SYNTHESIS OF FORMAL CONTROL STRATEGIES FOR TRAFFIC SYSTEMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

KEMAL ÇAĞRI BARDAKÇI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

FEBRUARY 2018

Approval of the thesis:

SYNTHESIS OF FORMAL CONTROL STRATEGIES FOR TRAFFIC SYSTEMS

submitted by **KEMAL ÇAĞRI BARDAKÇI** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Halit Oğuztüzün
Head of Department, **Computer Engineering** _____

Assist. Prof. Dr. Ebru Aydın Göl
Supervisor, **Computer Engineering Department, METU** _____

Examining Committee Members:

Prof. Dr. Halit Oğuztüzün
Computer Engineering Department, METU _____

Assist. Prof. Dr. Ebru Aydın Göl
Computer Engineering Department, METU _____

Assist. Prof. Dr. Ömer Özgür Tanrıöver
Computer Engineering Department, Ankara University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: KEMAL ÇAĞRI BARDAKÇI

Signature :

ABSTRACT

SYNTHESIS OF FORMAL CONTROL STRATEGIES FOR TRAFFIC SYSTEMS

Bardakçı, Kemal Çağrı
M.S., Department of Computer Engineering
Supervisor : Assist. Prof. Dr. Ebru Aydın Göl

February 2018, 89 pages

The problems caused by traffic congestion affect human life adversely. Wasted amount of time, fuel and money as well as adverse effects to environment are examples of these problems which reduce life quality. Studies on traffic management underline importance of traffic network control mechanisms. Different configurations of roads and signalized intersections complicate interactions among vehicles and pedestrians. Hence, traffic control systems, which need to satisfy complex specifications, should be constructed to serve complex traffic network features. In this dissertation, we study the problem of synthesizing a signal control strategy for a traffic system from Linear Temporal Logic (LTL) specifications.

We focus on scalability issue in formal control of large traffic systems and propose to tackle it by decomposing the main problem into smaller problems. The developed decomposition algorithm partitions main traffic system into subsystems and derives a specification for each subsystem from main specification. In addition, we derive additional constraints on the signals lying on boundaries of subsystems to ensure fair-

ness. We employ abstraction based techniques to find control strategies for subsystems by considering dynamics of adjacent subsystems. We show that the controllers found for each system guarantee that overall traffic system satisfies given specification. Moreover, we use bounded LTL, and we analyze effects of bounds on resulting set of satisfying initial states. Furthermore, this dissertation incorporates various optimization criteria into control synthesis. In particular, we developed novel methods to synthesize strategies minimizing the total number of switches and minimizing the maximum vehicle denisty in any link.

Keywords: Linear Temporal Logic, Formal Control, Signalized Traffic Control, Network Partitioning (Decomposition), Cyber-Physical Systems

ÖZ

TRAFİK SİSTEMLERİ İÇİN FORMEL KONTROL STRATEJİLERİNİN SENTEZLENMESİ

Bardakçı, Kemal Çağrı
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Yrd. Doç. Dr. Ebru Aydın Göl

Şubat 2018, 89 sayfa

Trafik sıkışıklığı, dünyadaki metropollerdeki gibi ülkemizde İstanbul ve Ankara gibi büyük şehirlerde insan yaşamını olumsuz etkileyecek seviyelere ulaşmış durumdadır. Trafik sıkışıklığı, sebep olduğu kaybolan zaman, boşa giden yakıt/ para, çevre üzerindeki olumsuz etkilerle insanların yaşam kalitesini düşürmektedir. Bu konuda yapılan araştırmalar, trafik ağı kontrolünün ne denli önemli olduğunu göstermektedir. Çok sayıda araç ve yayanın, farklı yol ve ışık yapılandırmalarındaki etkileşimleri karmaşıklaşmakta, dolayısıyla karmaşık gereksinimleri sağlayacak trafik sistemlerinin oluşturulması gerekmektedir. Bu alanda yapılan çalışmalarla yol ve ışık bazındaki karmaşık gereksinimleri, oluşturulan trafik sistemi modeli üzerinde kontrol edebilecek farklı mekanizmalar geliştirilmeye çalışılmıştır. Bu tezde, büyük bir sinyalize trafik sistemi üzerinde, tanımlanmış karmaşık doğrusal zamansal mantık gereksinimlerini sağlayacak bir kontrolcü sentezleme problemi üzerine çalışmalar yapılmıştır.

Yaptığımız çalışmalarda, özellikle büyük trafik sistemlerinin formel kontrolünde ortaya çıkan ölçeklenebilirlik sorunu üzerine odaklanarak, bu sorunu ana problemi daha

küçük problemlere ayrıştıracak çözümler üretmekteyiz. Geliştirilen bölüntüleme algoritması ana trafik sistemini alt sistemlere bölüp, ana gereksinimden her bir alt sistem için gereksinim kümesi elde etmektedir. Bunun yanında, alt sistemler arasında eşit ve adil dağılımın garanti edilmesi adına komşu sistemler arasındaki sınır sinyalleri üzerinde ek kısıtlar türetilmektedir. Alt sistemler için kontrol stratejileri üretilmesi sırasında ise soyutlama tekniklerinden yararlanılmaktadır. Bu şartlar altında, her bir alt sistem için kontrolcü üretiliyor olmasının, sistemin bütününün ana gereksinimi sağlamasını garantileyeceği ispatlanmaktadır. Ayrıca, sınırlandırılmış operatörler kullanılarak bunların gereksinimi sağlayan başlangıç durum kümesi üzerindeki etkileri incelenmektedir. Son olarak bu tez, çeşitli optimizasyon kriterlerini kontrol stratejileri üzerinde deneyimlemektedir. Örnek olarak, trafik sistemindeki toplam sinyal değişim sayısının minimize edilmesi, maksimum araç yoğunluğuna sahip yoldaki araç yoğunluğunun minimize edilmesi sayılabilir.

Anahtar Kelimeler: Doğrusal Zamansal Mantık, Formel Kontrol, Sinyalize Trafik Kontrolü, Ağ Bölüntüleme, Siber Fiziksel Sistemler

To My Family

ACKNOWLEDGMENTS

I am glad to present my gratefulness to all people helping me to overcome thesis experience.

First and foremost, I would like to express my inmost gratitude to my mentor, supervisor, Assist. Prof. Dr. Ebru Aydın Göl for her insight, encouragement, guidance, patience and support. It is a great honor for me to share her wisdom and knowledge. She always encourages me and shows me the direction when I felt lost inside the research and study.

Additionally, I would like to thank the examining committee members, Prof. Dr. Halit Oğuztüzün and Assist. Prof. Dr. Özgür Tanrıöver, for their advices, insightful comments and questions.

I thank my friends and work-fellows Ahmet Yapıcı, Osman Eren, Mehmet Şamlı, İsmail Akpolat and others for their intellectual support and companionship.

I am grateful to Sultan Arslan, chief of student affairs, for her valuable supports, patience and generosity.

I would like to thank my parents Alaaddin and Gülşen as well as Salim and Nuran, my sisters Sümeyye and Şule as well as Hazal for their supports to relieving my burden. I am also grateful to my grandparent Kemal and the deceased grandmother Fatma for their good wishes and prayers.

The last but not the least, I would like to thank my lover and supportive wife Merve Peyker who exerts herself with every effort and my little son Almir Alaaddin. I could not overcome the difficulties without their love and forbearance.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

The traffic density evolves into a serious problem with the increase in the individual usage of vehicles as well as reckless management of the traffic network. Hence, an efficient and realizable control of vehicular traffic network is an essential requirement to resolve this worldwide trouble (especially in metropolises). The TomTom Traffic Index based on November 2015 data in [77] reveals that while Mexico City and Bangkok are the most congested two cities in the world (among the cities with a population greater than 800,000) with the congestion level of 59 % and 57 % respectively, Istanbul is the third most congested city with the 50 % congestion level. The congestion level indicates the increase in overall travel times when compared to a Free Flow situation (an uncongested situation). In the same report, Istanbul has the congestion level of 62 % in morning peak time and 94 % in evening peak time whereas the congestion levels of morning and evening peak times 97 % and 94 % for Mexico City and 85 % and 114 % for Bangkok respectively. 94 % congestion level in evening peak time for Istanbul increases travel time from 30 minutes to 58 minutes. TomTom index also indicates that Ankara has the overall congestion level of 32 % and 42 %, 52 % in the morning, evening peak times whereas Izmir has 29 % overall congestion level and 37 % and 51 % congestion levels in morning and evening peak times.

Another report on the traffic congestion with time and money costs have been published as INRIX Global Traffic Scorecard in February 2017 [66]. When we look at

the results on this report we can see that Los Angeles, Moscow and New York is the first three most congested cities in 2016. The cost of congestion in these cities are also calculated in the report, *e.g.,* the annual cost of congestion in New York (2016) is $16.9 billion. The report does not include annual cost of congestion in Istanbul, however, the report underlines that Istanbul is one of 10 top congested cities in the world and it is also one of the top 5 congested cities in Europe. If we look at the cities which have similar INRIX congestion indices and the number of vehicles to the values in Istanbul, annual cost of traffic congestion in Istanbul is about $3 - 4 billion.

1.2 Research Objectives and Scope

The demand on the roadways with increasing individual cars, the congestion levels on the roadways and their high cost necessitate to develop efficient signal control strategies. The control of vehicle traffic systems is the subject of Cyber-Physical System (CPS). US National Science Foundation (NSF) defines CPS as:

> The engineered systems that are built from, and depend upon, the seamless integration of computational algorithms and physical components.

In this thesis, the complex control tasks for traffic system are defined as temporal specifications and techniques to produce feedback control strategies from these temporal objectives are developed. The specifications such as “the number of vehicles on a link does not exceed a threshold" and “as long as there exist entering vehicles to the traffic system, the number of vehicles exiting from system is above a threshold" can easily be represented as temporal logic formulae. The traffic system used in this dissertation can be categorized as CPS. Although traffic systems can be modelled as CPS, the existing controller synthesis techniques in traditional CPS is not applicable directly to traffic systems due to the scalability problems. The crucial point is the exponential increase in the computational complexity with the size of the mathematical models of the traffic systems.

In this dissertation, we aim to construct finite state representation of the large traffic systems using its dynamics and synthesize the control strategy from this model. While a control strategy is synthesized from temporal logic specifications, optimiza-

tion criteria such as minimizing total number of signal switches are also determined in this study. In other words, even though this thesis focuses on the control of traffic systems, the determination of the optimal / suboptimal control strategy problem in the field of formal control is also studied.

1.3 Structure of The Thesis

This thesis starts with Chapter 1 as an introduction which describes motivation behind this study, research objectives and scope informally.

In Chapter 2, we review and give information about the state-of-the-art studies in the fields of traffic system and its modelling approaches, formal methods and temporal logic usages and lastly the control of traffic systems using formal methods.

Preliminary information for this thesis is presented in Chapter 3. This chapter formally defines the traffic system, its mathematical model, Linear Temporal Logic and automaton concepts.

In Chapter 4, formal synthesis problem and the abstraction based solution approach are given. We introduce formal synthesis problem for traffic systems, abstraction based solution approach and our new solution approach that is based on partitioning traffic network and constructing corresponding subsystems and specifications.

Our proposed traffic network partitioning procedure and specification derivations for newly produced subsystems developments are presented in Chapter 5.

The proposed efficient abstraction techniques for traffic systems and the novel formal synthesis techniques for interacting subsystems are presented in Chapter 6.

In Chapter 7, we present the optimal and suboptimal controller synthesis for 2 different optimizaiton criteria including minimization of total number of switches in overall traffic network, minimization of the maximum vehicle density.

We highlight the experiments and their results for proposed controller synthesis techniques with various traffic system parameter sets and LTL specifications in Chapter 8. Furthermore, the proposed formal controller synthesis solutions are discussed and

compared with each other as well as other related state-of-the-art studies.

Chapter 9 summarizes the thesis and discusses further research directions..

CHAPTER 2

RELATED WORK

In recent years, as the number of vehicles has increased severely, it is necessary to have smart traffic control systems which provide efficient and effective usage of existing traffic infrastructure. Influential traffic network algorithms have been developed in order to balance / reduce traffic congestion and to improve efficiency / time-saving during travelling. This chapter briefly describes the related works that are relevant to the work presented in this dissertation.

2.1 Traffic Systems

We present commonly used traffic system models in this section. One of the very first traffic system control models was introduced by Webster in 1958. In this seminal work [83], the goal is to determine whether an optimum traffic signal schedule can be produced by gathering test data from the signalized intersections in London. Traffic system controllers can utilize 3 different types of signal procedures which are in the characteristics of pretimed, semi-actuated and fully-actuated as explained in [6]. In pretimed models, periods and occurrences of every traffic signal phase are predetermined, whereas at least one of the phases is guaranteed to be operated in semi-actuated models. However, in traffic responsive (fully-actuated) models the traffic signals are actuated according to the traffic conditions at incoming roads between minimum and maximum duration bounds.

Traffic system models have been also grouped into two different categories in terms of dependency of the signals. One of them is isolated strategy which evaluates every

traffic intersection one by one and the the other one is coordinated strategy which specifies one or more signalized intersections together.

The traffic system models for isolated intersection with fixed cycle length (pretimed) can be categorized as stage and phase approaches. Stage and phase approaches try to maximize the number of vehicles exit from intersection or to minimize the total delay for all vehicles in controlled intersection. In addition to optimization problems, these approaches try to find (sub)optimal cycle lengths (stage approach) [2, 3] and optimal setting for each phase (phase approach) [41]. However, the isolated strategies is not suitable for the large and congested traffic system models.

2.1.1 Traffic Flow Models

Traffic flow can be summarized as the study of the movements and interactions among the vehicles and employed traffic network platform with the characteristics of velocity, density, flow rate, volume, etc. While in the networks with very low density, the interactions between stakeholders is not important and negligible, in the networks with very high density, *i.e.,* saturated or oversaturated networks, flow dynamics is not applicable and the queueing theory is widely used on these systems. In congested but moving traffic networks, the flow model plays a significant role for the computations in the queues. Flow models are categorized as microscopic, mesoscopic and macroscopic in some studies whereas we use the categorization of microscopic, macroscopic and cellular / kinematic modelling approaches. The detailed analysis about traffic flow dynamics and models can be found in [60, 79].

2.1.1.1 Microscopic Flow Model

In the microscopic model, the position, velocity and acceleration values for each vehicle in the network is considered individually. In other words, in this traffic flow model, each car (or driver) acts individually with the space and time parameters and constraints by considering the behaviours of the cars (drivers) in its neighbourhood. The vehicle following (or car-following or leader-driver following) model, several simulation models and particle models are considered as microscopic. Cellular automata

model can also be considered in this category due to its car-based approach [64]. With the help of vehicle following model, almost every detail for each vehicle is gathered and exploited individually. In the studies of [65, 71], the flow is considered as a single lane and a vehicle is followed by another one and the behaviours of each vehicle are represented in detail with the constraints on the distance between two successive vehicles and the velocity of the following vehicle. This model is expanded into multiple lanes with lane switching and distinct types of vehicles. The kernel of several microscopic simulator models generally utilizes functions with discrete time step computations as in the examples of AIMSUN with Gipps model [11] and VISSIM with the Wiedemann model [33]. Some studies underline the convergence of microscopic and macroscopic models [16, 55].

2.1.1.2 Cellular or Kinematic Model

Cellular Automaton models describe the network (or system) cell by cell using discrete time steps and propagating the particles from one cell to another. The space (width or height or both for a cell predetermined) and time step variables are discrete in this model and each cell at each time contains at most one vehicle. The constraints and set of actions are determined according to the propagation of a vehicle from one cell to another cell. Nagel-Schreckenberg model [62] is usually considered as the standard in cellular automata model although similar models were used beginning from 1960s, *e.g.,* [25]. Stochastic cellular automata models are proposed with probabilistic actions such as pausing, accident, etc. in the studies [40, 87] and with the implementation of distribution (or partition) of state space in [82].

2.1.1.3 Macroscopic Flow Model

Macroscopic flow models include variables which represent several vehicles and / or drivers actions / behaviours, *e.g.,* vehicle density in a link, number of vehicles in an interval of link or directly in a link, flow ratio, inflow-outflow equations, number of exterior vehicles, etc. Proposed optimization problems utilized distinct macroscopic parameters such as average flow and average velocity in a link in [57], average ve-

locity in a link and link density as in [38], distance (with space partitioning) and average speed in a link as in the examples of [80, 81, 86]. One of the first examples of macroscopic flow models is first order model named Lighthill Whitham Richards (LWR) [58] which draws attention to the functional relation between density and velocity and utilizes the flow conservation formula.

The Cell Transmission Model (CTM) is another commonly used macroscopic model. This model contains location and time dependant continuous parameters such as density, flow ratio and speed. Discrete version of CTM adapts from its continuous counterpart by using Godunov method [34]. This method partitions the link and time to discretize the traffic system. Nonetheless, the discrete simulations obstruct the comprehension of the new solution heuristics for CTM. Hence, the continuous version of CTM that proposes the demand, supply and density as state variables but flow ratio and speed as calculated variables with the help of downstream and upstream points in [42, 43, 76]. Those continuous and discrete CTMs can be applicable for the link-based traffic parametrization. Intersection models including general version [43], separation of intersections model [26, 44] and association of the intersections model [26, 42, 56] represent supply and demand information of downstream and upstream links in detail with the main goal of simulation and analysis of traffic systems. CTM is useful for planning whole traffic network, formulation of signal optimization and real time information attainment purposes. Therefore, CTM model is used as the traffic network model in this study.

2.2 Formal Methods and Temporal Logic

The development of a system to operate as reliable (or verifiable) as possible is a crucial challenge in Cyber-Physical Systems (CPS). Formal methods are developed to overcome this challenge by providing formal - mathematical foundations (techniques, languages and tools) for CPS [22]. These systems are verifiable by means of their meticulous mathematical design with formal methods which takes part in software development phases such as specification, implementation and verification as in the studies [22, 51, 54]. Model checking is the most commonly used formal verification technique. The inputs for model checking techniques are mathematical model

of a system and formal specification for this system. Any specification violation is detected by examining whole state-space of the model and is returned as a trace of the model behaviour to elucidate how the system model violates the specification. The formal control approach has been extensively studied in the literature both in theory [8, 19] and in practice [18, 39, 52].

A commonly used specification language in formal control is temporal logic. Linear Temporal Logic (LTL) [68] and the Computation Tree Logic (CTL) [20] are the widely used temporal logics. These two logics are the subset of the CTL* and all propositions of both and much more are expressible by CTL* [30]. In addition, the extensions of LTL such as metric temporal logic, real time temporal logic, signal temporal logic are developed for specific purposes.

Various tasks including safety (nothing bad happens), responsiveness (if p happens then q happens), persistence (a good thing happens at some state and stay the same on rest of the states), recurrence (a good thing happens repeatedly) can easily be expressed in LTL. Satisfaction of a property is checked over all possible sequences of the states successively by automated model checkers. The qualitative model checking problem from temporal logic formula is solvable by discretizing the dynamic properties as a piecewise affine (or piecewise linear) systems as in [12, 27]. In addition, it is discretized on the basis of quantitative simulations and experimental data instead of symbolic representation with the definition of constraints on the chunks as the information derivation from traces as in [31]. Furthermore, in [29, 72] similar model checking techniques are studied with extra constraints on the prospective actions for parameter settings and perdurability. As LTL is able to demonstrate useful specifications in a wide spectrum for CPS and is benefited prevalently in state-of-the-art formal control studies, LTL is utilized in this thesis as a formal specification language.

Recent studies introduce the usage of assume guarantee approach [37] for the subparts of the decentralized systems. A correct by construction controller is generated by using a similar approach, *i.e.,* separating overall system into subsystems in a compositional manner to reduce dimensionality [63]. However, as this technique only considers fixed feedback gains (invariant sets) and a part of LTL specifications, it does not evaluate real time behaviours, thus, it increases conservatism excrescently.

Another correct by construction controller synthesis problem is proposed in a parametric and reactive manner [5,73]. In these studies, the control strategy is constructed compositionally from a user specified library of controllers with parameters. Nevertheless, this approach pretend that every piece of information about systems dynamics exist at any time. Therefore, it is not applicable to the realistic systems as it is presented and it is open to problems due to the user involvement in controller synthesis parametrization.

Formal control is used in the control communities to specify desired actions for a dynamical system such as hybrid systems, linear systems, piecewise affine systems etc. The common approach in control of such systems is to first construct an abstract model of the system in the form of a transition system, and then use automata theoretic techniques to synthesize a control strategy from the formal specifications on the abstract model [4,17,85]. In [15,45,67], approximate finite abstractions of nonlinear transition systems are constructed to synthesize control strategies from motion planning tasks expressed in LTL. In another study [48], the techniques for the synthesis of a control policy (from the abstraction of a linear system) guaranteeing the satisfaction of LTL specification by the original system is proposed. Moreover, for the non-deterministic abstraction of linear systems, synthesis of reactive control policy techniques is revealed in [50]. To solve the state space explosion problem caused by this approach, control strategy synthesis with receding horizon techniques is developed in [84]. Synthesis of control strategies with MDPs for LTL and Probabilistic CTL is provided in [28] and [53], respectively. On the other hand, suboptimal and robust control techniques are proposed in a few studies [46,74] and [59,78], respectively. The controller synthesis for LTL specification has doubly exponential time complexity in the length of the LTL formula [24,69]. In addition to this, the discrete abstract model construction possesses high computational cost for high dimensional systems. Therefore, the study in this thesis aims at developing efficient techniques for formal control of traffic systems.

2.3 The Control of Traffic Systems via Formal Methods

As the conventional optimal control techniques are computationally more complex for hybrid dynamic systems, the applicability to the very large traffic systems is intractable *e.g.,* Hamilton Jacobi Bellman (HJB) partial differential equation as discussed in detail [9, 10]. Model predictive control (MPC) with real time optimization for fully-actuated traffic system implementations - such as Split Cycle and Offset Optimization Technique (SCOOT) and Optimization policies for adaptive control (OPAC) - are not feasible to the very large scale and real time traffic systems as discussed in [1, 13, 64]. Intersection-based distributed optimization techniques for the traffic control are also inadequate to reduce congestion in large systems [36]. Although real time, interconnected and hierarchical adaptive control of traffic systems [61] reduces the computational complexity and congestion, this type of solutions does not ensure specifications globally in a formal manner. In this thesis, it is ensured that given overall specifications are satisfied even though the signal control strategies are determined in a decentralized manner.

In this dissertation, we develop formal control strategies for traffic systems from Linear Temporal Logic (LTL) specifications. This traffic system can be categorized as hybrid dynamic model. Hybrid dynamic models contain continuous and discrete variables which make the traffic models mathematically more complex. Several constraints for the traffic dynamics also increases this complexity *e.g.,* constraints for the traffic signal periods. A variety of studies demonstrates the methods to synthesize formal control strategies for hybrid dynamic systems from complex specifications [7, 19, 32, 35, 49, 75]. As traffic systems have several characteristics in common with hybrid dynamic systems, high level specifications can be identified for traffic system such as congestion and flow scheduling. The formal control of traffic systems from temporal logic specifications is proposed in [23]. This study underlines that the control strategies - synthesized with correct by construction technique - guarantees the satisfaction of complex LTL specifications. Besides, it provides efficient methods to compute finite state abstraction of the traffic system. Nonetheless, the computational difficulty in state space discretization for high dimensional systems and the complexity of finding a control strategy for a large finite model can be regarded as the

deficiencies in this study.

Another approach is to construct MPC strategies for traffic systems from temporal logic (especially Signal Temporal Logic (STL)) specifications [70]. This approach cannot ensure safety specifications and it is insufficient for the exterior vehicle joins. Although traffic systems can be modelled as hybrid dynamic systems, the existing methods in this field is not applicable directly to traffic systems due to the scalability problems. The crucial point is the exponential increase in the computational complexity with the system size as well as curse of dimensionality of the mathematical models of the traffic systems. In this dissertation, we aim to construct finite state representation of the traffic system using its dynamics and synthesize the control strategy from this model. While a control strategy is synthesized from LTL specifications, optimization criteria such as minimizing total number of signal switch are also determined in this study. In other words, in this thesis, even though the focus is on the control of traffic systems, the determination of the optimal / suboptimal control strategy problem in the field of formal control is also studied.

Another technique of decomposition of traffic systems into subsystems is proposed in [47], where flow constraints between adjacent subsystems are specified. These constraints bound the number of vehicles in the roads that connect adjacent subsystems. In this thesis, instead of flow constraints, we introduce constraints on the signal durations for the intersections of adjacent subsystems. As a discrete time model is used, we can search for all possible constraints, whereas, the constraint search in [47] is done via discretization of the road capacities. In addition, tight flow constraints might result in biased control strategies that actuate the connection roads more than the others. Furthermore, we propose two methods to reduce conservatism caused by the partitioning: 1) using the upper bounds on the roads of the adjacent subsystems, 2) using the specifications of the adjacent subsystems. The first method is similar to flow constraints. However, instead of introducing new bounds, we use the bounds specified by the user. The second method further reduces the conservatism by considering the temporal properties that the adjacent subsystems have to satisfy during the controller synthesis. Consequently, our framework allows for richer interactions between adjacent subsystems. Finally, while similar system partition definitions are used in this work and in [47], we define an algorithm to construct it.

2.4 Main Contributions of The Thesis

As specified in 2.3 the major problem in the formal control of very large traffic systems is state-space explosion problem. Due to this problem, formal control techniques can not be directly applied to traffic models. Furthermore, the applicability of the traffic controllers to the real life scenario is also another crucial challenge. In addition to all these, uncertainties in traffic system models and the desire to synthesize optimal or suboptimal controller strategies further complicates the issue. We have developed new techniques which help overcome these problems. The main contributions of this thesis include the followings:

- The primary contribution of this work is to produce a scalable synthesis method. The developed method is based on dividing the traffic system into smaller subsystems and deriving specifications for these systems. We define a formal algorithm to construct traffic system partition. Furthermore, additional constraints on the signals lying on the boundaries of subsystems are introduced to ensure fairness. It is proved that "if each subsystem satisfies its own specification, then the overall traffic system also satisfies given specification". Hence, the correctness of the syntesized strategy is guaranteed. The correctness result is reached by considering all possible behaviours of the adjacent systems.
- Second main contribution of this thesis is the proposal of two methods to reduce conservatism caused by the partitioning:

  1. using the upper bounds on the roads of the adjacent subsystems,
  2. using the specifications of the adjacent subsystems

  The first method is similar to setting bounds to flow parameters. However, instead of introducing new bounds, we use the bounds specified by the user. The second method further reduces the conservatism by considering the temporal properties that the adjacent subsystems have to satisfy during the controller synthesis. Consequently, our newly proposed framework allows for richer interactions between adjacent subsystems.
- Another contribution is reducing computation cost of the transition system construction. A major advantage of our construction based on link transition sys-

tem is that the link transition systems only constructed once. Thus, the abstraction computation based on the actual system dynamics is done only once. Thanks to this, the transition systems for the subsystems are constructed using only the link transition systems. If we can not find a solution for the overall system, the following approaches can be followed to reduce the conservatism of the partitioning based approach 1) perform a search over the constraints defined for the boundary signals, 2) iteratively merge neighbouring subsystems. In both cases, we reuse the link transition systems hence avoid additional computation based on the system dynamics.

- Another contribution of this thesis is the use of bounded LTL operators in specifications, and analyzing the effects of the bounds on the resulting set of satisfying initial states. We conduct experiments on the effects of the bounded specifications and satisfying volume of the synthesized controller.

- The final contribution of this dissertation is incorporating various optimization criteria into the control synthesis. In particular, the methods are developed to syntesize optimal and suboptimal strategies for minimization of total number of switches in overall traffic network, minimization of the maximum vehicle density.

CHAPTER 3

TRAFFIC SYSTEM DEFINITION AND SPECIFICATIONS

3.1 Traffic Systems

A traffic system is composed of a set of links $\mathcal{L}$ and a set of signals $\mathcal{V}$. The system is modeled as a discrete-time queue with finite capacity. The number of vehicles in a link l at a time t is denoted by

$$x_l[t] \in [0, x_l^{cap}], \forall l \in \mathcal{L},$$

where x_l^{cap} refers to the vehicle capacity for link l. The number of vehicles which link l can send to link l' depends on the current number of vehicles in link l as well as the available space on the link l'. The number of vehicles demanded to be sent by a link l at time t is formulated as

$$z_l^{out}[t] = min\{x_l[t], c_l\}, \tag{3.1}$$

where c_l represents the saturation flow of link l, i.e., the maximum number of vehicles which can exit from link l at a unit of time. The amount of space supplied by link l is formulated as

$$z_l^{in}[t] = x_l^{cap} - x_l[t]. \tag{3.2}$$

The supply ratio α_{lk} is used to determine the space of link k that is reserved for link l when vehicles from multiple links can flow to k. Similarly, when vehicles from a link can flow to multiple links, the demand ratio β_{lk} is used to compute the number of vehicles sent from link l to link k.

The number of vehicles which exit from a link at a discrete time is computed using the signal configuration, the number of vehicles requested to be sent by current link,

the number of vehicles accepted by other links which are downstream to the current link and formulated as:

$$f_l^{out}(x_l[t], s_l[t]) = s_l[t] \min\left\{ z_l^{out}[t], \min_{k, \beta_{lk} \neq 0} \left\{ \frac{\alpha_{lk}}{\beta_{lk}} z_k^{in}[t] \right\} \right\}, \tag{3.3}$$

where $s_l[t] \in \{0,1\}$ is computed from the mode of the signal which controls the vehicle flow from link l. $s_l[t]$ is 1 if the link is actuated for the selected mode. The dynamics of the number of vehicles on the links are expressed as:

$$\begin{aligned} x_l[t+1] &= F_l(\mathbf{x}[t], \mathbf{s}[t], d_l) \\ &= x_l[t] + d_l[t] - f_l^{out}(x_l[t]) + \sum_{k \in \mathcal{L}} \beta_{kl} f_k^{out}(x_k[t], s_k[t]), \end{aligned} \tag{3.4}$$

where d_l represents the number of vehicles joined from other roads which are not modelled in the traffic system and $d_l^{min} \leq d_l \leq d_l^{max}$.

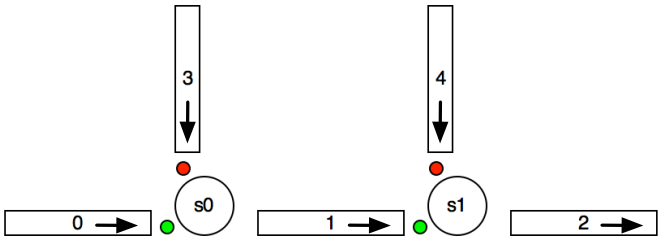


Figure 3.1: A traffic network composed of links $l_0, l_1, \ldots, l_4$ and signals s_0, s_1. The flow directions of the links are shown with arrows.

Example 3.1.1. *An example traffic system is shown in Figure 3.1. The system contains horizontal links l_0, l_1, l_2 and vertical links l_3, l_4 and signals s_0, s_1. The parameters for the traffic system is given as follows:*

$$\begin{aligned} &x_i^{cap} = 40 \text{ for } i \in \{0,1,2\}; x_j^{cap} = 20 \text{ for } j \in \{3,4\}. \\ &c_i = 20 \text{ for } i \in \{0,1\}; c_2 = 15; c_j = 10 \text{ for } j \in \{3,4\}. \\ &\beta_{ij} = 0.75 \text{ for } i\text{-}j \in \{0\text{-}1, 1\text{-}2\}; \beta_{kl} = 0.3 \text{ for } k\text{-}l \in \{3\text{-}1, 4\text{-}2\}. \\ &\alpha_{31}^0 = 0.8; \alpha_{ij}^k = 1 \text{ for } i\text{-}j\text{-}k \in \{0\text{-}1\text{-}0, 1\text{-}2\text{-}1, 4\text{-}2\text{-}1\}. \\ &d_i^{max} = 4.99 \text{ for } i \in \{0,3,4\}. \end{aligned}$$

where x_i^{cap} and c_i denotes the capacity of the link l_i and saturation flow for the link l_i, respectively. α_{ij}^k denotes the supply ratio of the link j to the link i when the signal k is green for the link i. All signals has two modes, either the horizontal or the vertical links are actuated.

3.2 Transition Systems

Transition system is used as finite abstractions of the traffic system.

Definition 3.2.1. *A transition system (TS) is a tuple of* $T = (\mathcal{Q}, \mathcal{S}, \rightarrow, I, O, o)$*, where*

- $\mathcal{Q}$ *is a set of states,*
- $\mathcal{S}$ *is a set of inputs,*
- $\rightarrow \subseteq \mathcal{Q} \times \mathcal{S} \times \mathcal{Q}$ *is a transition relation,*
- $I \subseteq \mathcal{Q}$ *is a set of initial states,*
- O *is a set of atomic propositions (observations),*
- $o : \mathcal{Q} \rightarrow 2^O$ *is a labelling function (observation map).*

A transition $(q, s, q') \in \rightarrow$ *shows that the system state can change to* q' *from* q *when control input* s *is applied.* $q \xrightarrow{s} q'$ *is used to denote* $(q, s, q') \in \rightarrow$*.*

A trajectory of a transition system is an infinite sequence $q_0, q_1, q_2, \ldots$ *such that* $q_i \in \mathcal{Q}$*, and* $q_i \xrightarrow{s} q_{i+1}$ *for some* $s \in \mathcal{S}$ *for all* $i = 0, 1, \ldots$*. The trajectory produces an infinite word* $o(q_0)o(q_1)\ldots$*. Such a word can be checked against an LTL formula defined over* O*. A trajectory satisfies a formula if the corresponding word satisfies the formula.* $T = (\mathcal{Q}, \mathcal{S}, \rightarrow)$ *notation is used when* I, O, o *are not needed.*

Definition 3.2.2. *Given a state* q *and a control input* s*, the set of the successor states of* q *after executing* s *is defined as follows:*

$$Post(q, s) = \{q' \mid (q, s, q') \in \rightarrow\} \tag{3.5}$$

Therefore, $Post(q, s)$ *contains all states that can be reached from state* q *by applying control input* s*. We can also lift up the notation to the set of actions:*

$$Post(q) = \bigcup_{s \in \mathcal{S}} Post(q, s) \tag{3.6}$$

$Post(s)$ *contains all states reachable from state* q *by applying any control input.*

A TS is deterministic if:

1. $|I| = 1$, which is, TS has exactly one initial state,
2. $\forall q \in Q, \forall s \in \mathcal{S}; |Post(q,s)$, see Definition 3.2.2 $| \leq 1$; which is for all states and inputs TS has at most one successor state.

Otherwise, a TS can be categorized as non-deterministic.

Definition 3.2.3. *A finite memory controller (FMC) for a TS (see Definition 3.2.1) is a tuple of $C = (\mathcal{M}, f_n, f_u)$, where $\mathcal{M}$ is a finite set of memory elements (modes), $f_n : Q \times \mathcal{M} \to Q$ is a next state function and $f_u : Q \times \mathcal{M} \to \mathcal{S}$ is an update function. f_u selects a control action based on the current state of the TS and memory value of the controller C, and f_n updates the memory of the controller. Thus, $C(q[0], \ldots, q[t]) = f_u(m[t], q[t])$ where $m[t]$ is defined as $m[t+1] = f_n(m[t], q[t])$ for $t \geq 0$ ($s[t]$ is also formulated as $s[t] = f_u(m[t], q[t])$).*

A FMC (see Definition 3.2.3) $C : Q^+ \to \mathcal{S}$ of a transition system maps a finite sequence of states of the system $q_0, \ldots, q_d$, $d \in \mathbb{N}$ to a control input $s \in \mathcal{S}$.

Given a finite transition system T (see Definition 3.2.1), an LTL formula Φ over its set of observations O, the problem of finding a set of states Q^Φ and a finite memory controller C^Φ such that all the trajectories of the controlled system that originate from Q^Φ satisfies the formula Φ is known as the formal controller syntesis problem [14]. The problem can be solved by constructing an automaton from the specification, taking the product of the automaton and the transition system and then solving a game on the product [14]. This synthesis algorithm is double exponential with the length of Φ.

3.3 System Specifications (Linear Temporal Logic)

We use Linear Temporal Logic (LTL) specifications for the links and signals of the traffic system. LTL allows us to express a wide range of system properties succinctly. Atomic state property of LTL expresses the boolean property which is true or false over state of the variables in this logic, *e.g.,* $p =$ the traffic light is green.

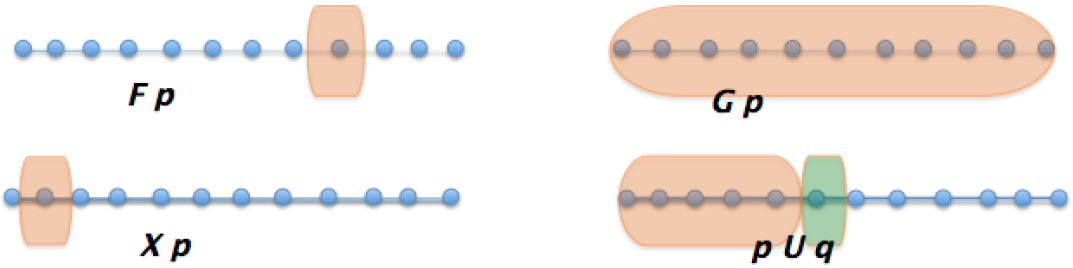


Figure 3.2: Basic LTL operators $\mathbf{F}p$ = Eventually p, $\mathbf{G}p$ = Globally (Always) p, $\mathbf{X}p$ = Next p and $p\mathbf{U}q$ = p until q

3.3.1 Syntax

The syntax for an LTL formulae is defined as:

$\phi ::= \top \mid \bot \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \mathbf{X}\phi \mid \mathbf{F}\phi \mid \mathbf{G}\phi \mid \phi_1 \mathbf{U} \phi_2$,

where

- $\top$ denotes the logical constant "True",
- $\bot$ denotes the logical constant "False",
- p denotes an atomic proposition,
- $\neg\phi$ - not, denotes the negation of ϕ,
- $\phi_1 \wedge \phi_2$ - and, denotes the conjunction of ϕ_1 and ϕ_2,
- $\phi_1 \vee \phi_2$ - or, denotes the disjunction of ϕ_1 and ϕ_2,
- $\mathbf{X}\phi$ - next, denotes the next state (or next time) ϕ,
- $\mathbf{F}\phi$ - eventually, denotes the at least one state in the future beginning from now ϕ,
- $\mathbf{G}\phi$ - globally, denotes all states in the future beginning from now ϕ,
- $\phi_1 \mathbf{U} \phi_2$ - until, denotes ϕ_1 is true for all states from now at least until the state where ϕ_2 is true.

The precedence of execution (binding strength) of the LTL operators is as follows in descending order: 1) $\mathbf{G}, \mathbf{F}, \mathbf{X}, \neg$(negation) the same for these 4 operators, 2)$\wedge$, 3)$\vee$, 4)$\mathbf{U}$.

3.3.2 Semantics

The semantics of an LTL formula over a set of atomic propositions AP is defined over infinite words $w \in (2^{AP})^{\omega}$. Let $w, i \models \phi$ means that w satisfies the LTL formula ϕ at position i of the w and the satisfaction relation is recursively defined as:

- $w, i \not\models \bot$ - $\bot$ (False) is never satisfied
- $w, i \models \top$ - $\top$ (True) is always satisfied
- $w, i \models p$ - p (atomic proposition) is satisfied if and only if (iff) p is true in state w[i] (or $p \in w$[i])
- $w, i \models \phi_1 \wedge \phi_2$ - iff $w, i \models \phi_1$ and $w, i \models \phi_2$
- $w, i \models \mathbf{X}\phi$ - iff $w, i+1 \models \phi$
- $w, i \models \mathbf{F}\phi$ - iff $w, j \models \phi$ for at least one j where $i \leq j$
- $w, i \models \mathbf{G}\phi$ - iff $w, j \models \phi$ for all values of j where $i \leq j$
- $w, i \models \phi_1$ $\mathbf{U}$ ϕ_2 - iff there exists a $j \geq 0$ where $w, j \models \phi_2$ and $w, i \models \phi_1$, for all $0 \leq i < j$.

The semantics of the basic LTL operators are illustrated in Figure 3.2, the basic operators for the LTL are globally (always) p ($\mathbf{G}p$) which means that proposition p holds for all states of the trace starting from initial state, eventually p ($\mathbf{F}p$) which states the truth of p in at least one of the states, next p ($\mathbf{X}p$) which necessitates the trueness of p in the next (successive) state and p until q ($p\mathbf{U}q$) which means that p is true until q is true, *i.e.,* there exists a position i in which q is true, and p is true at all positions before i.

For instance, LTL formula $\mathbf{FG}o1$ expresses the property that observation $o1 \in O$ eventually holds at a point in the future and continues to hold for all future time after

this point. This interpretation, and the construction of valid LTL formulas is made precise in the following. Given a finite set of observations O, LTL formulas are interpreted over infinite sequences of subsets of O, that is, over 2^O which denotes the set of all subsets of O. The word $o(q_0), o(q_1), \ldots$ generated by a trajectory of a TS = $(Q, \mathcal{S}, \rightarrow, I, O, o)$ as defined in 3.2.1 can be checked against an LTL formula ϕ defined over O. We say that a trajectory $q_0, q_1, \ldots$ satisfies ϕ if the corresponding trace satisfies ϕ.

A word w satisfies an LTL formula ϕ if $w, 0 \models \phi$, which is denoted by $w \models \phi$. The set of all words satisfied by an LTL formula ϕ is called the language defined by ϕ and denoted by $\mathrm{L}(\phi)$. The set of all words that satisfy an LTL formula is accepted by a *deterministic Rabin automaton* [21]. We consider specifications expressed in a fragment of LTL, namely dLTL (deterministic LTL). The set of all words that satisfy a dLTL formula is accepted by a *deterministic Büchi automaton* [21]. dLTL is sufficient for the traffic system specifications described here. However, as discussed later, the proposed methods can be extended to full LTL at the cost of increased complexity.

Definition 3.3.1. *A Büchi automaton (BA) is a tuple of* $\mathcal{A} = (Q^{\mathcal{A}}, \Sigma, \delta, Q_0^{\mathcal{A}}, \mathcal{F})$*, where*

- $Q^{\mathcal{A}}$ *is a finite set of states,*
- Σ *is a finite alphabet of input symbols,*
- $\delta : Q^{\mathcal{A}} \times \Sigma \rightarrow 2^{Q^{\mathcal{A}}}$ *is a transition function,*
- $Q_0^{\mathcal{A}} \subseteq Q^{\mathcal{A}}$ *is a set of initial states,*
- $\mathcal{F} \subseteq Q^{\mathcal{A}}$ *is a set of accept (final) states.*

$\mathcal{A}$ is called deterministic if $|Q_0^{\mathcal{A}}| \leq 1$ *and* $|\delta(q^{\mathcal{A}}, \sigma)| \leq 1$ *for all states* $q^{\mathcal{A}} \in Q^{\mathcal{A}}$ *and all input symbols* $\sigma \in \Sigma$*.*

The size of $\mathcal{A}$ is the sum of the number of states and the number of transitions:

$$|\mathcal{A}| = |Q| + \sum_{q^{\mathcal{A}} \in Q^{\mathcal{A}}} \sum_{\sigma \in \Sigma} |\delta(q^{\mathcal{A}}, \sigma)|. \tag{3.7}$$

The automaton is defined on the input symbols of alphabet Σ and may start one of the states defined by set $Q_0^{\mathcal{A}}$. “Transition function" and “transition relation" notions

can be used interchangeably for δ throughout the thesis. We can identify transition function δ with the relation $\rightarrow \subseteq Q^{\mathcal{A}} \times \Sigma \times Q^{\mathcal{A}}$ as:

$$q^{\mathcal{A}} \xrightarrow{\sigma} q'^{\mathcal{A}} \text{ iff } q'^{\mathcal{A}} \in \delta(q^{\mathcal{A}}, \sigma)$$

which states that when we read the input symbol σ we can move from the state $q^{\mathcal{A}}$ to the state $q'^{\mathcal{A}}$ of the automaton.

The semantics of Büchi automaton are defined over infinite words in Σ^{ω}. A run for infinite word $w = \sigma_1\sigma_2 \ldots \in \Sigma^{\omega}$ in A is an infinite sequence of states $q_0^{\mathcal{A}} q_1^{\mathcal{A}} \ldots$ where $q_0^{\mathcal{A}} \in Q_0^{\mathcal{A}}$ and $q_i^{\mathcal{A}} \xrightarrow{\sigma_{i+1}} q_{i+1}^{\mathcal{A}}$ for all $i \geq 0$. The run is called accepting run if it visits the set of final (accepting) states $\mathcal{F}$ infinitely many times.

Definition 3.3.2. *A product automaton (it may also be denoted as product transition system) is the product of transition system $T = (Q, \mathcal{S}, \rightarrow, I, O, o)$ in 3.2.1 and automata $\mathcal{A} = (Q^{\mathcal{A}}, \Sigma, \delta, Q_0^{\mathcal{A}}, \mathcal{F})$ in 3.3.1.*

Then, product automaton is a tuple of $\mathcal{PA} = (Q^{\mathcal{P}}, \mathcal{S}, \rightarrow^{\mathcal{P}}, Q_0^{\mathcal{P}}, \mathcal{F}^{\mathcal{P}})$, where

- $Q^{\mathcal{P}} = Q \times Q^{\mathcal{A}}$ *set of states,*
- $\mathcal{S}$ *is a finite set of control inputs,*
- $\rightarrow^{\mathcal{P}}$ *: is the smallest relation defined by the rule* $\dfrac{q \xrightarrow{s} p \bigwedge q^{\mathcal{A}} \xrightarrow{o(q)} p^{\mathcal{A}}}{(q, q^{\mathcal{A}}) \xrightarrow{s}{}^{\mathcal{P}} (p, p^{\mathcal{A}})}$,

  i.e., $(q, q^{\mathcal{A}}) \xrightarrow{s}{}^{\mathcal{P}} (p, p^{\mathcal{A}})$ *if* $((q, q^{\mathcal{A}}), s, (p, p^{\mathcal{A}})) \in \xrightarrow{s}{}^{\mathcal{P}}$,
- $Q_0^{\mathcal{P}} = \{(q_0, q^{\mathcal{A}}) \mid (q_0 \in I) \bigwedge (\exists q_0^{\mathcal{A}} \in Q_0^{\mathcal{A}}$ *subject to* $q_0^{\mathcal{A}} \xrightarrow{o(q_0)} q^{\mathcal{A}})\}$*, is a set of initial states,*
- $\mathcal{F}^{\mathcal{P}} = Q \times \mathcal{F}$ *is a set of accept (final) states.*

A trajectory of product automaton, produced by a sequence of control inputs $s_0 s_1 \ldots$, is an infinite sequence $(q_0, q_0^{\mathcal{A}})(q_1, q_1^{\mathcal{A}}) \ldots$ such that $(q_0, q_0^{\mathcal{A}}) \in Q_0^{\mathcal{P}}$ and $(q_k, q_k^{\mathcal{A}}) \xrightarrow{s_k}{}^{\mathcal{P}} (q_{k+1}, q_{k+1}^{\mathcal{A}})$ for all $k = 0, 1, \ldots$. The trajectory $(q_0, q_0^{\mathcal{A}})(q_1, q_1^{\mathcal{A}}) \ldots$ is called accepting if it visits $\mathcal{F}^{\mathcal{P}}$ infinitely many times.

We use $W^{\mathcal{P}} \subseteq Q^{\mathcal{P}}$ to denote the winning region of the set of product automaton states, *i.e.,* $W^{\mathcal{P}}$ denotes the set of states where infinitely many visits to $\mathcal{F}^{\mathcal{P}}$ are guaranteed.

Definition 3.3.3. *A feedback control strategy $C : \bigcup_{t\in\mathbb{N}} C_t$ is a set of functions $C_t : W^{\mathcal{P}^{t+1}} \to \mathcal{S}$ where $s_t = C_t(q_0^{\mathcal{P}}, q_1^{\mathcal{P}}, \ldots, q_t^{\mathcal{P}})$ for set of control actions $\mathcal{S}$ and a subset of $Q^{\mathcal{P}}$ in product automata. We consider the feedback control strategy as a control automaton adapted from Büchi automaton and transition system. Thus, feedback control strategy is defined also as $C = (\mathcal{Q}, Q^{\mathcal{A}}, Q_0^{\mathcal{A}}, \mathcal{S}, f_n, f_u)$, where*

- *$\mathcal{Q}$ is a set of states (equal with set of states of transition system T),*
- *$Q^{\mathcal{A}}$ is set of states of Büchi automaton,*
- *$\mathcal{S}$ is set of control actions of transition system T,*
- *$Q_0^{\mathcal{A}}$ is set of initial states of Büchi automaton,*
- *$f_n : \mathcal{Q} \times Q^{\mathcal{A}} \to Q^{\mathcal{A}}$ is a next state function,*
- *$f_u : \mathcal{Q} \times Q^{\mathcal{A}} \to \mathcal{S}$ is an update function.*

f_u selects a control action based on the current state $q^{\mathcal{P}} = (q, q^{\mathcal{A}})$ of the control (or product) automaton if $(q, q^{\mathcal{A}}) \in W^{\mathcal{P}}$, and f_n updates the memory of the controller.

We define atomic propositions for links and signals as follows

$$x_l \leq D \quad \text{and} \quad s_l \tag{3.8}$$

where D is a constant integer and s_l holds simply when $s_l = 1$. For instance, if the proposition for the congestion is defined as “the number of vehicles (x_l) in a link exceeds the upper bound $\bar{x}_l$ for this link", the requirement that there is never a congestion in traffic system can be formulated as $\bigwedge_{l\in\mathcal{L}} \mathbf{G}\ x_l \leq \bar{x}_l$. Another example is to prevent continual high density and this requirement can be formulated as $\bigwedge_{l\in\mathcal{L}} \mathbf{G}\ \mathbf{F}\ x_l \leq \underline{x}_l$ (always eventually x_l will be less than or equal to $\underline{x}_l$). The final example is to provide fairness for signals (prevent the situation when one link is actuated for long time periods, while the others blocked) and can be formulated as $\mathbf{G}\ (\neg s_l \wedge \mathbf{X}\ \neg s_l \wedge \mathbf{X}\ \mathbf{X}\ \neg s_l \rightarrow \mathbf{X}\ \mathbf{X}\ \mathbf{X}\ s_l)$ for 3 time periods. Variations of those such as “do not block a link for long time periods when there are vehicles in the link" and other specifications can easily be expressed in LTL.

CHAPTER 4

CONTROLLER SYNTHESIS PROBLEM AND PROPOSED APPROACH

In this chapter, the formal control strategy synthesis problem for traffic systems is detailed formally. The abstraction based approach for the solution of this problem is outlined which is the commonly used approach proposed in state-of-the-art studies. Furthermore, our new approach, constructing subsystems and subspecifications and then generating abstractions of these systems is presented.

4.1 Problem Formulation

In this section, the Linear Temporal Logic (LTL) control problem for a traffic system is formulated, and the formal control strategy synthesis for a finite transition system (TS) is summarized.

Problem 1. *Given a traffic system* $(\mathcal{L}, \mathcal{V})$*, an LTL formula* Φ *of the form*

$$\Phi = \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_p \tag{4.1}$$

over a set of atomic propositions O defined as in (3.8) *over* $(\mathcal{L}, \mathcal{V})$*, find a control strategy and a set of initial states such that the trajectories of the controlled system originating from there satisfies* Φ*. A trajectory of the system satisfies* Φ *if the corresponding sequence of sets of atomic propositions satisfies* Φ*.*

In order to produce a Finite Memory Controller (FMC) (see Definition 3.2.3) for the given Problem 1, a finite representation of the traffic system is constructed as a TS

explained in detail in Chapter 6. A finite TS can be defined as a fully observable non-deterministic labeled transition system with finitely many states, control inputs and observations. An LTL verification for a TS is defined as follows: Given a finite TS and an LTL formula over its sets of atomic propositions, we check whether the language, *i.e.,* all possible words, of the finite TS starting from all initial states satisfies the LTL formula. This is a preliminary for LTL controller synthesis in this dissertation. In this thesis, an FMC for a TS from LTL formula is constructed by finding a set of initial states and a control strategy for all initial states such that the produced language of the finite TS satisfies the formula.

Example 4.1.1. *An LTL specification Φ^{ex} for the traffic system defined in Example 3.1.1 is given below. The goal in synthesis problem for this example is to find a feedback control strategy for the signals s_0 and s_1 such that the system trajectories satisfies formula Φ^{ex}:*

$$\Phi^{ex} = \Phi_1 \wedge \Phi_2 \wedge \Phi_3 \wedge \Phi_4$$
$$\Phi_1 = \mathbf{G}(x_0 < 30 \wedge x_1 < 30 \wedge x_3 < 15)$$
$$\Phi_2 = \mathbf{G}(x_2 < 30 \wedge x_4 < 15)$$
$$\Phi_3 = \mathbf{GF}(x_0 < 20 \wedge x_1 < 20 \wedge x_3 < 10)$$
$$\Phi_4 = \mathbf{GF}(x_2 < 20 \wedge x_4 < 10)$$

4.2 Abstraction-based Approach

The synthesis of correct-by-design controllers for hybrid dynamical systems like traffic systems is solved by first obtaining a finite state abstraction. Then, applying a game-based algorithm for synthesizing a control strategy for this abstraction to satisfy a Linear Temporal Logic (LTL) specification. The dynamics of traffic systems demonstrates a structure that enables construction of finite state abstractions in an efficient way. [14]. In particular, a finite state abstraction of the traffic system is constructed by only using linear operations (instead of polyhedral operations). The abstraction constructed with this method over-approximates the traffic system. The over-approximation guarantees that every violating property of the behaviours in the original traffic system with the given inputs is also present int he abstraction. Fur-

thermore, the automata based control synthesis approach from temporal logic specifications guarantees that the abstraction and the underlying system satisfy the specification. The synthesis algorithm, explained and illustrated in Chapter 6, depends on automata theory and uses fixed point algorithms to compute finite memory control strategies.

The discrete-time dynamical system for traffic network is in the form of as specified in Equation (3.4) where at time step t, $\mathbf{x}[t]$ is the number of vehicles on the link, $\mathbf{s}[t]$ is the signals in intersections and $d[t]$ is the exterior number of vehciles, the set of exogenous vehicle flow ($x_l[t+1] = F_l(\mathbf{x}[t], \mathbf{s}[t], d_l)$). This form models the real traffic system and we construct a finite abstraction of the system by partitioning its state space.

We use the symbols $\mathbb{R}_{\geq 0}$ and $\mathbb{Z}_{\geq 0}$ to denote the set of nonnegative real numbers and nonnegative integers, respectively.

$\mathcal{D} \subset \mathbb{R}^{\mathcal{L}}_{\geq 0}$

denotes the set of exterior vehicle flows to the traffic system. A maximum capacity for any link l is defined as $x_l^{cap} \in \mathbb{R}_{\geq 0}$ and the number of vehicles in this link at a time step $t \in \mathbb{Z}_{\geq 0}$ is defined as $x[t] \in [0, x_l^{cap}]$. Then we define state space of the traffic system as

$$\mathcal{X} = \prod_{l \in \mathcal{L}} [0, x_l^{cap}] \subset \mathbb{R}^{\mathcal{L}}_{\geq 0}. \tag{4.2}$$

$\mathcal{Q}$ is a finite index set projected from the domain $\mathcal{X}$. Then we define a rectangular grid partition for the continuous state space $\mathcal{X}$ and denote finite set of partitions as

$$\mathcal{Y} = \{\mathcal{X}_q\}_{q \in \mathcal{Q}}, \tag{4.3}$$

which satisfies $\mathcal{X} = \bigcup_{q \in \mathcal{Q}} \mathcal{X}_q$ and $\mathcal{X}_q \bigcap \mathcal{X}'_q = \emptyset$ for all q and q'. X. Each state of the finite abstraction is duly admitted by the each partition of the continuous state space.

Definition 4.2.1. *Given a partition $\mathcal{Y} = \{\mathcal{X}_q\}_{q \in \mathcal{Q}}$ of $\mathcal{X}$ in Equation* (3.4)*, a finite state abstraction can be defined as*

$\mathcal{T} = (\mathcal{Q}, \mathcal{S}, \rightarrow)$

where $\mathcal{Q}$ is a finite set of discrete states, $\mathcal{S}$ is a finite set of control inputs and $\rightarrow$ is the

transition function which maps input symbols (signals) and current states to a next state, i.e., $Q \times S \to 2^Q$.

This transition function guarantees that if there exists $d \in \mathcal{D}$, $x \in \mathcal{X}_q$ *and* $F(x,s,d) \in \mathcal{X}'_q$ *then* $q' \in\!\!\to (q,s)$ *for any* $q, q' \in Q$ *and* $s \in S$. *A finite state abstraction executes as* $q[t+1] \in\!\!\to (q[t], \boldsymbol{s}[t])$ *for all* $t \geq 0$.

4.3 Our New Synthesis Approach: Construction of Subsystems and Specifications

The solution we developed to solve Prob. 1 is based on:

1. decomposing the main problem into subproblems,
   and for every subproblem
   (a) constructing a finite abstraction as a TS for the subsystem,
   (b) solving formal synthesis problem for the abstract system,
   (c) and then combining controllers to generate a solution for the original system.

We first decompose the main traffic system $(\mathcal{L}, \mathcal{V})$ and specification formula Φ into a set of smaller systems $(\mathcal{L}^1, \mathcal{V}^1), \ldots, (\mathcal{L}^n, \mathcal{V}^n)$ and formulas $\Phi^1, \ldots, \Phi^n$ with the guarantee that if each $(\mathcal{L}^i, \mathcal{V}^i)$ satisfies Φ^i then $(\mathcal{L}, \mathcal{V})$ satisfies Φ. By this way, we aim at solving the scalability problem inherent in formal synthesis, hence synthesize control strategies for large traffic systems. Our decomposition method guarantees that each signal is controlled by a unique subsystem. The signals that lie in between two or more subsystems affect all of them and called *boundary signals*. While decomposing overall specification into the smaller pieces, we derive new specifications for the boundary signals to ensure fairness, i.e. to guarantee that the subsystem that controls the signal does not block the other subsystems continuously. The details of the decomposition method is given in Chapter 5.

Once the subsystems are defined, we compute a finite abstraction for each system $(\mathcal{L}^i, \mathcal{V}^i)$ as a TS T^i using the methods presented in [47] and solve formal synthesis

problem for the abstraction from its specification Φ^i [14, 47]. The main difference of our abstraction and synthesis algorithms from the existing ones [14, 47] is that we incorporate the derived constraints, and the properties (specifications and dynamics) of the links that are not in $\mathcal{L}^i$ but affect the evolution the links from $\mathcal{L}^i$ into the abstraction and the synthesis algorithms. The details of these methods are given in Chapter 6.

CHAPTER 5

TRAFFIC NETWORK PARTITIONING AND SPECIFICATION DERIVATION

As it was discussed before, solving formal synthesis problem for a large traffic system by applying the abstraction and formal synthesis algorithms directly is almost impossible due to the state-space explosion problem, i.e., due to the memory requirements for the abstraction and the computation time required to solve the synthesis problem. In this chapter, we present detailed description of the proposed partitioning algorithm and give information about how to derive specifications after partitioning.

5.1 Partitioning Algorithm

We propose to decompose the traffic system and LTL specifications into smaller parts. While conforming to the specifications and features of other subsystems, we produce one controller for each subsystem by considering its own links, signals and specifications. Our decomposition algorithm generates subsystems and their specifications by partitioning links and intersections with respect to the given specifications and network connectivity. Furthermore, we introduce new constraints on the boundary signals at the intersections of two subsystems. The constraints on a boundary signal v with $\mathcal{S}^v$ modes is denoted by

$$\{s_{min}, s_{max}\}_{s \in \mathcal{S}^v}, \tag{5.1}$$

which state minimum and maximum durations for each mode $s \in \mathcal{S}^v$, i.e., when the mode switches to s it has to stay in this mode at least s_{min} time units and it can not

stay in this mode longer than s_{max} time units. The subsystem that owns the boundary signal controls it under the given constraints, whereas the other subsystem generates a control strategy by considering all possible sequences for the boundary signal (non-deterministically).

Our partitioning algorithm assigns specifications to each subsystem on the basis of its links, signals and structure of the specifications while preserving overall specification. Furthermore, we interpolate new assumptions on the boundary signals at the intersections of two subsystems as follows: 1) the subsystem assigned with one of boundary signals obeys the assumptions while generating a controller for itself, 2) the other subsystem evaluates the boundary signal as satisfying assumptions non-deterministically. Thus, one of the interconnected subsystems controls the boundary signal whereas the other subsystem generates a control strategy by considering all possible sequences for the boundary signal. Given an LTL formula in the form of (4.1), we denote the set of links and signals included in Φ_i as $\mathcal{L}_{\Phi_i} \subseteq \mathcal{L}$ and $\mathcal{V}_{\Phi_i} \subseteq \mathcal{V}$, respectively (i.e. $l_k \in \mathcal{L}_{\Phi_i}$ if x_k appears in Φ_i).

Definition 5.1.1. *Given a traffic system ($\mathcal{L}$, $\mathcal{V}$), and a specification Φ as in* (4.1)*, a set of systems $\{(\mathcal{L}^1, \mathcal{V}^1), \ldots (\mathcal{L}^n, \mathcal{V}^n)\}$ is called a valid partition of ($\mathcal{L}$, $\mathcal{V}$) with respect to Φ if conditions 1-4 are satisfied:*

1. *for each Φ_i, there is a subsystem $(\mathcal{L}^j, \mathcal{V}^j)$ such that $\mathcal{L}_{\Phi_i} \subseteq \mathcal{L}^j$ and $\mathcal{V}_{\Phi_i} \subseteq \mathcal{V}^j$,*
2. *$\mathcal{L} = \mathcal{L}^1 \cup \ldots \cup \mathcal{L}^n$ and $\mathcal{L}^i \cap \mathcal{L}^j = \emptyset$ if $i \neq j$,*
3. *$\mathcal{V} = \mathcal{V}^1 \cup \ldots \cup \mathcal{V}^n$ and $\mathcal{V}^i \cap \mathcal{V}^j = \emptyset$ if $i \neq j$,*
4. *each subsystem $(\mathcal{L}^j, \mathcal{V}^j)$ is a connected system, i.e., there is a path between each link and signal that only contains the links and signals owned by this subsystem.*

A partition satisfying conditions 1-4 is constructed using adjacency graph for system $(\mathcal{L}, \mathcal{V})$ and formula Φ as follows:

First, the sets of links $\mathcal{L}_{\Phi_i}$ and signals $\mathcal{V}_{\Phi_i}$ are constructed for each Φ_i. Subsystems consisting of a single link or intersection are defined for each link l and signal v that

does not appear in any formula Φ_i, which forms the initial subsystem sets:

$$\begin{aligned}\mathcal{P} =& \{(\mathcal{L}_{\Phi_i}, \mathcal{V}_{\Phi_i}) \mid i = 1\ldots, p\} \bigcup \\ & \{(\{l\}, \emptyset) \mid l \in \mathcal{L}, l \notin \cup_{i=1,\ldots,p} \mathcal{L}_{\Phi_i}\} \bigcup \\ & \{(\emptyset, \{v\}) \mid v \in \mathcal{V}, v \notin \cup_{i=1,\ldots,p} \mathcal{V}_{\Phi_i}\}\end{aligned} \tag{5.2}$$

Note that this is not necessarily a valid partition as these sets can intersect. In the second step, the intersecting subsystems are merged iteratively, i.e. steps *i* and *ii* are applied if $\mathcal{L}_a \cap \mathcal{L}_b \neq \emptyset$ or $\mathcal{V}_a \cap \mathcal{V}_b \neq \emptyset$ for some $(\mathcal{L}_a, \mathcal{V}_a), (\mathcal{L}_b, \mathcal{V}_b) \in \mathcal{P}$:

$$\begin{aligned} i &: (\mathcal{L}_{ab}, \mathcal{V}_{ab}) = (\mathcal{L}_a \cup \mathcal{L}_b, \mathcal{V}_a \cup \mathcal{V}_b) \\ ii &: \mathcal{P} = (\mathcal{P} \setminus \{(\mathcal{L}_a, \mathcal{V}_a), (\mathcal{L}_b, \mathcal{V}_b)\}) \cup \{(\mathcal{L}_{ab}, \mathcal{V}_{ab})\} \end{aligned}$$

The iteration terminates when $\mathcal{L}_a \cap \mathcal{L}_b = \emptyset$ and $\mathcal{V}_a \cap \mathcal{V}_b = \emptyset$ for all $(\mathcal{L}_a, \mathcal{V}_a), (\mathcal{L}_b, \mathcal{V}_b) \in \mathcal{P}$ with $a \neq b$. By (5.2) and the termination condition, it is straightforward to verify that the resulting partition satisfies conditions 1,2 and 3.

In the last step, the connectivity for each $(\mathcal{L}_a, \mathcal{V}_a) \in \mathcal{P}$ is checked. If the condition is not met by some $(\mathcal{L}_a, \mathcal{V}_a)$, then required signals and links are found via a shortest path algorithm and the corresponding subsystems $(\mathcal{L}_b, \mathcal{V}_b)$ are merged with $(\mathcal{L}_a, \mathcal{V}_a)$ iteratively as shown in steps *i* and *ii* above. The iteration ends when all the subsystems are connected. Note that the subsystems obtained by merging does not violate conditions 1,2 and 3. The final set of subsystems obtained at this step is denoted by $\mathcal{P} = \{(\mathcal{L}^i, \mathcal{V}^i) \mid i = 1\ldots, n\}$.

5.2 Specification Derivation for Subsystems

In this section, we describe how the link and signal-based specifications are derived for the constructed subsystems. In addition, different techniques to determine those newly derived specifications are explained.

For each subsystem ($\mathcal{L}^i$, $\mathcal{V}^i$) defined in 5.1, the specification is defined as follows

$$\Phi^i = \bigwedge_{\mathcal{L}_{\Phi_j} \subseteq \mathcal{L}^i or \mathcal{V}_{\Phi_j} \subseteq \mathcal{V}^i \, j=1,\ldots,p} \Phi_j \tag{5.3}$$

Dependency sets Given a subsystem ($\mathcal{L}^i$, $\mathcal{V}^i$), the set of links and intersections that are not included in ($\mathcal{L}^i$, $\mathcal{V}^i$) but affect the evolution of some link $l \in \mathcal{L}^i$ are called *dependency sets* and denoted as $\mathcal{L}^{D-i}$ and $\mathcal{V}^{D-i}$, respectively. To formally define the dependency sets for subsystems, we first define the dependency sets $\mathcal{L}^{D-l}$ and $\mathcal{V}^{D-l}$ for a link l.

Let $down(l)$ and $up(l)$ denote the downstream and upstream intersections of a link l (e.g. $down(l_1)$ is s_1 and $up(l_1)$ is s_0 in Fig. 3.1). Similarly, let $down(v)$ and $up(v)$ denote the set of downstream and upstream links of intersection v, respectively (e.g. $down(s_1)$ is $\{l_2\}$ and $up(s_1)$ is $\{l_1, l_4\}$). An intersection affect link l if it is downstream to l or upstream to l:

$$\mathcal{V}^{D-l} = \{down(l), up(l)\} \tag{5.4}$$

According to the system dynamics (3.4), a link $\bar{l}$ affects link l if it is downstream to l ($\bar{l}$'s upstream intersection is l's downstream intersection) or it is upstream to l ($\bar{l}$'s downstream intersection is l's upstream intersection) or their upstream intersections are the same.

$$\begin{aligned}\mathcal{L}^{D-l} = \{\bar{l} \,|&up(\bar{l}) = down(l) \\ &\text{or } down(\bar{l}) = up(l) \text{ or } up(\bar{l}) = up(l)\}.\end{aligned} \tag{5.5}$$

The dependency sets of a subsystem ($\mathcal{L}^i$, $\mathcal{V}^i$) is formally defined as

$$\mathcal{L}^{D-i} = \{\bar{l} \in \mathcal{L} \setminus \mathcal{L}^i \mid \exists l \in \mathcal{L}^i \text{ s.t. } \bar{l} \in \mathcal{L}^{D-l}\}, \tag{5.6}$$

$$\mathcal{V}^{D-i} = \{v \in \mathcal{V} \setminus \mathcal{V}^i \mid \exists l \in \mathcal{L}^i \text{ s.t. } v \in \mathcal{V}^{D-l}\}. \tag{5.7}$$

$\cup_{i=1,\ldots,n} \mathcal{V}^{D-i}$ is the set of signals that lie in the boundary of two or more subsystems. The decomposition algorithm guarantees that only one of the subsystems "owns", and hence, controls such signals. As described before, we derive constraints on the minimum and maximum mode durations of such signals (Equation (5.1)). These constraints can be expressed as an LTL formula or encoded to a transition system modeling the signal. As explained in the next section, the latter approach is followed in this work for simplicity.

Example 5.2.1. *We applied the decomposition algorithm to the traffic system from Example 3.1.1 and formula Φ^{ex} from Ex. 4.1.1. As all the links appear in some for-*

mula and no signals appear in a formula, the initial subsystem set (5.2) *is composed of 5 sets (e.g.* $\mathcal{L}_{\Phi_1} = \{l_0, l_1, l_3\}, \mathcal{L}_{\Phi_2} = \{l_2, l_4\}$*). After merging the intersecting subsystems, we obtained the following set*

$$\{(\{l_0, l_1, l_3\}, \emptyset), (\{l_2, l_4\}, \emptyset), \\ (\emptyset, \{s_0\}), (\emptyset, \{s_1\})\}$$

After merging these according to connectivity analysis, we obtained two subsystems $(\mathcal{L}^1, \mathcal{V}^1)$ *and* $(\mathcal{L}^2, \mathcal{V}^2)$*:*

$$\mathcal{L}^1 = \{l_0, l_1, l_3\},\ \mathcal{V}^1 = \{s_0\}, \\ \mathcal{L}^2 = \{l_2, l_4\},\ \mathcal{V}^2 = \{s_1\}.$$

The corresponding specifications are

$$\Phi^1 = \Phi_1 \wedge \Phi_3 \text{ and } \Phi^2 = \Phi_2 \wedge \Phi_4.$$

The derived dependency sets are $\mathcal{L}^{D-1} = \{l_2\}$, $\mathcal{V}^{D-1} = \{s_1\}$, $\mathcal{L}^{D-2} = \{l_1\}$, $\mathcal{V}^{D-2} = \emptyset$*.* s_1 *is a boundary signal owned by subsystem-2. The constraint defined for* s_1 *is* $\{(2,3),(1,2)\}$ *(for horizontal and vertical actuation, respectively, see* (5.1)*).*

As it can be seen in 5.2.1, the signal s_1 is regarded as member of $\mathcal{V}_{\overline{2}}$ but it is external signal of $\mathcal{V}_{\overline{1}}$ because this signal is a boundary signal between two subsystems. Although s_1 signal is in the set of controlled signals of subsystem 2 ($\mathcal{V}_{\overline{2}}$), we need to define constraints for this signal as external signal for subsystem 1 to control each subsystem by respecting the behaviors of the other. The details of this approach are given in the Section 6.

CHAPTER 6

FORMAL CONTROLLER SYNTHESIS

The abstraction and synthesis algorithms and the main correctness results are presented in this section. As explained in Chapter 5 (Section 5.1), we decompose the traffic system $(\mathcal{L}, \mathcal{V})$ and the specification Φ into subsystems $\{(\mathcal{L}^i, \mathcal{V}^i)\}_{i=1,\ldots,n}$ and specifications $\{\Phi^i\}_{i=1,\ldots,n}$ (4.1), define dependency sets $\mathcal{L}^{D-i}$, $\mathcal{V}^{D-i}$ for each subsystem and derive constraints on the boundary signals.

6.1 Abstraction of Traffic Model

We construct finite abstraction of a system (or subsystem) $(\mathcal{L}, \mathcal{V})$ via a grid partition of the state space as follows:

$$\{0, x^{l,1}, \ldots, x^{l,M_l} = x_l^{cap}\}_{l \in \mathcal{L}}.$$

It is assumed that each threshold D from the atomic propositions (3.8) appears in the corresponding list, hence the partition is observation preserving and the observation maps are well defined. In the following, we describe how the abstract model, i.e, a transition system T^l, is constructed for a link $l \in \mathcal{L}$, and then define abstractions for subsystems from such link transition systems.

We first define a projection function $J_{A \to B}(\cdot)$, which is used to simplify the presentation of the results given in this section. For sets of variables $A = \{x_1, \ldots, x_m\}$ and B with $B \subseteq A$, the projection of a valuation $[a_1, \ldots, a_m]$ of variables in A to the valuation of the variables in B is denoted by $J_{A \to B}([a_1, \ldots, a_m])$, where a_i appears in $J_{A \to B}([a_1, \ldots, a_m])$ if $x_i \in B$.

6.1.1 Link Transition System

Definition 6.1.1 (Link TS)**.** *Given a traffic system ($\mathcal{L}$, $\mathcal{V}$), grid partition $\{0, x^{\bar{l},1}, \ldots, x^{\bar{l},M_{\bar{l}}} = x^{cap}_{\bar{l}}\}_{\bar{l}\in\mathcal{L}}$, transition system for link $l \in \mathcal{L}$ is defined as*

$$T^l = (Q^l, \mathcal{S}^l, \rightarrow^l),$$

- $Q^l = \{q^l_1, \ldots, q^l_{M_l}\}$. *$q^l_i$ represents region $[x^{l,i-1}, x^{l,i})$ in the given grid partition. The region represented by a state q is denoted by R_q.*
- $\mathcal{S}^l = \Pi_{\bar{l}\in\mathcal{L}^{D-l}} Q^{\bar{l}} \times \Pi_{v\in\mathcal{V}^{D-l}} S_v$ *($\mathcal{L}^{D-l}$ and $\mathcal{V}^{D-l}$ are defined in* (5.5) *and* (5.4)*,respectively.)*
- $(q^l_a, (\mathbf{q}, \mathbf{s}), q^l_b) \in \rightarrow^l$, *if there exists $d_l \in [d^{min}_l, d^{max}_l]$ and $\mathbf{x}_a, \mathbf{x}_b \in \mathcal{X}$ such that*
  1. $J_{\mathcal{L}\rightarrow\{l\}}(\mathbf{x}_a) \in R_{q^l_a}$
  2. $J_{\mathcal{L}\rightarrow\{l\}}(\mathbf{x}_b) = x^l \in R_{q^l_b}$
  3. $x^l = F_l(\mathbf{x}_a, \mathbf{s}, d_l)$
  4. *for each $\bar{l} \in \mathcal{L}^{D-l}$ it holds that $J_{\mathcal{L}\rightarrow\{\bar{l}\}}(\mathbf{x}_a) \in R_{\bar{q}}$, where $\bar{q} = J_{\mathcal{L}\rightarrow\{\bar{l}\}}(\mathbf{q})$.*

Link transition system T^l for link l represents all possible transitions with respect to the states of the finite models of links $\mathcal{L}^{D-l}$ and signals $\mathcal{V}^{D-l}$ that affect the evolution of the number of vehicles on link l. We use the algorithm proposed in [23] for the computation of the transition relation ($\rightarrow^l$). It is shown that the transition relation can be obtained with linear operations when the state space of the traffic system is divided into rectangular regions [23].

6.1.2 Signal Transition System

Definition 6.1.2 (Signal TS)**.** *Given signal $v \in \mathcal{V}$ and its set of modes $\mathcal{S}^v$, fairness constraints as defined in* (5.1)*, signal transition system is defined as*

$$T^v = \{Q^v, \mathcal{S}^v, \rightarrow^v\},$$

- $Q^v = \cup_{s\in\mathcal{S}_v}\{(s,1), \ldots, (s, s_{max})\}$

- $((s,i),s',(s',1)) \in \rightarrow^v$ *if* $i \geq s_{min}$ *and* $s \neq s'$*, and* $((s,i),s,(s,i+1)) \in \rightarrow^v$ *if* $i < s_{max}$

The amount of time passed since the last mode change is encoded in the states of T^v, and its transitions ensure that the fairness constraints are satisfied. A signal transition system $T^v = \{Q^v, S^v, \rightarrow^v\}$ is defined for each signal, where S^v denotes the set of signal modes, and the states are defined to count the time passed since the last mode change. Given a signal constraint in the form (5.1), a transition system with $v_{g,max} + v_{r,max}$ states constructed.

6.1.3 Subsystem Transition System

Definition 6.1.3 (Subsystem TS)**.** *Given a subsystem* $(\mathcal{L}^i, \mathcal{V}^i)$*, the dependence sets* $\mathcal{L}^{D-i}, \mathcal{V}^{D-i}$*, and the corresponding link and signal transition systems, transition system for the subsystem is defined as*

$$T^i = (Q^i, \mathcal{S}^i, \rightarrow^i, O^i, o^i),$$

where $\mathcal{T}^i = \mathcal{L}^i \cup \mathcal{V}^i \cup \mathcal{L}^{D-i} \cup \mathcal{V}^{D-i}$ *denote the set of links and signals that appear in the transition system and*

- $Q^i = \Pi_{l \in \mathcal{L}^i} Q^l \times \Pi_{l \in \mathcal{L}^{D-i}} Q^l \times \Pi_{v \in \mathcal{V}^i} Q^v \times \Pi_{v \in \mathcal{V}^{D-i}} Q^v$
- $\mathcal{S}^l = \Pi_{v \in \mathcal{V}^l} \mathcal{S}^v$
- $(\mathbf{q}, \mathbf{s}, \mathbf{q}') \in \rightarrow^i$ *if (let* $J_{\mathcal{T}^i \rightarrow X}(q)$ *for* $q \in Q^i$ *denote the states of links* $x \in X$ *and signals* $x \in X$ *of the corresponding transition systems* T^x*)*
  1. *for each* $v \in \mathcal{V}^i$ *it holds that* $(J_{\mathcal{T}^i \rightarrow \{v\}}(\mathbf{q}), J_{\mathcal{V}^i \rightarrow \{v\}}(\mathbf{s}), J_{\mathcal{T}^i \rightarrow \{v\}}(\mathbf{q}') \in \rightarrow^v$
  2. *for each* $l \in \mathcal{L}^i \cup \mathcal{L}^{D-i}$*, there exists* $\mathbf{s}' \in \mathcal{S}^l$ *such that* $J_{\mathcal{T}^i \rightarrow \mathcal{T}^i \cap (\mathcal{L}^{D-l} \cup \mathcal{V}^{D-l})}(\mathbf{s}) = J_{\mathcal{L}^{D-l} \cup \mathcal{V}^{D-l} \rightarrow \mathcal{T}^i \cap (\mathcal{L}^{D-l} \cup \mathcal{V}^{D-l})}(\mathbf{s}')$ *and* $(J_{\mathcal{T}^i \rightarrow \{l\}}(\mathbf{q}), \mathbf{s}', J_{\mathcal{T}^i \rightarrow \{l\}}(\mathbf{q}') \in \rightarrow^l$

The set of propositions $O^i \subseteq O$ *(see Prob.1) is the set of propositions defined over the links from the set* $\mathcal{L}^i \cup \mathcal{L}^{D-i}$ *or signals from the set* $\mathcal{V}^i \cup \mathcal{V}^{D-i}$*, and the observation map* o^i *is defined accordingly.*

Signals from $\mathcal{V}^i$ and $\mathcal{V}^{D-i}$ are represented in T^i. However, only signals from the set $\mathcal{V}^i$ are controlled by T^i, the rest can change modes non-deterministically (controlled by other subsystems). Note that signals from both of the sets are guaranteed to satisfy the constraints (5.1) via the signal transition systems.

6.2 Formal Synthesis for The Finite Systems

The formal synthesis problem for T^i from LTL formula Φ^i asks for finding a set of states Q^i_{sat} and a feedback control strategy C_{Φ^i} such that the closed loop trajectories of T^i satisfies Φ^i. Note that, the LTL formula Φ^i obtained from the decomposition algorithm for subsystem-i is over O^i. To solve the synthesis problem, we apply the Büchi game based approach. As it is well-documented in literature [14], we omit the details, and present a short summary:

1. Construct a deterministic Büchi automaton A_{Φ^i} from Φ^i,
2. Construct the synchronous product of A_{Φ^i} and T^i, which results in a non-deterministic Büchi automaton,
3. Finally, find a set of initial states Q^i_{sat} and control strategy C_{Φ^i} by solving a Büchi game on the product.

For computational efficiency, the overall approach can be applied using a deterministic Büchi automaton and solving a Büchi game if Φ^i is a deterministic LTL specification [14]. We compute a set of satisfying states Q^i_{sat} and a finite memory feedback control strategy C_{Φ^i} for each subsystem-i $(\mathcal{L}^i, \mathcal{V}^i)$ and it's finite abstraction T^i. At time step k, given the last m states $q_{k-m}, \ldots, q_k$, control strategy C_{Φ^i} produces the signal modes (control inputs) for signals in $\mathcal{V}^i$ $(C_{\Phi^i} : (Q^i)^+ \to \mathcal{V}^i)$. The number of states required by the strategy is bounded and depends on the automaton A_{Φ^i} [14].

Satisfying initial states. Let $T = (Q, \mathcal{S}, \to, O, o)$ be the transition system constructed for $(\mathcal{L}, \mathcal{V})$ as in Definition 6.1.3. A state $q \in Q$ of this system is marked as satisfying if all the projections of this state to the states of the subsystems are satisfying:

$$Q_{sat} = \{\mathbf{q} \in Q \mid \forall i \in \{1, \ldots, n\}. J_{\mathcal{L}\cup\mathcal{V}\to\mathcal{T}^i}(\mathbf{q}) \in Q^i_{sat}\}. \tag{6.1}$$

We use the control strategies $C_{\Phi^1}, \ldots, C_{\Phi^n}$, to produce the signal modes for each signal in $\mathcal{V}$ for trajectories of the traffic system originating from

$$\mathcal{X}^{\Phi} = \prod_{i=1,\ldots,n} \left(\bigcup_{q \in Q_{sat}} R_q \right). \tag{6.2}$$

Note that the resulting strategy is decentralized and each controller produces a control signal based on the local information. Next, we prove the correctness of our solution: the trajectories of the traffic system $(\mathcal{L}, \mathcal{V})$ originating from $\mathcal{X}^{\Phi}$ satisfy formula Φ. The proof is based on a finite model T of the whole traffic network.

Theorem 6.2.1. *For a traffic system $(\mathcal{L}, \mathcal{V})$, a set of atomic propositions O defined as in* (3.8) *and formula Φ* (4.1)*, let $\{(\mathcal{L}^i, \mathcal{V}^i)\}_{i=1,\ldots,n}$ be a valid partition as defined in Definition 5.1.1, and $\{\Phi^i\}_{i=1,\ldots,n}$ be the corresponding subsystem formulas as defined in* (5.3)*. Furthermore, for each $i = 1, \ldots, n$, let T^i be the transition system constructed as in Definition 6.1.3, and Q^i_{sat}, C_{Φ^i} be the set of initial states and control strategy, respectively, obtained from formal controller synthesis for T^i from Φ^i. Finally, let $T = (Q, \mathcal{S}, \rightarrow, O, o)$ be the transition system constructed for $(\mathcal{L}, \mathcal{V})$ as in Definition 6.1.3. Then, the trajectories of T in closed loop with $\{C_{\Phi^i}\}_{i=1,\ldots,n}$ originating from Q_{sat}* (6.1) *satisfy Φ.*

Proof. Let $\mathcal{T}$ denote $\mathcal{L} \cup \mathcal{V}$ and $\mathcal{T}^i$ denote $\mathcal{L}^i \cup \mathcal{V}^i$ throughout the proof. Let $\mathbf{q}_0 \in Q_{sat}$, and $\mathbf{q}_0\mathbf{q}_1\mathbf{q}_2\ldots$ be a trajectory produced by the controlled system and $\mathbf{s}_0\mathbf{s}_1\mathbf{s}_2\ldots$ be the corresponding control sequence, i.e., for each $k = 0, 1, 2, \ldots$

$$\begin{aligned} J_{\mathcal{V}\rightarrow\mathcal{V}^i}(\mathbf{s}_k) = C_{\Phi^i}(&J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_{k-N_i}), \\ &J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_{k-N_i+1}), \ldots, J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_k)) \quad i = 1, \ldots, n. \end{aligned} \tag{6.3}$$

where N_i is the number of states required to be known by C_{Φ^i}. The control input $\mathbf{s}_k$ as defined in (6.3) is well-defined, since the signal sets $\mathcal{V}^i$ do not intersect and their union equals to $\mathcal{V}$ by Definition 5.1.1, and each C_{Φ^i} produce controls for $\mathcal{V}^i$. Consider the projection of the trajectory onto Q^i

$$J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_0)J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_1)J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_2)\ldots \tag{6.4}$$

By definition of Q_{sat}, $J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}_0) \in Q^i_{sat}$. By Definition 6.1.3, for any $k \geq 0$, $(\mathbf{q}_k, \mathbf{s}_k, \mathbf{q}_{k+1}) \in \to$ implies that

$$(J_{\mathcal{T}\to\{l\}}(\mathbf{q}_k), \mathbf{s}'_k, J_{\mathcal{T}\to\{l\}}(\mathbf{q}_{k+1}) \in \to^l$$

for each link $l \in \mathcal{L}$, where

$$\mathbf{s}'_k = (J_{\mathcal{T}\to\mathcal{L}^{D-l}}(\mathbf{q}_k), J_{\mathcal{V}\to\mathcal{V}^{D-l}}(\mathbf{s}_k)).$$

Hence $(J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}_k), J_{\mathcal{V}\to\mathcal{V}^i}(\mathbf{s}_k), J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}_{k+1})) \in \to^i$. Observing that $J_{\mathcal{V}\to\mathcal{V}^i}(\mathbf{s}_k)$ is produced by C_{Φ^i} at each time step, we conclude that the projected trajectory (6.4) is a trajectory of T^i that originates from Q^i_{sat} and is produced in closed loop with C_{Φ^i}, hence satisfies Φ^i. The decomposition algorithm guarantees that the atomic propositions appear in subformulas Φ^i do not intersect. Hence, for each $i = 1, \ldots, n$, the corresponding trajectory (6.4) satisfies Φ^i and $\Phi = \wedge_{i=1,\ldots,n}\Phi^i$. Consequently, $\mathbf{q}_0\mathbf{q}_1\mathbf{q}_2\ldots$ satisfies Φ. □

Theorem 6.2.1 and its proof indicate that the trajectories of the finite transition system T constructed for the overall traffic system satisfies Φ when the control strategies synthesized for the subsystems are used. Note that T simulates the traffic system (3.4). Consequently, as proven in [23], the trajectories of the traffic system originating from $\mathcal{X}^\Phi$ (6.2) satisfy the specification.

The proposed solution increases the conservatism in the abstraction based approaches. In other words, the developed synthesis algorithm might fail to find a strategy, when there is one for the overall transition system T. The first source of the additional conservatism is the signal constraints (5.1). In the algorithm, a control strategy for the subsystem that is adjacent to a boundary signal but does not own it, is generated by considering all possible behaviors of the boundary signal. It would be unlikely to find such a strategy if the signal could block some links continuously. Consequently, these constraints are necessary for the developed synthesis algorithm. This conservatism can be reduced by performing a search on the boundary constraints. Since the constraints are integer valued, a simple grid search would suffice.

The second source of conservatism is that a subsystem considers all possible behaviors of links in $\mathcal{L}^{D-i}$ including the ones that violate Φ. Two methods are presented

to reduce this limitation. The correctness proofs for both of the methods follow the main arguments given in the proof of Theorem 6.2.1.

6.2.1 Eliminating States Violating Safety Properties (ESVSP)

Consider link l, subsystems i and j, assume that $l \in \mathcal{L}^{D-i}$ and $l \in \mathcal{L}^{j}$, and $\mathbf{G}x_l < \bar{x}_l$ is part of Φ^j. In this case, while all reachable states of l including the ones with $x_l > \bar{x}_l$ will be considered in subsystem-i, the synthesized control strategy for subsystem-j will guarantee that such states will not be reached. For these types of formulas, we prune T^i by removing all states that violate $x_l < \bar{x}_l$ and the transitions of such states. Note that pruning is not applied to T^j with $l \in \mathcal{L}^j$ as we need to find strategies avoiding such states.

The definition of the link transition system given in 6.1.1 is reorganized for the method **(ESVSP)**. We define set of states for each link for **(ESVSP)** as

$$Q^{l,(\mathbf{ESVSP})} = [\mathbf{q} \in Q^l] \tag{6.5}$$

where the region represented by a state $\mathbf{q}$ is denoted by $R_{\mathbf{q}}$. In Definition 6.1.1, fourth item of transition relation is also adapted as

$$\begin{aligned} &\text{for each } \bar{l} \in \mathcal{L}^{D-l} \text{ it holds that } J_{\mathcal{L}\to\{\bar{l}\}}(\mathbf{x}_l) \in R_{\bar{\mathbf{q}}}, \\ &\text{where } R_{\bar{\mathbf{q}}} \subseteq [0, \bar{\mathbf{x}}_l] \text{ and } \bar{\mathbf{q}} = J_{\mathcal{L}\to\{\bar{l}\}}(\mathbf{q}). \end{aligned} \tag{6.6}$$

Now, as in theorem 6.2.1 we prove that the trajectories of the traffic system $(\mathcal{L}, \mathcal{V})$ originating from $\mathcal{X}^{\Phi}$ satisfy formula Φ when **(ESVSP)** method is used in controller synthesis.

Theorem 6.2.2. *For a traffic system $(\mathcal{L}, \mathcal{V})$, a set of atomic propositions O defined as in* (3.8) *and formula Φ* (4.1)*, let $\{(\mathcal{L}^i, \mathcal{V}^i)\}_{i=1,\dots,n}$ be a valid partition as defined in Definition 5.1.1, and $\{\Phi^i\}_{i=1,\dots,n}$ be the corresponding subsystem formulas as defined in* (5.3)*. Furthermore, for each $i = 1, \dots, n$, let T^i be the transition system constructed as in Definition 6.1.3, and Q^i_{sat}, C_{Φ^i} be the set of initial states and control strategy, respectively, obtained from formal controller synthesis for T^i from Φ^i when **(ESVSP)** method is used.*

Finally, let $T = (Q, \mathcal{S}, \rightarrow, O, o)$ be the transition system constructed for $(\mathcal{L}, \mathcal{V})$ as in Definition 6.1.3. Then, the trajectories of T in closed loop with $\{C_{\Phi^i}\}_{i=1,\ldots,n}$ originating from Q_{sat} (6.1) *satisfy* Φ.

Proof. Let $\mathcal{T}$ denote $\mathcal{L} \cup \mathcal{V}$ and $\mathcal{T}^i$ denote $\mathcal{L}^i \cup \mathcal{V}^i$ throughout the proof. Let $\mathbf{q}_0 \in Q_{sat}$, and $\mathbf{q}_0\mathbf{q}_1\mathbf{q}_2\ldots$ be a trajectory produced by the controlled system and $\mathbf{s}_0\mathbf{s}_1\mathbf{s}_2\ldots$ be the corresponding control sequence, i.e., for each $k = 0, 1, 2, \ldots$

$$J_{\mathcal{V}\rightarrow\mathcal{V}^i}(\mathbf{s}_k) = C_{\Phi^i}(J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_{k-N_i}),$$
$$J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_{k-N_i+1}),\ldots,J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_k)) \quad i = 1,\ldots,n.$$

where N_i is the number of states required to be known by C_{Φ^i}. The control input $\mathbf{s}_k$ as defined in (6.3) is well-defined, since the signal sets $\mathcal{V}^i$ do not intersect and their union equals to $\mathcal{V}$ by Definition 5.1.1, and each C_{Φ^i} produce controls for $\mathcal{V}^i$. Consider the projection of the trajectory onto Q^i

$$J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_0)J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_1)J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_2)\ldots \tag{6.7}$$

By definition of Q_{sat}, $J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_0) \in Q^i_{sat}$. By Definition 6.1.3, for any $k \geq 0$, $(\mathbf{q}_k, \mathbf{s}_k, \mathbf{q}_{k+1}) \in \rightarrow$ implies that

$$(J_{\mathcal{T}\rightarrow\{l\}}(\mathbf{q}_k), \mathbf{s}'_k, J_{\mathcal{T}\rightarrow\{l\}}(\mathbf{q}_{k+1}) \in \rightarrow^l$$

for each link $l \in \mathcal{L}$, where

$$\mathbf{s}'_k = (J_{\mathcal{T}\rightarrow\mathcal{L}^{D-l}}(\mathbf{q}_k), J_{\mathcal{V}\rightarrow\mathcal{V}^{D-l}}(\mathbf{s}_k)).$$

Hence $(J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_k), J_{\mathcal{V}\rightarrow\mathcal{V}^i}(\mathbf{s}_k), J_{\mathcal{T}\rightarrow\mathcal{T}^i}(\mathbf{q}_{k+1})) \in \rightarrow^i$.

Now, assume that $l_{sp} \in \mathcal{L}^{D-i}$ and $l_{sp} \in \mathcal{L}^j$ for subsystems i and j. Assume that there is a trajectory $\mathbf{q}_0\mathbf{q}_1\mathbf{q}_2\ldots\mathbf{q}_m\ldots$ of the controlled system and in this trajectory, q_m violates the safety property $x_{l_{sp}} < \bar{x}_{l_{sp}}$ for link l_{sp}:

$$l_{sp} \in \mathcal{L}^{D-i} J_{\mathcal{L}\rightarrow\{l_{sp}\}}(\mathbf{q}_m = q_m^{l_{sp}}) \in R_{q_m} \text{ where } R_{\mathbf{q}_m^{l_{sp}}} \not\subseteq [0, \bar{x}_{l_{sp}}].$$

However, this trajectory cannot be part of the solution because the controller C_{Φ^j} will not produce the required control input due to the violating property of the state $\mathbf{q}_m$. Therefore, we can eliminate such transitions in T^i as in (6.6).

Observing that subsystem-i will guarantee that $\mathbf{q}_m$ will not be reachable, we conclude that the projected trajectory (6.7) is a trajectory of T^i that originates from Q^i_{sat} which uses Equation (6.5) and (6.6). Furthermore, this trajectory is produced in closed loop with C_{Φ^i}, hence satisfies Φ^i. The decomposition algorithm guarantees that the atomic propositions appear in subformulas Φ^i do not intersect. Hence, for each $i = 1,\ldots,n$, the corresponding trajectory (6.7) satisfies Φ^i and $\Phi = \wedge_{i=1,\ldots,n}\Phi^i$. Consequently, $\mathbf{q}_0\mathbf{q}_1\mathbf{q}_2\ldots$ where each $\mathbf{q}$ with $R_{\mathbf{q}} \subseteq \Pi_{l\in L}[0,\bar{x}_l]$ satisfies Φ.

□

6.2.2 Eliminating Transitions Leading to a Trap State (ETLTS)

Consider a link $l \in \mathcal{L}^i$, assume that $\Phi^i = \Phi^{i,\mathcal{L}^i-l} \wedge \Phi^{i,l}$ and $\Phi^{i,l}$ is over propositions defined on l. We construct Büchi automaton defined as in 3.3.1 from $\Phi^{i,l}$. In the Büchi automaton, we mark the states that are not accepting final state and that do not have a transition to another state as “trap state". Note that such states can not be part of any accepting run since they are not accepting and they do not have outgoing transitions. Then, we take the product of this Büchi automaton with T^l as in Definition 3.3.2. We eliminate the transitions leading to a “trap state" in the product automaton. We use the product automaton instead of T^l to synthesize controller for a subsystem with $l \in \mathcal{L}^{D-j}$. Note that the eliminated transitions will be avoided by the control strategy generated for subsystem-i.

The set of trap states in the Büchi automaton from $\Phi^{i,l}$ is a subset of $Q^{\mathcal{A}}$ defined in 3.3.1 and denoted as:

$$Q^{\mathcal{A}}_{tp} = \{q^{\mathcal{A}} \in Q^{\mathcal{A}} \mid \bigcup_{\sigma\in\Sigma}(q^{\mathcal{A}} \notin \mathcal{F} \bigwedge \delta(q^{\mathcal{A}},\sigma) = \emptyset)\}. \tag{6.8}$$

$\delta(q^{\mathcal{A}},\sigma) = \emptyset$ in Equation (6.8) can also be denoted as $(\neg(\exists q^{\mathcal{A}}_z \text{ s.t. } q^{\mathcal{A}} \xrightarrow{\sigma} q^{\mathcal{A}}_z))$. It is apparent that $Q^{\mathcal{A}}_{tp} \cap \mathcal{F} = \emptyset$, *i.e.,* any trap state cannot be a member of set of accepting final states in automaton as stated before.

Now, the cartesian product of states of the TS and the states of the automaton, which contains marked “trap states", is taken as defined in 3.3.2. We use an iterative procedure to determine and eliminate the transitions leading to a marked trap state. Firstly,

we define the set of trap states in the product automaton as:

$$Q_{tp}^{\mathcal{P}} = \{(q, q_{tp}^{\mathcal{A}}) \in Q^{\mathcal{P}} \mid q_{tp}^{\mathcal{A}} \in Q_{tp}^{\mathcal{A}}\}. \tag{6.9}$$

We use $q^{\mathcal{P}}$ and $(q, q^{\mathcal{A}})$ interchangeably in equations because the set of product automaton states are obtained as a cartesian product of the TS and Büchi automaton states. Then, the set of trap states in product automaton is obtained by adding states which are in the form of:

$$(q, q^{\mathcal{A}}) \notin \mathcal{F}^{\mathcal{P}} \text{ and } \forall s \in \mathcal{S}^{\mathcal{P}}, \rightarrow^{\mathcal{P}}((q, q^{\mathcal{A}}), s) \cap Q_{tp}^{\mathcal{P}} \neq \emptyset. \tag{6.10}$$

All product automaton states in the form of $q^{\mathcal{P}} = (q, q_{tp}^{\mathcal{A}})$ where $q_{tp}^{\mathcal{A}} \in Q_{tp}^{\mathcal{A}}$ are marked for the next procedure. In the next stage, all product automaton transitions $\rightarrow^{\mathcal{P}}((q, q_k^{\mathcal{A}}), s_{tp}, (q, q_{tp}^{\mathcal{A}}))$ are eliminated from the product automaton of subsystem-i. The same procedure is applied for subsystem-j with $l \in \mathcal{L}^{D-j}$ during the computation of the product of its automaton and its transition system T^j. Consequently, the behaviours of the neighbour links that are not allowed by the specification are eliminated in a subsystem.

Theorem 6.2.3. *For a traffic system $(\mathcal{L}, \mathcal{V})$, a set of atomic propositions O defined as in* (3.8) *and formula Φ* (4.1)*, let $\{(\mathcal{L}^i, \mathcal{V}^i)\}_{i=1,\ldots,n}$ be a valid partition as defined in Definition 5.1.1, and $\{\Phi^i\}_{i=1,\ldots,n}$ be the corresponding subsystem formulas as defined in* (5.3)*. Furthermore, for each $i = 1, \ldots, n$, let T^i be the transition system constructed as in Definition 6.1.3, $\mathcal{A}^i$ be the Büchi automaton as in 3.3.1, and $\mathcal{PA}^i$ be the product of T^i and $\mathcal{A}^i$ as defined in 3.3.2 and Q_{sat}^i, C_{Φ^i} be the set of initial states and control strategy, respectively, obtained from formal controller synthesis for $\mathcal{PA}^i$ from Φ^i when **(ETLTS)** method is used.*

Finally, let $\mathcal{PA} = (Q^{\mathcal{P}}, \mathcal{S}, \rightarrow^{\mathcal{P}}, Q_0^{\mathcal{P}}, o^{\mathcal{P}}, \mathcal{F}^{\mathcal{P}})$ be the product automaton constructed for $(\mathcal{L}, \mathcal{V})$ as in Definition 3.3.2. Then, the trajectories of $\mathcal{PA}$ in closed loop with $\{C_{\Phi^i}\}_{i=1,\ldots,n}$ originating from Q_{sat} (6.1) *satisfy Φ.*

Proof. Let $\mathcal{T}$ denote $\mathcal{L} \cup \mathcal{V}$ and $\mathcal{T}^i$ denote $\mathcal{L}^i \cup \mathcal{V}^i$ throughout the proof. Let $\mathbf{q}_0^{\mathcal{P}} = (\mathbf{q}_0, \mathbf{q}_0^{\mathcal{A}})$ where $\mathbf{q}_0 \in Q_{sat}$ and $\mathbf{q}_0^{\mathcal{A}} \in Q_0^{\mathcal{A}}$, and $\mathbf{q}_0^{\mathcal{P}} \mathbf{q}_1^{\mathcal{P}} \mathbf{q}_2^{\mathcal{P}} \ldots$ be a product automaton path including a state $\mathbf{q}_i^{\mathcal{P}} = (\mathbf{q}_i, \mathbf{q}_i^{\mathcal{A}})$ where $\mathbf{q}_i^{\mathcal{A}} \in \mathcal{F}$ visited for infinitely many indices i. In addition, let the product automaton path $\mathbf{q}_0^{\mathcal{P}} \mathbf{q}_1^{\mathcal{P}} \mathbf{q}_2^{\mathcal{P}} \ldots$ be produced by the controlled system and $\mathbf{s}_0 \mathbf{s}_1 \mathbf{s}_2 \ldots$ be the corresponding control sequence, i.e., for each

$k = 0, 1, 2, \ldots$

$$J_{\mathcal{V}\to\mathcal{V}^i}(\mathbf{s}_k) = C_{\Phi^i}(J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{P}}_{k-N_i}), \\ J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{P}}_{k-N_i+1}), \ldots, J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{P}}_{k})) \quad i = 1, \ldots, n. \tag{6.11}$$

where N_i is the number of states required to be known by C_{Φ^i}. The control input $\mathbf{s}_k$ as defined in (6.11) is well-defined, since the signal sets $\mathcal{V}^i$ do not intersect and their union equals to $\mathcal{V}$ by Definition 5.1.1, and each C_{Φ^i} produce controls for $\mathcal{V}^i$. Consider the projection of the trajectory onto $Q^{\mathcal{P},i}$

$$J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{P}}_0) J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{P}}_1) J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{P}}_2) \ldots \tag{6.12}$$

As $\mathbf{q}^{\mathcal{P}}_j = (\mathbf{q}_j, \mathbf{q}^{\mathcal{A}}_j)$, we derive also $J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}_j)$ and $J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{A}}_j)$. By definition of Q_{sat}, $J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}_0) \in Q^i_{sat}$. By definition of product automaton, $J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}^{\mathcal{A}}_0) \in Q^{\mathcal{A}}_0$. By Definition 6.1.3, and 3.3.2 for any $k \geq 0$, $((\mathbf{q}_k, \mathbf{q}^{\mathcal{A}}_k), \mathbf{s}_k, (\mathbf{q}_{k+1}, \mathbf{q}^{\mathcal{A}}_{k+1})) \in \rightarrow^{\mathcal{P}}$ implies that

$$(J_{\mathcal{T}\to\{l\}}(\mathbf{q}_k, \mathbf{q}^{\mathcal{A}}_k), \mathbf{s}'_k, J_{\mathcal{T}\to\{l\}}(\mathbf{q}_{k+1}, \mathbf{q}^{\mathcal{A}}_{k+1}) \in \rightarrow^{\mathcal{P},l}$$

for each link $l \in \mathcal{L}$, where

$$\mathbf{s}'_k = (J_{\mathcal{T}\to\mathcal{L}^{D-l}}(\mathbf{q}_k, \mathbf{q}^{\mathcal{A}}_k), J_{\mathcal{V}\to\mathcal{V}^{D-l}}(\mathbf{s}_k)).$$

Hence $(J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}_k, \mathbf{q}^{\mathcal{A}}_k), J_{\mathcal{V}\to\mathcal{V}^i}(\mathbf{s}_k), J_{\mathcal{T}\to\mathcal{T}^i}(\mathbf{q}_{k+1}, \mathbf{q}^{\mathcal{A}}_{k+1})) \in \rightarrow^i$.

At the beginning of product automaton construction, there are several marked states computed as in (6.8) and (6.9). Assume that a product automaton state $q^{\mathcal{P}}_m$ in the trajectory $\mathbf{q}^{\mathcal{P}}_0 \mathbf{q}^{\mathcal{P}}_1 \mathbf{q}^{\mathcal{P}}_2 \ldots \mathbf{q}^{\mathcal{P}}_m \ldots$ has outgoing transitions to a state $q^{\mathcal{P}}_m$ has a transition $q^{\mathcal{P}}_{tp} = (q_i, q^{\mathcal{A}}_{tp})$. As there is no outgoing transition from the state $q^{\mathcal{A}}_{tp}$ and the transition function of product automaton in 3.3.2 provides transition if the current state of $\mathcal{A}$ (Büchi automaton) can make transition to another state of $\mathcal{A}$, $q^{\mathcal{P}}_{tp}$ cannot be never part of the accepting run in product automaton. On the other hand, those trap states are marked for such a link $l_{cp} \in \mathcal{L}^{D-i} and l_{cp} \in \mathcal{L}^j$ which is a boundary link with more complex propositions than just a safety property for subsystems i and j. The state $q^{\mathcal{P}}_m$ violates actually a property for link l_{cp} for the controlled system, *i.e.,* specification of $\mathbf{G}(x_{l_{cp}} > \bar{x}_{l_{cp}} \wedge \mathbf{X}\, x_{l_{cp}} > \bar{x}_{l_{cp}} \rightarrow \mathbf{X}\,\mathbf{X}\, x_{l_{cp}} < \bar{x}_{l_{cp}})$ is violated such that $x_{l_{cp}} \geq \bar{x}_{l_{cp}}$ for the last 3 successive states. The product automaton state at index m for subsystem-i and

link l_{cp} is $\mathbf{q}_m^{\mathcal{P}} = (\mathbf{q}_m, \mathbf{q}_m^{\mathcal{A}})$ such that:

$$l_{cp} \in \mathcal{L}^{D-i} J_{\mathcal{L}\to\{l_{cp}\}}(\mathbf{q}_m = q_m^{l_{cp}}) \in R_{q_m} \text{ where } R_{\mathbf{q}_m^{lsp}} \not\subseteq [0, \bar{x}_{l_{cp}}]$$
$$\text{and } \mathbf{q}_m^{\mathcal{A}} \overset{o(\mathbf{q}_m)}{\to}^{\mathcal{A}} q_{tp}^{\mathcal{A}}.$$

However, the path in product automaton (with the projected trajectory in T and projected run on $\mathcal{A}$) cannot be part of the solution because the controller C_{Φ^j} will not produce the required control input due to the violating property of the state $\mathbf{q}_m$. Therefore, we can eliminate such transitions in the product automaton $Q^{\mathcal{P},i}$ with the corresponding $T^{i,l}$ and $\mathcal{A}^{i,l}$ as in (6.10).

Observing that subsystem-i will guarantee that $\mathbf{q}_m^{\mathcal{P}}$ will not be reachable, we conclude that when we take projection of the states in projected product automaton path (6.12) we get a trajectory of T^i that originates from Q_{sat}^i. In addition, the projected run onto $\mathcal{A}$ from product automaton states in (6.12) provides $\mathbf{q}_0^{\mathcal{A}} \in Q_0^{\mathcal{A}}$ and $\mathbf{q}_i^{\mathcal{A}} \in \mathcal{F}$ for infinitely many indices i. Furthermore, this trajectory is produced in closed loop with C_{Φ^i}, hence satisfies Φ^i. The decomposition algorithm guarantees that the atomic propositions appear in subformulas Φ^i do not intersect. Hence, for each $i = 1, \ldots, n$, the corresponding trajectory (6.4) satisfies Φ^i and $\Phi = \wedge_{i=1,\ldots,n}\Phi^i$. Consequently, $\mathbf{q}_0^{\mathcal{P}}\mathbf{q}_1^{\mathcal{P}}\mathbf{q}_2^{\mathcal{P}}\ldots$ where each $\mathbf{q}^{\mathcal{P}}$ with $R_{\mathbf{q}^{\mathcal{P}}} \subseteq \Pi_{l\in L}[0, \bar{x}_l]$ satisfies Φ.

□

(ETLTS) covers **(ESVSP)**. In addition to specifications of the form $\mathbf{G}x_l < \bar{x}_l$, **(ETLTS)** allows us to consider more complex specifications defined over a link l such as $\mathbf{G}(x_l > \bar{x}_l \wedge \mathbf{X}\, x_l > \bar{x}_l \to \mathbf{X}\,\mathbf{X}\, x_l < \bar{x}_l)$ when we synthesize a control strategy for a subsystem that depend on l, $l \in \mathcal{L}^{D-j}$. Hence, as shown in Ex. 6.2.1, it reduces conservatism. However, it requires additional product computation, and increases the transition system sizes (T^j), consequently it is computationally more expensive.

Remark 6.2.1. *A major advantage of our construction based on link transition system is that the link transition systems only constructed once, hence, the abstraction computation based on the actual system dynamics* (3.4) *is done only once. Then, the transition systems for the subsystems are constructed using only the link transition systems. If we can not find a solution for the overall system, the following approaches*

can be followed to reduce the conservatism of the partitioning based approach 1) perform a search over the constraints defined for the boundary signals, 2) iteratively merge neighbouring subsystems. In both cases, we reuse the link transition systems hence avoid additional computation based on the system dynamics.

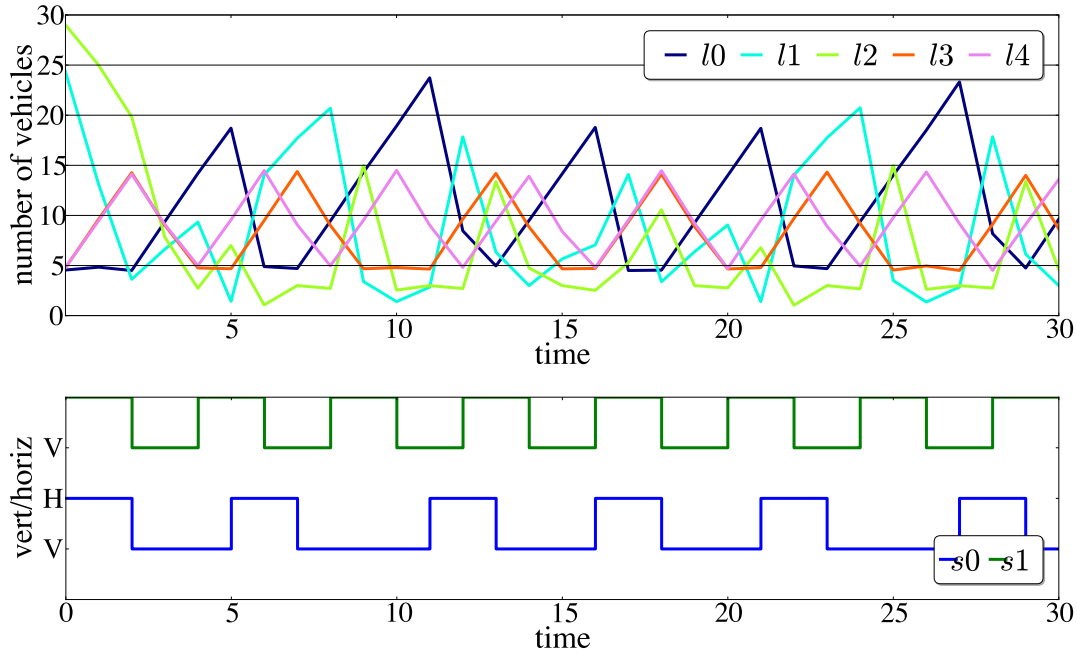


Figure 6.1: The number of vehicles on the links and the states of the signals (Vertical / Horizontal) during 30 time steps.

Example 6.2.1. *We applied the formal synthesis technique detailed in this section to the subsystems obtained in Example. 5.2.1.*

The number of states in T (abstraction for the overall system) would be approximately 8192 for a grid size of 5*. For method* ***(ETLTS)****, fortunately, for the same grid size, we attain 4115 states in the product of* Φ^1 *and* T^1 *for subsystem-1* ($\mathcal{L}^1, \mathcal{V}^1$) *and 138 in the product of* Φ^2 *and* T^2 *for subsystem-2* ($\mathcal{L}^2$, $\mathcal{V}^2$) *by implementing the abstraction method described here and on-the-fly reachability simplification during the product computation. The ratio of the satisfying region over* $\mathcal{X}$ *is* 0.0747*. For method* ***(ES-VSP)*** *for the same traffic system, the numbers of states in the products for subsystem-1 and subsystem-2 were 553 and 69, respectively. The ratio of the satisfying region is* 0.0512*. As the ratios indicate, the second method finds a larger set of satisfying initial states, and thus, reduces conservatism.*

The number of vehicles on each link and the state of the signals during 30 time steps

*are illustrated in Fig. 6.1. The computation of the link transition systems took 0.14 and 0.18 seconds for **(ESVSP)** and **(ETLTS)**, respectively. The construction of the product automata and solving Büchi games took 0.66 seconds for method **(ESVSP)** and 3.98 seconds for **(ETLTS)** in total. The computation time and the possible parallelization demonstrate the efficiency of the developed methods and the adaptability for the large traffic systems.*

We produce solution for also overall system without partitioning. We obtain 5288 product states for overall system by implementing abstraction method described in 4.3. The ratio of satisfying region over $\mathcal{X}$ is 0.2042 whereas this ratio is 0.0747 for **(ETLTS)**, it is about one third of without partition solution. The number of satisfying initial states is 1673 for without partition solution, whereas this number is 612 when we solve the system by partitioning and implementing **(ETLTS)** method. Total time for the product construction and Büchi game is 16.93 seconds without partition. The computation time for **(ETLTS)** method with partition is one fifth of the computation time for without partition. For a larger traffic system with more number of subsystems when the overall system is partitioned, the computation time will increase exponentially with the length of the LTL formula.

6.3 Bounded Specifications & Effects on The Set of Satisfying Initial States

As addressed before, state space explosion is a major problem in formal control. In this thesis, we propose to tackle this problem via partitioning. However, as discussed in previous section, this approach introduce additional conservatism. In the previous section, we introduced two methods to reduce this conservatism. These approaches aimed at pruning transitions with respect to infeasible behaviours of the neighbour subsystems. In this section, we propose to use bounded temporal operators to further reduce the conservatism. In particular, our goal is to limit the time bound on the eventually (**F**) operator and use these bounds on the neighbouring systems to further limit the feasible set of behaviours.

For instance, consider formula

$$\Psi = \mathbf{GF}x_0 < 20 \wedge \mathbf{GF}x_1 < 20 \wedge \mathbf{GF}x_2 < 20 \wedge \mathbf{GF}x_3 < 10 \wedge \mathbf{GF}x_4 < 10$$

for the traffic system in Example 3.1.1. We extend this formula by adding bounds to the **F** as $\mathbf{GF}N_{main}x_i < 20$ for the main roads and $\mathbf{GF}N_{side}x_j < 10$ for the side roads. Each formula in given format is transformed to a formula with **X** operator. For example, $\mathbf{GF}2x_i < 20$ is transformed to

$$\mathbf{G}(x_i < 20 \vee (\mathbf{X}x_i < 20 \vee \mathbf{XX}x_i < 20)).$$

We implement this approach to overall traffic system without partition and subsystems constructed by partitioning algorithm. We observe also the effects of bounded **F** operator on the controller synthesis for both **(ESVSP)** and **(ETLTS)** approaches. One of the observations during the experiments with **(ESVSP)** method is that while the increase in N_{main} increases number of satisfying initial states for its own subsystem, other subsystems have never been affected with this change. The reason is that **(ESVSP)** method does not have any effect on the specifications except the safety specifications ($\mathbf{G}\, x_l < \bar{x}_l$). Therefore, the details of trivial results for **(ESVSP)** method are omitted in this thesis.

We synthesized controllers also with the **(ETLTS)** method using bounded **F** operator on the partitioned subsystems obtained in Example. 5.2.1. The results for number of satisfying initial states and satisfying volume of the product automata are presented in Tables 6.1, 6.2 and 6.3. As in the results of **(ESVSP)** method, the number of satisfying initial states in the product automata increases when the N_{Main} parameter is increased until $N_{Main} = 3$. Therefore, it is best to use either **F** operator without bound or with the bound parameters $N_{Main} = 3$ and $N_{Side} = 1$ for the method **(ETLTS)** if there is no other effect than presented in Tables 6.1, 6.2 and 6.3.

Table6.1: **(ETLTS)** effects of bounded **F** operator on partitioned system - Subsystem 1.

N_{Main}	N_{Side}	SatInitSt	Time(sec.)
WB	WB	317	3.87
1	1	228	1.52
2	1	312	3.33
3	1	317	5.40
4	1	317	6.95
10	1	317	15.50

We also examined whether we can find an example of which a control strategy cannot

Table6.2: **(ETLTS)** effects of bounded **F** operator on partitioned system - Subsystem 2.

N_{Main}	N_{Side}	SatInitSt	Time(sec.)
WB	WB	12	0.11
1	1	8	0.06
2	1	8	0.08
3	1	12	0.16
4	1	12	0.19
10	1	12	0.27

Table6.3: **(ETLTS)** effects of bounded **F** operator on partitioned system - Overall Data

N_{Main}	N_{Side}	SatInitSt	SatVol	SatVolRat	Time(sec.)
WB	WB	612	1912500	0.0747	3.98
1	1	344	1075000	0.0420	1.58
2	1	404	1262500	0.0493	3.41
3	1	612	1912500	0.0747	5.56
4	1	612	1912500	0.0747	7.13
10	1	612	1912500	0.0747	15.78

be synthesized for one of the subsystems or overall system when we use bounded **F** operator, whereas a control strategy is found for the same subsystem and specifications without bound on **F** operator. Nonetheless, we could not find such example for the traffic system with 5 links by changing specifications and traffic dynamics. For instance, we used different N_{Main} parameters (N_{Main_1} for subsystem1 and N_{Main_2} for subsystem-2) for the links in subsystem-1 and subsystem-2 in order to observe the change in the number of satisfying initial states. We presented the results in Table 6.4 where $SatInitSt^1$ and $SatInitSt^2$ denotes the number of satisfying initial states in the product automaton of subsystem-1 and subsystem-2, respectively.

We made experiments with the fixed values of $N_{Main_1} = 2$,$N_{Side_1} = 1$ and $N_{Side_2} = 1$. We compared the numbers of satisfying initial states for subsystem-1 and subsystem-2 with changing $1 \leq N_{Main_2} \leq 8$ values. We observed that while $SatInitSt^1$ is 320 if $N_{Main_2} = 1$, this number decreases to 312 if $N_{Main_2} \geq 2$. We compared the numbers of satisfying initial states in the product automaton synthesized with bound and without bound on **F** operator. For the same traffic system, specifications and partitioning, $SatInitSt^1$ is 317 without bound on **F** operator. Hence, we can deduce that it

is possible to reduce the number of satisfying initial states for one of the subsystems even though its N_{Main} and N_{Side} values are fixed and the number of satisfying initial states of the other subsystem(s) increases. Thanks to this, it is also possible to have configurations where the use of bounds on **F** operator can provide that we can find controller for each subsystems whereas we cannot find controller for at least one of the subsystems using specifications without bound on **F** operator.

Table6.4: **(ETLTS)** effects of bounded distinct N_{Main} parameters for **F** operator on partitioned system

N_{Main_1}	N_{Main_2}	$SatInitSt^1$	$SatInitSt^2$	SatVolRat	Time(sec.)
WB	WB	317	12	0.0747	3.98
1	1	228	8	0.0420	1.58
1	2	225	8	0.0420	2.18
1	3	225	12	0.0629	2.96
2	1	320	8	0.0493	2.14
2	2	312	8	0.0493	3.37
2	3	312	12	0.0739	4.79
3	1	321	8	0.0498	2.38
3	2	317	8	0.0498	3.92
3	3	317	12	0.0747	5.48

CHAPTER 7

OPTIMAL CONTROL

The performance criteria such as the assurance of the certain limits for the number of vehicles in a link can be expressed as Linear Temporal Logic (LTL) formulas easily. Nevertheless, it is crucial to select the best one among the satisfying control strategies with respect to the larger scale of the criteria for the traffic system. In this chapter, we present the optimization criteria with this aim. We utilize the formal synthesis technique as detailed in Chapter 6 and integrate the optimization problem with the specified criteria. We introduce traffic system dynamics in a compact form as discrete-time system:

$$x_{t+1} = F(x_t, s_t, d_t), \tag{7.1}$$

$x_t \in \mathcal{X}$ is the state, $\mathcal{X}$ is in the form of (4.2), $s_t \in \mathcal{S}$ is the control input, $d_t \in \mathcal{D}$ is the exogenous vehicle flow, $\mathcal{D} \subset \mathbb{R}_{\geq 0}^{|\mathcal{L}|}$ at time $t \in \mathbb{Z}_+$ and $F : \mathcal{X} \times \mathcal{S} \times \mathcal{D} \to \mathcal{X}$ is the piecewise affine dynamics as in (3.4). A finite trajectory of states x, control inputs s and exogenous flow d starting from time t with a planning horizon of length H is denoted as:

$$\begin{aligned} x_t^H(x_t, s_t^H, d_t^H) &= x_{t+1}, \ldots, x_{t+H}, \\ s_t^H &= s_t, \ldots, s_{t+H-1}, \\ d_t^H &= d_t, \ldots, d_{t+H-1}. \end{aligned} \tag{7.2}$$

Then, for each time step we select control input s_t^H which optimizes the cost $W(x_t^H, s_t^H)$ where $W : \mathcal{X}^H \times \mathcal{S}^H \to \mathbb{R}_+$ and $W(\cdot,\cdot)$ maps trajectories and control inputs to $\mathbb{R}_+$.

Problem 2. *Given a traffic system* $(\mathcal{L}, \mathcal{V})$*, LTL specification* Φ (4.1) *with atomic*

propositions O in the form of (3.8)*, and a finite horizon cost function* $W(\cdot,\cdot) : \mathcal{X}^H \times \mathcal{S}^H \rightarrow \mathbb{R}_+$*, find a control strategy* C^W *such that the resulting system trajectory satisfies formula* Φ *while minimizing the cost at each time step.*

In order to solve the optimal control synthesis problem (Problem 2), we first solve formal synthesis problem as outlined in Chapter 6 with a minor extension. During solving the Büchi game, for a product automaton state, we store all control actions that can produce a satisfying run. We synthesize the feedback control strategies as defined in 3.3.3. The control strategy $\mathcal{C}^s$ we obtained from the modified Büchi game is set valued $\mathcal{C}^s : (Q \times Q^{\mathcal{A}}) \rightarrow 2^{\mathcal{S}}$, where $Q \times Q^{\mathcal{A}}$ is the set of product automaton states (see Definition 3.3.2) and $2^{\mathcal{S}}$ is the power set of $\mathcal{S}$. Note that $\mathcal{C}^s(q^{\mathcal{P}})$ gives all admissible control actions for a product automaton state $q^{\mathcal{P}} = (q, q^{\mathcal{A}})$. Hence, the optimization problem reduces to choosing the control action s from $\mathcal{C}^s(q^{\mathcal{P}})$ that minimizes the given cost $W(\cdot,\cdot)$.

In the remainder of this chapter, we define two cost criteria, namely minimization of the signal switches and minimization of the maximum vehicle density, and present algorithms to produce signals from $\mathcal{C}^s$ minimizing these cost criteria.

7.1 Minimization of Total Number of Switches

A control input (signal) switches between 0 and 1 values. The total number of switches along a trajectory of length H is defined as :

$$W(x_t^H, s_t^H) = \sum_{i=0}^{H} SC(s_{t+i-1}, s_{t+i}) \tag{7.3}$$

where

$$SC(s, s') = \sum_{v \in \mathcal{V}} \mid s_v - s'_v \mid \tag{7.4}$$

As the system is non-deterministic, during the online optimization we consider all possible reachable states. Given a state $(q, q^{\mathcal{A}})$ and a control action s, the set of states that can be reached from $(q, q^{\mathcal{A}})$ in one step when s is applied is defined as:

$$Post^{\mathcal{P}}((q, q^{\mathcal{A}}), s) = \{(p, p^{\mathcal{A}}) \mid ((q, q^{\mathcal{A}}), s, (p, p^{\mathcal{A}})) \in \overset{s}{\rightarrow}^{\mathcal{P}}\}. \tag{7.5}$$

Our aim is to minimize the cost function defined in the form of (7.3). We first derive a cost function for the product automaton states that mimic the cost given in (7.3). Given a sequence of product automaton states $\mathbf{p}_t^{\mathcal{P},H} = (q_t, q_t^{\mathcal{A}}) \dots (q_{t+H}, q_{t+H}^{\mathcal{A}}) \dots$, and sequence of control actions $s^H = s_{t-1}, \dots s_{t+H-1}$, the derived cost is defined as:

$$W^{SP}(\mathbf{p}_t^{\mathcal{P},H}, s^H) = \sum_{i=0}^{H} SC(s_{t+i-1}, s_{t+i}). \tag{7.6}$$

Then, the optimal control action minimizing (7.6) for horizon $H = 1$ is defined as:

$$s_t^* = \underset{s_t \in C^s(q_t, q_t^{\mathcal{A}})}{argmin} \quad SC(s_{t-1}, s_t). \tag{7.7}$$

Next, we define an optimization problem miminizing the worst case cost due to the non-determinism of the product automaton. In particular, we need to take into account all possible successor states from $Post^{\mathcal{P}}$ (see (7.5)) at each step in the given horizon. First, we define the following cost function for the product automaton states that gives an upper bound on the cost given in (7.6) for a given control strategy C^H (note that C^H is not set valued):

$$\begin{aligned} W^{CS}((q_t, q_t^{\mathcal{A}}), s_{t-1}, H, C^H) &= SC(s_{t-1}, C^H(q_t, q_t^{\mathcal{A}})) + \\ \max_{((q_{t+1}, q_{t+1}^{\mathcal{A}}) \in Post^{\mathcal{P}}((q_t, q_t^{\mathcal{A}}), C^H(q_t, q_t^{\mathcal{A}})))} & W^{CS}((q_{t+1}, q_{t+1}^{\mathcal{A}}), C^H(q_t, q_t^{\mathcal{A}}), H-1, C^H). \\ \text{and } W^{CS}((q_t, q_t^{\mathcal{A}}), s_{t-1}, 1, C^H) &= SC(s_{t-1}, C^H(q_t, q_t^{\mathcal{A}})) \end{aligned} \tag{7.8}$$

Then the optimization problem becomes finding the optimal control function of length H from the set of all feasible control functions $\mathcal{C}^{*,H,s_{t-1},(q_t,q_t^{\mathcal{A}})}$ which is derived from $\mathcal{C}^s$. Based on this derivation, the optimal cost is defined as:

$$W^{*,H,s_{t-1},(q_t,q_t^{\mathcal{A}})} = \min_{C^H \in \mathcal{C}^{s,H}} \quad W^{CS}((q_t, q_t^{\mathcal{A}}), s_{t-1}, H, C^H) \tag{7.9}$$

Finally, when we combine (7.8) and (7.9), we obtain the optimal control action at time t as:

$$s_t^* = \underset{s_t \in C^s(q_t, q_t^{\mathcal{A}})}{argmin} \quad SC(s_{t-1}, s_t) + \gamma \underset{(q_{t+1}, q_{t+1}^{\mathcal{A}}) \in Post^{\mathcal{P}}((q_t, q_t^{\mathcal{A}}), s_t)}{W^{*,H,s_t,(q_{t+1},q_{t+1}^{\mathcal{A}})}} \tag{7.10}$$

We scaled the future cost with $\gamma \in (0, 1]$ to reduce the effect of the future costs due to the non-determinism. The optimal signal defined in (7.10) is same as the one minimizing (7.9) when $\gamma = 1$. When we reduce the value of γ we reduce the effects of the future steps for $H > 1$.

We propose Algorithm 1 in a recursive manner in order to solve the minimization problem for total number of switches. Assume that we have the current product automaton state $q^{\mathcal{P}_t} = (q_t, q_t^{\mathcal{A}})$ where q_t is the transition system state and $q_t^{\mathcal{A}}$ is the Büchi automaton state.. We have also the previous control action s_{t-1}. s_t denotes any admissible control action in from C^s at the current step, *i.e.,* $s_t \in C^s((q_t, q_t^{\mathcal{A}}))$.

As we have non-deterministic product automaton, we get a list of possible next states from the $Post^{\mathcal{P}}((q_{t+1}, q_{t+1}^{\mathcal{A}}), s_t))$. We cannot make choice among these states. Therefore, we try to optimize the worst case cost among the candidate post states. Thus, in order to simplify worst case optimization problem, we consider it as an optimization problem for the maximal system, *i.e.,* select the maximum number of switches from each branch of the possible post states. Finally, we sum these maximal values with the current number of switches and find a minimum among those sums.

We ran Algorithm 1 with the cost function described as in 7.9 for the traffic system in Example 3.1.1 during 30 time steps. We synthesized control strategies with both **(ESVSP)** and **(ETLTS)**. As the obtained results for both methods demonstrates close resemblance to each other, we revealed the blended results for both methods in Figure 7.1. We chose optimal control input among the set of control inputs produced by the control strategy C^s using receding horizon approach with different length of horizons. We compared the results for each value of $H \in \{1,2,3\}$ as well as the results without optimization. We fixed the initial state and obtained exterior vehicle flow as $d \in [4.50, 4.99]$ for the experiments. The number of switches were calculated as the average value of the 20 experiments in Figure 7.1a. In Figure 7.1b, we calculated step counts per switch of one signal for the traffic system with 5 links and 2 signals ($number\_of\_signals = 2$) during 30 time steps ($total\_step = 30$) as:

$$step\_count = \frac{total\_switch\_count([25,39])}{number\_of\_signals(2) \times total\_step(30)}.$$

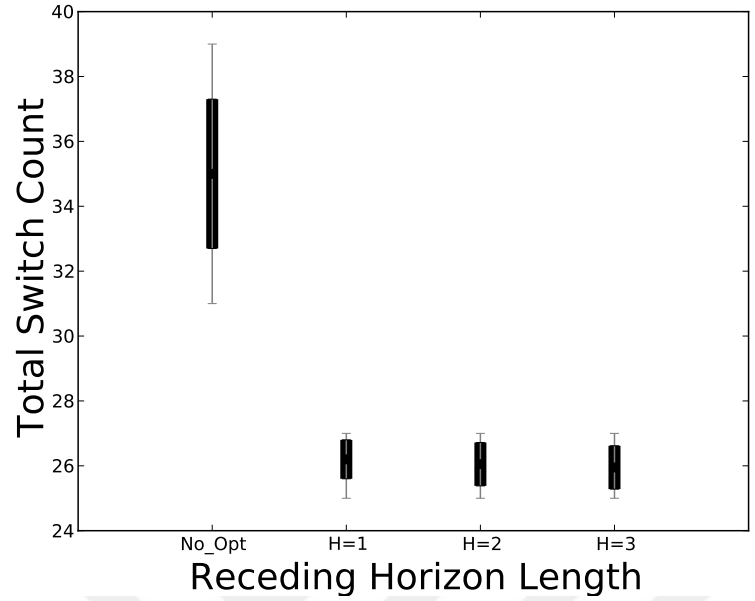
As the traffic system used in these tests includes only 2 control inputs, the results do not show enough improvement on the reduction of total number of switches. The example traffic system do not show decrease on the average number of switches with the increase of the optimization horizon H. The main reason of this is the non-determinism of the system. As $Post^{\mathcal{P}}(q^{\mathcal{P}}, s)$ returns a list of next states with the given control action, we consider the maximum cost associated with those states and as H

Algorithm 1 SwitchMinimizationRHC – MinSW ($q_t^{\mathcal{P}}$, s_{t-1}, PA, H, γ, C^s)

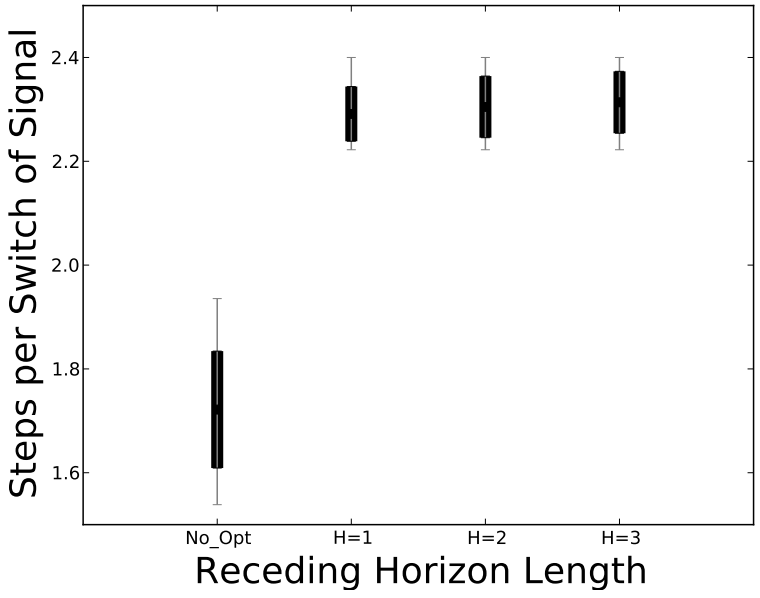
Input: $q_t^{\mathcal{P}} = (q_t, q_t^{\mathcal{A}})$ {current product automaton state},
s_{t-1} {previous control action},
PA {overall product automaton structure},
H {receding horizon length},
γ {weight constant for future cost},
C^s {set of feasible control actions for each PA state}.

Output: $W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$, s_t^* {optimal cost, optimal control action}

$C^{*,H,s_{t-1},q_t^{\mathcal{P}}} = PA(C^s, q_t^{\mathcal{P}})$ {set of admissible control actions for curr. state}
$W^{*,H,s_{t-1},q_t^{\mathcal{P}}} = \infty$, $s_t^* = \emptyset$
for each $s_t \in C^{*,H,s_{t-1},q_t^{\mathcal{P}}}$ **do**
  if H == 1 **then**
    if $SC(s_{t-1}, s_t) < W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$ {$SC(s_{t-1}, s_t)$ as in Equation (7.4)} **then**
      $W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$, $s_t^* = SC(s_{t-1}, s_t)$, s_t
    end if
  else if $H \geq 2$ **then**
    $W^{max,s_t} = 0$
    for each $q_{t+1}^{\mathcal{P}} \in Post^{\mathcal{P}}(q_t^{\mathcal{P}}, s_t)$ **do**
      W^{CS}, s^z = MinSW ($q_{t+1}^{\mathcal{P}}$, s_t, PA, $H-1$, γ, C^s)
      if $W^{CS} > W^{max,s_t}$ **then**
        $W^{max,s_t} = W^{CS}$
      end if
    end for
    if $SC(s_{t-1}, s_t) + \gamma \times W^{max,s_t} < W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$ **then**
      $W^{*,H,s_{t-1},q_t^{\mathcal{P}}} = SC(s_{t-1}, s_t) + \gamma \times W^{max,s_t}$
      $s_t^* = s_t$
    end if
  end if
end for
Return $W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$, s_t^*.



(a)



(b)

Figure 7.1: During 30 time steps, total number of signal switches in (a) and step counts per switch of one signal in (b) are shown as the average of 20 experiments without optimization (No_Opt) and with optimization *i.e.,* receding horizon lengths $H = 1, 2, 3$.

increases, all possible action sequences converge to the same set. Another reason is that after the 25^{th} time step, the algorithm begins to control the number of switches which will not be tested anymore, *i.e.,* 31^{th} time step is not under control. The final reason is the small number of control inputs, 2 for the sample traffic system. Hence, the maximum number of switches in one turn is 2. While the algorithm with $H = 1$ only controls the next step of the simulation and just selects the control input if the number of switches is zero for the next step, the algorithm with $H = 5$ does not only consider the minimum number of switches for the next step and tries to find minimum number of switches among the sum of the current number switches with the worst-case (maximum) number of switches for the next 4 steps. The algorithm with $H = 5$ selects possibly control input s_{min} with switch number 4 within next 5 steps, whereas s_{min} causes 1 switch for the next step. Therefore, for the traffic systems with small number of control inputs, it is hard to observe the effects of the increase in receding horizon length H. In Chapter 8, we make experiments with larger traffic systems with more control inputs. Thus, the effects of the increase in receding horizon length H on the minimization algorithm appear clearly.

7.2 Minimization of Maximum Vehicle Density

In this section, we aim at minimizing the maximum vehicle density. Proposed approach is almost the same with the approach in the criterion of minimization of the number of switches. The difference is in the computation of the cost function. x_t^H, s_t^H is the system trajectory as in (7.2) and the maximum density along this trace is defined as:

$$W(x_t^H, s_t^H) = \max_{i=0}^{H}\big(\max_{l\in\mathcal{L}}(\frac{x_{l,t+i}}{x_l^{cap}})\big) \tag{7.11}$$

As in the previous case, we compute an upper bound on this cost (7.12) via the abstraction. The derived cost for a product automaton states trace of length H $\mathbf{p}_t^{\mathcal{P},H} = (q_t, q_t^{\mathcal{A}})\dots(q_{t+H}, q_{t+H}^{\mathcal{A}})\dots$ (we dropped the signal sequence as it does not effect the cost directly) is defined as:

$$\begin{aligned} W(\mathbf{q}_t^H) &= \max_{i=0}^{H}\big(VD(\mathbf{q}_{t+i})\big) \text{ where} \\ VD(\mathbf{q}) &= \max_{l\in\mathcal{L}}\big(\frac{\overline{x}_l}{q_l^{cap}} \text{ where } R_{q_l} = [\underline{x}_l, \overline{x}_l] \text{ (see 6.1.1)}\big) \end{aligned} \tag{7.12}$$

For a trajectory of system x_t^H, s_t^H and the corresponding trajectory of the product automaton, it is straightforward to prove that $W(x_t^H) <= W^P(q_t^H)$. In the optimization, for computational reasons, we only consider W^P.

We compare the possible vehicle densities obtained from the states of the control strategy $\mathcal{C}^s$ during the next H steps. We use the notations $q^{\mathcal{P}}$ and $(q, q^{\mathcal{A}})$ interchangeably to denote states of the product automaton. We utilize the $Post^{\mathcal{P}}$ function as defined in Equation (7.5). We aim to minimize the cost in the form of (7.12) for the maximum vehicle density among all links in the given step. We define a generic cost function for the states of product automaton to compute a similar cost given in (7.12). The optimal control action minimizing (7.12) for horizon $H = 1$ is defined as:

$$s_t^* = \underset{s_t\in\mathcal{C}^s(q_t^{\mathcal{P}})}{argmin} \quad VD(\mathbf{q}_{t+i+1}^{\mathcal{P}}). \tag{7.13}$$

Next, we define an optimization problem miminizing the worst case density cost due to the non-determinism of the product automaton. In particular, we need to take into account all possible successor states from $Post^{\mathcal{P}}$ (see (7.5)) at each step in the given

horizon. First, we define the following cost function for the product automaton states that gives an upper bound on the cost given in (7.12) for a given control strategy C^H (note that C^H is not set valued):

$$W^{VD}(q_t^{\mathcal{P}}, s_{t-1}, H, C^H) = \max_{(q_{t+1}^{\mathcal{P}} \in Post^{\mathcal{P}}(q_t^{\mathcal{P}}, C^H(q_t^{\mathcal{P}})))} \Big(\big(VD(q_{t+1}^{\mathcal{P}})\big), \big(\max(W^{VD}(q_{t+1}^{\mathcal{P}}, C^H(q_t^{\mathcal{P}}), H-1, C^H))\big) \Big)$$

$$\text{and } W^{VD}(q_t^{\mathcal{P}}, s_{t-1}, 1, C^H) = \max_{(q_{t+1}^{\mathcal{P}} \in Post^{\mathcal{P}}(q_t^{\mathcal{P}}, C^H(q_t^{\mathcal{P}})))} \Big(\big(VD(q_{t+1}^{\mathcal{P}})\big), \tag{7.14}$$

where $q_{t+1}^{\mathcal{P}} \in Post^{\mathcal{P}}(q_t^{\mathcal{P}}, C^H(q_t^{\mathcal{P}}))$. Then, the optimization problem becomes finding the optimal control function of length H from the set of all feasible control functions $\mathcal{C}^{*,H,s_{t-1},q_t^{\mathcal{P}}}$ which is derived from $\mathcal{C}^s$. Based on this derivation, the optimal cost is defined as:

$$W^{*,H,s_{t-1},q_t^{\mathcal{P}}} = \min_{C^H \in \mathcal{C}^{s,H}} \quad W^{VD}(q_t^{\mathcal{P}}, s_{t-1}, H, C^H) \tag{7.15}$$

Finally, when we combine (7.14) and (7.15), we obtain the optimal control action at time t as:

$$s_t^* = \underset{s_t \in C^s(q_t^{\mathcal{P}})}{argmin} \quad \max \Big(\big(VD(q_{t+1}^{\mathcal{P}})\big) \big(\underset{q_{t+1}^{\mathcal{P}} \in Post^{\mathcal{P}}(q_t^{\mathcal{P}}, s_t)}{W^{*,H,s_t,q_{t+1}^{\mathcal{P}}}} \big) \Big) \tag{7.16}$$

Algorithm 2 summarizes whole optimization procedure to minimize maximum vehicle density on the links.

We made experiments implementing Algorithm 2 with the cost function described as in 7.15 for the traffic system in Example 3.1.1 during 30 time steps. We synthesized control strategies with both **(ESVSP)** and **(ETLTS)** methods. The presented vehicle density results were calculated as the average of 20 experiments. The results are shown in Figure 7.2. As the obtained results for both methods were almost equal, we revealed the blended results for both methods in Figure 7.2. We chose optimal control input among the set of control inputs produced by the control strategy $\mathcal{C}^s$ using receding horizon approach with different length of horizons. We compared the results for each value of $H \in \{1,2,3\}$ as well as the results without optimization. We fixed the initial state and obtained exterior vehicle flow as $d \in [4.50, 4.99]$ for the experiments.

Algorithm 2 DensityMinimizationRHC – MinVD ($q_t^{\mathcal{P}}$, s_{t-1}, PA, H, γ, $\mathcal{C}^s$)

Input: $q_t^{\mathcal{P}} = (q_t, q_t^{\mathcal{A}})$ {current product automaton state},

s_{t-1} {previous control action},

PA {overall product automaton structure},

H {receding horizon length},

γ {weight constant for future cost},

$\mathcal{C}^s$ {set of feasible control actions for each PA state}.

Output: $W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$, s_t^* {optimal cost, optimal control action}

$\mathcal{C}^{*,H,s_{t-1},q_t^{\mathcal{P}}} = PA(\mathcal{C}^s, q_t^{\mathcal{P}})$ {set of admissible control actions for curr. state}

$W^{*,H,s_{t-1},q_t^{\mathcal{P}}} = \infty$, $s_t^* = \emptyset$

for each $s_t \in \mathcal{C}^{*,H,s_{t-1},q_t^{\mathcal{P}}}$ **do**

  $W^{max,s_t} = 0$

  for each $q_{t+1}^{\mathcal{P}} \in Post^{\mathcal{P}}(q_t^{\mathcal{P}}, s_t)$ {x_{t+1} derived as $R_{q_{t+1}^{\mathcal{P}}} = [\underline{x}_{t+1}, \overline{x}_{t+1}]$} **do**

    if $VD(x_{t+1}, q_{t+1}^{\mathcal{P}}) > W^{max,s_t}$ {$VD(x_{t+1}, q_{t+1}^{\mathcal{P}})$ as in (7.12)} **then**

      $W^{max,s_t} = VD(x_{t+1}, q_{t+1}^{\mathcal{P}})$

    end if

  end for

  if H == 1 **then**

    if $W^{max,s_t} < W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$ **then**

      $W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$, $s_t^* = W^{max,s_t}$, s_t

    end if

  else if $H \geq 2$ **then**

    for each $q_{t+1}^{\mathcal{P}} \in Post^{\mathcal{P}}(q_t^{\mathcal{P}}, s_t)$ **do**

      W^{VD}, s^z = MinVD ($q_{t+1}^{\mathcal{P}}$, s_t, PA, $H-1$, γ, $\mathcal{C}^s$)

      if $W^{VD} > W^{max,s_t}$ **then**

        $W^{max,s_t} = W^{CS}$

      end if

    end for

    if $\max\left(VD(x_{t+1}, q_{t+1}^{\mathcal{P}}), W^{max,s_t}\right) < W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$ **then**

      $W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$, $s_t^* = \max\left(VD(x_{t+1}, q_{t+1}^{\mathcal{P}}), W^{max,s_t}\right)$, s_t

    end if

  end if

end for

Return $W^{*,H,s_{t-1},q_t^{\mathcal{P}}}$, s_t^*.

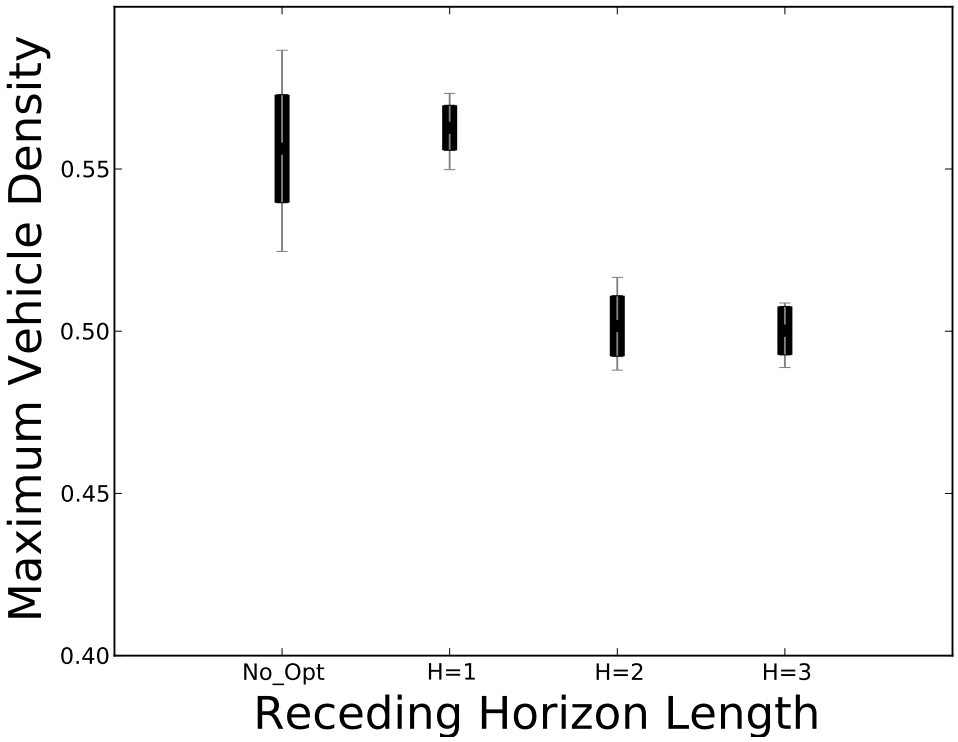


Figure 7.2: Maximum vehicle densities are shown as the average of 20 experiments without optimization (No_Opt) and with receding horizon lengths $H = 1, 2, 3$.

The results revealed that optimal control with receding horizon length $H = 2$ and $H = 3$ reduced the maximum vehicle density by $10 - 13\%$. This reduction was not seen in the results for $H = 1$ because traffic system used in these experiments contained only 2 control inputs. The example traffic system did not show respectable amount of decrease on the amaximum vehicle density with the increase of the optimization horizon H. The main reason of this is the non-determinism of the system. As $Post^{\mathcal{P}}(q^{\mathcal{P}}, s)$ returned a list of next states with the given control action, we considered the maximum cost associated with those states and as H increased, all possible action sequences converged to the same set. Another reason was that after the 25^{th} time step, the algorithm begins to control the densities which will not be tested anymore, *i.e.,* 31^{th} time step was not under control. The final reason was the small number of control inputs, 2 for the sample traffic system. Hence, the maximum number of switches in one turn is 2. Therefore, for the traffic systems with small number of control inputs, it was hard to observe the effects of the increase in receding horizon length H. In Chapter 8, we will make experiments with larger traffic systems with more control inputs. Thus, the effects of the increase in receding horizon length H on the minimization algorithm may appear clearly.

CHAPTER 8

EXPERIMENTS AND RESULTS

The software package implementing the methods developed in this dissertation is available at `http://user.ceng.metu.edu.tr/~ebru/wordpress/home/fc-ts/` and freely downloadable. The source codes of this study have been developed in Python by using open source codes for basic processes. The input parameters for this program are:

1. traffic system file describing capacity, saturation flow, exterior flow, turn and supply ratio parameters,
2. proposition file describing atomic propositions for each link and signal,
3. overall specification file describing Linear Temporal Logic (LTL) specifications as in 8.2,
4. number of links and signals in the traffic system,
5. the method for controller synthesis ((**ESVSP)** or **(ETLTS)**),
6. step count for simulation,
7. optimization criteria,
8. the receding horizon length if optimization criteria mod is specified.

This program builds a traffic network model and produce partition for it. Then, an abstraction of the partitioned subsystems (links and signal) is built as transition system (TS). In addition, it builds Büchi automaton from derived sub specifications and also the product of TS and automaton. Furthermore, the program plays a Büchi game to find recurrent sets, satisfying initial strategy and finally a control strategy. Finally, the program produces solution for the Problems 1 and 2 in the form of optimal feedback control strategy as the set of states and control signals satisfying all sub specifications

for the abstraction of the subsystems. Our tests were run on an MacBook Pro with 16 GB RAM and 2,2 GHz Intel Core i7 processor.

8.1 Case Study for Traffic System with 9 Links and 4 Signals

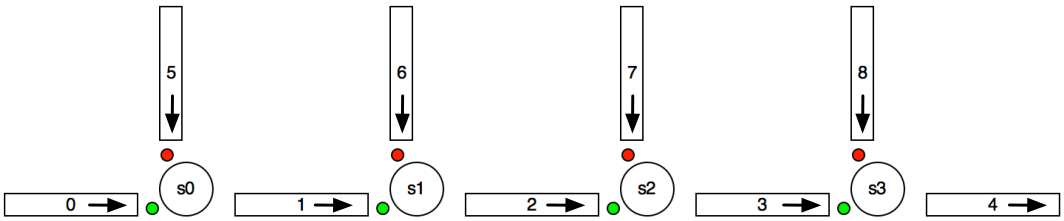


Figure 8.1: A traffic network composed of links $l_0, l_1, \ldots, l_8$ and signals s_0,s_1,s_2,s_3. The flow directions of the links are shown with arrows.

The example traffic network in our first case study consists of 9 links and 4 signals as shown in Fig. 8.1. The network contains horizontal links l_0, l_1, l_2, l_3, l_4 and vertical links l_5, l_6, l_7, l_8 and signals s_0, s_1, s_2, s_3. The capacity, saturation flow, exterior flow, turn and supply ratio parameters for this traffic system are as follows:

$$
\begin{aligned}
& x_i^{cap} = 40 \text{ for } i \in \{0,1,2,3,4\}; x_j^{cap} = 20 \text{ for } j \in \{5,6,7,8\}. \\
& c_2 = 15; c_i = 20 \text{ for } i \in \{0,1,3,4\}; c_j = 10 \text{ for } j \in \{5,6,7,8\}. \\
& \beta_{ij} = 0.75 \text{ for } i\text{-}j \in \{0\text{-}1, 1\text{-}2, 2\text{-}3, 3\text{-}4\}; \beta_{kl} = 0.3 \text{ for } k\text{-}l \in \\
& \{5\text{-}1, 6\text{-}2, 7\text{-}3, 8\text{-}4\}. \alpha_{51}^0 = 0.8; \alpha_{ij}^k = 1 \text{ for} \\
& i\text{-}j\text{-}k \in \{0\text{-}1\text{-}0, 1\text{-}2\text{-}1, 2\text{-}3\text{-}2, 3\text{-}4\text{-}3, 6\text{-}2\text{-}1, 7\text{-}3\text{-}2, 8\text{-}4\text{-}2\}. \\
& d_i^{max} = 5 \text{ for } i \in \{0,5,6,7,8\}; d_j^{max} = 0 \text{ for } j \in \{1,2,3,4\}.
\end{aligned}
\tag{8.1}
$$

where α_{ij}^k denotes the supply ratio of the link j to the link i when the signal k is green for the link i. All signals has two modes, either the horizontal or the vertical links are actuated.

An LTL specification Φ^{ex} for the traffic system defined in Figure 8.1 with parameters as in 8.1 is given below. The goal in synthesis problem for this example is to find a feedback control strategy for the signals s_0, s_1, s_2 and s_3 such that the system trajectories satisfies formula Φ^{ex}:

$$\Phi^{ex} = \Phi_1 \wedge \Phi_2 \wedge \Phi_3 \wedge \Phi_4 \wedge \Phi_5 \wedge \Phi_6 \tag{8.2}$$
$$\Phi_1 = \mathbf{G}(x_0 < 40 \wedge x_1 < 30 \wedge x_2 < 30 \wedge x_5 < 15 \wedge x_6 < 15)$$
$$\Phi_2 = \mathbf{G}(x_3 < 30 \wedge x_4 < 35 \wedge x_7 < 15 \wedge x_8 < 15)$$
$$\Phi_3 = \mathbf{GF}x_0 < 25 \wedge \mathbf{GF}x_1 < 20 \wedge \mathbf{GF}x_2 < 20$$
$$\Phi_4 = \mathbf{GF}x_5 < 10 \wedge \mathbf{GF}x_6 < 10$$
$$\Phi_5 = \mathbf{GF}x_3 < 20 \wedge \mathbf{GF}x_4 < 25$$
$$\Phi_6 = \mathbf{GF}x_7 < 10 \wedge \mathbf{GF}x_8 < 10$$

We applied the decomposition algorithm to the traffic system in Figure 6.2 and formula Φ^{ex} from 8.2. As all the links appear in some formula and no signals appear in a formula, the initial sub-system set (5.2) is composed of 12 sets (e.g. $\mathcal{L}_{\Phi_1} = \{l_0, l_1, l_2, l_5, l_6\}, \mathcal{L}_{\Phi_2} = \{l_3, l_4, l_7, l_8\}$). After merging the intersecting subsystems, we obtained the following set

$$\begin{aligned}\{(\{l_0, l_1, l_2, l_5, l_6\}, \emptyset), (\{l_3, l_4, l_7, l_8\}, \emptyset),\\ (\emptyset, \{s_0\}), (\emptyset, \{s_1\}), (\emptyset, \{s_2\}), (\emptyset, \{s_3\})\}\end{aligned} \tag{8.3}$$

After merging these according to connectivity analysis, we obtained two subsystems $(\mathcal{L}^1, \mathcal{V}^1)$ and $(\mathcal{L}^2, \mathcal{V}^2)$:

$$\begin{aligned}\mathcal{L}^1 = \{l_0, l_1, l_2, l_5, l_6\}, \mathcal{V}^1 = \{s_0, s_1\},\\ \mathcal{L}^2 = \{l_3, l_4, l_7, l_8\}, \mathcal{V}^2 = \{s_2, s_3\}.\end{aligned} \tag{8.4}$$

The corresponding specifications are

$$\Phi^1 = \Phi_1 \wedge \Phi_3 \wedge \Phi_4 \text{ and } \Phi^2 = \Phi_2 \wedge \Phi_5 \wedge \Phi_6.$$

The derived dependency sets are $\mathcal{L}^{D-1} = \{l_3\}$, $\mathcal{V}^{D-1} = \{s_2\}$, $\mathcal{L}^{D-2} = \{l_2\}$, $\mathcal{V}^{D-2} = \emptyset$. s_2 is a boundary signal owned by subsystem-2. The constraint defined for s_2 is $\{(2,3),(1,2)\}$ (for horizontal and vertical actuation, respectively, see (5.1)).

We applied the formal synthesis technique detailed in Section 6.2 to the subsystems obtained in 8.3.

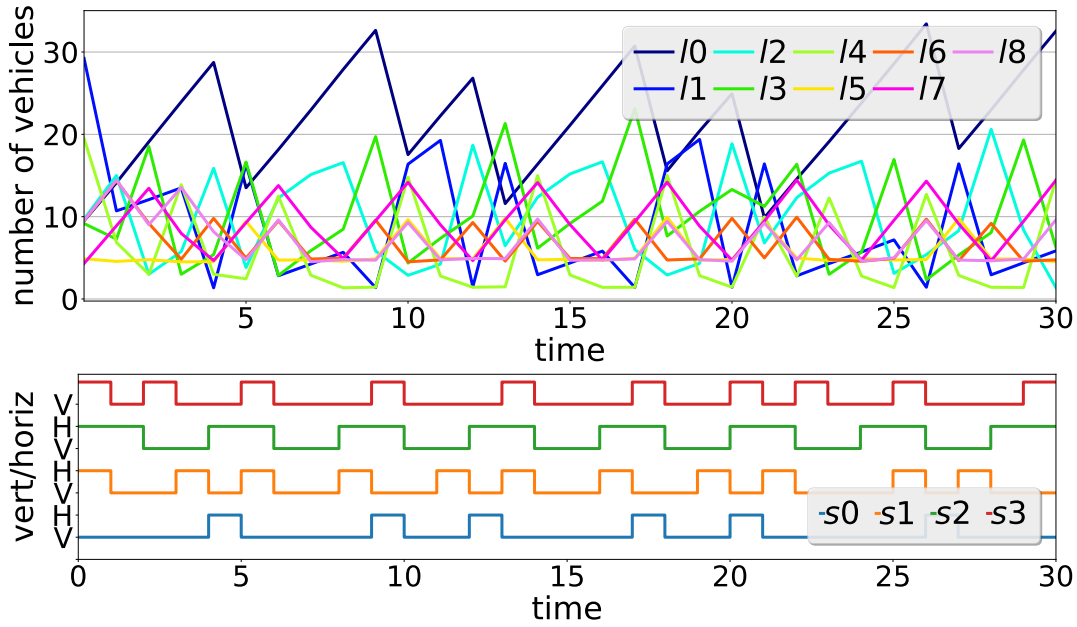


Figure 8.2: The number of vehicles on the links and the states of the signals (Vertical / Horizontal) during 30 time steps.

The number of states in T (abstraction for the overall system) would be approximately 8.4 millions for a grid size of 5. For method **(ETLTS)**, fortunately, for the same grid size, we attain 59125 states in the product of Φ^1 and T^1 for subsystem-1 ($\mathcal{L}^1, \mathcal{V}^1$) and 3113 in the product of Φ^2 and T^2 for subsystem-2 ($\mathcal{L}^2$, $\mathcal{V}^2$) by implementing the abstraction method described here and on-the-fly reachability simplification during the product computation. The ratio of the satisfying region over $\mathcal{X}$ (only considering links) is 0.0230 For method **(ESVSP)** for the same traffic system, the numbers of states in the products for subsystem-1 and subsystem-2 were 20567 and 1656, respectively. The ratio of the satisfying region is 0.0176. As the ratios indicate, **(ETLTS)** finds a larger set of satisfying initial states, and thus, reduces conservatism.

The number of vehicles on each link and the state of the signals during 30 time steps are illustrated in Fig. 8.2. The computation of the link transition systems took 2.03 seconds. The construction of the product automata and solving Büchi games took 108 seconds for method **(ESVSP)** and 550 seconds for **(ETLTS)** in total. The computation time and the possible parallelization demonstrate the efficiency of the developed methods and the adaptability for the large traffic systems.

Now, we present the controller synthesis results for the bounded specifications. As

addressed in Section 6.3 the increase or decrease in bounds of **F** operator did not affect the satisfying volume of other subsystems for **(ESVSP)**. The reason is that **(ESVSP)** method shows effects on the safety specifications (**G** $x_l < \bar{x}_l$). Hence, we did not present the results for **(ESVSP)**.

We synthesized controllers with the **(ETLTS)** using bounded **F** operator on the partitioned subsystems obtained in (8.4). We presented the results for number of satisfying initial states and satisfying volume of the product automata are in Tables 8.1, 8.2 and 8.3. As in the results of **(ESVSP)** method, the number of satisfying initial states in the product automata increased trivially when the N_{Main} parameter is increased until $N_{Main} = 3$. Therefore, it was best to use either **F** operator without bound or with the bound parameters $N_{Main} = 3$ and $N_{Side} = 1$ for the method **(ETLTS)** without considering other effects. Nevertheless, when the $N_{Main} \geq 2$ there is no increase in the satisfying initial states and satisfying volume. Hence, we did not include the results for different values of N_{Main} as well as the results for changing values of N_{Side} because N_{Side} did not any effect on the number of satisfying initial states and satisfying volume. The bounds reported in Table 8.3 and others prove a known phenomena in formal control that unbounded eventually operator captures all possible bounds. In addition, these bounds show an important characteristic of the synthesized control strategy: the best strategy producing the largest set of satisfying initial states can guarantee that the liveness constraint will be satisfied with period 2.

Table8.1: **(ETLTS)** effects of bounded **F** operator on partitioned system - Subsystem 1.

N_{Main}	N_{Side}	SatInitSt	Time(sec.)
WB	WB	7064	450.28
1	1	6054	220.59
2	1	7064	636.44
3	1	7064	970.49

We examined whether we can find an example of which a control strategy cannot be synthesized for one of the subsystems or overall system when we use bounded **F** operator, whereas a control strategy is found for the same subsystem and specifications without bound on **F** operator. For instance, we used different N_{Main} parameters (N_{Main_1} for subsystem-1 and N_{Main_2} for subsystem-2) in order to observe the change

Table8.2: **(ETLTS)** effects of bounded **F** operator on partitioned system - Subsystem 2.

N_{Main}	N_{Side}	SatInitSt	Time(sec.)
WB	WB	258	10.21
1	1	252	6.08
2	1	258	8.80
3	1	258	9.44

Table8.3: **(ETLTS)** effects of bounded **F** operator on partitioned system - Overall Data

N_{Main}	N_{Side}	SatInitSt	SatVol	SatVolRat	Time(sec.)
WB	WB	166848	325,875,000,000	0.0199	460.50
1	1	130560	255,000,000,000	0.0156	226.68
2	1	166848	325,875,000,000	0.0199	645.24
3	1	166848	325,875,000,000	0.0199	979.94

in the number of satisfying initial states. We presented the results in Table 8.4 where $SatInitSt^1$ and $SatInitSt^2$ denotes the number of satisfying initial states in the product automaton of subsystem-1 and subsystem-2, respectively.

We made experiments with the fixed values of $N_{Side_1} = 1$ and $N_{Side_2} = 1$. We compared the numbers of satisfying initial states for subsystem-1 and subsystem-2 with changing N_{Main_1} and N_{Main_2} values. We observed that while $SatInitSt^1$ is 9524 for $N_{Main_1} = 3$ if $N_{Main_2} = 1$, this number decreases to 7064 if $N_{Main_2} \geq 2$. We compared the numbers of satisfying initial states in the product automaton synthesized with bound and without bound on **F** operator. For the same traffic system, specifications and partitioning, $SatInitSt^1$ is 7064 without bound on **F** operator. Hence, we can deduce that it is possible to reduce the number of satisfying initial states for one of the subsystems even though its N_{Main} and N_{Side} values are fixed and the number of satisfying initial states of the other subsystem(s) increases. Thanks to this, it is also possible to have configurations where we can find controller for each subsystem with bounded **F** whereas we cannot find controller for at least one of the subsystems using specifications without bound on **F** operator.

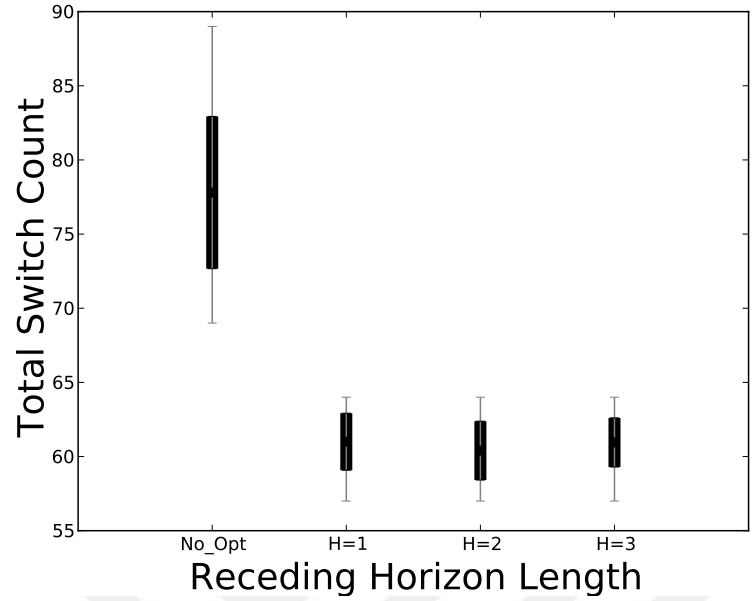
Now, we illustrate results for the optimal control with the aim of minimization of signal switches for the traffic system in 8.1 during 30 time steps. We synthesized

Table8.4: **(ETLTS)** effects of bounded **F** with distinct N_{Main} parameters for each subsystem on partitioned system

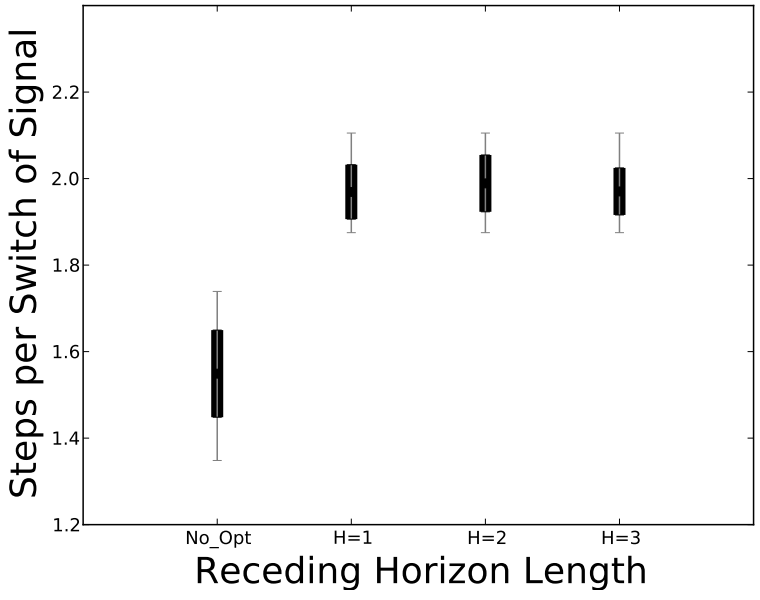
N_{Main_1}	N_{Main_2}	$SatInitSt^1$	$SatInitSt^2$	SatVolRat	Time(sec.)
WB	WB	7064	258	0.0199	460.50
1	1	6054	252	0.0156	211.59
1	2	6028	258	0.0161	335.97
1	3	6028	258	0.0161	663.82
2	1	9122	252	0.0217	384.23
3	1	9524	252	0.0218	297.20
2	2	7064	258	0.0199	639.32
2	3	7064	258	0.0199	878.24
3	2	7064	258	0.0199	771.09
3	3	7064	258	0.0199	958.29

control strategies with both **(ESVSP)** and **(ETLTS)**. We compared the results for each value of $H \in \{1,2,3\}$ as well as the results without optimization. In order to produce results for $H = 3$, a considerable amount of time was required. We fixed the initial state and obtained exterior vehicle flow as $d \in [4.50, 4.99]$ for the experiments. The number of switches were calculated as the average value of the 20 experiments in Figure 8.3a and average step counts required per switch of one signal is revealed in Figure 8.3b.

The results showed that the number of signal switches reduced by $25-30\%$ when optimal control was used with $H = 1,2,3$. On the other hand, the example traffic system do not show decrease on the average number of switches with the increase of the optimization horizon H. The main reason of this is the non-determinism of the system. As $Post^{\mathcal{P}}(q^{\mathcal{P}}, s)$ returns a list of next states with the given control action, we consider the maximum cost associated with those states and as H increases, all possible action sequences converge to the same set. Another reason is that after the 25^{th} time step, the algorithm begins to control the number of switches which will not be tested anymore, *i.e.,* 31^{th} time step is not under control. We demonstrate the results for traffic system with 16 links and 8 signals in 8.2 and compare with the results of this section.



(a)



(b)

Figure 8.3: During 30 time steps, total number of signal switches in (a) and step counts per switch of one signal in (b) are shown as the average of 20 experiments without optimization (No_Opt) and with optimization *i.e.,* receding horizon lengths $H = 1, 2, 3$.

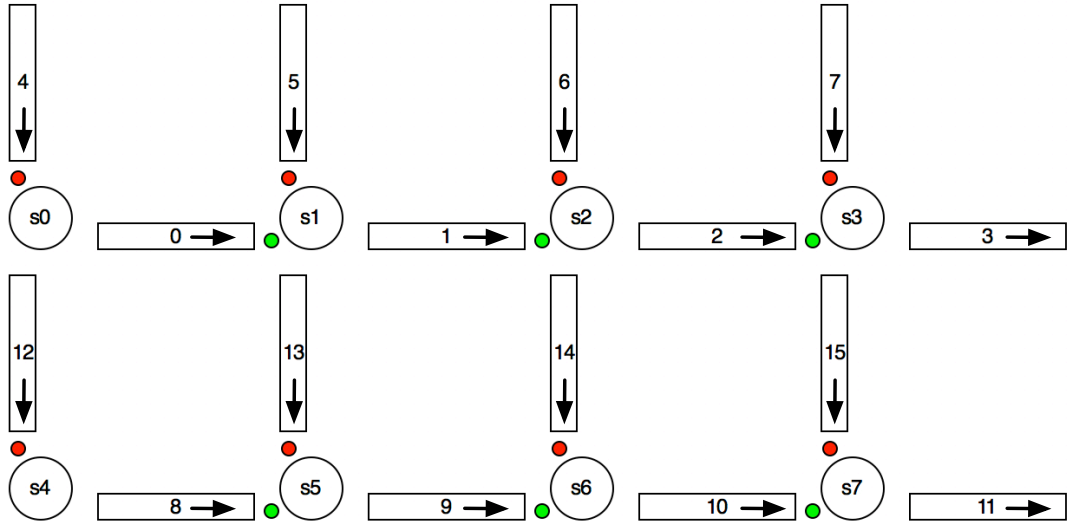


Figure 8.4: A traffic network composed of links $l_0, l_1, \ldots, l_{15}$ and signals $s_0, s_1, \ldots, s_7$. The ratio of the number of vehicles to the link capacity at a moment during simulation is shown on the links.

8.2 Case Study for Traffic System with 16 Links and 8 Signals

The example traffic network in our second case study consists of 16 links and 8 signals as shown in Fig. 8.4. The capacity, saturation flow, exterior flow, turn and supply ratio parameters for this traffic system are as follows:

$$x_i^{cap} = \begin{cases} 40, & \text{if } i \in \{0,1,2,3,8,9,10,11\} \\ 20, & \text{otherwise} \end{cases}$$

$$c_i = \begin{cases} 20, & \text{if } i \in \{0,1,2,3,8,9,10,11\} \\ 10, & \text{otherwise} \end{cases}$$

$$d_i^{max} = \begin{cases} 5, & \text{if } i \in \{0,4,5,6,7,12\} \\ 10, & \text{if } i \in \{8\} \\ 0, & \text{otherwise} \end{cases}$$

$$\beta_{ij} = \begin{cases} 0.9, & \text{if } i\text{-}j \in \{14\text{-}10, 15\text{-}11\} \\ 0.7, & \text{if } i\text{-}j \in \{0\text{-}1, 1\text{-}2, 2\text{-}3, 10\text{-}11, 12\text{-}8, 13\text{-}9\} \\ 0.5, & \text{if } i\text{-}j \in \{4\text{-}12, 5\text{-}13, 8\text{-}9, 9\text{-}10\} \\ 0.4, & \text{if } i\text{-}j \in \{4\text{-}1, 6\text{-}2, 6\text{-}14, 7\text{-}3, 7\text{-}15\} \\ 0.2, & \text{otherwise} \end{cases}$$

$$\alpha_{ij}^{k} = \begin{cases} 0.9, & \text{if } i\text{-}j\text{-}k \in \{13\text{-}9\text{-}5, 14\text{-}10\text{-}6, 15\text{-}11\text{-}7, 6\text{-}14\text{-}2, 7\text{-}15\text{-}3\} \\ 1, & \text{otherwise} \end{cases}$$

The overall LTL specifications for the traffic system given in Fig. 8.4 are as follows:

$$\begin{aligned}
&\Phi^r = \Phi_1 \wedge \Phi_2 \wedge \Phi_3 \wedge \Phi_4 \wedge \Phi_5 \wedge \Phi_6 \wedge \Phi_7 \wedge \Phi_8 \\
&\Phi_1 = \mathbf{G}(x_0 < 30 \wedge x_1 < 30) \wedge \mathbf{GF}x_0 < 20 \wedge \mathbf{GF}x_1 < 20 \\
&\Phi_2 = \mathbf{G}(x_4 < 15 \wedge x_5 < 15) \\
&\Phi_3 = \mathbf{G}(x_2 < 35 \wedge x_3 < 35) \wedge \mathbf{GF}x_2 < 25 \wedge \mathbf{GF}x_3 < 25 \\
&\Phi_4 = \mathbf{G}(x_6 < 15 \wedge x_7 < 15) \\
&\Phi_5 = \mathbf{G}(x_8 < 30 \wedge x_9 < 30) \wedge \mathbf{GF}x_8 < 20 \wedge \mathbf{GF}x_9 < 20 \\
&\Phi_6 = \mathbf{G}(x_{12} < 15 \wedge x_{13} < 15) \\
&\Phi_7 = \mathbf{G}(x_{10} < 35 \wedge x_{11} < 35) \wedge \mathbf{GF}x_{10} < 25 \wedge \mathbf{GF}x_{11} < 25 \\
&\Phi_8 = \mathbf{G}(x_{14} < 15 \wedge x_{15} < 15)
\end{aligned}$$

By using decomposition methods detailed in Section 5.1, we obtained 4 subsytems $(\mathcal{L}^1, \mathcal{V}^1)$, $(\mathcal{L}^2, \mathcal{V}^2)$, $(\mathcal{L}^3, \mathcal{V}^3)$, $(\mathcal{L}^4, \mathcal{V}^4)$:

$$\begin{aligned}
&\mathcal{L}^1 = \{l_0, l_1, l_4, l_5\}, \quad \mathcal{V}^1 = \{s_0, s_1\}, \\
&\mathcal{L}^2 = \{l_2, l_3, l_6, l_7\}, \quad \mathcal{V}^2 = \{s_2, s_3\}, \\
&\mathcal{L}^3 = \{l_8, l_9, l_{12}, l_{13}\}, \quad \mathcal{V}^3 = \{s_4, s_5\}, \\
&\mathcal{L}^4 = \{l_{10}, l_{11}, l_{14}, l_{15}\}, \quad \mathcal{V}^4 = \{s_6, s_7\}.
\end{aligned}$$

The corresponding specifications for each subsystem are

$$\Phi^1 = \Phi_1 \wedge \Phi_2, \quad \Phi^2 = \Phi_3 \wedge \Phi_4,$$
$$\Phi^3 = \Phi_5 \wedge \Phi_6, \quad \Phi^4 = \Phi_7 \wedge \Phi_8.$$

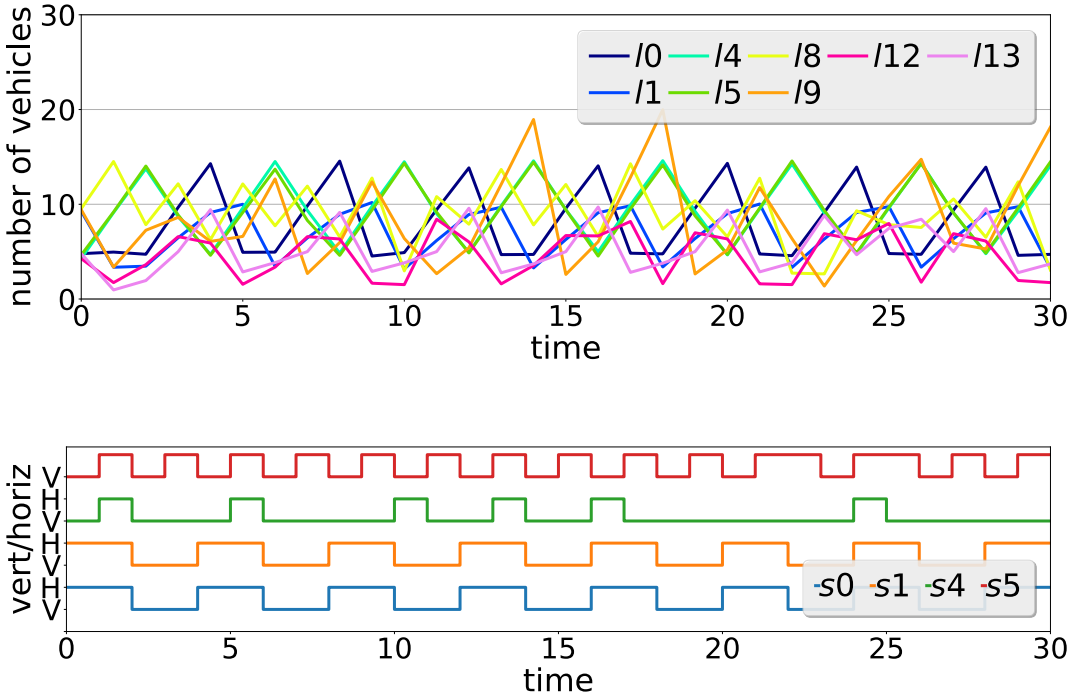


Figure 8.5: During 30 time steps, the number of vehicles on the links and the states of the signals for the subsystem-1 and subsystem-3. To enhance visibility, the links and signals are shown in two graphs.

The computation of the finite models took 13 seconds. In this example, we used method **(ESVSP)**. The total time for constructing the product of Büchi with TS and solving Büchi game for subsystems 1,2,3,4 were 16, 5, 613, and 112 seconds, respectively. The number of satisfying initial states/the number of states on the product were 29/7179, 40/1678, 756/66150 and 186/29106, for subsystems 1,2,3,4, respectively. The number of satisfying states for the overall traffic system (see(6.1)) is 256. The vehicle counts on each link and the state of the signals for the traffic system in closed loop with the synthesized controllers during 30 time steps are shown in Fig. 8.5 and 8.6. Fig. 8.5 contains the links and signals of subsystem-1 and subsystem-3, whereas Fig. 8.6 contains the links and signals of subsystem-2 and subsystem-4. We did not show all links and all signals in one graph because it was difficult to observe the results, hence we revealed them in 2 graphs.

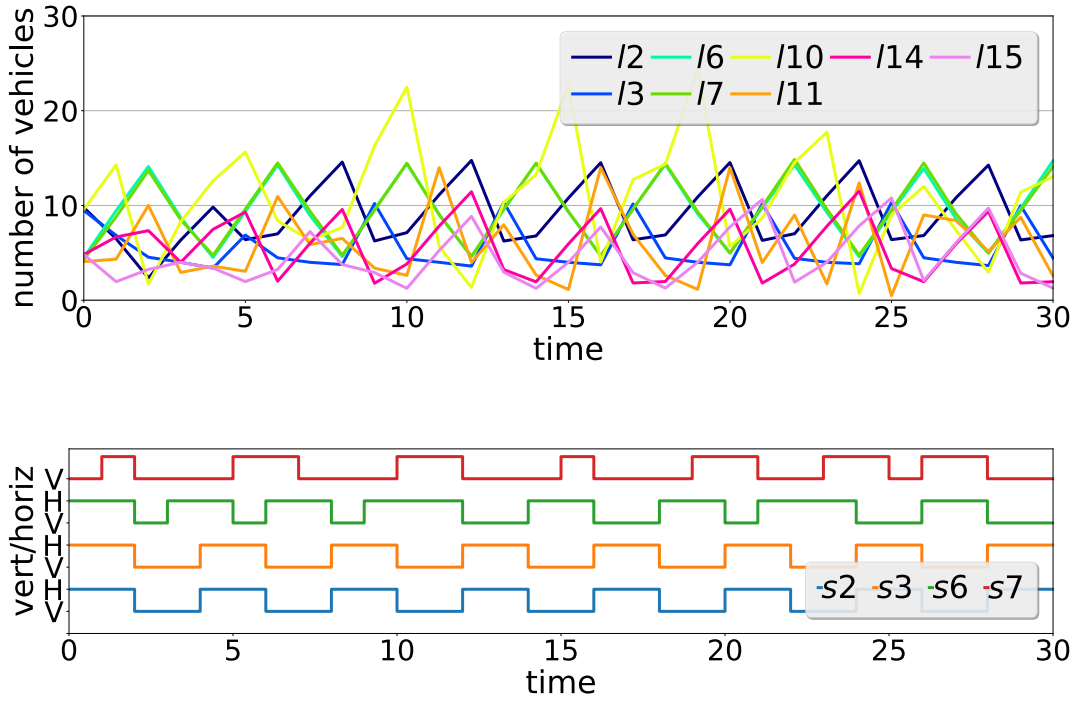


Figure 8.6: During 30 time steps, the number of vehicles on the links and the states of the signals for the subsystem-2 and subsystem-4. To enhance visibility, the links and signals are shown in two graphs.

The results will be added for the controller synthesis with bounded **F** specifications and optimization criteria.

CHAPTER 9

SUMMARY AND CONCLUSIONS

We studied the problem of synthesizing a signal control strategy for a traffic system from Linear Temporal Logic specifications. In particular, we focused on the scalability issue in the formal control of large traffic systems and proposed to tackle it by decomposing the main problem into smaller problems. We presented scalable controller synthesis techniques for traffic systems by decomposing the traffic system and LTL specifications into smaller parts while preserving requirements. We partitioned the main traffic system into subsystems with the developed decomposition algorithm. Then, we derived a specification for each of the subsystems from the main specification. In addition, additional constraints were derived on the signals lying on the boundaries of subsystems to ensure fairness. Abstraction based techniques were employed to find control strategies for subsystems. During the computation of a finite abstraction, the dynamics and specifications of adjacent systems were also considered. We proved the correctness of the solution by showing that if a strategy is found for each subsystem, then the overall traffic system satisfies the given specification when these controllers generate the corresponding signals. As each controller produces a control signal based on the local information available to its subsystem, the resulting strategy is decentralized.

The conservatism in the abstraction method was increased with the proposed solution. In other words, while a control strategy can be found for the overall transition system, proposed synthesis algorithm via partitioning might not produce any control strategy. One of the reason for the source of conservatism is that a subsystem considered all actions for the boundary links owned actually by another subsystem. These actions

could also be violating properties for the overall specification. For this purpose, we have developed two different techniques to reduce conservatism arisen from partitioning. These methods were 1) eliminating states violating safety properties **(ESVSP)** and 2) eliminating transitions leading to a trap state **(ETLTS)**. **(ESVSP)** method enabled us to reduce the conservatism on the controller synthesis by eliminating states violating properties of boundary link. The states of boundary link was controlled by first subsystem, hence other subsystems including this link in its dependency set eliminate violating states for the properties of such link. This method also reduced the computation time for the controller synthesis. **(ETLTS)** method also enabled us less conservative control strategy synthesis by marking automata states which have no outgoing transition and are not accepting final. When a product of this marked automaton and transition system was taken, the transitions which are outgoing to marked states were eliminated. Thus, the satisfying volume ratio in the product automaton state-space volume increased. **(ETLTS)** method is less conservative as it is detailed in Section 6.2, whereas this method increases the computation time because all state-space of the transition system and Büchi automata are used in the computation of the product automata. Thus, while we did not consider the aforementioned states and transitions during Büchi game, because of the determination of such states and transition and overall state space usage the computation time was increased. We showed the effectiveness of the proposed techniques on non-trivial examples with 9 and 16 links.

We proposed a technique to reduce conservatism by adding bounds for the **F** operator in LTL specification. We managed to increase the number of satisfying initial states in the overall state-space volume of the product automaton. Moreover, for the method **(ETLTS)** we observed that while the number of satisfying initial states in one of the subsystem decreased, we got the increased number of satisfying initial states for another subsystem. Thanks to this, it is possible to adjust the bounds for the **F** operator so as to synthesize control strategies even though any control strategy is not found from the original LTL specification for the partitioned systems.

We computed all feasible control actions satisfying overall specification for the abstract model of traffic system. Then, we proposed to find optimal control strategy among the computed ones by means of receding horizon framework. We have devel-

oped algorithms to produce optimal control strategies for two different optimization criteria *e.g.,* minimization of the total number of signal switches and minimization of the maximum density of vehicles among all links for the given horizon length. The proposed algorithms minimized the overall cost over the controllable product automaton trajectories. These algorithms optimized control strategy synthesis efficiently without enforcement of the other issues such as stability. The correctness of the all proposed techniques have been guaranteed formally in this dissertation.

REFERENCES

[1] K. Aboudolas, M. Papageorgiou, A. Kouvelas, and E. Kosmatopoulos. A rolling-horizon quadratic-programming approach to the signal control problem in large-scale congested urban road networks. *Transportation Research Part C: Emerging Technologies*, 18(5):680–694, 2010.

[2] R. E. Allsop. Sigset: a computer program for calculating traffic signal settings. *Traffic Engineering & Control*, 1971.

[3] R. E. Allsop. Sigcap: A computer program for assessing the traffic capacity of signal-controlled road junctions. *Traffic Engineering & Control*, 17(Analytic), 1976.

[4] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(7):971–984, 2000.

[5] R. Alur, S. Moarref, and U. Topcu. Compositional synthesis with parametric reactive controllers. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, pages 215–224. ACM, 2016.

[6] D. E. Associates. *Traffic Control Systems Handbook*. Dunn Engineering Associates, Westhampton Beach, 2005.

[7] E. Aydin Gol, M. Lazar, and C. Belta. Language-guided controller synthesis for discrete-time linear systems. In *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control*, pages 95–104. ACM, 2012.

[8] C. Baier, J.-P. Katoen, and K. G. Larsen. *Principles of model checking*. MIT press, 2008.

[9] S. Baldi, I. Michailidis, E. B. Kosmatopoulos, and P. A. Ioannou. A" plug and play" computationally efficient approach for control design of large-scale nonlinear systems using cosimulation: a combination of two" ingredients". *IEEE Control Systems*, 34(5):56–71, 2014.

[10] S. Baldi, I. Michailidis, V. Ntampasi, E. B. Kosmatopoulos, I. Papamichail, and M. Papageorgiou. Simulation-based synthesis for approximately optimal urban traffic light management. In *American Control Conference (ACC), 2015*, pages 868–873. IEEE, 2015.

[11] J. Barceló and J. Casas. Dynamic network simulation with aimsun. *Simulation approaches in transportation analysis*, pages 57–98, 2005.

[12] G. Batt, M. Page, I. Cantone, G. Goessler, P. Monteiro, and H. De Jong. Efficient parameter search for qualitative models of regulatory networks using symbolic model checking. *Bioinformatics*, 26(18):i603–i610, 2010.

[13] A. L. Bazzan. Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multi-Agent Systems*, 18(3):342–375, 2009.

[14] C. Belta, B. Yordanov, and E. A. Gol. Formal methods for discrete-time dynamical systems, 2017.

[15] A. Bhatia, M. R. Maly, L. E. Kavraki, and M. Y. Vardi. Motion planning with complex goals. *IEEE Robotics & Automation Magazine*, 18(3):55–64, 2011.

[16] E. Bourrel and J.-B. Lesort. Mixing microscopic and macroscopic representations of traffic flow: Hybrid model based on lighthill-whitham-richards theory. *Transportation Research Record: Journal of the Transportation Research Board*, (1852):193–200, 2003.

[17] M. S. Branicky, V. S. Borkar, and S. K. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE transactions on automatic control*, 43(1):31–45, 1998.

[18] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri. Nusmv: A new symbolic model verifier. In *International conference on computer aided verification*, pages 495–499. Springer, 1999.

[19] E. Clarke, A. Fehnker, Z. Han, B. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald. Abstraction and counterexample-guided refinement in model checking of hybrid systems. *International journal of foundations of computer science*, 14(04):583–604, 2003.

[20] E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Workshop on Logic of Programs*, pages 52–71. Springer, 1981.

[21] E. M. Clarke, D. Peled, and O. Grumberg. *Model checking*. MIT Press, 1999.

[22] E. M. Clarke and J. M. Wing. Formal methods: State of the art and future directions. *ACM Computing Surveys (CSUR)*, 28(4):626–643, 1996.

[23] S. Coogan, E. A. Gol, M. Arcak, and C. Belta. Traffic network control from temporal logic specifications. *IEEE Transactions on Control of Network Systems*, 3(2):162–172, 2016.

[24] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the ACM (JACM)*, 42(4):857–907, 1995.

[25] M. Cremer and J. Ludwig. A fast simulation model for traffic flow on the basis of boolean operations. *Mathematics and Computers in Simulation*, 28(4):297–303, 1986.

[26] C. F. Daganzo. The cell transmission model, part ii: network traffic. *Transportation Research Part B: Methodological*, 29(2):79–93, 1995.

[27] H. De Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bulletin of mathematical biology*, 66(2):301–340, 2004.

[28] X. C. D. Ding, S. L. Smith, C. Belta, and D. Rus. Ltl control in uncertain environments with probabilistic satisfaction guarantees. *IFAC Proceedings Volumes*, 44(1):3515–3520, 2011.

[29] A. Donzé, T. Ferrere, and O. Maler. Efficient robust monitoring for stl. In *International Conference on Computer Aided Verification*, pages 264–279. Springer, 2013.

[30] E. A. Emerson and J. Y. Halpern. "sometimes" and "not never" revisited: on branching versus linear time temporal logic. *Journal of the ACM (JACM)*, 33(1):151–178, 1986.

[31] F. Fages and A. Rizk. On temporal logic constraint solving for analyzing numerical data time series. *Theoretical Computer Science*, 408(1):55–65, 2008.

[32] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

[33] M. Fellendorf. Vissim: A microscopic simulation tool to evaluate actuated signal control including bus priority. In *64th Institute of Transportation Engineers Annual Meeting*, pages 1–9. Springer, 1994.

[34] S. K. Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Matematicheskii Sbornik*, 89(3):271–306, 1959.

[35] E. A. Gol, M. Lazar, and C. Belta. Temporal logic model predictive control. *Automatica*, 56:78–85, 2015.

[36] J. Gregoire, X. Qian, E. Frazzoli, A. De La Fortelle, and T. Wongpiromsarn. Capacity-aware backpressure traffic signal control. *IEEE Transactions on Control of Network Systems*, 2(2):164–173, 2015.

[37] T. A. Henzinger, S. Qadeer, and S. K. Rajamani. You assume, we guarantee: Methodology and case studies. In *International Conference on Computer Aided Verification*, pages 440–451. Springer, 1998.

[38] B. Heydecker and J. D. Addison. Analysis and modelling of traffic flow under variable speed limits. *Transportation Research Part C: Emerging Technologies*, 19(2):206–217, 2011.

[39] G. Holzmann. *Spin model checker, the: primer and reference manual*. Addison-Wesley Professional, 2003.

[40] P. Huang, L. Kong, and M. Liu. The study on the one-dimensional random traffic flow model. *Acta Physica Sinica*, 50(1):30–36, 2001.

[41] G. Improta and G. Cantarella. Control system design for an individual signalized junction. *Transportation Research Part B: Methodological*, 18(2):147–167, 1984.

[42] W.-L. Jin. Continuous kinematic wave models of merging traffic flow. *Transportation research part B: methodological*, 44(8):1084–1103, 2010.

[43] W.-L. Jin. A kinematic wave theory of multi-commodity network traffic flow. *Transportation Research Part B: Methodological*, 46(8):1000–1022, 2012.

[44] W.-L. Jin. Analysis of kinematic waves arising in diverging traffic flow models. *Transportation Science*, 49(1):28–45, 2014.

[45] S. Karaman and E. Frazzoli. Sampling-based motion planning with deterministic μ-calculus specifications. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 2222–2229. IEEE, 2009.

[46] S. Karaman and E. Frazzoli. Incremental sampling-based algorithms for optimal motion planning. *Robotics Science and Systems VI*, 104, 2010.

[47] E. S. Kim, M. Arcak, and S. A. Seshia. Compositional controller synthesis for vehicular traffic networks. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 6165–6171. IEEE, 2015.

[48] M. Kloetzer and C. Belta. A fully automated framework for control of linear systems from temporal logic specifications. *IEEE Transactions on Automatic Control*, 53(1):287–297, 2008.

[49] M. Kloetzer and C. Belta. Automatic deployment of distributed teams of robots from temporal logic motion specifications. *IEEE Transactions on Robotics*, 26(1):48–61, 2010.

[50] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas. Temporal-logic-based reactive mission and motion planning. *IEEE transactions on robotics*, 25(6):1370–1381, 2009.

[51] T. Kropf. *Introduction to formal hardware verification*. Springer Science & Business Media, 2013.

[52] M. Kwiatkowska, G. Norman, and D. Parker. Prism 4.0: Verification of probabilistic real-time systems. In *Computer aided verification*, pages 585–591. Springer, 2011.

[53] M. Lahijanian, S. B. Andersson, and C. Belta. Temporal logic motion planning and control with probabilistic satisfaction guarantees. *IEEE Transactions on Robotics*, 28(2):396–409, 2012.

[54] K. Lano. *The B Language and Method: a guide to practical formal development*. Springer Science & Business Media, 2012.

[55] J. A. Laval and L. Leclercq. A mechanism to describe the formation and propagation of stop-and-go waves in congested freeway traffic. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 368(1928):4519–4541, 2010.

[56] J. Lebacque and M. Khoshyaran. First order macroscopic traffic flow models for networks in the context of dynamic assignment. In *Transportation Planning*, pages 119–140. Springer, 2002.

[57] M. Z. Li. A generic characterization of equilibrium speed-flow curves. *Transportation Science*, 42(2):220–235, 2008.

[58] M. J. Lighthill and G. B. Whitham. On kinematic waves. ii. a theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, pages 317–345, 1955.

[59] R. Majumdar, E. Render, and P. Tabuada. Robust discrete synthesis against unspecified disturbances. In *Proceedings of the 14th international conference on Hybrid systems: computation and control*, pages 211–220. ACM, 2011.

[60] A. D. May. *Traffic flow fundamentals*. 1990.

[61] P. Mirchandani and L. Head. A real-time traffic signal control system: architecture, algorithms, and analysis. *Transportation Research Part C: Emerging Technologies*, 9(6):415–432, 2001.

[62] K. Nagel and M. Schreckenberg. A cellular automaton model for freeway traffic. *Journal de physique I*, 2(12):2221–2229, 1992.

[63] P. Nilsson and N. Ozay. Synthesis of separable controlled invariant sets for modular local control design. In *American Control Conference (ACC), 2016*, pages 5656–5663. IEEE, 2016.

[64] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang. Review of road traffic control strategies. *Proceedings of the IEEE*, 91(12):2043–2067, 2003.

[65] L. A. Pipes. An operational analysis of traffic dynamics. *Journal of applied physics*, 24(3):274–281, 1953.

[66] G. C. . B. Pishue. INRIX 2016 global traffic scorecard. `http://inrix.com/resources/inrix-2016-global-traffic-scorecard/`. Accessed: 2017-10-23.

[67] E. Plaku, L. E. Kavraki, and M. Y. Vardi. Motion planning with dynamics by a synergistic combination of layers of planning. *IEEE Transactions on Robotics*, 26(3):469–482, 2010.

[68] A. Pnueli. The temporal logic of programs. In *Foundations of Computer Science, 1977., 18th Annual Symposium on*, pages 46–57. IEEE, 1977.

[69] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 179–190. ACM, 1989.

[70] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia. Model predictive control with signal temporal logic specifications. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 81–87. IEEE, 2014.

[71] A. Reuschel. Vehicle movements in a platoon. *Oesterreichisches Ingenieur-Archir*, 4:193–215, 1950.

[72] A. Rizk, G. Batt, F. Fages, and S. Soliman. Continuous valuations of temporal logic specifications with applications to parameter optimization and robustness measures. *Theoretical Computer Science*, 412(26):2827–2839, 2011.

[73] M. Rungger and M. Zamani. Compositional construction of approximate abstractions. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, pages 68–77. ACM, 2015.

[74] S. L. Smith, J. Tumova, C. Belta, and D. Rus. Optimal path planning for surveillance with temporal-logic constraints. *The International Journal of Robotics Research*, 30(14):1695–1708, 2011.

[75] P. Tabuada and G. J. Pappas. Linear time logic control of discrete-time linear systems. *IEEE Transactions on Automatic Control*, 51(12):1862–1877, 2006.

[76] T.-Q. Tang, L. Caccetta, Y.-H. Wu, H.-J. Huang, and X.-B. Yang. A macro model for traffic flow on road networks with varying road conditions. *Journal of Advanced Transportation*, 48(4):304–317, 2014.

[77] Tomtom. TomTom traffic index. `https://www.tomtom.com/en_gb/trafficindex/list?citySize=LARGE&continent=ALL&country=ALL`. Accessed: 2017-10-22.

[78] U. Topcu, N. Ozay, J. Liu, and R. M. Murray. On synthesizing robust discrete controllers under modeling uncertainty. In *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control*, pages 85–94. ACM, 2012.

[79] M. Treiber and A. Kesting. Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, 2013.

[80] F. van Wageningen-Kessels, H. van Lint, S. Hoogendoorn, and K. Vuik. Lagrangian formulation of multiclass kinematic wave model. *Transportation Research Record: Journal of the Transportation Research Board*, (2188):29–36, 2010.

[81] F. van Wageningen-Kessels, Y. Yuan, S. P. Hoogendoorn, H. Van Lint, and K. Vuik. Discontinuities in the lagrangian formulation of the kinematic wave model. *Transportation Research Part C: Emerging Technologies*, 34:148–161, 2013.

[82] F. Wang, L. Li, J.-M. Hu, Y. Ji, R. Ma, and R. Jiang. A markov-process inspired ca model of highway traffic. *International Journal of Modern Physics C*, 20(01):117–131, 2009.

[83] F. V. Webster. Traffic signal settings. Technical report, 1958.

[84] T. Wongpiromsarn, U. Topcu, and R. M. Murray. Receding horizon temporal logic planning. *IEEE Transactions on Automatic Control*, 57(11):2817–2830, 2012.

[85] B. Yordanov, J. Tumova, I. Cerna, J. Barnat, and C. Belta. Temporal logic control of discrete-time piecewise affine systems. *IEEE Transactions on Automatic Control*, 57(6):1491–1504, 2012.

[86] Y. Yuan, J. Van Lint, R. E. Wilson, F. van Wageningen-Kessels, and S. P. Hoogendoorn. Real-time lagrangian traffic state estimator for freeways. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):59–70, 2012.

[87] H. Zhu, H. Ge, and S. Dai. A new cellular automaton model for traffic flow with different probability for drivers. *International Journal of Modern Physics C*, 18(05):773–782, 2007.