N}^1$'s assumption, then these contract parameters are inconsistent.

A subset $\mathcal{P}_v \subseteq \mathcal{P}$ of the parameter space that corresponds to all consistent network-wide parameters. Set $\mathcal{P}_v$ is enumerated via a depth first traversal over a tree with depth N and branching factors m_i. The traversal aggressively prunes branches of the contract space as soon as a contract inconsistency parameter is identified.

Contract Transitions and Recursive Feasibility

A contract transition is valid when each sub-network can promise its future self the ability to satisfy the new contract via a transition from the old contract. Given two consistent contract parameters $p, \hat{p} \in \mathcal{P}_v$ where p is for use over $[kT, (k+1)T]$ and $\hat{p}$ is for use over $[(k+1)T, (k+2)T]$, a switch from p to $\hat{p}$ is *valid* if the following element-wise inequality holds:

$$\bigwedge_{l \in \mathcal{L}} x^l_*[(k+1)T] \leq \hat{x}^l_*[(k+1)T]. \tag{6.37}$$

The left side is the terminal state guarantee from p and the right is the initial state assumption of $\hat{p}$. Recursive feasibility as in (6.7.3) is a special case when $p = \hat{p}$.

We define the contract parameters *Viable Graph* (VG) as a directed graph $(\mathcal{P}_v, \mathcal{E}_v)$, where $\mathcal{P}_v$ is the set of nodes, and $\mathcal{E}_v \subseteq \mathcal{P}_v \times \mathcal{P}_v$ is the set of edges such that $\forall (p, \hat{p}) \in \mathcal{E}_v$, the switch from p to $\hat{p}$ is valid. We denote $\mathcal{E}^p_v = \{(p, \hat{p}) | (p, \hat{p}) \in \mathcal{E}_v\}$. A node p is a dead-end if $\mathcal{E}^p_v = \emptyset$. If no dead-end is reached, then there always exists a consistent contract with feasible transition options to other contracts, which by construction implies the following statement.

Proposition 17. Given a infinite-time contract parameter sequence $p_0, p_1, \cdots$, where p_k is used for control synthesis in the time interval $[kT, (k+1)T]$, the specification (6.29) is satisfied if $(p_k, p_{k+1}) \in \mathcal{E}_v$ and $p_k \in \mathcal{P}_v, \forall k \in \mathbb{N}$.

6.7.5 Optimal Contract Coordination

The recursive feasibility property and contract consistency require that no dead-end node in VG is reached. By recursively removing the dead-end nodes and the edges leading to them, we obtain a fixed point which characterizes the *viable kernel graph* (VKG) $(\mathcal{P}_{v,\kappa}, \mathcal{T}_{v,\kappa})$, where $\mathcal{P}_{v,\kappa} \subset \mathcal{P}_v$ and $\mathcal{T}_{v,\kappa} \subseteq \mathcal{P}_{v,\kappa} \times \mathcal{P}_{v,\kappa}$ with the extra property that $\forall p \in \mathcal{P}_{v,\kappa}, \mathcal{E}^p_{v,\kappa} \neq \emptyset$. Once a parameter contract of a node in VKG is chosen, there always exist a feasible handover of the contract to another node in VKG, establishing infinite-time recursive feasibility and consistency.

Each contract p^i corresponds to a cost J^i_* for network $\mathcal{N}^i$, which is the delay induced if control sequence $u^i_*[kT, (k+1)T)$ is applied starting from $x^i_*[kT]$ under the demand assumptions $d^i_*[kT, (k+1)T)$. It follows from monotonicity properties that J^i_* is a upper-bound for possible costs in real-time implementation. Given a contract parameter $p = (p^1_{k_1}, \cdots, p^N_{k_N})$, the sum of associated contract costs is

$$c(p) := \sum_{i=1}^{N} J^i_*(p_{k_i}). \tag{6.38}$$

Now we determine which contract parameter from VKG nodes to choose at each time $kT, k \in \mathbb{N}$. In order to aim for optimality, we choose the contract parameter for which the infinite-horizon cost $c_\infty(p) = \sum_{k=0}^{\infty} \alpha^k c(p_k)$ is minimal, where $p = p_0$, and $\alpha \in (0, 1)$ is a discount factor to make the cost properly defined. Denote the optimum cost to go from p by c^*_∞ which follows from Bellman's equation [Bertsekas, 1995]:

$$c^*_\infty(p) = c(p) + \alpha \min_{p' \in \mathcal{E}^p_{v,\kappa}} (J^*_\infty(p')) \tag{6.39}$$

In order to find $c^*_\infty(p), p \in \mathcal{P}_{v,\kappa}$, (6.39) is cast as a linear program. Finally, at time kT we choose the optimal contract parameter $p^* \in \mathcal{P}_{x[KT]}$ with minimum $c^*_\infty(p)$.

6.7.6 Contract Mining

A delicate tradeoff exists between conservative assumptions which can accommodate an influx of vehicles and aggressive guarantees which quickly dissipate vehicles in the network. We present a heuristic to generate a set of assume-guarantee pairs for each sub-network.

Given a fixed assumption and $\mathcal{N}^i$, a miner is a bounded horizon optimization algorithm that computes a control trajectory that induces minimal guarantees. Guarantee parameters $x^i_*[(k+1)T], y^i_*[kT,(k+1)T)$ are minimal if contract satisfaction as in (32) is infeasible for any smaller guarantee pair such that $\hat{x}^i_*[kT] \leq x^i_*[kT]$ and $\hat{y}^i_*[kT,(k+1)T) \leq y^i_*[kT,(k+1)T)$ with element-wise inequality. The miner's optimization objective can be any monotone function of $\mathcal{X} \times \mathcal{U}$; we opt to minimize a combination of the l_1 and l_∞ norms. After mining, a guarantee is propagated into an assumption for adjacent networks via (6.34) with the equality is replaced with an inequality, and the mining continues.

(1) provides pseudocode which generates `MaxIter` guarantees for every sub-network. The contract sets are initially empty. Infeasibility of the mining algorithm triggers a multiplicative decrease in the initial conditions and disturbance by a factor $\gamma \in (0,1)$ until the contract is satisfied. Propagation can be visualized over the bottom of (6·6) where inter-network promises (solid edges) and a network's recursive feasibility constraint (dashed edges) are both updated. Both contract parameters and the control sequence are saved.

6.7.7 example

Example 16. (6·6)'s network is used to evaluate the efficacy of the dynamic contracts system. (1) is used to generate a set of 25 contract parameters for each sub-network.

Algorithm 1 Guarantee Mining Algorithm

```
Set $N$ = Number of Sub-networks
for all $i = 1, \dots,$ MaxIter do
    for all $i \in \{1, \dots, N\}$ do
        while True do
            Feas, $x[\cdot]$, $u[\cdot]$, $y[\cdot]$= mine($\mathcal{N}^i$,$x[0]$, $d[\cdot]$)
            if Feas then
                Break
            $x[0] := \gamma x[0]$, $d[\cdot] := \gamma d[\cdot]$
        Add to $\mathcal{N}^i$s contract set
        Propagate Guarantee
```

Table 6.2: Accumulated Delays for Different Control Methods

Networks Experiencing Full Demand	**Dynamic Contracts Fixed Control**	**Fixed Contracts MPC**	**Dynamic Contracts MPC**
All	784	753	747
(1,4)	354	381	346
(2,4)	248	244	226
(1,2,4)	513	513	495
(1,2,3)	670	646	635
(1,2)	405	398	394
(1,3)	498	507	460
(1)	238	249	229
(2)	154	122	104
(4)	115	86	85

There are $|\mathcal{P}_v| = 1363$ consistent contract parameters, of which 664 were members of the viability kernel $\mathcal{P}_{v,\kappa}$. All optimization problems were posed and solved using `Gurobi`'s mixed integer linear program solver [Gurobi Optimization, 2016]. We used the method in (6.7.4) to change contract parameters as a feedback of system state every T time steps. We simulated the network for 30 time steps (5 rounds of contract transitions). The satisfaction of the specification was implicitly implied by the fact that the MPC optimization problem was feasible at all times. Sample results are illustrated in (6·7).

Eq. (6.2) shows the accumulated delay for different network conditions and control architectures. The first column shows the subset of networks that experience a fully adversarial demand, e.g. the $(1,3)$ column means that $\mathcal{N}^1, \mathcal{N}^3$ experience the maximum number of incoming vehicles and $\mathcal{N}^2, \mathcal{N}^4$ experience no exogenous de-

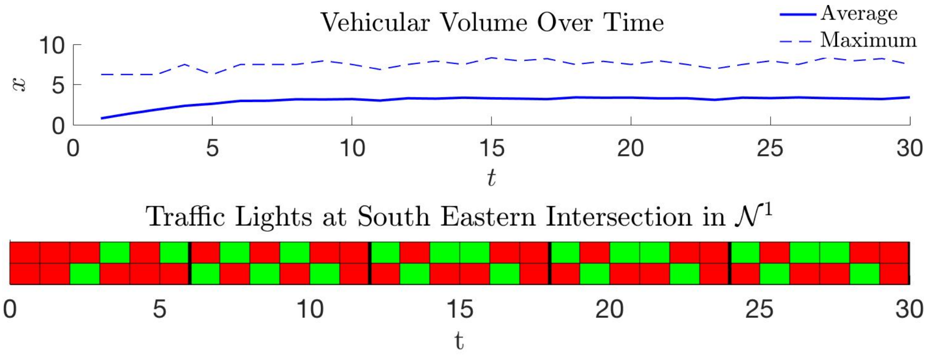


Figure 6·7: Example 16: [Top] State over Time. [Bottom] The traffic lights history for the links in South-Eastern Intersection of $\mathcal{N}^1$. The lower color is for the light corresponding to the link in North-South direction and the upper one stands for the link in East-West direction. The pedestrian liveness requirement (both lights simultaneously getting red) is satisfied in each round of contract transitions (shown by thick vertical block lines).

mand. As expected, the cumulative delay decreases when the network load decreases. The fixed controller executes a control sequence without state feedback in the interval $(kT, (k+1)T)$, but permits contract switches every T steps. The MPC controller with fixed contracts achieves similar objective values, but is not strictly better than the fixed control with dynamic contracts, suggesting that contract constraints are a major impediment for achieving a lower delay. The dynamic contracts with MPC column outperforms both other control strategies. The performance gain is greater when the exogenous demand has an asymmetric profile, when dynamic contracts assign higher priority to sub-networks experiencing higher demand.

Chapter 7

Robust Distributed Set-Invariance

Centralized control of large-scale networked systems requires all the subsystems to communicate with a central coordinator, an entity which has to promptly compute control decisions for all subsystems, making centralized control impractical. Distributed control policies - where the computation and communication loads of subsystems are limited - are preferred in practice.

In this chapter, we consider a linear system subject to additive disturbances. Polytopic set-invariance is the main objective. It is well-known that all invariance-inducing controllers may not be described using a finite number of parameters [Blanchini, 1999]. We use the framework in [Raković et al., 2007] to characterize convex sets of parameters guaranteeing set-invariance. We propose a method to impose structural constraints on the parameters. Unlike the traditional approaches discussed earlier, we require that subsystems act as relay nodes while passing information in the network. The delay of such relaying processes is taken into account in the design of the controller. In this chapter, we establish the following two main results:

- Given a directed communication graph describing the structural constraints of the network, *our method designs control policies using linear programming.* The number of constraints and variables scale polynomially with the problem size.
- When structural constraints are not given, we find a minimal communication

graph - in the sense that a weighted sum of (one-way) communication links is minimized - for which a distributed invariance-inducing control policy exists. The problem can be both solved exactly using a mixed-integer linear program or approximately solved using linear-programming relaxations.

7.1 Problem Formulation

A networked control system $\mathcal{S}$ is defined as a set of interconnected subsystems. The discrete-time evolution of $s \in \mathcal{S}$ is given as:

$$x_s[t+1] = A_s x_s[t] + B_s u_s[t] + w_s[t] + \sum_{s' \in \mathcal{S}, s' \neq s} \zeta_{s's}[t], \tag{7.1a}$$

$$\zeta_{s's}[t] = A_{s's} x_{s'}[t] + B_{s's} u_{s'}[t], \tag{7.1b}$$

where $x_s[t] \in \mathbb{R}^{n_s}$, $u_s[t] \in \mathbb{R}^{m_s}$, $w_s[t] \in \mathbb{R}^{n_s}$, are the state, control and additive disturbance of system s, respectively, and $\zeta_{s's}[t]$ is the dynamical influence of s' on s at time $t \in \mathbb{N}$. Matrices $A_s \in \mathbb{R}^{n_s \times n_s}$, $B_s \in \mathbb{R}^{n_s \times m_s}$ are constant and correspond to the internal dynamics of s, while $A_{s's} \in \mathbb{R}^{n_s \times n_{s'}}$, $B_{s's} \in \mathbb{R}^{n_s \times m_{s'}}$ are constant matrices characterizing the influence of s' on s. Given a particular ordering of the subsystems in $\mathcal{S}$ as $(s_1, s_2, \cdots, s_N)$, where $N = |\mathcal{S}|$, the states, controls and disturbances of $\mathcal{S}$ are denoted by x, u, and w, respectively, where:

$$x = \begin{pmatrix} x_{s_1} \\ \vdots \\ x_{s_N} \end{pmatrix}, u = \begin{pmatrix} u_{s_1} \\ \vdots \\ u_{s_N} \end{pmatrix}, w = \begin{pmatrix} w_{s_1} \\ \vdots \\ w_{s_N} \end{pmatrix}, \tag{7.2}$$

We have $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $w \in \mathbb{R}^n$, where

$$n = \sum_{s \in \mathcal{S}} n_s, m = \sum_{s \in \mathcal{S}} m_s. \tag{7.3}$$

The evolution of $\mathcal{S}$ is written in the following compact form:

$$x[t+1] = Ax[t] + Bu[t] + w[t], \tag{7.4}$$

where $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m}$ are unambiguously constructed from (7.1) and (7.2).

Definition 34. A directed communication graph is defined as the tuple $\mathcal{G} = (\mathcal{S}, \mathcal{L})$, where $\mathcal{S}$ (the set of subsystems) is the set of vertices and $\mathcal{L} \subseteq \mathcal{S} \times \mathcal{S}$ is a set of ordered pairs. Subsystem s' is able to transmit information to subsystem s if and only if $(s', s) \in \mathcal{L}$.

Given $\mathcal{G} = (\mathcal{S}, \mathcal{L})$, we define the k^{th} power of $\mathcal{G}$ as $\mathcal{G}^k = (\mathcal{S}, \mathcal{L}^k)$, $k \in \mathbb{N}_+$, such that $(s, s') \in \mathcal{L}^k$ if and only if there exists a walk from s to s' on $\mathcal{G}$ with length less than or equal to k. Note that $G^1 = G$. As a special case for $k = 0$, we define $\mathcal{G}^0$ and $\mathcal{L}^0$ such that $(s, s) \in \mathcal{L}^0, \forall s \in \mathcal{S}$ (self-loops). In other words, every subsystem has access to its own information.

Assumption 5. At time $t \in \mathbb{N}$, system s knows $x_{s'}[t-k+1]$ and $u_{s'}[t-k]$ if and only if $(s', s) \in \mathcal{L}^k$.

Assumption 5 requires that subsystems act as relay nodes while passing information in the network. Each relay node induces a one time step delay. Assumption 5 is not restrictive in most applications. We require each subsystem to have some additional memory to store the history of state and controls of its own and some other subsystems. Fortunately, as it will be made clear later, only a finite (usually small) number of recent states and controls is sufficient for our purpose. In this chapter, every link beyond an immediate neighbor corresponds to one unit time delay. More complex delay behavior can be accommodated in our framework by adding virtual relaying nodes (see, e.g., [Lamperski and Lessard, 2015]).

We are given the following polytopes:

$$\mathbb{X} := \{x | H_x x \leq h_x\}, \tag{7.5a}$$

$$\mathbb{U} := \{u | H_u u \leq h_u\}, \tag{7.5b}$$

$$\mathbb{W} := \{w | H_u w \leq h_w\}, \tag{7.5c}$$

where $H_x \in \mathbb{R}^{q_x \times n}, H_u \in \mathbb{R}^{q_u \times m}, H_w \in \mathbb{R}^{q_w \times n}$, and $h_x \in \mathbb{R}_+^{q_x}, h_u \in \mathbb{R}_+^{q_u}, h_w \in \mathbb{R}_+^{q_w}$. Note that we assume $\mathbb{X}, \mathbb{U}$, and $\mathbb{W}$ contain the origin.

Definition 35 (Centralized Policy). A centralized control policy is defined as $\mu^c : \mathbb{R}^n \to \mathbb{R}^m$, where

$$u[t] = \mu^c(x[t]). \tag{7.6}$$

Definition 36 (Distributed Policy). Given a networked control system $\mathcal{S}$ with communication graph $\mathcal{G}$, and a positive integer $K \geq 1$, a distributed control strategy of memory K is defined as a set of functions $\mu^d := \{\mu_s\}_{s \in \mathcal{S}}$ such that for all $t \geq K$:

$$u_s[t] = \mu_s\Big(\begin{array}{l} \Big\{ \{x_{s'}[t-k+1]\}_{(s',s) \in \mathcal{L}^k}, \\ \{u_{s'}[t-k]\}_{(s',s) \in \mathcal{L}^k} \Big\}_{k \in \{1, \cdots, K\}} \end{array} \Big), \tag{7.7}$$

where $\mu_s : \mathbb{R}^{\eta_s} \to \mathbb{R}^{m_s}, \eta_s = \sum_{k=1}^{K} \sum_{(s',s) \in \mathcal{L}^{k-1}} n_{s'} + \sum_{k=1}^{K} \sum_{(s',s) \in \mathcal{L}^k} m_{s'}$.

Definition 36 does not explain how to compute controls for $t < K$. We shift the start time to K and make the following assumption.

Assumption 6. System (9.1) is initialized at time $t = K$ with $x[K] = 0$, $x[k] = 0, u[k] = 0, \forall k \in [0, K-1]$.

Assuming the initial condition to be zero is restrictive but simplifies our analysis. We can drop Assumption 6 at the expense of adding an initial coordination between the subsystems. The details are explained in 7.2.3. The second part of Assumption 6 is not restrictive as we can always shift the start of time to K and assign arbitrary values to the past.

Definition 37 (Correctness). Given a networked control system $\mathcal{S}$ as (7.1), (9.1), polytypic sets $\mathbb{X}$, $\mathbb{U}$, $\mathbb{W}$ as (9.9), a communication graph $\mathcal{G}$, and a positive integer K, a (distributed) control μ (of memory K) is *correct* if for all allowable sequences $w[K], w[K+1], \cdots, w[t] \in \mathbb{W}, \forall t \geq K$, we have $x[t] \in \mathbb{X}$ and $u[t] \in \mathbb{U}, \forall t \in \mathbb{N}$.

Definition 38 (Margin of Correctness). Given a correct control policy μ, the margin of correctness $\rho^* \in [0, 1]$ is defined as the maximum value of ρ for which μ remains correct when $\mathbb{X} \leftarrow (1-\rho)\mathbb{X}$ and $\mathbb{U} \leftarrow (1-\rho)\mathbb{U}$.

The margin of correctness has a straightforward interpretation. If $\rho^* = 0$, it implies that correctness is lost if $\mathbb{X}$ or $\mathbb{U}$ are shrunk around the origin. If $\rho^* = 1$, it indicates that the state and controls can be always zero, which essentially requires $\mathbb{W} = \{0\}$. More complicated definitions for margin of correctness are also possible. For instance, one may consider different scaling variables for components in $\mathbb{X}$ and $\mathbb{U}$ and define the margin as a weighted sum of them.

7.1.1 Problem Statement

We formulate two problems. In both, we are given a networked control system $\mathcal{S}$ as (7.1), (9.1), polytypic sets $\mathbb{X}$, $\mathbb{U}$, $\mathbb{W}$ as (9.9), and a positive integer K. In practice, K is a design parameter which determines the complexity of the controller. We usually start from small values of K and make it larger until feasibility/satisfactory performance is reached.

Problem 9 (Optimal Strategy Design). Given a communication graph $\mathcal{G}$, design a correct distributed control policy μ of memory K with the maximum margin of correctness ρ^*.

Problem 10 (Optimal Graph Design). Find a communication graph $\mathcal{G} = (\mathcal{S}, \mathcal{L})$ for which a correct control policy exists such that the following cost function is minimized:

$$J(\mathcal{G}) = \sum_{s \in \mathcal{S}} \sum_{s' \in \mathcal{S}} c_{s's} \mathcal{I}\left((s', s) \in \mathcal{L}\right), \tag{7.8}$$

where $c_{s's} \in \mathbb{R}_+^n$ is the cost of establishment of one-way communication link from s' to s, and $\mathcal{I}$ is the indicator function that designates 1 (respectively, 0) if its argument is true (respectively, false).

7.2 Parameterized Set-Invariance

In this section, we present the family of parameterized controllers in [Raković et al., 2007]. We do not, yet, impose structural constraints. The key idea of this chapter is outlined in Sec. 7.2.3, where we show that a memoryless piecewise affine invariance-inducing control policy can be converted to a linear controller with memory, paving the path to impose structural requirements in Sec. 7.3.

7.2.1 Convex Parameterization

Lemma 3. [Raković et al., 2007] Let $\Theta := (\theta_0, \theta_1, \cdots, \theta_{K-1})$, where $\theta_k \in \mathbb{R}^{m \times n}, k = 0, \cdots, K-1$, be a $m \times nK$ matrix of parameters such that the following condition holds:

$$A^K + A^{K-1}B\theta_0 + \cdots + AB\theta_{K-2} + B\theta_{K-1} = 0. \tag{7.9}$$

Define the following set:

$$\begin{aligned}\Omega_\Theta := \; & (A^{K-1} + A^{K-2}B\theta_0 + \cdots + B\theta_{K-2})\mathbb{W} \\ & \oplus(A^{K-2} + A^{K-3}B\theta_0 + \cdots + B\theta_{K-3})\mathbb{W} \\ & \oplus \cdots \oplus (A + B\theta_0)\mathbb{W} \oplus \mathbb{W}\end{aligned} \tag{7.10}$$

Then there exists $\mu : \mathbb{R}^n \to \mathbb{R}^m$ such that

$$\forall x \in \Omega_\Theta, \{Ax + B\mu(x)\} \oplus \mathbb{W} \subseteq \Omega_\Theta.$$

Proof. For all $x \in \Omega_\Theta$, there exists $w^{K-1}, w^{K-2}, \cdots, w^0 \in \mathbb{W}$ such that

$$\begin{aligned}x = \; & (A^{K-1} + A^{K-2}B\theta_0 + \cdots + B\theta_{K-2})w_x^{K-1} + \\ & (A^{K-2} + A^{K-3}B\theta_0 + \cdots + B\theta_{K-3})w_x^{K-2} \\ & + \cdots + (A + B\theta_0)w_x^1 + w_x^0.\end{aligned} \tag{7.11}$$

Now let the $\mu^c(x)$ be the following control input:

$$\mu^c(x) = \theta_{K-1}w_x^{K-1} + \theta_{K-2}w_x^{K-2} + \cdots + \theta_0 w_x^0. \tag{7.12}$$

Denote the new disturbance hitting system by w^+. The subsequent state is

$$\begin{aligned}x^+ = \; & Ax + B\mu^c(x) + w^+ = \\ & (A^K + A^{K-1}B\theta_0 + \cdots + B\theta_{K-1})w_x^{K-1} \\ & +(A^{K-1} + A^{K-3}B\theta_0 + \cdots + B\theta_{K-2})w_x^{K-2} \\ & + \cdots + (A + B\theta_0)w_x^0 + w^+.\end{aligned} \tag{7.13}$$

Substituting (7.9) in (7.13) results in:

$$\begin{aligned} x^{+} = \;& (A^{K-1} + A^{K-2}B\theta_0 + \cdots + B\theta_{K-2})w_x^{K-2} \\ & +(A^{K-2} + A^{K-3}B\theta_0 + \cdots + B\theta_{K-2})w_x^{K-3} \\ & + \cdots + (A + B\theta_0)w_x^0 + w^{+}. \end{aligned} \tag{7.14}$$

A quick inspection of (7.10) and (7.14) verifies $x^{+} \in \Omega_\Theta$. □

Notice that (7.9) is not restrictive since Θ is non-empty for controllable (A, B) and K greater than its controllability index. The set Ω_Θ is a *robust control invariant* (RCI) set. Following (7.12), the set of all possible controls is

$$\Psi_\Theta := \bigoplus_{k=0}^{K-1} \theta_k \mathbb{W}. \tag{7.15}$$

In order to have a correct control policy, we require $\Omega_\Theta \subseteq \mathbb{X}$ and $\Psi_\Theta \subseteq \mathbb{U}$.

Lemma 4. [Raković et al., 2007] The set of parameters $\Theta, \alpha \in \mathbb{R}_+$, for which $\Omega_\Theta \subseteq \alpha\mathbb{X}$ and $\Psi_\Theta \subseteq \alpha\mathbb{U}$ is convex.

Proof. The proof is based on an extension of Farkas's Lemma: Given sets $\mathbb{S} = \{s \in \mathbb{R}^n | H_s s \leq h_s\}$, and $Y \subset \{y \in \mathbb{R}^q | H_y y \leq h_y\}$, and matrices $L_i \in \mathbb{R}^{q \times n}$, $i = 1, \cdots, \kappa$, then $\bigoplus_{i=1}^{\kappa} L_i \mathbb{S} \subseteq \alpha\mathbb{Y}$, is equivalent to the following set of constraints:

$$Z_i H_s = H_y L_i, i = 1, \cdots, \kappa, \sum_{i=1}^{\kappa} Z_i h_s \leq \alpha h_y, \tag{7.16}$$

where $Z_i \in \mathbb{R}^{q \times n}$, $i = 1, \cdots, \kappa$, are appropriately sized matrices with non-negative entries. Convexity follows from the linearity of the constraints. □

Remark 4. The family of RCI sets introduced in [Raković et al., 2007] is not necessarily equivalent to the set of all RCI sets. In particular, there is no guarantee that one can find the maximal RCI set using this approach, which is not surprising as the problem of finding the maximal RCI set is not decidable, in general [Blanchini, 1999]. Nevertheless, the set of RCI sets in [Raković et al., 2007] is quite rich as they are generated using piecewise affine feedback laws rather than the much more limited class of linear control laws.

7.2.2 Centralized Policy

Given Θ, the map from x to u requires finding values for $w_x^{K-1}, \cdots, w_x^0$, subject to (7.11). This is accomplished by solving a linear/quadratic program. A centralized policy can be constructed as:

$$u[t] = \mu^c(x[t]) = \begin{array}{ll} \underset{u}{\arg\min} & \pi(u) \\ \text{s.t.} & (7.11), (7.12), w_x^k \in \mathbb{W}, \\ & k = 0, \cdots, K-1, \end{array} \tag{7.17}$$

where $\pi : \mathbb{R}^m \to \mathbb{R}$ is a user-defined convex linear/quadratic cost function. It is well-known that μ^c becomes a piecewise affine function.

Remark 5. An alternative way to make a set-invariance control policy distributed is using the state-of-the-art distributed convex optimization techniques to solve (7.17), such as alternating direction method of multipliers (ADMM) [Boyd et al., 2011]. However, a substantial communication and computation effort is required to perform the iterations in ADMM [Summers and Lygeros, 2012].

7.2.3 Linear Delay Policy

The following result states that any memoryless piecewise affine policy obtained from (7.17) can be converted into a linear policy with memory K.

Theorem 12. Let Θ such that $\Omega_\Theta \subseteq \mathbb{X}$ and $\Psi_\Theta \subseteq \mathbb{U}$. Then a control policy in which control decisions for $t \geq K$ are given as:

$$\begin{aligned} u[t] = \;& \theta_{K-1} w[t-K] + \theta_{K-2} w[t-K+1] \\ & + \cdots + \theta_0 w[t-1] \end{aligned} \tag{7.18}$$

is correct if the following condition holds:

$$\begin{aligned} x[K] = \;& (A^{K-1} + A^{K-2} B\theta_0 + \cdots + B\theta_{K-2}) w[0] + \\ & (A^{K-2} + A^{K-3} B\theta_0 + \cdots + B\theta_{K-3}) w[1] \\ & + \cdots + (A + B\theta_0) w[K-2] + w[K-1]. \end{aligned} \tag{7.19}$$

Proof. We prove correctness by showing that

$$\begin{aligned} & x[t] = (A^{K-1} + A^{K-2} B\theta_0 + \cdots + B\theta_{K-2}) w[t-K] \\ & +(A^{K-2} + A^{K-3} B\theta_0 + \cdots + B\theta_{K-3}) w[t-K+1] \\ & + \cdots + (A + B\theta_0) w[t-2] + w[t-1]. \end{aligned} \tag{7.20}$$

for all $t \geq K$. We prove by induction. For $t = K$, the statement is assumed true as (7.19). We prove the inductive step using (7.9) to arrive in:

$$\begin{aligned} & x[t+1] = Ax[t] + Bu[t] + w[t] \\ & = (A^{K-1} + A^{K-2} B\theta_0 + \cdots + B\theta_{K-2}) w[t-K+1] \\ & +(A^{K-2} + A^{K-3} B\theta_0 + \cdots + B\theta_{K-3}) w[t-K+2] \\ & + \cdots + (A + B\theta_0) w[t-1] + w[t]. \end{aligned}$$

It follows from (7.10) and (7.12) that $x[t] \in \Omega_\Theta, u[t] \in \Psi_\Theta, \forall t \geq K$, and the proof is complete. □

Eq. (7.18) is a linear policy based on disturbances. However, disturbances are not assumed to be directly measurable. Using (9.1), we can replace disturbances by state and controls to obtain a more useful form of (7.18):

$$\begin{aligned} u[t] = \ & (-\theta_{K-1}A)x[t-K] \\ & +(\theta_{K-1} - \theta_{K-2}A)x[t-K+1] \\ & +(\theta_{K-2} - \theta_{K-3}A)x[t-K+2] \\ & + \cdots + (\theta_1 - \theta_0 A)x[t-1] + \theta_0 x[t] \\ & -\big(\theta_{K-1}Bu[t-K] + \theta_{K-2}Bu[t-K+1] \\ & + \cdots + \theta_0 Bu[t-1]\big). \end{aligned} \tag{7.21}$$

Since all zeros is a trivial solution to (7.19), a simple way to make (7.19) true is holding Assumption 6. For any initial condition $x[K] \in \Omega_\Theta$, we can find hypothetical values for $w[0], w[1], \cdots, w[K-1]$ such that (7.19) holds by solving a linear program. However, solving such a linear program may require a central entity. We may use distributed linear program solvers (see Remark 5) to accomplish this task. Therefore, Assumption 6 is relaxable given arbitrary initial conditions, as long as they lie in the RCI set. Note that if the initial condition is outside of the (maximal) RCI set, satisfying the set-invariance objective is impossible.

7.3 Control with Structural Constraints

Here we provide the solution to Problem 1. We impose structural requirements on (7.21) based on Assumption 5. We define the following sets of matrices:

$$\begin{aligned} \mathbb{S}^x\left((\mathcal{S}, \mathcal{L})\right) := \ & \Big\{G \in \mathbb{R}^{m \times n} \Big| u_{[i]} \in s, x_{[j]} \in s', \\ & (s', s) \notin \mathcal{L} \Rightarrow G_{[i,j]} = 0\Big\}, \end{aligned} \tag{7.22a}$$

$$\begin{aligned} \mathbb{S}^u\left((\mathcal{S}, \mathcal{L})\right) := \ & \Big\{G \in \mathbb{R}^{m \times n} \Big| u_{[i]} \in s, u_{[j]} \in s', \\ & (s', s) \notin \mathcal{L} \Rightarrow G_{[i,j]} = 0\Big\}, \end{aligned} \tag{7.22b}$$

where $x_{[i]} \in s$ ($u_{[i]} \in s$) is interpreted as whether i'th component of x (u) belongs to subsystem s. Sets in (7.22) are convex. The coefficients that relate a component of

$u[t]$ in (7.21) to a component of $x[t-k+1], u[t-K], k = 1, \cdots, K$, has to be zero if it violates Assumption 5, which is formally stated as follows:

$$\left\{\begin{array}{l} \theta_{K-1}A \in \mathbb{S}^x(\mathcal{G}^{K+1}) \\ \theta_{K-1} - \theta_{K-2}A \in \mathbb{S}^x(\mathcal{G}^K) \\ \vdots \\ \theta_1 - \theta_0 A \in \mathbb{S}^x(\mathcal{G}^2) \\ \theta_0 \in \mathbb{S}^x(\mathcal{G}^1) \end{array}\right., \left\{\begin{array}{l} \theta_{K-1}B \in \mathbb{S}^u(\mathcal{G}^K) \\ \vdots \\ \theta_0 B \in \mathbb{S}^u(\mathcal{G}^1) \end{array}\right. \tag{7.23}$$

Finally, the solution to Problem 1 is found by solving the following linear program:

$$\begin{array}{rll} \{\rho^*, \Theta^*\} = & \underset{\Theta,\rho}{\arg\max} & \rho \\ & \text{subject to} & (7.9), (7.23), \\ & & \Omega_\Theta \subseteq (1-\rho)\mathbb{X}, \\ & & \Psi_\Theta \subseteq (1-\rho)\mathbb{U}. \end{array} \tag{7.24}$$

Complexity

The number of variables and constraints in (7.24) scales linearly with respect to K, n, m, and the number of rows in $\mathbb{X}, \mathbb{U}, \mathbb{W}$. In practice, representation complexity of sets in (9.9) scale polynomially in n and m, while the exact degree of growth depends on the application. Thus, taking the complexity of the interior-point linear programming methods into account, the overall complexity of our solution to Problem 1 increases polynomially with respect to the problem size.

7.4 Structure Design

Here we provide the solution to Problem 2.

7.4.1 Binary encoding

We need to make binary decisions on whether s' is connected to s on $\mathcal{G}$. This task is captured by introducing binary $N(N-1)$ binary variables $b_{s's} \in \{0,1\}, s, s' \in \mathcal{S}$. Note that $b_{ss} = 1, \forall s \in \mathcal{S}$. We define the adjacency matrix of $\mathcal{G}$ as $\mathcal{B}(\mathcal{G}) \in \mathbb{B}^{N\times N}$ such that $\mathcal{B}(\mathcal{G})_{[s's]} := b_{s's}$. The following property follows from the basic properties of powers of adjacency matrix in graph theory:

$$\mathcal{B}(\mathcal{G}^k) = \sum_{p=1}^{k} \mathcal{B}(\mathcal{G}^p), \tag{7.25}$$

where both summation and multiplication are defined in a Boolean sense, i.e., for $b_1, b_2 \in \mathbb{B}$ we have $b_1 b_2 = b_1 \wedge b_2$ and $b_1 + b_2 = b_1 \vee b_2$. (e.g., $1 + 1 = 1$).

Given $\mathcal{B} \in \mathbb{B}^{N\times N}$, we define $\mathcal{B}^x \in \mathbb{B}^{m\times n}$ and $\mathcal{B}^u \in \mathbb{B}^{m\times m}$ such that

$$\mathcal{B}^x((\mathcal{S},\mathcal{L}))_{[i,j]} = b_{s's}, u_{[i]} \in s_i, x_{[j]} \in s_{[j]}, \tag{7.26a}$$

$$\mathcal{B}^u((\mathcal{S},\mathcal{L}))_{[i,j]} = b_{s's}, u_{[i]} \in s_i, u_{[j]} \in s_{[j]}. \tag{7.26b}$$

Given a matrix $C \in \mathbb{R}^{m\times n}$, and $k \in \mathbb{N}_+$ the following relation holds:

$$C \in \mathbb{S}^z(\mathcal{G}^k) \Leftrightarrow -M\mathcal{B}^z(\mathcal{G}) \leq C \leq M\mathcal{B}^z(\mathcal{G}), \tag{7.27}$$

where $z = x, u$, and M is a sufficiently large positive number that is greater than $\max_{i,j} |C_{[i,j]]}|$. The constraints in (7.25), (7.27), are mixed binary-linear constraints. We need only to declare the entries in $\mathcal{B}(\mathcal{G})$ as binaries - there are $N(N-1)$ of them - and all the other relations in (7.25) are captured using continuous auxiliary variables declared over $[0,1]$ - which constraints enforce them take values from $\{0,1\}$. Encoding

Boolean functions using mixed binary-linear constraints is a standard procedure (see, e.g., [Bemporad and Morari, 1999]) and the details are not presented here.

7.4.2 Graph Optimization

Finally, we find the optimal communication graph $\mathcal{G}^*$ - the solution to Problem 2 - as the following mixed-integer linear program (MILP):

$$\begin{array}{rll} \{\Theta^*, \mathcal{B}(\mathcal{G}^*)\} = & \underset{\Theta, b_{s's}, s, s' \in \mathcal{S}}{\arg\min} & \sum_{s,s' \in \mathcal{S}} b_{s's} \\ & \text{subject to} & (7.9), (7.23), (7.27), \\ & & \Omega_\Theta \subseteq \mathbb{X}, \Psi_\Theta \subseteq \mathbb{U}. \end{array} \tag{7.28}$$

Note that (7.28) provides both a communication graph and a corresponding RCI set and distributed control policy parameterized by Θ^*. Note that we can combine Problem 1 and Problem 2 by adding an additional term to the cost function in (7.28) to promote greater margin of correctness. The trade-off between sparser graph and greater margin of correctness can be controlled by designating weights to the corresponding terms.

Complexity

Unlike (7.24), solving (7.28) is NP-hard. MILP solvers use branch and bound techniques to explore optimal solutions by solving linear-program relaxations of the original problem. In order to find suboptimal but (arbitrary) faster solutions, a simple approach is terminating the MILP solver early - after it has an incumbent feasible solution. (see Fig. 7·4 in the examples).

7.5 Examples

We have developed a python script that solves Problem 1 and Problem 2 given the system, specification and relevant parameters by the user. This script, as well as codes for the examples below, are publicly available in `github.com/sadraddini/distinct`.

7.5.1 Coupled Double Integrators

We consider $N = 5$ double-integrators with state and control couplings. Fo all $s, s' \in \mathcal{S}$, we assign the following values to (7.1):

$$A_s = \begin{pmatrix} 1+\epsilon & 1 \\ -\epsilon & 1+\epsilon \end{pmatrix}, A_{s's} = \begin{pmatrix} \epsilon & -\epsilon \\ -\epsilon & \epsilon \end{pmatrix},$$

$$B_s = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, B_{s's} = \begin{pmatrix} -\epsilon \\ \epsilon \end{pmatrix},$$

where ϵ is a constant characterizing the degree of coupling. We explore the behavior of solutions versus multiple values of $\epsilon > 0$. For any $\epsilon > 0$, at least one of the eigenvalues of A lies out of the unit circle. Thus A is unstable. We let $\mathbb{X} = \mathcal{B}_\infty^{10}, \mathbb{U} = 2\mathcal{B}_\infty^5, \mathbb{W} = \eta\mathcal{B}_\infty^{10}$, where η is also a constant we vary in this example.

Structured Control

We solve Problem 1. We are given a communication graph that is circular, as illustrated in Fig. 7·1. We consider both the directed and the undirected case. The results for various values of K, η, ϵ, are shown in Table 7.1. As expected, the margins are smaller when coupling and disturbances are greater, and communications are directed. Also, higher values of K usually correspond to better performance. For $K < 6$, we could not find a solution for the directed graph. Projections of the RCI set

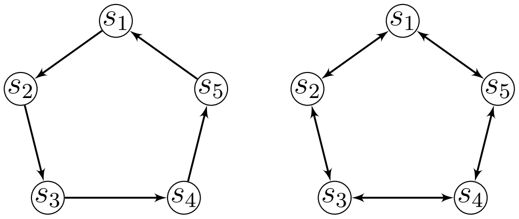


Figure 7·1: Circular graphs: [Left]: directed [Right]: undirected

Table 7.1: Margins of Correctness for Graphs in Fig. 7·1

K	η	ϵ	ρ^*	
			Directed	Undirected
6	0.05	0.05	0.27	0.75
6	0.1	0.1	Infeasible	0.33
6	0.1	0.01	0.02	0.58
4	0.05	0.01	Infeasible	0.79
4	0.05	0.05	Infeasible	0.75
6	0.05	0.01	0.51	0.79

and sample trajectories are illustrated in Fig. 7·2. It is observed that the undirected circular communication graph is able to keep the state closer to zero, while the RCI set and trajectories of the directed graph get closer to the boundaries of $\mathbb{X}$. All the computations in Table 7.1 were performed using Gurobi linear program solver on a dual core 3GHz MacBook Pro. The computation times were all less than a second.

Graph Design

We solve Problem 2. We let $c_{s's} = 1, \forall s, s' \in \mathcal{S}, s \neq s'$. Various optimal graphs corresponding to different values are shown in Fig. 7·3. We often obtained graphs that were strongly connected. However, in case the couplings are sufficiently weak, fully decentralized solutions were found, as shown in the null graph in Fig. 7·3 (f). The computations were performed using Gurobi MILP solver on a 3GHz dual core MacBook Pro. As discussed in Sec. 7.4.2, MILP solvers explore solutions using branch and bound techniques. Two instances of the best incumbent solution versus time is

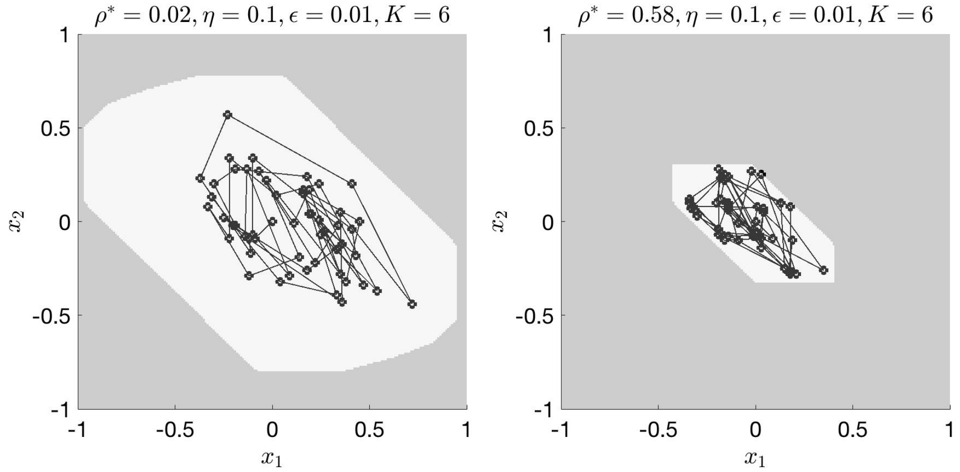


Figure 7·2: Projection of the RCI set on the state-space of s_1 and a sample trajectory of 60 time steps. RCI sets correspond to [Left]: directed [Right]: undirected graphs in Fig. 7·1.

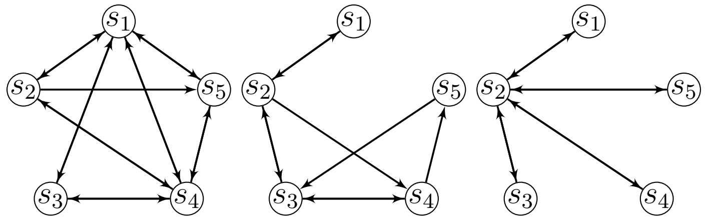
shown in Fig. 7·4. We can obtain suboptimal solutions by early manual termination.

7.5.2 Platooning

We adopt a simplified version of the model in [Sadraddini et al., 2017]. A platoon is a string of N_p autonomous vehicles following a leader l. We have $N = N_p$ subsystems. System (9.1) and sets (9.9) are constructed from what is described below. The state of each follower vehicle $s \in \mathcal{S}$ is $x_s = (d_s, v_s)$, where d_s represents the distance from the preceding vehicle and v_s is its velocity in the leader's frame. The evolution is given by:

$$\begin{aligned} d_s[t+1] &= d_s[t] - v_s[t] + v_{s'}[t] + \delta_s^x[t], \\ v_s[t+1] &= v_s[t] + u_s[t] + \delta_s^v[t] + \delta_l^v[t], \end{aligned} \tag{7.29}$$

where $s' \in \mathcal{S} \cup \{l\}$ is the preceding vehicle, $\delta_s^v[t] \in [-\varepsilon, \varepsilon], \delta_s^x[t] \in \frac{1}{10}[-\varepsilon, \varepsilon]$, are the disturbances hitting a follower vehicle, and $\delta_l^v[t] \in [-\varepsilon, \varepsilon]$ is the disturbance hitting the leader, which makes the frame non-inertial. We vary ε in this example. Note that (7.29) is a quite adversarial model since we consider independent disturbances



(a) $K = 3$

$\eta = 0.2, \epsilon = 0.06$

$J(\mathcal{G}^*) = 14$, CT=82s

(b) $K = 6$

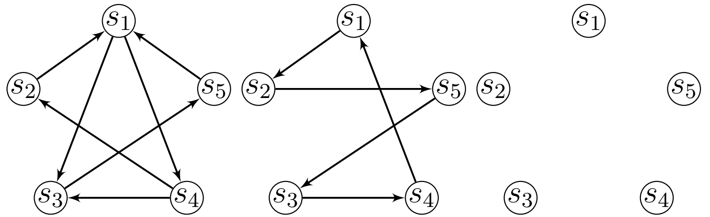
$\eta = 0.2, \epsilon = 0.1$

$J(\mathcal{G}^*) = 9$, CT=321s

(c) $K = 4$

$\eta = 0.2, \epsilon = 0.05$

$J(\mathcal{G}^*) = 8$, CT=213s



(d) $K = 4$

$\eta = 0.1, \epsilon = 0.1$

$J(\mathcal{G}^*) = 7$, CT=721s

(e) $K = 4$

$\eta = 0.1, \epsilon = 0.02$

$J(\mathcal{G}^*) = 5$, CT=226s

(f) $K = 6$

$\eta = 0.1, \epsilon = 0.01$

$J(\mathcal{G}^*) = 0$, CT=1s

Figure 7·3: Optimal communication graphs for different values of K (complexity of the controller), η (maximum magnitude of the allowed disturbances) and ϵ (the degree of dynamical couplings). "CT" stands for computation time.

affecting the distance evolution. The objective is to avoid rear-end avoid collisions for all times by writing $d_s[t] \geq -0.5, \forall t \geq 0$, (a distance offset is performed in a way that $d_s < -0.5$ implies collision), and $\sum_{\mathcal{S}} d_s[t] \leq \frac{1}{2} N_p, \forall t \in \mathbb{N}$ - the length of the platoon is always bounded. We also have bounded controls: $u_s[t] \in [-1, 1], \forall s \in \mathcal{S}, \forall t \in \mathbb{N}$. System (7.29) and specification are put into the form (9.1), and (9.9), respectively. Note that the number of rows in $\mathbb{W}$ scale quadratically with the platoon size - $\mathbb{W}$ has a more complicated shape than a box [Sadraddini et al., 2017].

Structured Control

We solve Problem 1. Consider a communication graph that every vehicle sends information to its follower (see Fig. 7·5 (b)). We set $\epsilon = 0.05$. We observe that the minimum K such that a feasible solution is found is $N_p + 1$. The results are shown

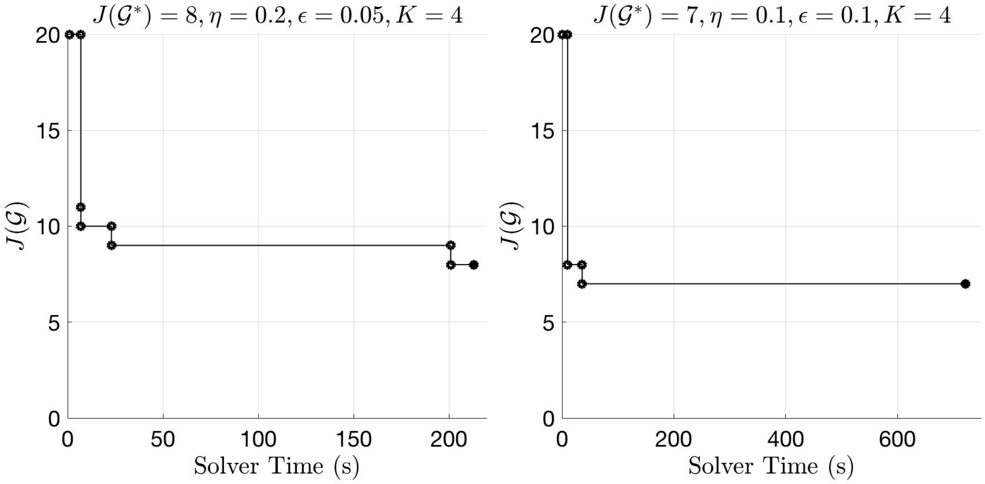


Figure 7·4: The costs of incumbent feasible solutions versus time.

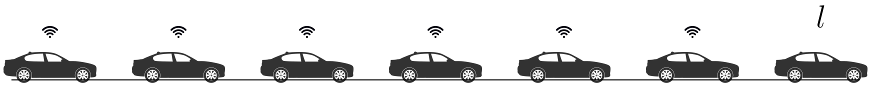
Table 7.2: Margins of Correctness for Predecessor Following

N_p	3	4	5	6	8	10	12	15
ρ^*	0.727	0.726	0.723	0.721	0.716	0.710	0.704	0.697
CT(s)	–	–	0.05	0.1	0.7	3	13	133

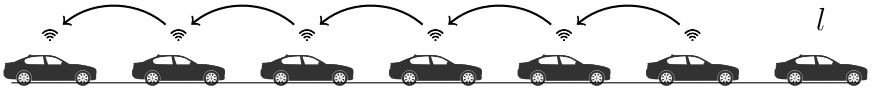
in Table. 7.2 for $K = N_p + 1$. It is observed that the margin of correctness gradually decreases with the platoon size, highlighting the fundamental limits of predecessor following [Sabău et al., 2017].

Graph Design

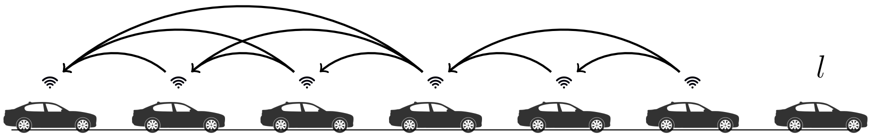
We solve Problem 2. We let $N = 6$ and $c_{s_i,s_j} = (i-j)^2$ to penalize longer communication links. Some particular optimal graphs are shown in Fig. 7·5. It is observed that for small disturbances, no communication is needed at all, but in order to attenuate heavier disturbances without violating collision and platoon length constraints, more communication links are required.



(a) $\varepsilon = 0.15, K = 8, J(\mathcal{G}^*) = 0$, CT=7s



(b) $\varepsilon = 0.1800, K = 8, J(\mathcal{G}^*) = 5$, CT=81s



(c) $\varepsilon = 0.1836, K = 8, J(\mathcal{G}^*) = 26$, CT=32s

Figure 7·5: Optimal communication graphs for platooning. The leader is considered as an adversary with bounded acceleration range.

Part III

Learning-based Formal Synthesis

Chapter 8

Formal Methods for Adaptive Control

In this chapter, we introduce a formal perspective on adaptive control. One particular limitation of current adaptive (self-learning) control methods is handling systems that involve discontinuities. Most adaptive control techniques rely on the continuity of the model and its parameterization. In many realistic models, state, control or parameters take values from both continuous and discrete domains. Within methods that do not entirely depend on the continuity of the model, a promising direction is using multiple models/controllers [Morse, 1996, Narendra and Xiang, 2000, Anderson et al., 2001, Hespanha et al., 2003], where the objective is to achieve stability via designing a switching law to coordinate the controllers. Model reference adaptive control (MRAC) of specific forms of scalar input piecewise affine systems were studied in [di Bernardo et al., 2013, di Bernardo et al., 2016]. However, it is still not clear how to deal with general discrete or hybrid systems.

We consider discrete-time systems with constant but initially unknown parameters. We describe system specifications using linear temporal logic (LTL). As in any other adaptive control technique, we require an online parameter estimator. Our parameter estimator maps the history of the evolution of the system to the set of "all possible" parameters, which contains the actual parameters. We embed the parameterized system in a (non-deterministic) parametric transition system (PTS), from which we construct a (non-deterministic) adaptive transition system (ATS) that contains all the possible combinations of transitions with the what is returned by the parameter estimator. The main results and contributions of this chapter are as follows:

- For finite systems, the LTL adaptive control problem reduces to a Rabin game [Grädel et al., 2002] on the product of the finite ATS and the Rabin automaton corresponding to the LTL specification. The method is correct by design and it is complete, i.e. it finds a solution if one exists;
- For infinite systems, we construct finite quotient ATSs by partitioning the state and the parameter space and quantizing the control space. Once an adaptive control strategy is found for the quotient, it is guaranteed that it will also ensure the satisfaction of the LTL formula for the original infinite system. The method may be conservative.

This chapter is organized as follows. The problem is formulated in Sec. 8.1. We define PTSs in Sec. 8.2. Technical details for the solutions for finite and infinite systems are explained in Sec. 8.3 and 8.4, respectively. Two case studies are presented in Sec. 8.5.

8.0.1 Transition Systems

Definition 39. A transition system is defined as the tuple $\mathcal{T} = (X, U, \beta, \Pi, O)$, where:

- X is a (possibly infinite) set of states;
- U is a (possibly infinite) set of control inputs;
- β is a transition function $\beta : X \times U \to 2^X$;

- $\Pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ is a finite set of atomic propositions;
- $O : X \to 2^{\Pi}$ is an observation map.

We assume that $\mathcal{T}$ is *non-blocking* in the sense that $|\beta(x,u)| \neq 0$ for all $x \in X, u \in U$. [1] A transition system $\mathcal{T}$ is *deterministic* if $|\delta(x,u)| = 1, \forall x \in X, \forall u \in U$, and is *finite* if X and U are finite sets. A trajectory of $\mathcal{T}$ is an infinite sequence of visited states $x_0 x_1 x_2 \cdots$. The infinite word produced by such a trajectory is $O(x_0)O(x_1)O(x_2)\cdots$. Note that the alphabet here is 2^{Π}. The set of all infinite words that can be generated by $\mathcal{T}$ is a subset of $(2^{\Pi})^{\omega}$.

Definition 40. A control strategy Λ is a function $\Lambda : X^* \times U^* \to U$ that maps the history of visited states and applied controls to an admissible control input, where $u_k = \Lambda(x_0 \cdots, x_k, u_0 \cdots, u_{k-1}), \forall k \in \mathbb{N}$.

Definition 41. Given a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$, a control strategy Λ and a set of initial states $X_0 \in X$, we define:

$$L(\mathcal{T}, \Lambda, X_0) := \Big\{ \begin{array}{l} O(x_0)O(x_1)\cdots \in (2^{\Pi})^{\omega} \Big| \\ x_0 \in X_0, x_{k+1} \in \beta(x_k, u_k), k \in \mathbb{N} \Big\}, \end{array}$$

where $u_k = \Lambda(x_0 \cdots, x_k, u_0 \cdots, u_{k-1})$.

8.0.2 Quotient Transition System

Consider a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$. A (finite) set $Q \subset 2^X$ is a (finite) partition for X if 1) $\emptyset \notin Q$, 2) $\bigcup_{q \in Q} q = X$, and 3) $q \cap q' = \emptyset, \forall q, q' \in Q, q \neq q'$. A partition Q is *observation preserving* if for all $q \in Q$, we have $O(x) = O(x'), \forall x, x' \in q$.

---

[1] If $\mathcal{T}$ is blocking, we can make it non-blocking by adding an additional state x^{sink} such that for all $x \in X, u \in U, |\beta(x,u)| = 0$, we have $x^{sink} = \beta(x,u)$. Also, we add transitions $x^{sink} = \beta(x^{sink}, u), \forall u \in U$. In order to prevent blocking, we find a control strategy such that x^{sink} is not reachable.

Definition 42. Given a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$ and an observation preserving partition Q for X, the *quotient transition system* is defined as the tuple $\mathcal{T}_Q = (Q, U, \beta_Q, \Pi, O_Q)$ such that:

- for all $q \in Q$, we have $q' \in \beta_Q(q, u)$ if and only if $\exists x \in q$, $\exists x' \in q'$ such that $x' \in \beta(x, u)$;
- for all $q \in Q$, we have $O_Q(q) = O(x)$ for any $x \in q$.

Given a control strategy for the quotient $\Lambda_Q : Q^* \times U^* \to U$, and a set of initial conditions Q_0, we construct $\Lambda^{(Q)} : X^* \to U$ such that $\Lambda^{(Q)}(x_0 \cdots x_k) = \Lambda_Q(q_0 \cdots q_k)$, $x_i \in q_i$, $0 \leq i \leq k$, $k \in \mathbb{N}$, and $X_0^{(Q)} = \{x_0 | x_0 \in q_0, q_0 \in Q_0\}$. It is easy to show that $L(\mathcal{T}, \Lambda^{(Q)}, X_0^{(Q)}) \subseteq L(\mathcal{T}_Q, \Lambda_Q, Q_0)$, which stems from the fact that $\mathcal{T}_Q$ simulates $\mathcal{T}$. We refer to $L(\mathcal{T}_Q, \Lambda_Q, Q_0) \setminus L(\mathcal{T}, \Lambda^{(Q)}, X_0^{(Q)})$ as the set of spurious infinite words (SIW). In order to have $L(\mathcal{T}, \Lambda^{(Q)}, X_0^{(Q)}) = L(\mathcal{T}_Q, \Lambda_Q, Q_0)$ (empty SIW), a sufficient condition is that $\mathcal{T}_Q$ and $\mathcal{T}$ are *bisimilar* [Belta et al., 2017]. For infinite X, there is no general guarantee that a finite Q exists such that $\mathcal{T}_Q$ is bisimilar to $\mathcal{T}$. In order to "shrink" SIW, Q is refined. At the most extreme case, SIW remains nonempty unless $Q = X$. Further details on simulation and bisimulation relations are not required for this chapter and the interested reader is referred to the related works in the literature, such as [Fernandez and Mounier, 1991, Tabuada, 2008, Belta et al., 2017].

8.0.3 LTL Control

Given a finite transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$ and an LTL formula φ over Π, we are interested in finding a control strategy Λ and the largest set of initial conditions $X_0^{\max}$ such that $L(\mathcal{T}, \Lambda, X_0^{\max}) \subseteq L(\varphi)$. In other words, we require φ to be satisfied for all trajectories that are allowed by the non-determinism in $\mathcal{T}$.

Definition 43. Given a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$ and a DRA $\mathcal{R}_\varphi = (S, s^0, \mathcal{A}, \alpha, \Omega)$ corresponding to LTL formula φ, the product automaton $\mathcal{T}^P_\varphi = \mathcal{T} \otimes \mathcal{R}_\varphi$ is defined as the tuple $\left(X^P, X^{P,0}, U, \beta^P, \Omega^P\right)$, where:

- $X^P = X \times S$ is the set of product states;
- $X^{P,0} = \{(x, s^0) | x \in X\}$ is the set of initial product states;
- U is the set of control inputs;
- $\beta^P : X^P \times U \to 2^{X^P}$ is the product transition function, where $x^{P'} \in \delta(x^P, u)$, $x^P = (x, s), x^{P'} = (x', s')$, if and only if $x' \in \beta(x, u)$ and $s' = \alpha(s, O(x))$.
- $\Omega^P = \left\{(F_1^P, I_1^P), \cdots, (F_r^P, I_r^P)\right\}$ is a finite set of pairs of sets of states, where $F_i^P = \{(x, s) | x \in X, s \in F_i\}, I_i^P = \{(x, s) | x \in X, s \in I_i\}, i = 1, \cdots, r$.

The product automaton $\mathcal{T}^P_\varphi$ is a (non-deterministic) automaton (with control inputs) capturing both the transitions in $\mathcal{T}$ and the acceptance condition of φ. The solution to the problem of finding a control strategy to satisfy φ is accomplished by solving the Rabin game on the product automaton. The details are not presented here but can be found in [Chatterjee and Henzinger, 2012]. It can be shown that the control strategy is memoryless on the product automaton in the form $\Lambda : X \times S \to U$. In other words, the history of the system is incorporated into the state of the Rabin automaton. The largest set of admissible initial conditions $X_0^{\max}$ corresponds to the winning region of the Rabin game.

If the transition system $\mathcal{T}$ is infinite, a finite quotient is constructed. If U is infinite, it can be quantized to obtain a finite set [2]. It is straightforward to show

---

[2] An alternative (better) approach was proposed in [Yordanov et al., 2012] for piecewise affine systems, where the authors computed a finite set of sets of control inputs that enabled transitions with minimal non-determinism in the quotient system.

that if a control strategy satisfying φ exists for the finite quotient, it also satisfies φ if implemented on the original system. However, unless the quotient and the original transition system are bisimilar, the non-existence of a control strategy for the quotient does not indicate that one does not exist for the original system. Hence the approach of using finite quotients may be conservative [Tabuada, 2008, Belta et al., 2017].

8.1 Problem Formulation and approach

We are interested in discrete-time systems of the following form:

$$\begin{aligned} x^{+} = & \; F(x, u, \theta, d), \\ y_i = & \; \mu_i(x), i = 1, \cdots, m, \end{aligned} \tag{8.1}$$

where $x \in X$ is the state, $u \in U$ is the control input, $\theta \in \Theta$ represents the parameters of the system, $d \in D$ is the disturbance (adversarial input), $F : X \times U \times \Theta \times D \to X$ is the system evolution function, and $y_i, i = 1, \cdots, m$, are Boolean system outputs, where $\mu_i : X \to \mathbb{B}$. We define the set of atomic propositions $\Pi = \{\pi_1, \cdots, \pi_m\}$ such that $x \models \pi_i \Leftrightarrow \mu_i(x) = \text{True}, i = 1, \cdots, m$. The sets X, U, Θ, D are the admissible sets for states, controls, parameters and disturbances respectively. All sets may be finite or infinite. System (8.1) is finite if X, U, Θ, D are all finite.

Example 17. A prominent class of systems encountered in adaptive control are parameterized linear systems, where $F(x, u, \theta, d) = A(\theta)x + B(\theta)u + d$. We have $X \subset \mathbb{R}^{n_x}$, $U \subset \mathbb{R}^{n_u}$, $\Theta \subset \mathbb{R}^{n_\theta}$, $D \subset \mathbb{R}^{n_d}$. A, B are matrices with appropriate dimensions that depend on θ. It is also common to assume that the outputs are Boolean evaluations of linear predicates $\mu_i = (r_i^T x \leq \rho_i)$, where $r_i \in \mathbb{R}^n$, and $\rho_i \in \mathbb{R}$. Thus, each proposition π_i defines a closed half space in $\mathbb{R}^{n_x}$.

As mentioned in the introduction, we distinguish between the uncertainty in parameters and disturbances. Disturbances usually have unknown (fast) variations in time. In this chapter, we assume that θ is a constant but its value θ^* is initially unknown. If we treat the uncertainties in parameters and disturbances in the same way, we are required to design control strategies that are robust versus all values in both Θ and D. This approach is severely conservative and often fails to find a solution. The key idea of adaptive control is to take advantage of the fact that θ^* can be (approximately) inferred from the history of the evolution of the system. Therefore, adaptive control is often significantly more powerful than pure robust control and it is also more difficult to design and analyze. In engineering applications, parameters are related to the physical attributes of the plant whereas disturbances are related to effects of stochastic nature such as imperfect actuators/sensors and perturbations in the environment.

Problem 11. Given system (8.1) and an LTL formula φ over Π, find a control strategy $\Lambda : X^* \times U^* \to U$ and a set of initial states $X_0 \subseteq X$ such that all the trajectories of the closed loop system starting from X_0 satisfy φ.

Our aim is to convert Problem 12 to an LTL control problem described in Sec.8.0.3 and use the standard tools for Rabin games. To this end, we need to incorporate adaptation into control synthesis. The central tool to any adaptive control technique is parameter estimation. Note that an adaptive control strategy has the form $\Lambda : X^* \times U^* \to U$, since parameters are estimated using the history of the evolution of the system. We take the following approach to convert Problem 12 into an LTL control problem. We embed system (8.1) in a parametric transition system (PTS), which is defined in Sec. 8.2. We construct a finite adaptive transition system (ATS) from a finite PTS. An ATS is an ordinary transition system as in Sec. 8.0.1, but parameters are also incorporated into its states and transitions in appropriate way, which is explained in Sec. 8.3. We deal with an infinite PTS by constructing a finite quotient PTS in Sec. 8.4.

8.2 Parametric Transition System

Definition 44. A parametric transition system (PTS) is defined as the tuple $\mathcal{T}^{\Theta} = (X, U, \Theta, \gamma, \Pi, O)$, where:

- X is a (possibly infinite) set of states;
- U is a (possibly infinite) set of control inputs;
- Θ is a (possibly infinite) set of parameters;
- γ is a transition function $\gamma : X \times U \times \Theta \to 2^X$.
- $\Pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ is a finite set of atomic propositions;
- $O : X \to 2^{\Pi}$ is an observation map.

The only difference between a PTS and a transition system is that its transitions depend on parameters. Note that if $|\Theta| = 1$, a PTS becomes a transition system.

Now we explain how to represent (8.1) in the form of a PTS. The sets X, U, Θ are inherited from (8.1) (which is why we have used the same notation). The transition function γ is constructed such that

$$\gamma(x, u, \theta) = \left\{ F(x, u, \theta, d) \middle| d \in D \right\}. \tag{8.2}$$

The observation map $O : X \to 2^{\Pi}$ is given by:

$$O(x) = \left\{ \pi_i \middle| \mu_i(x) = \text{True}, i = 1 \cdots, m \right\}. \tag{8.3}$$

Therefore, $\mathcal{T}^{\Theta} = (X, U, \Theta, \gamma, \Pi, O)$ captures everything in system (8.1). We refer to $\mathcal{T}^{\Theta}$ as the *embedding* of (8.1). One can interpret a PTS as a (possibly infinite) family of transition systems. The actual transitions are governed by a single parameter θ^*, which is initially unknown to the controller. Therefore, the controller has to find out which transition system is the ground truth.

8.3 Control Synthesis for Finite Systems

In this section, we assume the PTS embedding system (8.1) is finite.

8.3.1 Parameter Estimation

Definition 45. A *parameter estimator* Γ is a function

$$\Gamma : X^* \times U^* \to 2^{\Theta}_{-\emptyset} \tag{8.4}$$

that maps the history of visited states and applied controls to a subset of parameters. We have $\vartheta_k = \Gamma(x_0 \cdots x_k; u_0 \cdots u_{k-1})$, where:

$$\vartheta_k = \left\{ \theta \in \Theta \middle| x_{i+1} \in \gamma(x_i, u_i, \theta), 0 \leq i \leq k-1 \right\}. \tag{8.5}$$

One can see that the parameter estimator (8.5) is “sound” in the sense that $\theta^* \in \vartheta_k, \forall k \in \mathbb{N}$. We have $\vartheta_0 = \Gamma(x_0) = \Theta$, by definition. Note that our definition of parameter estimator is different from the traditional ones, which are often in the form

$X^* \times U^* \to \Theta$, as they return only an estimate $\hat{\theta}$ rather than the set of all possible parameters. For our formal setup, it is vitally important that the controller take into account all possible ground truth parameters at all times. Otherwise, guaranteeing the specification is impossible. The following proposition enables us to make (8.5) recursive.

Proposition 18. The following recursive relation holds:

$$\vartheta_{k+1} = \left\{ \theta \in \vartheta_k \middle| x_{k+1} \in \gamma(x_k, u_k, \theta) \right\}. \tag{8.6}$$

Proof. Substitute ϑ_k from (8.5):

$$\begin{aligned} & \left\{ \theta \in \vartheta_k \middle| x_{k+1} \in \gamma(x_k, u_k, \theta) \right\} \\ = & \left\{ \theta \in \Theta \middle| \theta \in \vartheta_k, x_{k+1} \in \gamma(x_k, u_k, \theta) \right\} \\ = & \left\{ \theta \in \Theta \middle| x_{i+1} \in \gamma(x_i, u_i, \theta), 0 \leq i \leq k \right\} = \vartheta_{k+1}. \end{aligned}$$

□

Corollary 6. The set of estimated parameters never grows: $\vartheta_{k+1} \subseteq \vartheta_k, \forall k \in \mathbb{N}$.

Therefore, we obtain a recursive parameter estimator $\Gamma_{rec} : 2^{\Theta}_{-\emptyset} \times X \times U \times X \to 2^{\Theta}_{-\emptyset}$ as $\vartheta_{k+1} = \Gamma_{rec}(\vartheta_k, x_k, u_k, x_{k+1})$. Note that Γ_{rec} is deterministic.

8.3.2 Adaptive Transition System

As mentioned in the introduction, a primary challenge of provably correct adaptive control is coupling parameter estimation and control synthesis. In order to combine these two, we provide the following definition.

Definition 46. Given a PTS $\mathcal{T}^{\Theta} = (X, U, \Theta, \gamma, \Pi, O)$, we define the adaptive transition system (ATS) as the tuple $\mathcal{T}^{adp} = \left(X^{adp}, U, \gamma^{adp}, \Pi, O^{adp}\right)$, where U, Π are inherited

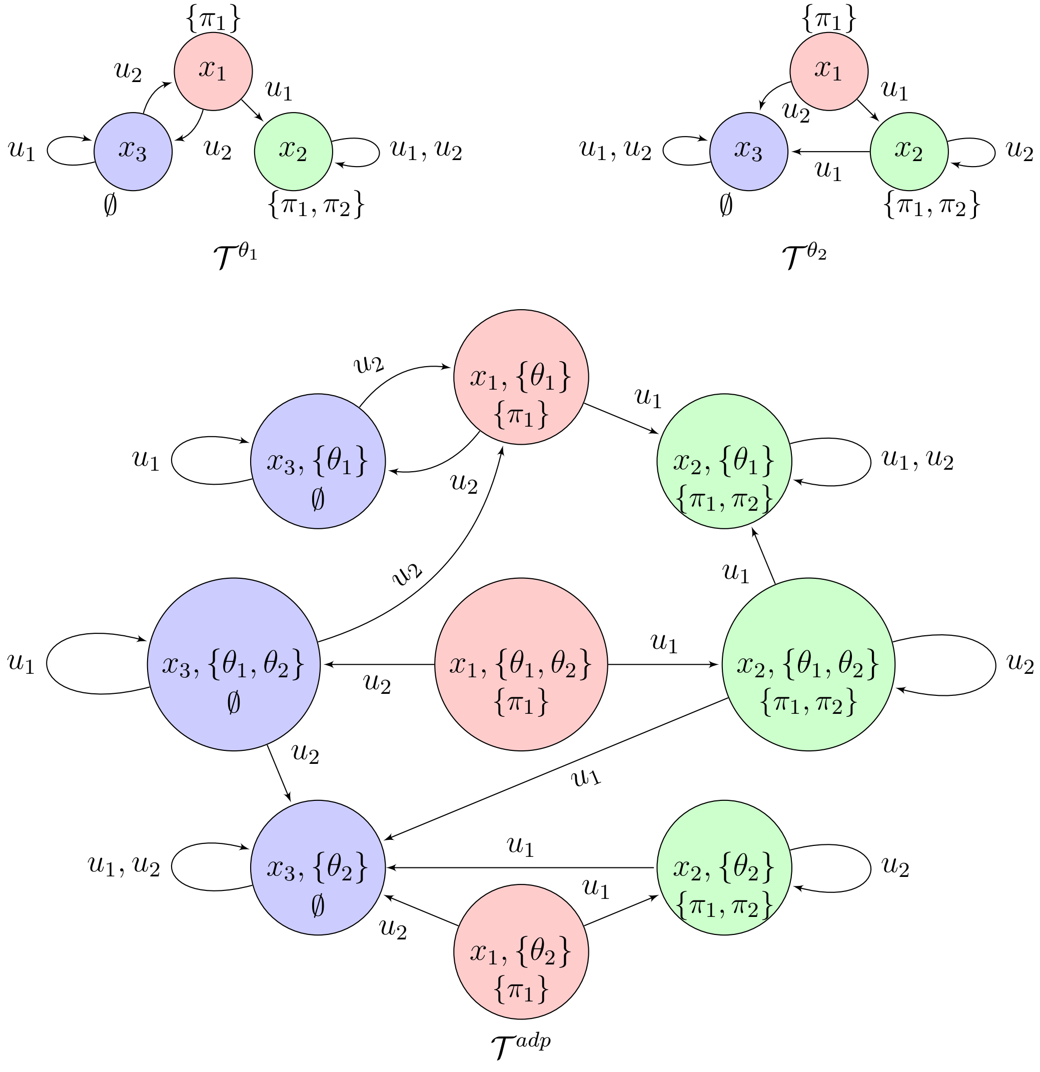


Figure 8·1: Example 18: [Top] A PTS with two possible parameters θ_1, θ_2, and the corresponding transition systems [Bottom] The corresponding ATS from $\mathcal{T}^{\Theta}$ with the same meaning and

- $X^{adp} \subseteq X \times 2^{\Theta}_{-\emptyset}$ is the set of states;
- $\gamma^{adp} : X^{adp} \times U \to 2^{X^{adp}}$ is the transition function, where we have $(x', \vartheta') \in \gamma^{adp}((x, \vartheta), u)$ if and only if $x' \in \gamma(x, u)$ and $\vartheta' = \Gamma_{rec}(\vartheta, x, u, x')$;
- $O^{adp} : X^{adp} \to 2^{\Pi}$ is the observation function where $O^{adp}(x, \vartheta) = O(x), \forall x \in X, \vartheta \in 2^{\Theta}_{-\emptyset}$.

Example 18. Consider a PTS with $X = \{x_1, x_2, x_3\}, U = \{u_1, u_2\}$, and $\Theta = \{\theta_1, \theta_2\}$, and $\Pi = \{\pi_1, \pi_2\}$, where $O(x_1) = \{\pi_1\}$, $O(x_2) = \{\pi_1, \pi_2\}$, and $O(x_3) = \emptyset$. The transition systems corresponding to θ_1 and θ_2 are illustrated in Fig. 8·1 [top]. The ATS corresponding is shown in Fig. 8·1 [Bottom].

The number of states in the ATS is upper-bounded by $|X|(2^{|\Theta|} - 1)$, which shows an exponential explosion with the number of parameters. Fortunately, not all states in $X \times 2^{\Theta}_{-\emptyset}$ are reachable from the set $\{(x, \theta) | x \in X, \theta \in \Theta\}$, which is the set of possible initial states in the ATS. Algorithm 2 constructs the ATS consisting of only these reachable states.

8.3.3 Control Synthesis

Finally, given an ATS $\mathcal{T}^{adp}$ and an LTL formula φ, we construct the product automaton $\mathcal{T}^{adp} \otimes \mathcal{R}_{\varphi}$ as explained in Sec. 8.0.3, and find the memoryless control strategy on $\mathcal{T}^{adp} \otimes \mathcal{R}_{\varphi}$ by solving the Rabin game. We also find the largest set of admissible initial conditions $X_0^{adp,\max}$ as the winning region of the Rabin game. In order to find $X_0^{\max}$, we perform the following projection:

$$X_0^{\max} = \left\{ x_0 \middle| (x_0, \Theta) \in X_0^{adp,\max} \right\}. \tag{8.7}$$

Algorithm 2 Procedure for Constructing ATS from a PTS

```
Require: $\mathcal{T}^{\Theta} = (X, U, \Theta, \gamma, \Pi, O)$
  $X^{adp,new} = \{(x, \Theta) | x \in X\}$
  $X^{adp} = X^{adp,new}$
  while $X^{adp,new} \neq \emptyset$ do
      $X^{adp,new} \leftarrow \emptyset$
      for $(x, \vartheta) \in X^{adp}$ do
          for $u \in U$ do
              $\gamma^{adp}((x, \vartheta), u) = \emptyset$
              $\vartheta' = \emptyset$
              for $\theta \in \vartheta$ do
                  for $x' \in \gamma(x, u, \vartheta)$ do
                      for $\theta' \in \vartheta$ do
                          if $x' \in \gamma(x, u, \theta')$ then
                              $\vartheta' \leftarrow \vartheta' \cup \theta'$
                      $\gamma^{adp}((x, \vartheta), u) \leftarrow \gamma^{adp}((x, \vartheta), u) \cup (x', \vartheta')$
                      if $(x', \vartheta') \notin X^{adp}$ then
                          $X^{adp,new} \leftarrow X^{adp,new} \cup (x', \vartheta')$
                          $X^{adp} \leftarrow X^{adp} \cup (x', \vartheta')$
                          $O^{adp}(x', \vartheta') = O(x')$
  return $\mathcal{T}^{adp} = \left(X^{adp}, U, \gamma^{adp}, \Pi, O^{adp}\right)$
```

The adaptive control strategy takes the memoryless form $\Lambda : X \times 2^{\Theta}_{-\emptyset} \times S \to U$, which maps the current state in the PTS, the set of current possible ground truth parameters and the state in the Rabin automaton to an admissible control action.

Theorem 13. Given a finite system (8.1), an initial condition $x_0 \in X$, an LTL formula over Π, there exists a control strategy $\Lambda^* : X^* \times U^* \to U$ such that $O(x_0)O(x_1)\cdots \models \varphi$, $\forall \theta \in \Theta, \forall d_k \in D$, $x_{k+1} = F(x_k, u_k, \theta, d_k), \forall k \in \mathbb{N}$, if and only if $x_0 \in X_0^{\max}$. .

Proof. (sketch) The completeness property follows from two facts. First, the solutions to Rabin games on finite automata are complete. Second, every possible behavior of a finite PTS embedding (8.1) and parameter estimator (8.5) is captured in the ATS. If $x_0 \notin X_0^{\max}$, then it can be shown that there exists a $\theta \in \Theta$ and a disturbance sequence $d_0 d_1 \cdots$ such that there does not exist any control strategy to satisfy the LTL specification. $\square$

8.4 Control Synthesis for Infinite Systems

In this section, we assume that PTS embedding (8.1) is not finite, which means that at least one of the sets X, U, Θ is infinite. We provide the general solution for the case when all sets are infinite. We note that the approach in this section is still preliminary and we leave further investigation to our future work.

We consider a finite observation preserving (see Sec. 8.0.2) partition $Q_X = \{q_X^1, \cdots, q_X^{p_X}\}$ for X and a finite partition $Q_\Theta = \{q_\Theta^1, \cdots, q_\Theta^{p_\Theta}\}$ for Θ. We also quantize U to obtain a finite $U_{\text{qtz}} = \{u_{qtz}^1, \cdots, u_{qtz}^{p_u}\}$. In this chapter, we do not consider any particular guideline for how to partition and leave this problem to our future work. In general, the finer the partitions, the less conservative the method is with a price of higher computational effort. "Smart" partition refinement procedures were studied in [Yordanov et al., 2013, Nilsson and Ozay, 2014].

Once partitions and quantizations are available, we compute the transitions. We denote the successor (post) of set q_X, under parameter set q_Θ and control u by

$$\text{Post}(q_X, q_\Theta, u) := \Big\{x \in X \Big| \exists x \in q_X, \exists \theta \in q_\Theta, x \in \gamma(x, \theta, u)\Big\}. \tag{8.8}$$

A computational bottleneck is performing the post computation in (8.8). For additive parameters, the post computation is exact for piecewise affine systems using polyhedral operations [Yordanov et al., 2012]. For multiplicative parameters, an over-approximations of post can be computed [Yordanov and Belta, 2008], which introduces further conservativeness but retains correctness. Finally, we construct the quotient PTS from the infinite PTS. The procedure is outlined in Algorithm 3.

Algorithm 3 Constructing quotient PTS from infinite PTS

```
Require: $\mathcal{T}^\Theta = (X, U, \Theta, \gamma, \Pi, O)$
Require: $Q_X, Q_\Theta, U_{\text{quantized}}$
for $q_X \in Q_X$ do
    $O^Q(q_X) = O(x)$ for some $x \in q_X$
    for $q_\Theta \in Q_\Theta$ do
        for $u_{qtz} \in U_{qtz}$ do
            $X_{\text{post}} = \text{Post}(q_X, q_\Theta, u)$
            $\gamma^Q(q_X, u_{qtz}, q_\Theta) = \emptyset$
            for $q'_X \in Q_X$ do
                if $X_{\text{post}} \cap q'_X \neq \emptyset$ then
                    $\gamma^Q(q_X, u_{qtz}, q_\Theta) \leftarrow \gamma^Q(q_X, u_{qtz}, q_\Theta) \cup q'_X$
return $\mathcal{T}^{Q,\Theta} = \big(Q_X, U_{\text{quantized}}, Q_\Theta, \gamma^Q, \Pi, O^Q\big)$
```

8.5 Case Studies

We present two case studies. The first one is a simple finite deterministic system. The second case study involves a linear parameterized system that is infinite and non-deterministic due to the presence of additive disturbances.

Example 19.

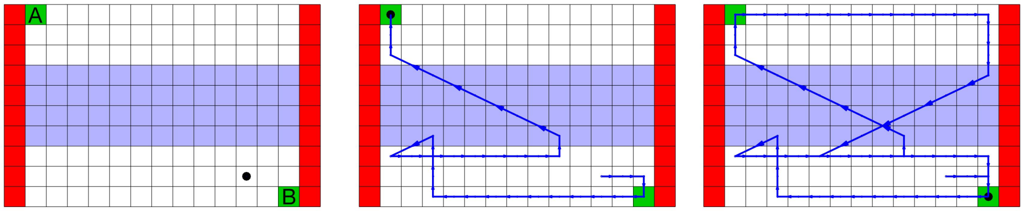


Figure 8·2: Example 19: (Left): The Robot (shown in black) and its environment. (Middle): Snapshots of the executed Motion at time $k = 33$, and (Right) $k = 62$. The robot satisfies the specification.

We consider a robot motion planning problem. The environment is modeled as a finite number of cells illustrated in Fig. 8·2. Each cell corresponds to a state in X. We have $|X| = 150$. The set of control inputs is given by $U = \{$ **left, right, up, down**$\}$, where the transition enabled by each input corresponds to its unambiguous meaning. There exists an constant drift in the horizontal direction in the purple region, but its direction to left or right and its intensity are unknown. The set of possible drifts is $\Theta = \{+2, +1, 0, -1, -2\}$, where positive sign corresponds to the left direction. At each time, if the robot is in a purple cell, the drift is added to its subsequent position. For example, if the robot applies $u =$**right**, and $\theta^* = 2$, the robot actually ends up in a cell to the left. Similarly, if $u =$**up** and $\theta^* = -2$, the robot moves a cell up and two cells to the right. The red cells are "unsafe" regions that must be avoided, and the green cells A, B are "interesting" regions, which have to be persistently visited. The LTL formula describing this specification is:

$$\varphi = \mathbf{GF}A \ \wedge \ \mathbf{GF}B \ \wedge \ \mathbf{G}(\neg \text{unsafe}).$$

We implemented the procedure outlined in Sec. 8.3. It is worth to note that there does not exist a pure robust control solution to this problem. In other words, if the robot ignores estimating the drift, it can not find a control strategy. For example, if the robot enters the purple region around the middle and persistently applies **up**, a maximum drift in either direction can drive the robot into the unsafe cells before it exits the purple region. Therefore, the only way the robot can fulfill the specification is to learn the drift. The robot first enters the drifty region to find out its value and then moves back and re-plans its motion. Notice that this procedure is fully automated using the solution of the Rabin game on the product $\mathcal{T}^{adp} \otimes \mathcal{R}_\varphi$. Two snapshots of the executed motion for the case $\theta^* = +2$ are shown in Fig. 8·2.

Example 20. Consider a one-dimensional linear system of the following form:

$$x^+ = (1+\theta_1)x + \theta_2 u + \theta_3 + d, \tag{8.9}$$

where $\theta_1 \in [-0.5, 0.5]$, $\theta_2 \in [1,2]$, and $\theta_3 \in [-0.2, 0.2]$ are fixed parameters, and $d \in D$, is the additive disturbance, $D = [-0.1, 0.1]$. The set of admissible control inputs is $U = [-1,1]$. We desire to restrict x to the $[-1,1]$ interval for all times, which is described by the following LTL formula:

$$\varphi = \mathbf{G}(x \leq 1) \wedge \mathbf{G}(x \geq -1).$$

We have $\Theta = [-0.5, 0.5] \times [1,2] \times [-0.2, 0.2]$. We partitioned the intervals of θ_1, θ_2, θ_3, and X into 2,2,4, and 10 evenly spaced intervals, respectively. Thus, we have partitioned Θ into 16 cubes ($|Q_\Theta| = 16$) and X into 10 intervals ($|Q_X| = 10$). U is quantized to obtain $U_{qtz} = \{-1, -0.8, \cdots, 0.8, 1\}$. We implemented Algorithm 3 to obtain the quotient PTS and Algorithm 2 to find the corresponding ATS. The computation times were 0.1 (Algorithm 3) and 152 (Algorithm 2) seconds on a 3.0 GHz MacBook Pro. Even though $|X \times 2^{Q_\Theta}_{-\emptyset}| = 655350$, the number of reachable states

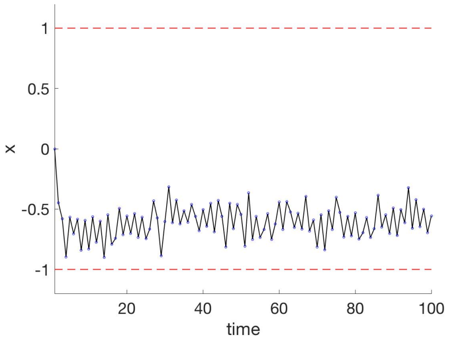


Figure 8·3: Example 20: trajectory of the system versus time, which is always between -1 and 1.

obtained from Algorithm 2 was 14146.

We solved the safety game on the ATS, which took less than a second and found a winning region containing 14008 states. The winning region in the state-space is $X_0 = [-0.6, 0.6]$. Since the solution is conservative, $X_0^{\max}$ may be larger if a finer partitioning is used. We also found that the winning region is empty if we had sought a pure robust control strategy. We simulated the system for 100 time steps starting from $x_0 = 0$. The values of disturbances at each time are chosen randomly with a uniform distribution over D. We observe that the specification is satisfied, and the sets given by the parameter estimator shrink over time and always contain the ground truth parameter, which in this case is $\theta_1^* = 0.45$, $\theta_2^* = 1.11$, $\theta_3^* = -0.18$. The results are shown in Fig. 8·4.

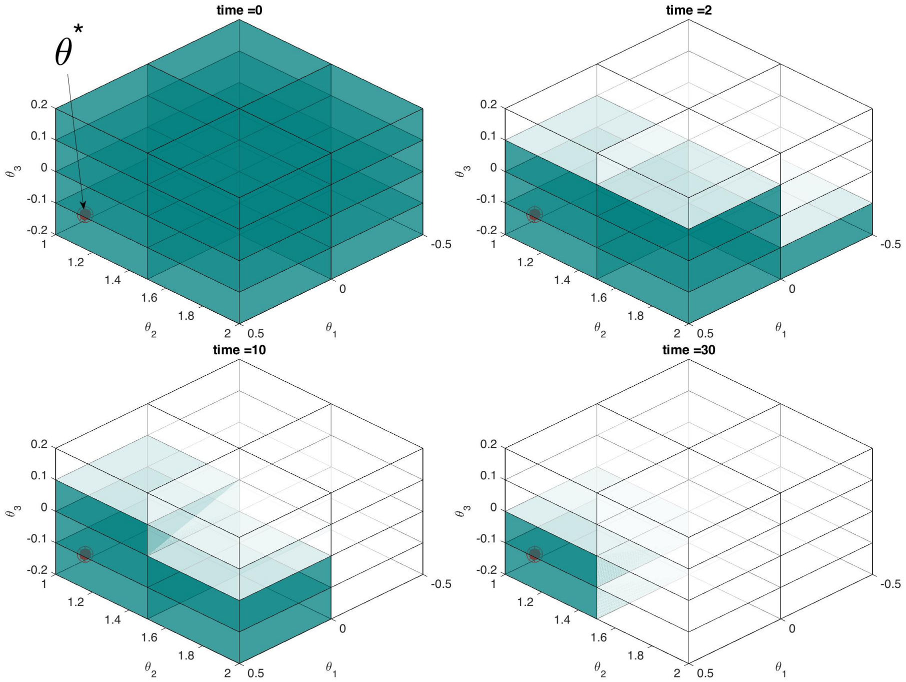


Figure 8·4: Example 20: Snapshots of ϑ_k at various times, which are illustrated by the shaded regions. They always contain the ground truth parameter $\theta_1^* = 0.45$, $\theta_2^* = 1.11$, $\theta_3^* = -0.18$.

Chapter 9

Formal Model Identification

In this chapter, a discrete-time system is considered from which we only assume a finite set of input-output data and bounds describing the continuity of the system. The latter, which is given by only 2 or 3 positive numbers, is essential to characterize the range of possible system behaviors. The goal is to design controllers from STL specifications over predicates on the state. If STL satisfaction is not possible, we are still interested in finding the least-violating controllers. Our main results and contributions are as follows:

- We fit a piecewise affine (PWA) model to data and continuity constants. PWA models are able to capture arbitrarily high degrees of nonlinearity by tuning the number of modes. Unlike existing works on hybrid model identification [Bemporad et al., 2005, Paoletti et al., 2007, Alur and Singhania, 2014], we provide a set-valued model which is non-deterministic in nature, and is guaranteed to contain all the behaviors of the concrete system. All the non-determinism is captured by polytopic additive disturbances. Since such PWA models are not unique, we find ones that have the smallest non-determinism - in a sense that is clarified in the chapter. The model identification technique in this chapter is based on solving a series of non-convex optimization problems, which are handled using mixed-integer linear programming (MILP).

- Once we have obtained a PWA model with additive polytopic disturbances, we design controllers from STL specifications using the tube-based approach in Chapter 4.

This chapter is organized as follows. The problem and the underlying assumptions are stated in Sec. 9.1. An overview of the approach is presented in Sec. 9.2. Technical details on model identification are provided in Sec. 9.3. An illustrative example is demonstrated in Sec. 9.4.

Notation

The set of natural, real, and non-negative real numbers are denoted by $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{R}_+$, respectively. The set of all finite sequences and infinite sequences that can be generated from an alphabet A are denoted by A^* and A^ω, respectively. A discrete-time real signal - simply referred to as *signal* in the rest of the chapter - is an infinite sequence $s[0]s[1]s[2]\cdots$, where $s \in (\mathbb{R}^n)^\omega$, $s[k] \in \mathbb{R}^n, k \in \mathbb{N}$. All time intervals in this chapter are interpreted in discrete-time: $[a,b] = \{a, a+1, \cdots, b\}, a, b \in \mathbb{N}, a < b$. Given sets $X, Y \subset \mathbb{R}^n$, their Minkowski sum is denoted by $X \oplus Y = \big\{x + y \big| x \in X, y \in Y\big\}$. The relation $\leq$ between two matrices of the same size is interpreted element-wise. The transpose of matrix M is denoted by M^T. The unit-vector in i^{th} direction and the vector of all ones in $\mathbb{R}^n$ are denoted by $e_{[i]}$ and 1_n, respectively. The absolute value of $x \in \mathbb{R}$ is shown by $|x|$, and the p-norm of $x \in \mathbb{R}^n$ is denoted by $\|x\|_p$. The unit p-norm ball is $\mathcal{B}_p := \{x \in \mathbb{R}^n \big| \, \|x\|_p \leq 1\}$. The convex hull of a set S is denoted by $\text{Convh}(S)$.

9.1 Problem Statement

Informally, the goal is to use data gathered from an unknown system to design a control policy such that an STL formula over predicates on state is satisfied for all closed-loop trajectories originating from a designated set of initial conditions. If STL satisfaction is not possible, we still want to compute the control policy resulting in the least worst-case STL violation. We formalize this problem in this section.

9.1.1 System and Assumptions

Definition 47. Given a workspace $X \subset \mathbb{R}^n$ and an admissible set of inputs $U \subset \mathbb{R}^m$, a control system $\mathcal{F}$ is defined as a *triadic relation*

$$\mathcal{F} \subset X \times U \times \mathbb{R}^n, \tag{9.1}$$

which is *left-total* in the sense that $\forall x \in X, \forall u \in U, \exists x^+ \in \mathbb{R}^n$ such that $(x, u, x^+) \in \mathcal{F}$.

Sets X and U are assumed to be compact and locally connected in their respective domain. Note that we have not assumed that X is invariant for all controls. It may be possible that $\exists (x, u, x^+) \in \mathcal{F}$ such that $x \in X, u \in U$, but $x^+ \notin X$. Keeping the state within the workspace is a non-trivial task that is an implicit objective of our problem. A control system $\mathcal{F}$ is *deterministic* if for all $(x_1, u_1, x_1^+), (x_2, u_2, x_2^+) \in \mathcal{F}$, $x_1 = x_2$ and $u_1 = u_2$ implies $x_1^+ = x_2^+$.

Definition 48. A control policy $\mu : X^* \to U$ is a function that determines the control input at time t as a feedback of the history of the system:

$$u[t] = \mu(x[0]x[1] \cdots x[t]). \tag{9.2}$$

Definition 49. Given a control system $\mathcal{F}$, a control policy μ, a set of initial conditions $X_0 \subseteq X$, we define the *closed-loop language* as $\mathcal{L} = \mathcal{L}(\mathcal{F}, X_0, \mu) \subset X^\omega$ such that $x[0]x[1]\cdots \in \mathbb{L}$ if and only if $x[0] \in X_0$ and

$$(x[k], \mu(x[0]x[1]\cdots x[k]), x[k+1]) \in \mathcal{F}, \forall k \in \mathbb{N}.$$

Assumption 7. (Closed-loop language non-emptiness) There exists a control policy μ such that for some $X_0 \subseteq X$, we have $\mathcal{L}(\mathcal{F}, X_0, \mu) \neq \emptyset$.

Assumption 7 is trivially essential for our purpose. Otherwise, it is not possible to keep the system in the workspace. Assumption 7 can be relaxed for applications that the objective can be accomplished in finite time and the system state is allowed to exit the workspace afterwards. In this chapter, our emphasis is on infinite-time properties and finite-time specifications are treated as a special case.

We assume no knowledge of $\mathcal{F}$, which we refer to as the *concrete* control system, except the following assumptions.

Assumption 8. (Data Points) We are given a set of N data points

$$\mathcal{D} := \left\{(x_i, u_i, x_i^+) \in \mathcal{F}\right\}_{i=1,\cdots,N}.$$

Assumption 8 is not restrictive as we are often able to estimate the system state, apply some control input, and measure the subsequent state. In this setting, we treat the system as an input-output black-box. Assumption 8 may also prove useful when some analytical form of $\mathcal{F}$ is available, but is too complex to use for control synthesis purposes. In this case, we may sample data points from $\mathcal{F}$ rather than use its analytical form.

Assumption 9. (Continuity bounds) We are given non-negative constants $\kappa_0, \kappa_x, \kappa_u$, referred to as continuity constants, such that for all $(x_1, u_1, x_1^+), (x_2, u_2, x_2^+) \in \mathcal{F}$, the following relation holds:

$$\left\| x_2^+ - x_1^+ \right\|_p \leq \kappa_0 + \kappa_x \left\| x_2 - x_1 \right\|_p + \kappa_u \left\| u_2 - u_1 \right\|_p , \tag{9.3}$$

where $p \geq 1$ is a choice of norm.

Constant κ_0 characterizes the degree of non-determinism in $\mathcal{F}$ and κ_x, κ_u characterize how continuous (in a Lipschitz sense) $\mathcal{F}$ is on X and U. If we know $\mathcal{F}$ is deterministic, we let $\kappa_0 = 0$. The assumption that the evolution of a physical system is continuous in state and controls is reasonable. Even many hybrid systems demonstrate continuity in the Lipschitz sense. Therefore, there always exists constants $\kappa_0, \kappa_x, \kappa_u$ such that (9.3) holds. The stronger assumption that we make in Assumption 9 is that we *know* the values of continuity constants. Note that $\kappa_0, \kappa_x, \kappa_u$ do not need to be the *best constants*. In other words, the inequality (9.3) does not need to be tight. Any upper-bound for the best values of $\kappa_0, \kappa_x, \kappa_u$ is sufficient for the soundness of the results in this chapter, but very large values obviously lead to excessive conservatism.

Any guarantee is provided against the values of constants in Assumption 9. Estimating the continuity constants using data points in $\mathcal{D}$ is not a sound approach since it is always possible to observe new data points that falsify the validity of the estimated constants. However, in practice, there might not be any other option than using $\mathcal{D}$ for estimating the continuity constants. One way to approach this issue is multiplying the tightest estimates for continuity constants by some safety factor, depending on the application. There exists several methods for estimating Lipschitz constants from data [Milanese and Novara, 2004, Calliess, 2017].

9.1.2 STL Specification

We are given a STL formula φ in the bounded-global form. The predicates of φ are considered to be linear over state:

$$\mathcal{P}_i := (\pi_i^T x \leq \zeta_i), \tag{9.4}$$

where $\pi_i \in \mathbb{R}^n$, $i = 1, \cdots, n_{\mathcal{P}}$, - $n_{\mathcal{P}}$ is the number of predicates, and $\zeta_i \in \mathbb{R}$. We also define

$$\Pi := \left(\pi_1^T \cdots \pi_{n_{\mathcal{P}}}^T\right)^T. \tag{9.5}$$

Matrix $\Pi \in \mathbb{R}^{n_{\mathcal{P}} \times n}$ characterizes the sensitivity of the predicates to changes in the state. We do not formulate predicates over controls but the state may be extended to include controls (see, e.g., [Rungger et al., 2013]).

9.1.3 Problem Formulation

Problem 12. Given data points $\mathcal{D}$ from a control system $\mathcal{F}$ as in (9.1), constants $\kappa_0, \kappa_x, \kappa_u$ corresponding to Assumption 9, an STL formula φ over predicates in the form of (9.4), find an optimal control policy μ^* and and a set of initial conditions $X_0^* \subseteq X$ such that:

$$\begin{array}{rll} (X_0^*, \mu^*) = & \underset{X_0, \mu}{\text{argmax}} & \epsilon \\ & \text{subject to} & \mathcal{L}(\mathcal{F}, X_0, \mu) \subseteq \mathcal{L}(\varphi, \epsilon). \end{array} \tag{9.6}$$

Our framework is able to accommodate slight variations of Problem 12. For instance, X_0 may be fixed by a user-specified set or point. We may also consider some weighted cost functions added to ϵ, such as penalizing the controls or distance from a reference trajectory.

9.2 Approach

Our solution to Problem 12 has two main steps. First, we construct a model from $\mathcal{D}$ and continuity constants. Our model has to serve two purposes: i) it has to contain all the behaviors of $\mathcal{F}$, as formalized shortly, and ii) has to be simple enough for control synthesis. We choose PWA models with user-specified number of modes - formally defined in Sec. 9.3.1. PWA models are able to capture arbitrarily high nonlinearities by increasing the number of modes in exchange for higher computational complexity - both in identifying the model and also synthesis based on the model. We focus on a a particular class of control strategies that are computationally tractable to compute. Therefore, completeness may be lost and we may obtain suboptimal solutions for Problem 12. However, we do not trade off correctness. Once our method returns some ϵ^* for Problem 12, we have the guarantee that all closed-loop trajectories of $\mathcal{F}$ are in ϵ^*-language of φ.

Definition 50. Given two systems $\mathcal{F} \subset X \times U \times \mathbb{R}^n$ and $\mathcal{G} \subset X \times U \times \mathbb{R}^n$, we say $\mathcal{G}$ *simulates* $\mathcal{F}$ if and only if $\mathcal{F} \subseteq \mathcal{G}$.

Definition 50 is reminiscent of simulation relation in concurrent systems [Tabuada, 2008, Belta et al., 2017]. Here we do not define simulation relation with respect to a particular equivalence class - here every state is equivalent only to itself and no abstraction is used. Simulation is a partial order relation since the following properties hold: $\mathcal{F}$ simulates $\mathcal{F}$; If $\mathcal{G}$ simulates $\mathcal{F}$ and $\mathcal{H}$ simulates $\mathcal{G}$, then $\mathcal{H}$ simulates $\mathcal{F}$; If $\mathcal{G}$ simulates $\mathcal{F}$ and vice-versa, then $\mathcal{F} = \mathcal{G}$. The following result holds from language inclusion properties of simulation relation [Belta et al., 2017].

Lemma 5. If $\mathcal{G}$ simulates $\mathcal{F}$, then for all μ and X_0 we have $\mathcal{L}(\mathcal{F}, X_0, \mu) \subseteq \mathcal{L}(\mathcal{G}, X_0, \mu)$.

9.3 Formal Model Identification

In this section, we introduce a method to identify a set-valued PWA model using data and continuity bounds.

Definition 51. Given data points $\mathcal{D} = \{(x_i, u_i, x_i^+)\}_{i=1,\cdots,N}$ from an unknown control system $\mathcal{F} \subset X \times U \times \mathbb{R}^n$, constants $\kappa_0, \kappa_x, \kappa_u$ corresponding to Assumption 9, the *tightest simulating* control system $\overline{\mathcal{F}} \subset X \times U \times \mathbb{R}^n$ is defined such that for all $(x, u, x^+) \in \overline{\mathcal{F}}$, the following holds:

$$x^+ \in \bigcap_{i=1}^{N} \left\{ \{x_i^+\} \oplus k_x \|x - x_i\|_p \mathcal{B}_p \oplus k_u \|u - u_i\|_p \mathcal{B}_p \oplus k_0 \mathcal{B}_p \right\}.$$

Lemma 6. The following properties hold: 1) $\mathcal{F} \subseteq \overline{\mathcal{F}}$; 2) For any $\mathcal{G}$ that is guaranteed to simulate $\mathcal{F}$, we have $\overline{\mathcal{F}} \subseteq \mathcal{G}$.

Proof. 1) The proof follows from two facts that establish $\mathcal{F} \subseteq \overline{\mathcal{F}}$. First, we require $(x_i, u_i, x_i^+) \in \overline{\mathcal{F}}, i = 1, \cdots, N$. Second, any point that is included in $\overline{\mathcal{F}}$ is sufficiently close to other data points in the sense of (9.3). 2) If there exists $\mathcal{G}$ simulating $\mathcal{F}$ such that $\overline{\mathcal{F}} \not\subset \mathcal{G}$, then there exists some $(x_s, u_s, x_s^+) \in \overline{\mathcal{F}}$ that is allowed to be in $\mathcal{F}$ by Assumption 9. Thus, $\mathcal{F} \subseteq \mathcal{G}$ does not necessarily hold.

We may use $\overline{\mathcal{F}}$ for control synthesis, but its representation is data-size dependent and is often too complex. We need simpler forms of systems that simulates $\overline{\mathcal{F}}$ (and hence $\mathcal{F}$).

9.3.1 PWA Systems

Definition 52. A control system $\mathcal{G} \subset X \times U \times \mathbb{R}^n$ is PWA if $\forall x \in X, \forall u \in U$, we have $(x, u, x^+) \in \mathcal{G}$ if and only if

$$x^+ \in \begin{cases} \{A_1 x + B_1 u + c_1\} \oplus W_1, & x \in X_1, \\ \vdots & \vdots \\ \{A_{\mathcal{M}} x + B_{\mathcal{M}} u + c_{\mathcal{M}}\} \oplus W_{\mathcal{M}}, & x \in X_{\mathcal{M}}, \end{cases} \tag{9.7}$$

where $X_i, i = 1, \cdots, \mathcal{M}$, are polyhedral sets with disjoint interiors, $\bigcup_{i=1}^{\mathcal{M}}(X_i) = X$, $\mathcal{M}$ is the number of modes, and $W_i \in \mathbb{R}^n, i = 1, \cdots, \mathcal{M}$, are polytopic sets of additive disturbances. Each mode is an affine system with constants $A_i \in \mathbb{R}^{n \times n}, B_i \in \mathbb{R}^{n \times m}$ and $c_i \in \mathbb{R}^n$.

In the rest of this section, we propose a method to solve the following subproblem:

SubProblem 1. Given data points $\mathcal{D}$ from control system $\mathcal{F} \subset X \times U \times \mathbb{R}^n$, constants $\kappa_0, \kappa_x, \kappa_u$ corresponding to Assumption 9, find a PWA control system $\mathcal{G}$, where an upper-bound for $\mathcal{M}$ is given, such that $\mathcal{F} \subseteq \mathcal{G}$ and minimize $\alpha(W_1, \cdots, W_{\mathcal{M}})$, where $\alpha : \left(2^{\mathbb{R}^n}\right)^{\mathcal{M}} \to \mathbb{R}$ is a cost function that promotes smaller disturbance sets..

The reason that we add a cost criteria to Subproblem 1 is that a PWA $\mathcal{G}$ that simulates $\mathcal{F}$ is not unique. In fact, by making the disturbance sets sufficiently large, $\mathcal{G}$ can simulate any system. Having large disturbance sets is undesirable for control synthesis. For computational purposes, we focus on simple forms of disturbance sets and α. For example, we let $W_i, i = 1, \cdots, M$, to be axis-aligned hyper-boxes and α is designed to penalize the length of their sides.

9.3.2 PWA Fitting

Here we simultaneously find values representing the sets X_i, and matrices A_i, B_i, c_i in (7.17) by solving an optimization problem. We find sets W_i afterwards, Consider $K \in \mathbb{N}_+$ hyperplanes - which we refer to as *guards*:

$$h_i^T x + 1 = 0, \tag{9.8}$$

where $h_i \in \mathbb{R}^n, i = i = 1, \cdots, K$. These guards partition X into at most 2^K polyhedral sets with disjoint interiors:

$$X_k = \left\{ x \in X \middle| h_i^T x + 1 \overset{(k,i)}{\sim} 0, i = 1, \cdots, K \right\}, k = 1, \cdots, 2^K, \tag{9.9}$$

where $\sim: \mathbb{N}_+ \times \mathbb{N}_+ \to \{\leq, \geq\}$ is defined in the following way: $\overset{(k,i)}{\sim}$ is $\geq$ if the i^{th} digit from the right of k written in binary numeral system is one, and $\leq$ otherwise. For example, for $k = 5$, we have $5 = (101)_2$. Hence $\overset{(5,1)}{\sim} = \geq$, $\overset{(5,2)}{\sim} = \leq$ and $\overset{(5,3)}{\sim} = \geq$. We interpret further digits on the left as zero: $\overset{(5,i)}{\sim} = \leq, i > 3$. Not all the sets in (9.9) may not be non-empty. The set of decision parameters are

$$\Theta := \left\{ \{A_k, B_k, c_k,\}_{i=1,\cdots,2^K}, \{h_i,\}_{i=1,\cdots,K} \right\}$$

The best values for Θ are found using the following optimization problem:

$$\begin{array}{rll} \Theta^* = & \underset{\Theta}{\arg\min} & \delta \\ & \text{subject to} & \left| x_j^+ - (A_i x_j + B_k u_j + c_k) \right| z_j^k \leq \delta 1_n, \\ & & z_j^k = 1 \Leftrightarrow x_j \in X_k, z_j^k = 0 \Leftrightarrow x_j \notin X_k, \\ & & X_k, k = 1, \cdots, 2^K, \text{ given by (9.9), (9.8),} \\ & & j = 1, \cdots, N, i = 1, \cdots, K, \end{array} \tag{9.10}$$

where δ is error. Eq. (9.10) finds the best PWA fit (additive disturbances are not yet considered) such that all data points are within $\delta\mathcal{B}_\infty$ of their predications.

Since all the values and sets are bounded, we show that (9.10) can be cast as a MILP problem using the *big-M method.* First, the expression $\left|x_j^+ - (A_i x_j + B_k u_j + c_k)\right| z_j^k \leq \delta 1_n$ is equivalent to the following set of constraints:

$$\begin{cases} -M(1-z_j^k) \leq x_j^+ - A_i x_j - B_k u_j - c_k + \delta_j^k \leq M(1-z_j^k), \\ -\delta 1_n \leq \delta_j^k \leq \delta 1_n, \end{cases}$$

where $\delta_j^k \in \mathbb{R}^n$ are auxilary variables and M is a sufficiently large positive number. Second, we need to capture (9.9). We define K binaries for each data point, introducing a total of NK binary variables. We have

$$b_j^i = \mathcal{I}(h_i^T x_j + 1 \geq 0) \Leftrightarrow \begin{cases} h_i^T x_j + 1 \leq M b_j^i, \\ h_i^T x_j + 1 \geq -M(1-b_j^i). \end{cases}$$

The relation $z_j^k = \mathcal{I}(x_j^k \in X_k)$ can be converted to: $z_j^k = \bigwedge_{i=1}^{K} \mathrm{Sgn}(k,i,b_j^i)$, where $\mathrm{Sgn}(k,i,b_j^i) = b_j^i$ if $\overset{k,i}{\sim} = \geq$, and is $\neg b_j^i$ otherwise. The conjunction and negation applying to integers is interpreted in a Boolean sense - e.g., $1 \wedge 0 = 0, 1 \wedge 1 = 0, \neg 1 = 0$, etc. The variables $z_j^k \in [0,1], j = 1, \cdots, N, k = 1, \cdots, 2^K$, are declared as continuous variables, but always take binary variables. Encoding Boolean operations as mixed-integer constraints is a standard procedure (see, e.g., [Bemporad and Morari, 1999]) and further details are not presented here.

9.3.3 Adjusting Disturbances

Now we find disturbance in (7.17) such that $\mathcal{G}$ simulates $\mathcal{F}$. We let every disturbance set to be represented by a hyper-box symmetric around the origin. The length of the side in q^{th} cartesian direction of W_i, $q = 1, \cdots, n$, is found using the following optimization problem:

$$\begin{array}{rll} {\eta_i}^*_{[q]} = & \underset{x,u,\eta}{\text{argmax}} & \left|e_{[q]}^T \eta_i\right| \\ & \text{subject to} & x^+ = x_j^+ + k_0\beta_j^0 + k_x\beta_j^x + k_u\beta_j^u \\ & & \beta_j^x \in \|x - x_j\|_p \mathcal{B}_p, \\ & & \beta_j^u \in \|u - u_j\|_p \mathcal{B}_p, \|\beta_j^0\|_\infty \in \mathcal{B}_\infty \\ & & x^+ = A_i x + B_i u + c_i + \eta_i \\ & & x \in X_i, u \in U, j = 1, \cdots, N. \end{array} \tag{9.11}$$

We let:

$$W_i = \{w \in \mathbb{R}^n | -\eta_i^* \leq w \leq \eta_i^*\} \tag{9.12}$$

Theorem 14. The PWA system constructed from solutions of (9.10), and (9.11), (9.12), $i = 1, \cdots, 2^K$, simulates $\mathcal{F}$.

Proof. Eq. (9.11) gives the farthest point (in the q^{th} direction) in $\overline{\mathcal{F}}$ from the model given by (9.10). Since the worst-case distances are considered in each direction, the sets in (9.12) establish $\overline{\mathcal{F}} \subseteq \mathcal{G}$. □

The following simple example shows that the distribution of data affects model identification.

Example 21. Consider fitting a line (a single mode PWA model) to one-dimensional data set of $N = 4$ points, shown by red circles in Fig. 9·1. There are no control inputs. The box represents X. The relation $\overline{\mathcal{F}}$ and the learned affine $\mathcal{G}$ are shown by cyan and yellow regions, respectively. In the figure to the left, points are closer to the edges of X, creating a vacuum of data in the center of X, which leads to large non-determinism in $\mathcal{G}$. On the other hand, data points on the right are more evenly-spaced, leading to less non-deterministic $\mathcal{G}$.

Remark 6. Overfitting arises when few data is used to decide about a large number of variables. Since our model identification is set-valued and takes into account

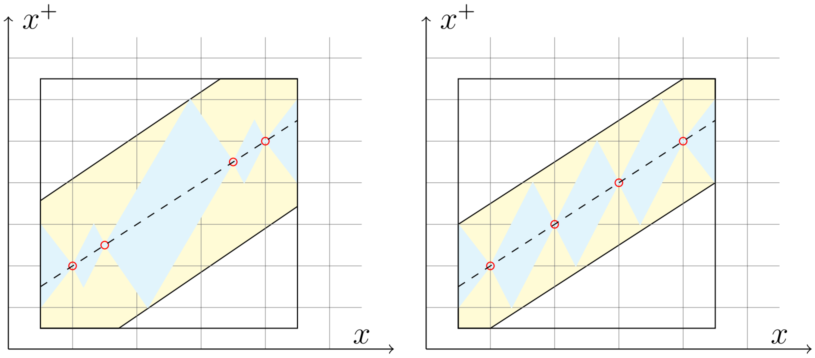


Figure 9·1: Example 21: The magnitude of non-determinism may depend on the distribution of data.

all "other" possible data, overfitting is never an issue. Even with one single data point, Definition (51) defines a set-valued model $\overline{\mathcal{F}}$ with very large non-determinism - virtually useless for control synthesis.

9.3.4 Computational Aspects

Eq. (9.10) is a MILP problem with NK binary variables and $\mathcal{O}(K \max(n^2, nm, Nn))$ continuous variables. The worst-case complexity scales exponentially with the number the number of binaries and polynomially with the number of continuous variables. MILPs finds the global optimum. For large problems, we may terminate a MILP early to obtain a suboptimal solution - after a feasible integer solution is found.

Eq. (9.11) is a non-convex optimization problem, that can also can be cast as a MILP when $p = 1, \infty$, where the number of its binary variables scales by $\mathcal{O}(\max(Nn, Nm))$ and the number of continuous variables and constraints scale similarly to (9.10). Validity of the continuity constants is implicit in the feasibility of (9.11) - if k_0, k_x, k_u are under-estimated then (9.11) may become infeasible.

In practice, exact solutions for (9.11) may unachievable. Heuristics can be used to over-approximate the sets in (9.12). The value of δ found in (9.10) provides an lower bound for the sides of disturbance sets. As (9.11) is sensitive to the number of data points, increasing the density of data decreases uncertainties in a linear fashion - as illustrated in Fig. 9·1. Thus, a good heuristic is to solve (9.11) for small data sets and accordingly adjust the values for larger data sets.

9.4 Example

Example 22. We adopt a controlled version of the genetic toggle switch model in [Gardner et al., 2000]. The control system $\mathcal{F}$ is constructed from:

$$\begin{aligned} x_{[1]}^{+} &\in \left\{ \tfrac{4}{5}x_{[1]} + \tfrac{1}{2(1+x_{[2]}^{3})} + e^{-\frac{1}{5}x_{[1]}[t]}u_{[1]} \right\} \oplus \tfrac{1}{20}[-1,1] \\ x_{[2]}^{+} &\in \left\{ \tfrac{4}{5}x_{[2]} + \tfrac{1}{2(1+x_{[1]}^{2})} + e^{-\frac{1}{5}x_{[2]}[t]}u_{[2]} \right\} \oplus \tfrac{1}{20}[-1,1], \end{aligned} \tag{9.13}$$

where $X = [0,1]^2$ and $U = [-1,1]^2$. The state components represent gene repressor concentrations. The goal is to oscillate the concentration levels. The STL specification is:

$$\varphi = \mathbf{G}_{[0,\infty]}\neg R_0 \wedge \mathbf{F}_{[0,10]}R_1 \wedge \mathbf{G}_{[10,\infty]}(\mathbf{F}_{[0,10]}R_1 \wedge \mathbf{F}_{[0,10]}R_2), \tag{9.14}$$

where R_0, R_1, R_2 are conjunctions of predicates that characterize the regions illustrated in Fig. 9·2 (left). Specification (9.14) states that "within 10 time units, R_1 has to be visited. Afterwards, R_1 and R_2 must be visited infinitely often while the time between two consecutive visits is never greater than 10. Also, always avoid R_0."

Note that (9.13) is unknown to the controller. We sampled 400 evenly-distributed data points from $X \times U$. The number of guards is set to $K = 2$. We used (9.10) - the computation time was about 5 minutes using Gurobi MILP solver - and obtained the following guards and affine modes:

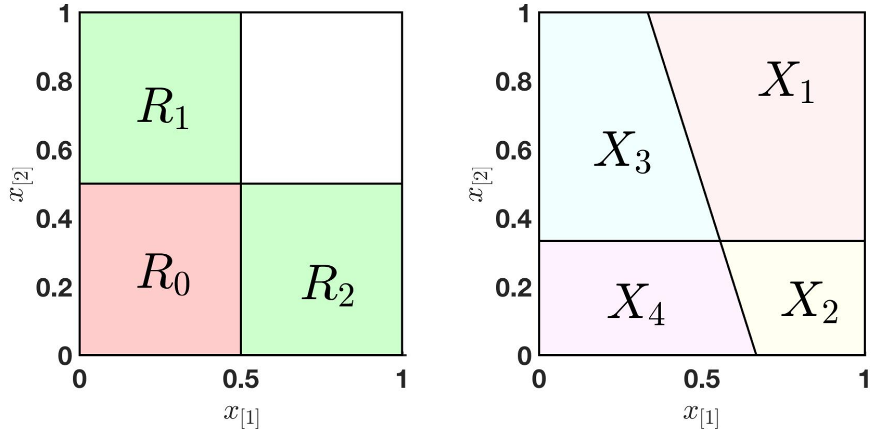


Figure 9·2: Example 22: (Left): Regions of interest. (Right): Computed polyhedral partition from (9.10).

$$h_1 = (-1.5, -0.5)^T, h_2 = (0, -3.0)^T, \delta^* = 0.07$$

$$A_1 = \begin{pmatrix} 0.75 & -0.38 \\ -0.3 & 0.78 \end{pmatrix}, B_1 = \begin{pmatrix} 0.85 & 0.0 \\ -0.0 & 0.85 \end{pmatrix}, c_1 = \begin{pmatrix} 0.66 \\ 0.56 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 0.79 & -0.04 \\ -0.35 & 0.73 \end{pmatrix}, B_2 = \begin{pmatrix} 0.85 & -0.01 \\ 0.01 & 0.97 \end{pmatrix}, c_2 = \begin{pmatrix} 0.5 \\ 0.6 \end{pmatrix},$$

$$A_3 = \begin{pmatrix} 0.76 & -0.4 \\ -0.17 & 0.83 \end{pmatrix}, B_3 = \begin{pmatrix} 0.98 & 0.01 \\ -0.01 & 0.86 \end{pmatrix}, c_3 = \begin{pmatrix} 0.66 \\ 0.48 \end{pmatrix},$$

$$A_4 = \begin{pmatrix} 0.87 & -0.01 \\ -0.15 & 0.81 \end{pmatrix}, B_4 = \begin{pmatrix} 0.96 & -0.0 \\ 0.01 & 0.97 \end{pmatrix}, c_4 = \begin{pmatrix} 0.49 \\ 0.5 \end{pmatrix},$$

The polyhedral partitions are shown in Fig. 9·2 (right). We set $k_0 = 0.05, k_x = 1.5, k_u = 1$, which is verified both against (9.13) and data. Using the procedure outline in Sec. 9.3.3, we solved (9.11) 8 times (2 for each mode) - the computation times were about 15 seconds for each case - and obtained:

$$W_1 = [-0.25, 0.25] \times [-0.33, 0.33],\ W_2 = [-0.13, 0.13] \times [-0.28, 0.28],$$
$$W_3 = [-0.28, 0.28] \times [-0.21, 0.21],\ W_4 = [-0.17, 0.17] \times [-0.23, 0.23].$$

The tube was computed for $\Gamma = 1$. We found $T = [-0.280.28] \times [-0.33, 0.33]$ and

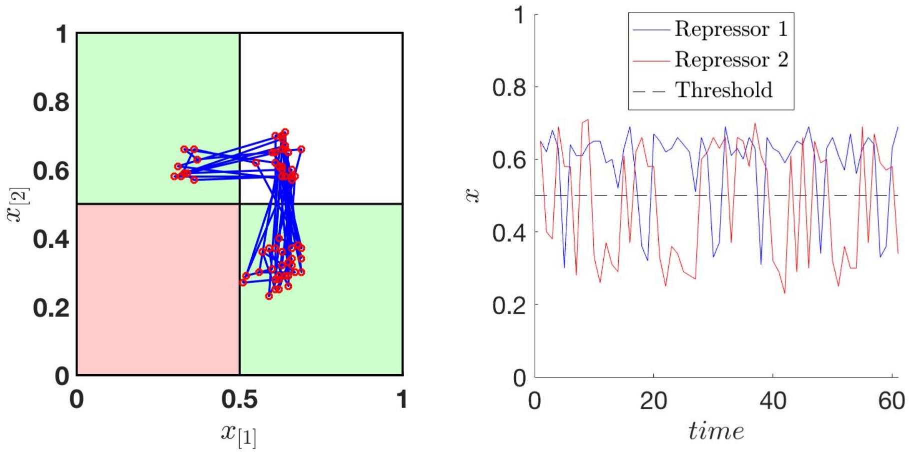


Figure 9·3: Example 22 a sample closed-loop trajectory

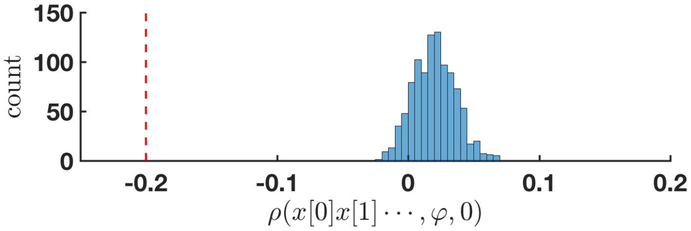


Figure 9·4: Example 22 histogram of STL scores.

$T_u = 0.39\mathbb{B}_\infty$. The nominal trajectory was computed for $x_0 = (0.65, 0.65)^T$, $\tau_0 = 10, \tau_p = 40$, took 4 seconds to solve, and resulted in $\epsilon^* = 0.13$. *Thus, we obtain the guarantee that all the closed-loop trajectories of* (9.13) *are in the* $-0.2(= 0.13-0.33)$-language set. A sample trajectory of (9.13) is shown in Fig. 9·3. In simulations, we observed STL scores were always greater than the guarantee (-0.2). A histogram of STL scores for 1000 simulations of 60 time steps is shown in Fig. 9·4. The differences between the worst-case theoretical STL score (denoted by red line) and the simulated ones highlight the conservativeness of the methods in this chapter. All simulations were performed by sampling disturbances in (9.13) uniformly from their respective domains.

Chapter 10

Conclusions

10.1 Summary of the thesis

In this thesis, we developed tools for formal synthesis of control strategies for systems with bounded uncertainties. We focused on linear time properties expressed as signal temporal logic. First, we developed a tube-based framework for optimal control of disturbed piecewise affine systems. The method was conservative and heavily relied on computational techniques for mixed-integer programs for both nominal trajectory design and tube design. However, the implementation requires a modest convex program to be solved online.

Regarding the mentioned difficulties, the focus was shifted toward systems and requirements with specific structures. We extensively studied control synthesis for a class of monotone systems, which was shown to include macroscopic traffic models. Using monotonicity, we showed that open-loop control policies are sufficient and almost necessary for satisfying a broad range of temporal logic specifications. The policies were synthesized quite efficiently for reasonably large systems. However, optimal control was not very scalable. We pursued contract-based design for distributed optimal control. Two approaches were outlined. First, the contracts were designed by decomposing robust control invariant sets. Second, compositional synthesis was used to find a rich library of contracts, where the best one was chosen in response to the real-time conditions. Its usefulness was demonstrated through examples of mixed urban-freeway networks. The second class of special systems was networked linear systems with polyhedral constraints. We designed distributed set-invariance policies subject to communication constraints. We also provided a method to find sparsest communication topologies such that invariance inducing policies were possible.

Finally, we focused on scenarios that the model is initially unknown. We developed learning-based techniques. First, we assumed that there exists a family of candidate models from which the controller infers the true one through recursive observations. Our method was complete for finite systems, whereas its usefulness for infinite systems was dependent on the quality of the finite abstractions. Second, we discussed constructing set-valued hybrid systems from input-output data. By assuming knowledge of continuity bounds, in a Lipschitz sense, we constructed models that were guaranteed to contain all the possible behaviors of the concrete system. Therefore, we obtained formal guarantees for the performance of applying the control policy designed for the learned PWA model to the concrete system.

10.2 Future Directions

Formal synthesis for automation of large-scale cyber-physical systems is still a largely open area [Seshia et al., 2016]. Even though the principles of contract-based design and compositional synthesis are not strictly dependent on the type of the system, applying them to systems without linearity or monotonicity is very difficult. Future research should develop tools for this matter.

Besides approaches discussed in this thesis, a possible opportunity to ease computational complexity is using event-driven-based formulation [Cassandras and Lafortune, 2009]. These approaches have the advantage that the complexity grows by the size of event space, which is usually much smaller than state space. However, it is still not clear how to combine optimal discrete-event control with formal specifications in systems with infinitely-many states.

Another area that is largely open is combining control theoretic aspects of artificial intelligence and formal methods. This thesis provided two approaches in this domain. Specifically, Chapter 8 provided the complete solution to learning-based control for finite systems. However, its scalability was poor. The data-driven model identification approach needed a lot of data to be successful. There are paradigms in machine learning, such as transfer learning [Raina et al., 2007], which are largely unexplored in control theory. Future research will focus on new ways for learning-based control with formal guarantees.

Appendix A

Appendix

A.1 Bounded-Global Formula

Theorem 15. Let $\mathbb{S}$ be the set of all STL formulas that can be written in the form:

$$\phi = \bigvee_{i=1}^{n_\phi} \varphi_{b,i} \wedge \mathbf{G}_{[\Delta_i,\infty]}\varphi_{g,i}, \tag{A.1}$$

where $\varphi_{b,i}, \Delta_i \geq h^{\varphi_{b,i}}, \varphi_{g,i}, i = 1, \cdots, n_\phi$, are bounded STL formulas. Then $\mathbb{S}$ is a subset of safety STL formulas that is closed under STL syntax with bounded temporal operators.

Proof. First, a quick inspection of (A.1) verifies that it is a safety STL formula. A predicate π is a bounded formula (with zero horizon) and is a special case of (A.1), hence $\pi \in \mathbb{S}$.

We also have the following property that relaxes the form in (A.1): *For all bounded STL formulas* φ_1, φ_2*, we have* $\varphi_1 \wedge \mathbf{G}_{[\Gamma,\infty)}\varphi_2 \in \mathbb{S}$*,* $\forall \Gamma \in \mathbb{N}$*.* Proof: The case for $\Gamma \geq h^{\varphi_1}$ is already in the form (A.1) with $n_\phi = 1$. If $\Gamma < h^{\varphi_1}$, we write $\mathbf{G}_{[\Gamma,\infty)}\varphi_2 = \mathbf{G}_{[\Gamma,h^{\varphi_1}]}\varphi_2 \wedge \mathbf{G}_{[h^{\varphi_1},\infty)}\varphi_2$. Now, define $\varphi_1 \wedge \mathbf{G}_{[\Gamma,h^{\varphi_1}]}\varphi_2$ as the new bounded formula and retain the form in (A.1) with $n_\phi = 1$.

We show that $\mathbb{S}$ is closed under STL syntax with bounded operators. The distributivity properties of Boolean connectives and temporal operators (see, e.g., [Huth, Michael and Ryan, Mark, 2004]) imply that: $\phi_1 \vee (\phi_2 \wedge \phi_3) = (\phi_1 \vee \phi_2) \wedge (\phi_2 \vee \phi_3)$, $\phi_1 \wedge (\phi_2 \vee \phi_3) = (\phi_1 \wedge \phi_2) \vee (\phi_2 \wedge \phi_3)$, $\mathbf{F}_I(\phi_1 \vee \phi_2) = (\mathbf{F}_I \phi_1) \vee (\mathbf{F}_I \phi_2)$, and $\mathbf{G}_I(\phi_1 \wedge \phi_2) = (\mathbf{G}_I \phi_1) \wedge (\mathbf{G}_I \phi_2)$, where ϕ_1, ϕ_2, ϕ_3 are temporal logic formulas and I is an interval.

1. $\phi_1, \phi_2 \in \mathbb{S} \Rightarrow \phi_1 \wedge \phi_2 \in \mathbb{S}, \phi_1 \vee \phi_2 \in \mathbb{S}$: this result easily follows from the distributivity properties of Boolean connectives mentioned above.

2. $\phi \in \mathbb{S} \Rightarrow \mathbf{F}_{\{t\}} \phi \in \mathbb{S}$: we use $\mathbf{F}_{\{t\}} \mathbf{G}_{[a,b]} = \mathbf{G}_{[t+a,t+b]}$ and distributivity to have (note that $\mathbf{F}_{\{t\}} = \mathbf{G}_{\{t\}}$)

$$\begin{aligned} & \mathbf{F}_{\{t\}}(\bigvee_{i=1}^{n_\phi}(\varphi_{b,i} \wedge \mathbf{G}_{[\Gamma_i,\infty]} \varphi_{g,i})) \\ = & \bigvee_{i=1}^{n_\phi}(\mathbf{F}_{\{t\}} \varphi_{b,i} \wedge \mathbf{G}_{[t+\Gamma_i,\infty]} \varphi_{g,i}). \end{aligned}$$

   Introducing $\mathbf{F}_{\{t\}} \varphi_{b,i}, i = 1, \cdots, n_\phi$, as new bounded STL formulas leads to the form in (A.1).

3. $\phi \in \mathbb{S} \Rightarrow \mathbf{F}_{[a,b]} \phi \in \mathbb{S}, \mathbf{G}_{[a,b]} \phi \in \mathbb{F}$: use $\mathbf{F}_{[a,b]} = \bigvee_{t \in [a,b]} \mathbf{F}_{\{t\}}$ and $\mathbf{G}_{[a,b]} = \bigwedge_{t \in [a,b]} \mathbf{F}_{\{t\}}$ to convert temporal operators to Boolean connectives.

4. $\phi_1, \phi_2 \in \mathbb{S} \Rightarrow \phi_1 \mathbf{U}_{[a,b]} \phi_2 \in \mathbb{S}$: use the STL semantics Definition 7 to substitute the bounded "until" operator using bounded "eventually" and bounded "always" operators:

$$\phi_1 \mathbf{U}_{[a,b]} \phi_2 = \bigvee_{t \in [a,b]} (\mathbf{G}_{[a,t] \phi_1} \wedge \mathbf{F}_{\{t\}} \phi_2).$$

Example 23. The "reach and stay" formula $\mathbf{F}_I \mathbf{G}_{[0,\infty)} \varphi$, where φ is a bounded formula, is equivalent to $\bigvee_{t \in I} \mathbf{G}_{[t,\infty)} \varphi$.

Remark 7. What remains to show that $\mathbb{S}$ is equivalent to the set of all safety STL formulas is having that $\phi \in \mathbb{S} \Rightarrow \mathbf{G}_{[\Gamma',\infty)} \phi \in \mathbb{S}, \forall \Gamma \in \mathbb{N}$, which is not true by restricting n_ϕ in (A.1) to be finite. Formulas that involve nested unbounded "always" operator and can not be further simplified, such as $\mathbf{G}_{[\Gamma',\infty)}(\varphi_1 \vee \mathbf{G}_{[\Gamma,\infty)}\varphi_2)$, are rarely encountered in applications.

$\square$

References

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

Abate, A., Prandini, M., Lygeros, J., and Sastry, S. (2008). Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. *Automatica*, 44(11):2724–2734.

Aksaray, D., Jones, A., Kong, Z., Schwager, M., and Belta, C. (2016). Q-learning for robust satisfaction of signal temporal logic specifications. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6565–6570. IEEE.

Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., and Topcu, U. (2017). Safe reinforcement learning via shielding. *arXiv preprint arXiv:1708.08611*.

Alur, R. and Dill, D. L. (1994). A theory of timed automata. *Theoretical computer science*, 126(2):183–235.

Alur, R., Henzinger, T. A., Lafferriere, G., and Pappas, G. J. (2000). Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(7):971–984.

Alur, R., Moarref, S., and Topcu, U. (2016). Compositional synthesis with parametric reactive controllers. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, pages 215–224. ACM.

Alur, R. and Singhania, N. (2014). Precise piecewise affine models from input-output data. In *2014 International Conference on Embedded Software (EMSOFT)*, pages 1–10. IEEE.

Anderson, B., Brinsmead, T., Liberzon, D., and Stephen Morse, A. (2001). Multiple model adaptive control with safe switching. *International journal of adaptive control and signal processing*, 15(5):445–470.

Angeli, D. and Sontag, E. D. (2003). Monotone control systems. *IEEE Transactions on Automatic Control*, 48(10):1684–1698.

Arastoo, R., Bahavarnia, M., Kothare, M. V., and Motee, N. (2016). Closed-loop feedback sparsification under parametric uncertainties. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 123–128. IEEE.

Aswani, A., Gonzalez, H., Sastry, S. S., and Tomlin, C. (2013). Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226.

Bacchus, F. and Kabanza, F. (2000). Using temporal logics to express search control knowledge for planning. *Artificial Intelligence*, 116(1-2):123–191.

Baier, C. and Katoen, J.-P. (2008). *Principles of model checking.* MIT press Cambridge.

Belta, C., Yordanov, B., and Aydin Gol, E. (2017). *Formal Methods for Discrete-Time Dynamical Systems.* Springer.

Bemporad, A., Garulli, A., Paoletti, S., and Vicino, A. (2005). A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, 50(10):1567–1580.

Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427.

Bertsekas, D. (1972). Infinite time reachability of state-space regions by using feedback control. *IEEE Transactions on Automatic Control*, 17(5):604–613.

Bertsekas, D. P. (1995). *Dynamic programming and optimal control*, volume 1 No. 2. Athena Scientific Belmont, MA.

Bertsekas, D. P. and Rhodes, I. B. (1971). On the Minimax Reachability of Target Sets and Target Tubes. *Automatica*, 7(2):233–247.

Blanchini, F. (1999). Set invariance in control–a survey. *Automatica*, 35(11):1747–1767.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122.

Calliess, J. P. (2017). Lipschitz optimisation for lipschitz interpolation. In *2017 American Control Conference (ACC)*, pages 3141–3146. IEEE.

Cassandras, C. G. and Lafortune, S. (2009). *Introduction to discrete event systems.* Springer Science & Business Media.

Chatterjee, K., Chmelík, M., and Tracol, M. (2016). What is decidable about partially observable markov decision processes with ω-regular objectives. *Journal of Computer and System Sciences*, 82(5):878–911.

Chatterjee, K. and Henzinger, T. A. (2012). A survey of stochastic ω-regular games. *Journal of Computer and System Sciences*, 78(2):394–413.

Como, G., Lovisari, E., and Savla, K. (2014). Throughput optimality and overload behavior of dynamical flow networks under monotone distributed routing. *IEEE Transactions on Control of Network Systems*, 5870(c):1–1.

Como, G., Lovisari, E., and Savla, K. (2016). Convexity and robustness of dynamic network traffic assignment for control of freeway networks. *IFAC-PapersOnLine*, 49(3):335–340.

Conte, C., Voellmy, N. R., Zeilinger, M. N., Morari, M., and Jones, C. N. (2012). Distributed synthesis and control of constrained linear systems. In *2012 American Control Conference (ACC)*, pages 6017–6022. IEEE.

Coogan, S. and Arcak, M. (2014). Dynamical properties of a compartmental model for traffic networks. In *2014 American Control Conference (ACC)*, pages 2511–2516. IEEE.

Coogan, S. and Arcak, M. (2015). Efficient finite abstraction of mixed monotone systems. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control, 58-67, 2015*, pages 58–67. ACM.

Coogan, S., Arcak, M., and Kurzhanskiy, A. a. (2016a). On the Mixed Monotonicity of FIFO Traffic Flow Models. In *55th IEEE Conference on Decision and Control*. IEEE.

Coogan, S., Gol, E. A., Arcak, M., and Belta, C. (2016b). Traffic network control from temporal logic specifications. *IEEE Transactions on Control of Network Systems*, 3(2):162–172.

Cortés, J. (2006). Finite-time convergent gradient flows with applications to network consensus. *Automatica*, 42(11):1993–2000.

Dallal, E. and Tabuada, P. (2015). On compositional symbolic controller synthesis inspired by small-gain theorems. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 6133–6138. IEEE.

Davey, B. A. and Priestley, H. A. (2002). *Introduction to lattices and order*. Cambridge university press.

De Leenheer, P. and Aeyels, D. (2001). Stabilization of positive linear systems. *Systems & Control Letters*, 44(4):259–271.

De Schutter, B. and Van den Boom, T. (2001). Model predictive control for max-min-plus-scaling systems. In *Proceedings of the 2001 American Control Conference*, volume 1, pages 319–324. IEEE.

di Bernardo, M., Montanaro, U., Ortega, R., and Santini, S. (2016). Extended hybrid model reference adaptive control of piecewise affine systems. *Nonlinear Analysis: Hybrid Systems*, 21:11–21.

di Bernardo, M., Montanaro, U., and Santini, S. (2013). Hybrid model reference adaptive control of piecewise affine systems. *IEEE Transactions on Automatic Control*, 58(2):304–316.

Dokhanchi, A., Hoxha, B., and Fainekos, G. (2014). On-line monitoring for temporal logic robustness. In *Runtime Verification*, pages 1–20. Springer.

Ehlers, R., Moarref, S., and Topcu, U. (2016). Risk-averse control of markov decision processes with ω-regular objectives. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 426–433. IEEE.

Emerson, E. A. (1990). Temporal and modal logic. In *Handbook of Theoretical Computer Science (Vol. B)*, chapter Temporal and Modal Logic, pages 995–1072. MIT Press, Cambridge, MA, USA.

Fainekos, G. E., Kress-Gazit, H., and Pappas, G. J. (2005). Temporal logic motion planning for mobile robots. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2020–2025. IEEE.

Farahani, S. S., Raman, V., and Murray, R. M. (2015). Robust model predictive control for signal temporal logic synthesis. *IFAC-PapersOnLine*, 48(27):323–328.

Fardad, M. and Jovanovic, M. R. (2014). On the design of optimal structured and sparse feedback gains via sequential convex programming. In *2014 American Control Conference (ACC)*, pages 2426–2431. IEEE.

Fattahi, S., Fazelnia, G., and Lavaei, J. (2015). Transformation of optimal centralized controllers into near-global static distributed controllers. In *2015 IEEE 54th Annual Conference on Decision and Control (CDC)*, pages 4915–4922. IEEE.

Fattahi, S., Lavaei, J., and Arcak, M. (2017). A scalable method for designing distributed controllers for systems with unknown initial states. In *2017 IEEE 56th Conference on Decision and Control (CDC)*, pages 4739–4746. IEEE.

Fazelnia, G., Madani, R., Kalbat, A., and Lavaei, J. (2017). Convex relaxation for optimal distributed control problems. *IEEE Transactions on Automatic Control*, 62(1):206–221.

Fernandez, J.-C. and Mounier, L. (1991). “on the fly” verification of behavioural equivalences and preorders. In *International Conference on Computer Aided Verification*, pages 181–191. Springer.

Fleck, J. L., Cassandras, C. G., and Geng, Y. (2016). Adaptive quasi-dynamic traffic light control. *IEEE Transactions on Control Systems Technology*, 24(3):830–842.

Fu, J. and Topcu, U. (2015). Computational methods for stochastic control with metric interval temporal logic specifications. In *2015 IEEE 54th Annual Conference on Decision and Control (CDC)*, pages 7440–7447. IEEE.

Furieri, L. and Kamgarpour, M. (2017). The value of communication in synthesizing controllers given an information structure. *arXiv preprint arXiv:1711.05324*.

Gardner, T. S., Cantor, C. R., and Collins, J. J. (2000). Construction of a genetic toggle switch in escherichia coli. *Nature*, 403(6767):339.

Gartner, N. (1983). *OPAC: A Demand-responsive Strategy for Traffic Signal Control*. Transportation Research Board, National Research Council.

Geng, Y. and Cassandras, C. G. (2015). Multi-intersection traffic light control with blocking. *Discrete Event Dynamic Systems*, 25(1-2):7–30.

Geroliminis, N. and Daganzo, C. F. (2008). Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings. *Transportation Research Part B: Methodological*, 42(9):759–770.

Ghaemi, R. and Del Vecchio, D. (2014). Control for safety specifications of systems with imperfect information on a partial order. *IEEE Transactions on Automatic Control*, 59(4):982–995.

Ghosh, S., Sadigh, D., Nuzzo, P., Raman, V., Donzé, A., Sangiovanni-Vincentelli, A. L., Sastry, S. S., and Seshia, S. A. (2016). Diagnosis and repair for synthesis from signal temporal logic specifications. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, pages 31–40. ACM.

Gol, E. A. and Belta, C. (2014). An additive cost approach to optimal Temporal Logic control. In *2014 American Control Conference (ACC)*, pages 1769–1774. IEEE.

Gol, E. A., Lazar, M., and Belta, C. (2014). Language-Guided Controller Design for Linear Systems. *IEEE Transactions on Automatic Control*, 59(5):1163–1176.

Grädel, E., Thomas, W., and Wilke, T. (2002). Automata, logics, and infinite games: a guide to current research. *Lecture Notes in Computer Science, Vol. 2500.*

Gregoire, J., Qian, X., Frazzoli, E., De La Fortelle, A., and Wongpiromsarn, T. (2015). Capacity-aware backpressure traffic signal control. *IEEE Transactions on Control of Network Systems*, 2(2):164–173.

Gurobi Optimization, I. (2016). Gurobi optimizer reference manual.

Haddad, W. M., Chellaboina, V., and Hui, Q. (2010). *Nonnegative and compartmental dynamical systems*. Princeton University Press.

Hafner, M. R. and Del Vecchio, D. (2011). Computational tools for the safety control of a class of piecewise continuous systems with imperfect information on a partial order. *SIAM Journal on Control and Optimization*, 49(6):2463–2493.

Heemels, W., Schutter, B. D., and Bemporad, A. (2001). Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091.

Henzinger, T. A. (2000). The theory of hybrid automata. In *Verification of Digital and Hybrid Systems*, pages 265–292. Springer.

Henzinger, T. A., Qadeer, S., and Rajamani, S. K. (1998). You assume, we guarantee: Methodology and case studies. In *International Conference on Computer Aided Verification*, pages 440–451. Springer.

Hespanha, J. P., Liberzon, D., and Morse, A. S. (2003). Overcoming the limitations of adaptive control by means of logic-based switching. *Systems & control letters*, 49(1):49–65.

Hirsch, M. W. (1985). Systems of differential equations that are competitive or cooperative II: Convergence almost everywhere. *SIAM Journal on Mathematical Analysis*, 16(3):423–439.

Hirsch, Morris W, Smith, S. H. (2005). Monotone maps: a review. *Journal of Difference Equations and Applications*, 4-5:379–398.

Hunt, P., Robertson, D., Bretherton, R., and Winton, R. (1981). Scoot-a traffic responsive method of coordinating signals. Technical report, Transport and Road Research Laboratory.

Huth, Michael and Ryan, Mark (2004). *Logic in Computer Science: Modelling and Reasoning About Systems*. Cambridge University Press.

Jones, C., Kerrigan, E. C., and Maciejowski, J. (2004). Equality set projection: A new algorithm for the projection of polytopes in halfspace representation. Technical report, Cambridge University Engineering Dept.

Karaman, S., Sanfelice, R. G., and Frazzoli, E. (2008). Optimal control of Mixed Logical Dynamical systems with Linear Temporal Logic specifications. In *2008 47th IEEE Conference on Decision and Control*, pages 2117–2122. IEEE.

Kerrigan, E. C. (2000). *Robust Constraint Satisfaction: Invariant Sets and Predictive Control*. PhD thesis, University of Cambridge, Department of Engineering.

Kerrigan, E. C. and Maciejowski, J. M. (2001). Robust feasibility in model predictive control: Necessary and sufficient conditions. In *Proceedings of IEEE Conference ond Decision and Control*, volume 1, pages 728–733. IEEE.

Kim, E. S., Arcak, M., and Seshia, S. A. (2015). Compositional controller synthesis for vehicular traffic networks. In *2015 IEEE 54th Annual Conference on Decision and Control (CDC)*, pages 6165–6171. IEEE.

Kim, E. S., Arcak, M., and Seshia, S. A. (2016). Directed Specifications and Assumption Mining for Monotone Dynamical Systems. In *19th ACM International Conference on Hybrid Systems: Computation and Control (HSCC)*, Vienna, Austria.

Kim, E. S., Arcak, M., and Seshia, S. A. (2017a). A small gain theorem for parametric assume-guarantee contracts. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, pages 207–216. ACM.

Kim, E. S., Arcak, M., and Seshia, S. A. (2017b). Symbolic control design for monotone systems with directed specifications. *Automatica*, 83:10–19.

Kim, E. S., Sadraddini, S., Belta, C., Arcak, M., and Seshia, S. A. (2017c). Dynamic contracts for distributed temporal logic control of traffic networks. In *2017 IEEE 56th Conference on Decision and Control (CDC)*, pages 3640–3645. IEEE.

Klein, J. and Baier, C. (2006). Experiments with deterministic ω-automata for formulas of linear temporal logic. *Theoretical Computer Science*, 363(2):182–195.

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

Koehler, S., Mehr, N., Horowitz, R., and Borrelli, F. (2016). Stable hybrid model predictive control for ramp metering. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1083–1088. IEEE.

Koymans, R. (1990). Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299.

Lahijanian, M., Almagor, S., Fried, D., Kavraki, L. E., and Vardi, M. Y. (2015). This Time the Robot Settles for a Cost: A Quantitative Approach to Temporal Logic Planning with Partial Satisfaction. In *The Twenty-Ninth AAAI Conference (AAAI-15)*, Austin, TX. AAAI.

Lahijanian, M., Wasniewski, J., Andersson, S. B., and Belta, C. (2010). Motion planning and control from temporal logic specifications with probabilistic satisfaction guarantees. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3227–3232. IEEE.

Lamperski, A. and Lessard, L. (2015). Optimal decentralized state-feedback control with sparsity and delays. *Automatica*, 58:143–151.

Li, X., Ma, Y., and Belta, C. (2017). Automata guided hierarchical reinforcement learning for zero-shot skill composition. *arXiv preprint arXiv:1711.00129*.

Lin, F., Fardad, M., and Jovanović, M. R. (2013). Design of optimal sparse feedback gains via the alternating direction method of multipliers. *IEEE Transactions on Automatic Control*, 58(9):2426–2431.

Liu, C., Herman, R., and Gazis, D. C. (1996). A review of the yellow interval dilemma. *Transportation Research Part A: Policy and Practice*, 30(5):333–348.

Lovisari, E., Como, G., and Savla, K. (2014). Stability of monotone dynamical flow networks. In *2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, pages 2384–2389. IEEE.

Maler, O. and Nickovic, D. (2004). Monitoring Temporal Properties of Continuous Signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, pages 152 – 166. Springer.

Mauro, V. and Di Taranto, C. (1990). Utopia. *Control, computers, communications in transportation*, pages 245–252. Politecnico di Torino.

May, R. (2007). *Theoretical ecology: principles and applications.* Oxford University Press.

Mayne, D. Q., Seron, M. M., and Raković, S. (2005). Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41(2):219–224.

Mehr, N., Sadigh, D., Horowitz, R., Sastry, S. S., and Seshia, S. A. (2017). Stochastic predictive freeway ramp metering from signal temporal logic specifications. In *2017 American Control Conference (ACC)*, pages 4884–4889. IEEE.

Meyer, P.-J., Girard, A., and Witrant, E. (2016). Robust controlled invariance for monotone systems: application to ventilation regulation in buildings. *Automatica*, 70:14–20.

Milanese, M. and Novara, C. (2004). Set membership identification of nonlinear systems. *Automatica*, 40(6):957–975.

Milner, R. (1989). *Communication and concurrency.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Mirchandani, P. and Head, L. (2001). A real-time traffic signal control system: Architecture, algorithms, and analysis. *Transportation Research Part C: Emerging Technologies*, 9(6):415–432.

Mitchell, I. M., Bayen, A. M., and Tomlin, C. J. (2005). A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957.

Morse, A. S. (1996). Supervisory control of families of linear set-point controllers-part i. exact matching. *IEEE Transactions on Automatic Control*, 41(10):1413–1431.

Narendra, K. S. and Xiang, C. (2000). Adaptive control of discrete-time systems using multiple models. *IEEE Transactions on Automatic Control*, 45(9):1669–1686.

Nilsson, P. and Ozay, N. (2014). Incremental synthesis of switching protocols via abstraction refinement. In *53rd IEEE Conference on Decision and Control (CDC)*, pages 6246–6253. IEEE.

Nilsson, P. and Ozay, N. (2016). Synthesis of separable controlled invariant sets for modular local control design. In *2016 American Control Conference (ACC)*, pages 5656–5663. IEEE.

Ouaknine, J. and Worrell, J. (2006). Safety metric temporal logic is fully decidable. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 411–425. Springer.

Paoletti, S., Juloski, A. L., Ferrari-Trecate, G., and Vidal, R. (2007). Identification of hybrid systems a tutorial. *European journal of control*, 13(2-3):242–260.

Pnueli, A. (1977). The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, 1977*, pages 46–57. IEEE.

Pola, G., Girard, A., and Tabuada, P. (2008). Approximately bisimilar symbolic models for nonlinear control systems. *Automatica*, 44(10):2508–2516.

Pola, G. and Tabuada, P. (2009). Symbolic models for nonlinear control systems: Alternating approximate bisimulations. *SIAM Journal on Control and Optimization*, 48(2):719–733.

Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. (2007). Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM.

Raković, S., Kerrigan, E. C., Mayne, D. Q., and Kouramas, K. I. (2007). Optimized robust control invariance for linear discrete-time systems: Theoretical foundations. *Automatica*, 43(5):831–841.

Raković, S. V., Grieder, P., Kvasnica, M., Mayne, D. Q., and Morari, M. (2004). Computation of invariant sets for piecewise affine discrete time systems subject to bounded disturbances. In *43rd IEEE Conference on Decision and Control (CDC), 2004*, volume 2, pages 1418–1423. IEEE.

Raković, S. V., Kern, B., and Findeisen, R. (2010). Practical set invariance for decentralized discrete time systems. In *2010 49th IEEE Conference on Decision and Control (CDC)*, pages 3283–3288. IEEE.

Raman, V., Donzé, A., Maasoumy, M., Murray, R. M., Sangiovanni-Vincentelli, A., and Seshia, S. A. (2014). Model predictive control with signal temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, pages 81–87. IEEE.

Raman, V., Donzé, A., Sadigh, D., Murray, R. M., and Seshia, S. A. (2015). Reactive synthesis from signal temporal logic specifications. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, pages 239–248. ACM.

Ramdani, N., Meslem, N., and Candau, Y. (2010). Computing reachable sets for uncertain nonlinear monotone systems. *Nonlinear Analysis: Hybrid Systems*, 4(2):263–278.

Rantzer, A. (2011). Distributed control of positive systems. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pages 6608–6611.

Rungger, M., Mazo, M., and Tabuada, P. (2013). Controller synthesis for linear systems and safe linear-time temporal logic. In *Proceedings of the 16th international conference on Hybrid systems: computation and control*, pages 333–342. ACM.

Rungger, M. and Zamani, M. (2015). Compositional construction of approximate abstractions. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, pages 68–77. ACM.

Sabău, Ş., Oară, C., Warnick, S., and Jadbabaie, A. (2017). Optimal distributed control for platooning via sparse coprime factorizations. *IEEE Transactions on Automatic Control*, 62(1):305–320.

Sadigh, D. and Kapoor, A. (2015). Safe control under uncertainty. *arXiv preprint arXiv:1510.07313.*

Sadigh, D., Kim, E. S., Coogan, S., Sastry, S. S., and Seshia, S. A. (2014). A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, pages 1091–1096. IEEE.

Sadraddini, S. and Belta, C. (2015). Robust temporal logic model predictive control. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 772–779.

Sadraddini, S. and Belta, C. (2016a). Feasibility envelopes for metric temporal logic specifications. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 5732–5737. IEEE.

Sadraddini, S. and Belta, C. (2016b). Safety Control of Monotone Systems with Bounded Uncertainties. In *55th IEEE Conference on Decision and Control*. IEEE.

Sadraddini, S., Sivaranjani, S., Gupta, V., and Belta, C. (2017). Provably safe cruise control of vehicular platoons. *IEEE Control Systems Letters*, 1(2):262–267.

Seshia, S. A., Hu, S., Li, W., and Zhu, Q. (2016). Design automation of cyber-physical systems: Challenges, advances, and opportunities. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905.

Sivaranjani, S., Sadraddini, S., Gupta, V., and Belta, C. (2017). Distributed control policies for localization of large disturbances in urban traffic networks. In *2017 American Control Conference (ACC)*, pages 3542–3547. IEEE.

Sivaranjani, S., Wang, Y.-S., Gupta, V., and Savla, K. (2015). Localization of disturbances in transportation systems. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 3439–3444. IEEE.

Smith, H. (2008). *Monotone dynamical systems: an introduction to the theory of competitive and cooperative systems*. Number 41 in Mathematical Surveys and Monographs. American Mathematical Soc.

Summers, T. H. and Lygeros, J. (2012). Distributed model predictive consensus via the alternating direction method of multipliers. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 79–84. IEEE.

Sutherland, W. A. (1975). *Introduction to metric and topological spaces*. Oxford University Press.

Svoreňová, M., Černá, I., and Belta, C. (2015). Optimal temporal logic control for deterministic transition systems with probabilistic penalties. *IEEE Transactions on Automatic Control*, 60(6):1528–1541.

Tabuada, P. (2008). *Verification and Control of Hybrid Systems* . Springer Science & Business Media.

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

Tomlin, C., Pappas, G. J., and Sastry, S. S. (1998). Conflict resolution for air traffic management: {A} study in multiagent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521.

Tumova, J., Hall, G. C., Karaman, S., Frazzoli, E., and Rus, D. (2013). Least-violating control strategy synthesis with safety rules. In *Proceedings of the 16th international conference on Hybrid systems: computation and control*, pages 1–10. ACM.

Wang, Z. and Ong, C. J. (2017). Distributed model predictive control of linear discrete-time systems with local and global constraints. *Automatica*, 81:184–195.

Wolff, E. M., Topcu, U., and Murray, R. M. (2014). Optimization-based trajectory generation with linear temporal logic specifications. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5319–5325. IEEE.

Wongpiromsarn, T., Topcu, U., and Murray, R. M. (2010). Receding horizon control for temporal logic specifications. In *HSCC'10: Proceedings of the 13th ACM International Conference on Hybrid Systems. Computation and Control*, pages 101–110. ACM.

Yordanov, B. and Belta, C. (2008). Formal analysis of piecewise affine systems under parameter uncertainty with application to gene networks. In *2008 American Control Conference*, pages 2767–2772. IEEE.

Yordanov, B., Tumova, J., Cerna, I., Barnat, J., and Belta, C. (2012). Temporal Logic Control of Discrete-Time Piecewise Affine Systems. *IEEE Transactions on Automatic Control*, 57(6):1491–1504.

Yordanov, B., Tumová, J., Cerná, I., Barnat, J., and Belta, C. (2013). Formal analysis of piecewise affine systems through formula-guided refinement. *Automatica*, 49(1):261–266.

Zamani, M., Pola, G., Mazo, M., and Tabuada, P. (2012). Symbolic models for nonlinear control systems without stability assumptions. *IEEE Transactions on Automatic Control*, 57(7):1804–1809.

CURRICULUM VITAE

Sadra Sadraddini

Phone: `6178177814` *Email*: `sadra@bu.edu` *Webpage*: `blogs.bu.edu/sadra`

EDUCATION

- *Doctor of Philosophy*, Mechanical Engineering
  Boston University, Boston, MA, 2013-January 2018 (defended on December 7th 2017)
  Advisor: Calin Belta
  Current GPA: 3.97/4

- *Master of Science*, Mechanical Engineering
  Boston University, Boston, MA, 2013-2017
  Advisor: Calin Belta

- *Bachelor of Science*, Mechanical Engineering
  Bachelor of Science, Aerospace Engineering (*Dual Major Program*)
  Sharif University of Technology, Tehran, Iran, 2009-2013
  Project Advisor: Mir Abbas Jalali
  Cumulative GPA: 17.47/20

Publications

Journals

J2. **Sadra Sadraddini**, Sivaranjani S, Vijay Gupta, Calin Belta, *Provably Safe Cruise Control of Vehicular Platoons*, IEEE Controls Systems Letters

J1. **Sadra Sadraddini**, Calin Belta, *Formal Synthesis of Control Strategies for Positive Monotone Systems*, IEEE Transactions on Automatic Control

Peer-Reviewed Conference Proceedings

C11. **Sadra Sadraddini**, Calin Belta, *Formal Guarantees in Data-Driven Model Identification and Control Synthesis*, Hybrid Systems: Computation and Control (HSCC), Porto, Portugal, 2018

C10. **Sadra Sadraddini**, Calin Belta, *Distributed Robust Set-Invariance for Interconnected Linear Systems*, American Control Conference (ACC), Milwaukee, WI, 2018

C9. Eric S. Kim, **Sadra Sadraddini**, Calin Belta, Murat Arcak, Sanjit Seshia, *Dynamic Contracts for Distributed Temporal Logic Control of Traffic Networks*, IEEE Conference on Control and Decision (CDC), Melbourne, Australia, 2017 (**Accepted**)

C8. **Sadra Sadraddini**, Calin Belta, *Formal Methods for Adaptive Control of Dynamical Systems*, IEEE Conference on Control and Decision (CDC), Melbourne, Australia, 2017 (**Accepted**)

C7. **Sadra Sadraddini**, Janos Rudan, Calin Belta, *Formal Synthesis of Distributed Optimal Traffic Control Policies* , International Conference on Cyber-Physical Systems (ICCPS), Pittsburgh, PA, 2017

C6. Sivaranjani S, **Sadra Sadraddini**, Vijay Gupta, Calin Belta, *Distributed Control Policies for Localization of Large Disturbances in Urban Traffic Networks* , American Control Conference (ACC), Seattle, WA, 2017

C5. **Sadra Sadraddini**, Calin Belta, *Feasibility Envelopes for Metric Temporal Logic Specifications*, IEEE Conference on Control and Decision (CDC), Las Vegas, NV, 2016

C4. Iman Haghighi, **Sadra Sadraddini**, Calin Belta, *Robotic Swarm Control From Spatio-Temporal Specifications* , IEEE Conference on Control and Decision (CDC), Las Vegas, NV, 2016

C3. **Sadra Sadraddini**, Calin Belta, *Safety Control of Monotone Systems with Bounded Uncertainties*, IEEE Conference on Control and Decision (CDC), Las Vegas, NV, 2016

C2. **Sadra Sadraddini**, Calin Belta, *A Provably Correct MPC Approach to Safety Control of Urban Traffic Networks*, American Control Conference (ACC), Boston, MA, 2016

C1. **Sadra Sadraddini**, Calin Belta, *Robust Temporal Logic Model Predictive Control*, 53rd Annual Allerton Conference on Communication, Control, and Computing, Urbana, IL, 2015

Invited Talks and Posters

P5. Sadra Sadraddini, *Control Synthesis for Performance-Critical Systems*, Distributed Robotics Lab, CSAIL MIT, October 2017

P4. Sadra Sadraddini, *Control Synthesis for Performance-Critical Systems*, Robot Locomotion Group, CSAIL MIT, October 2017

P3. Sadra Sadraddini, Calin Belta, *Distributed Robust Set-Invariance for Linear Interconnected Systems*, The 2nd Symposium on the COntrol of NEtwork Systems (SCONES), Boston, MA, 2017

P2. Sadra Sadraddini, Calin Belta, *Controlled Invariance for Uncertain Positive Monotone Systems* (Invited session extended abstract), 22nd Symposium on Mathematical Theory of Networks and Systems, Minneapolis, MN, 2016

P1. Sadra Sadraddini, Calin Belta, *Model Predictive Control of Urban Traffic Networks with Temporal Logic Constraints* (Tutorial Session Invited Talk), American Control Conference (ACC), Boston, MA, 2016 (Accepted)

Professional Experience

- Graduate Research Assistant, Hybrid and Networked Systems Lab, Boston University, Boston, MA, May 2014-Present
- Internship, Marine Engineering Lab., Sharif University of Technology, Tehran, Iran, Summer 2013
- Internship, Poladish Manufacturing Group, Tabriz, Iran, Summer 2012

Teaching Experience

- Teaching Assistant, Boston University, Boston, MA, Sep 2013-May 2014 and Summer 2016
  - Linear Algebra (College of Eng.)
  - Dynamics and Lab. (Department of Mech. Eng.)
- Teaching Assistant, Sharif University of Technology, Tehran, Iran, Sep 2012-May 2013
  - Flight Dynamics (Department of Aero. Eng.)
  - Mechanics of Materials Lab. (Department of Mech. Eng.)
  - Numerical Computations (Department of Mech. Eng.)
- Tutor for Physics and Astronomy Olympiad at Many High Schools in Tabriz, Tehran, and Kerman, Iran, 2007-2013

Honors and Awards

- Teaching Fellowship, Boston University, Department of Mechanical Engineering, Sep 2013-May 2014
- Fellowship, as a member of Iran's National Elites Foundation (INEF), 2008-2012, Iran
- Gold Medal, 2nd International Olympiad on Astronomy and Astrophysics, Bandung, Indonesia, 2008 (also the winner of *the best in theoretical competition award* and *the most creative solution in data analysis competition award*)
- Silver Medal, National Physics Olympiad, Iran, 2007
- Diploma III, 11th International Astronomy Olympiad, Mumbai, India, 2006
- Gold Medal, National Astronomy Olympiad, Iran, 2006
- Silver Medal, National Astronomy Olympiad, Iran, 2005

Review Responsibilities

- Journals
  - Automatica
  - IEEE Transactions on Automatic Control
  - IEEE Transactions on Robotics
  - IEEE Robotics and Automation Letters
  - IEEE Transactions on Aerospace and Electronic Systems
- Peer-Reviewed Conferences
  - IEEE conference on Decision and Control (CDC)
  - American Control Conference (ACC)
  - Hybrids Systems Computation and Control (HSCC)
  - International Conference on Cyber Physical Systems (ICCPS)
  - International Conference on Robotics and Automation (ICRA)
  - Workshop on the algorithmic foundations of robotics (WAFR)
  - Robotics Science and Systems (RSS)
  - Workshop on Distributed Estimation and Control in Networked Systems (NecSys)
  - IFAC Symposium on Control in Transportation Systems (CTS)

SKILLS

- *Programming:* Python (preferred), C++, MATLAB
- *Operating Systems:* Mac OS (preferred), Linux, Windows
- *Software/Tools*: Git, LaTeX, Microsoft Office, Arduino, Processing, SolidWorks, AutoCAD
- *Language Proficiency*: Azerbaijani, Persian (Native) English, Turkish (Fluent)