Zonglin Liu

Optimizing Control of Distributed Cyber-Physical Systems

kassel university press

Zonglin Liu

Optimizing Control of Distributed Cyber-Physical Systems

kassel university press

This work has been accepted by Faculty of Electrical Engineering/Computer Science of the University of Kassel as a thesis for acquiring the academic degree of Doktor der Ingenieurwissenschaften (Dr.-Ing.).

Supervisor: Prof. Dr.-Ing. Olaf Stursberg
Co-Supervisor: Prof. Dr.-Ing. Daniel Görges
Defense day: 24 March 2021

Bibliographic information published by Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at http://dnb.dnb.de.

Zugl.: Kassel, Univ., Diss. 2021
ISBN 978-3-7376-0976-0
DOI: https://doi.org/doi:10.17170/kobra-202107294443

https://kup.uni-kassel.de

Printing Shop: Print Management Logistik Service, Kassel
Printed in Germany

Contents

Summary

In this thesis, a set of modeling and control strategies are proposed for Cyber-physical systems (CPS), which aim at ensuring a safe, reliable, and highly performant operation of each local subsystem contained in the CPS. Modeling of CPS is challenging since not only must the tight interconnection of continuous and discrete dynamics of local subsystems be exactly represented, but so must also the interleaving structure between different subsystems. Optimal control of CPS, accordingly, should take into account not only the local mixed dynamics by local controller synthesis, but also the influence from other subsystems around.

To model a large variety of physical processes containing continuous and discrete behavior, a type of hybrid system HA is first introduced in this thesis. Compared to standard modeling techniques, such as piecewise affine (PWA) systems, the proposed HA is capable of encompassing both, autonomous switching and externally triggered switching between different continuous dynamics. By assuming each subsystem in CPS is modeled by HA, three different interleaving structures among the subsystems are considered in this thesis, namely: 1.) the influence from other subsystems is cast into a time-invariant change of the local HA; 2.) the influence is cast into an uncertain and time-varying change of the local HA; 3.) the influence is cast into coupling constraints to be jointly satisfied by all subsystems.

For the first case, in which no uncertainty is encountered, the major task of this thesis is to ensure the optimality of the local control strategy and the efficiency of the process to determine such a control strategy. Different methods are thus proposed to encode the hybrid dynamics of the HA, based on which the optimal control strategies are determined by solving mixed-integer programming (MIP) problems. For the second case, in which the local HA varies over time with uncertainties, means are introduced to ensure the robustness of the control strategies. For the online application of a selected strategy, another important task is to ensure the continuity of the operation despite uncertain changes of the environment. Regarding this problem, which refers to the concept of recursive feasibility, methods are introduced to preserve the continuity. For the last case, in which coupling constraints arise, a particular difficulty of controlling CPS is to provide such means that the coupling constraints are satisfied while the optimality is ensured. The most promising way is to employ a centralized solution, but the size, e.g., the number of subsystems, may quickly let it become intractable. Accordingly, a set of distributed solution strategies with feasibility and performance guaranties are introduced in this thesis. By applying these distributed strategies, the centralized problem is cast into a set of small-scale problems to be solved in parallel. This distributed scheme also ensures

a minor increase of the overall complexity when more subsystems are included in the CPS.

Last, besides the numerical experiments tested for each proposed method, a set of CPS-relevant practical studies are introduced for each considered case. Efficiency and reliability of the proposed methods are confirmed by all these tests and studies.

Acknowledgments

This thesis is a summary of my research during the last six years at the Institute of Control and System Theory of the University of Kassel. Six years are a long journey, and I would like to express gratitude towards all the people who accompanied me on the journey.

First and foremost, I would like to thank my supervisor Prof. Dr.-Ing. Olaf Stursberg, who welcomed me as a student assistant in 2014, and later offered me a PhD position in 2015. Thanks for the trust, patience, support, advice, and also numerous valuable discussions, which are the most precious gift one could get in the academic career. I would also like to thank Prof. Dr. rer. nat Arno Linneman, who re-sparked my interest to control theory when I first came to Germany in 2013. I am also grateful to Prof. Dr.-Ing. Daniel Görges for being on the thesis committee and examining my thesis, as well as Prof. Dr. rer. nat. Bernhard Sick for being on the defense committee. Furthermore, I would also like to thank for the financial support provided by the European Union through the H2020-project UnCoVerCPS, and also the possibility to get to know so much nice people during the project meetings.

I am also thankful to all my colleagues, Damian, Jan, Moritz, Jannik, and many others for the fruitful discussions and unforgettable trips. A special thank to the institute secretary, Mrs. Elena Rapp, who helped me in dealing with different administrative affairs, which were really not easy for a foreigner who has just come here.

Last but not least, I would like to thank my parents for their support over the years, and if there is one thing I would regret for the past years, that would be spending so little time with them since I was 15. Same thanks also to my mother-in-law, who gives me so much support since I came to Germany. Finally, my wife Charlen, who is also my best friend, this journey is because of you and also for you.

Kassel, April 2021 Zonglin Liu

Part I.

Introduction and (Theoretical) Background

1. Introduction and Problem Background

1.1. Introduction

Cyber-physical systems (CPS) have attracted great interest in recent years, not only because of their relevance for applications such as „Industrie 4.0", autonomous driving vehicles or human-robot collaboration, but also in academic research due to open system-theoretic questions on: 1.) how the tight interconnection of software and physical elements can be realized (leading to combined continuous and discrete dynamics) and 2.) how networked structures arising either from the coupling of the dynamic behavior of subsystems or from the communication of joint goals or specifications should be designed. Meanwhile, the challenges for modeling and control of CPS are further complicated as the sizes of systems are growing ever more: This is due to an increased complexity of the discrete dynamics interleaving with the continuous dynamics in each local subsystem, and to an increased number of subsystems interacting with each other.

A typical structure of CPS is illustrated in Fig. 1.1, where the plant of each local subsystem contains both continuous and discrete behaviors (the continuous part arises from modeling physical process and the discrete part represents some discrete decisions selected from a finite set), and a change of a local plant may also affect the evolution of other plants. Therefore, the local controllers have to coordinate with each other via the network in order to ensure that the local control objective is realized despite the complex dynamics and the influence of other subsystems. Some example scenarios of a CPS with this structure and taken from the domain of autonomous driving are illustrated in Fig. 1.2: The one on the left considers a vehicle overtaking problem, whereas the other one refers to a narrow passage problem at a building site. For both scenarios, a mixed decision consisting of both, a discrete decision (e.g. whether to change the lane or not, or who should drive first through the passage) and a continuous decision (e.g. velocity and acceleration over time) must be made by each local vehicle, while the behavior of other vehicles around must be taken into account during the local decision process through the network.

For the modeling of CPS, a general model class to describe the dynamics in each subsystem is often required, so that a large variety of the physical processes can be covered. As shown in the examples in Fig. 1.2, such a model should be able to

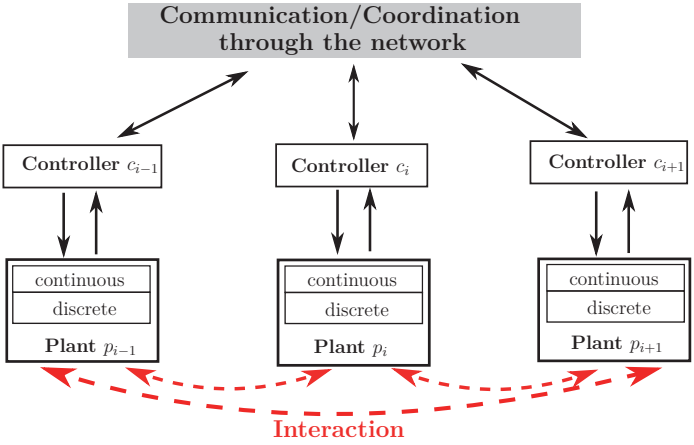

Figure 1.1.: Networked structure of CPS: each local subsystem $i \in \{1, \cdots, n_s\}$ consists of both continuous and discrete dynamics in the plant. Interactions between the subsystems are caused by either physical coupling or coupled cost criteria. For the local controller c_i, not only should the local mixed dynamics be considered, but also the interactions among the subsystems. It is thus necessary to communicate and coordinate local controlled behavior through a network according to certain protocol, such that the safety and optimality of all subsystems in CPS are guaranteed.

comprise both continuous and discrete dynamics. In addition, as the evolution of the continuous dynamics affects the evolution of the discrete part, or vice versa, this model should also be able to describe the interactions between the two dynamics, such as through suitable logic conditions. For the control part in CPS, a main goal is to guarantee the satisfaction of all safety properties of the overall system, while at the same time controller adaption in response to changes of the system or its local environment should enable high system performance in attaining the control goals.

Clearly, the tasks of modeling of and controller design for CPS are not independent of each other, but highly correlated. For example, as the use of an approximating model is sometimes preferred since it can lead to reduced system size (e.g. the use of linearized continuous dynamics instead of the original nonlinear one derived from the physical law, or neglecting some redundant discrete decisions), a larger set of uncertainties caused by approximation errors may arise and must be taken into account for controller synthesis. This may lead to relatively conservative control strategies and thus to worsening the performance, or even threatening the guaranty

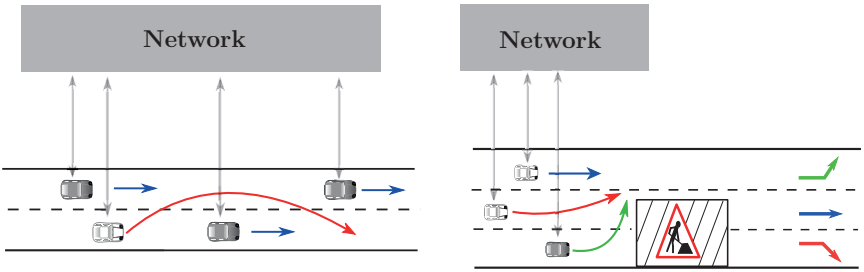

Figure 1.2.: Application of CPS in different autonomous driving problems: in both cases, local autonomous vehicles must communicate as well as coordinate their continuous and discrete behavior through the network in order to ensure a safe and reliable driving maneuver of all vehicles.

of safety of the controlled behavior.

Accordingly, the objective of this thesis is to develop new means for both modeling and control of CPS, such that a good control performance is achieved together with a guaranty of safety of the system. Before the details of these new means are exposed, existing methods in the given context are first surveyed below. Note that more detailed references to existing work are additionally provided at the beginning of each chapter.

1.1.1. Modeling of Hybrid Dynamics in CPS

To account for the requirements in CPS modeling, the class of hybrid systems is preferred in this thesis due to its high flexibility in describing the interleaving between the transitions among discrete states and the evolution of the continuous states. Typically, a change of the discrete state affects the continuous dynamics as well as the continuous state from which the further evolution starts, while the continuous state evolution determines the discrete successor state reached by the upcoming transition. An example is the automatic gear-shift of a vehicle, where the engine speed (continuous state) decides whether the gear (discrete state) should be shifted, and the chosen gear will also affect the evolution of engine speed.

According to different transition mechanisms of the discrete states, hybrid systems have been further classified into autonomously switching systems, switched systems (controlled switching systems), or the combination of both [37]. More details of this classification can be found at the beginning of the next chapter. Among the existing work in CPS modeling by using hybrid systems, the use of piecewise affine (PWA) systems, which belong to the class of switching systems, has attracted much attention in the last few years, see [33, 75, 21, 15]. There, the continuous state space

is partitioned into a set of subspaces by hyperplanes, and a discrete state is assigned to each of the subspaces. In different subspaces, different continuous dynamics are followed, and a change of the continuous dynamics is triggered by the state evolution crossing through a hyperplane constituting the boundary of a partition. Obviously, as the crossing of a boundary is the only way to change the discrete state in this scheme, physical processes in which the change of the discrete state is not bounded to crossing a hyperplane cannot be represented by using PWA systems. Thus, more general types of hybrid systems are required to model more general phenomena in CPS.

1.1.2. Controller Synthesis of CPS

Challenges in controller synthesis of CPS are manifold, but main objective remains nevertheless to ensure a safe, reliable and highly performant operation of the networked system. Especially for the networked structure in Fig. 1.1, the control strategy of each local subsystem must be adapted in order to overcome the difficulties of 1.) high complexity caused by different dynamics, parameters, objectives, and physical constraints defined for each subsystem; 2.) the uncertainty caused by incomplete knowledge during system modeling, measurement error of the changing environment and communication error/delay among the subsystems, and 3.) limited computational power which may make it difficult to find the optimal control strategy online.

Nevertheless, before any of these challenges can be addressed, a fundamental issue lies in encoding the transition mechanisms of the hybrid dynamics before the controller synthesis step. Note that for different transition mechanisms in given hybrid systems, the solution to the control problem (e.g. optimal control, robust control, etc.) is also different: for the optimal control of PWA systems (or the equivalent class of mixed logical dynamic (MLD) Systems [20, 19, 31, 75]), the hybrid dynamics in most cases is cast first into piecewise linear dynamics including real and binary variables. Then, a mixed integer programming (MIP) problem is solved in order to obtain the optimal control strategy. A similar procedure is possible for piecewise nonlinear dynamics, which then leads to mixed integer nonlinear programming (MINLP) problem [87, 125]. Another approach is to use dynamic programming (DP) or the maximum principle for the controller synthesis of switching or switched systems in continuous-time domain, see [31, 74, 139, 129]. The use of temporal logic [134, 57] to encode more elaborate logic specifications of the hybrid systems evolution were addressed in [80, 122, 62], but are mainly applied to PWA systems [166, 77, 170].

Note that most of these approaches (especially for discrete-time hybrid systems) need to use integer variables to encode the logic conditions behind the discrete and continuous state evolution, i.e., a MIP problem is formed, which is not convex, and it is also known that it cannot be solved in polynomial time [49]. To reduce the complexity of the controller synthesis, the strategy of Model Predictive Control

(MPC) [65, 109, 137, 108] is often applied, which only takes into account a finite time domain of the controlled behavior in any iteration, and thus reduces the problem size. The optimal control action is determined repeatedly in each iteration or time instance in MPC, until the control objective is achieved. The state and input constraints can also be easier handled via MPC than in traditional control strategies such as Linear Quadratic Regulators (LQR) [84, 10], and this is especially important for fulfilling safety requirements in CPS.

Robust Control of Uncertain CPS

Controller synthesis for local subsystems of CPS may encounter uncertainty, caused by either local modeling error or an inexact model to describe the interaction among the subsystems. For the former class of uncertainty arising in the local hybrid plant (e.g. random failures causing unexpected transitions from one discrete state to another, random resets of continuous states during the transitions, or disturbances affecting the continuous behavior), the work in [79, 3] derived a robust control law specific to PWA systems with additive disturbances through computation of robust controllable sets. For the same class of PWA systems, the authors in [115, 143] aimed at setting up a min-max problem to enforce the robustness. The work in [73, 67] preserved the robustness by using a robust invariant set to bound the effect of the disturbances for PWA systems and linear switched systems. In addition, the work in [113] proposed a robust switching law for autonomously switching systems, and the work in [2, 14, 15] focused on another class of uncertainty in which the discrete transition structure is probabilistic.

Besides the uncertainties affecting a local plant, a change of the environment (which may only be partially known) of the local subsystem may also yield uncertainties. This usually occurs when the impact of the interactions between subsystems cannot be exactly modeled nor predicted. Note that this type of uncertainty is even more critical when an online strategy like MPC is applied, since the change of the environment (usually leading to a change of the local state constraints) may lead to a loss of recursive feasibility and stability of the strategy. However, the controller synthesis problem for varying environments has rarely been considered in the past, with the following exceptions: The work in [163, 107] focused on MPC with time-varying input and state constraints, where the pattern of how the constraints change is assumed to be known a-priori. Techniques of explicit MPC rely on state-space partitioning, which has to be provided in offline computations [5] – considering all configurations (and thus different partitions) which may occur in applications like autonomous driving seems not realistic. The work in [106] introduced the method of homotheticly changing the terminal region in order to provide stability guarantees, despite changes of the state and input constraints. However, recursive feasibility was not addressed in that work, although it is an important pre-requisite of stability of predictive controllers [104]. In a more recent work in [145], a collision avoidance problem by using MPC is considered, where the obsta-

cles (representing the environment) are moving over time with uncertainties. The authors proposed means to ensure recursive feasibility, constraint satisfaction and robust collision avoidance for given problem settings.

Distributed Control of CPS

The distributed control of CPS has been proposed as an important class of techniques, when the control problem is too complicated to be solved in a centralized fashion (note that as communication and coordination between subsystems are required in the structure considered in Fig. 1.1, the notion of distributed control is used instead of decentralized control throughout this thesis). For two main types of coupling considered in CPS, i.e. criteria of coupled costs and coupled constrained sets, the development of distributed control strategies aims at decomposing the centralized problem into a set of small-scale sub-problems in order to reduce the problem size. Most of the discussions on distributed control algorithms are limited to linear dynamics and focus on how to achieve the global optimum in an iterative scheme with low communication frequency [70, 59, 52, 172], or for a time-varying communication graph [42, 117, 119]. For networked hybrid systems, where both real and integer variables are involved in the centralized program, only a few results have been addressed so far with respect to the distributed control problem. In [34], an iterative distributed MPC algorithm for hybrid systems was proposed, but the considered class of hybrid systems is far from general (linear systems with discrete valued inputs). In [71], a distributed control strategy of coupled PWA systems was proposed, but only a local optimum can be achieved through the proposed communication scheme.

Techniques for distributed optimization of MIP problems have also been developed in the past few years, e.g. in [162, 161, 60, 124], and they can possibly be applied to realize distributed control in CPS, since integer variables are applied to encode the hybrid dynamics. However, most of these techniques are limited to the case when the cost function (representing the control objective) is linear – although a quadratic one is more common in control methods such as MPC.

1.2. Outline of this Dissertation

In this thesis, a general class of hybrid system HA is first defined, which is able to describe a larger variety of transition mechanisms than is described in literature for existing methods, e.g., for PWA systems. Then, based on modeling each subsystem of a CPS by the considered general hybrid system, a series of controller synthesis problems and their solutions are proposed for different interaction schemes of CPS:

In Chapter 2, it is first assumed that the interaction between other subsystems and a subsystem i is static, and the influence to subsystem i can be cast into additional

deterministic constraints for subsystem i. By modeling the dynamics of subsystem i with the proposed HA, the local controller design thus takes all its local hybrid dynamics and constraints, and the additional constraints into account. An optimal control problem over a finite time horizon is thus formulated and considered in this chapter, where the transition mechanisms (including switching, controlled switching and conditional controlled switching) of local hybrid dynamics are translated into algebraic programs, to which existing solvers can be applied. As integer variables are used for the translation, which is crucial for the ability to encode logic constraints (e.g. the transition mechanisms) but leads to high complexity of the numerical programs, two translation schemes are proposed in this chapter to enhance efficiency. The first one requires to enumerate all possible discrete state sequences, while the other does not (advantages and disadvantages of the two schemes are also discussed in this chapter). Both schemes can ensure the exactness of the translation and thus the optimality of the control action obtained (for the discrete-time case). In both schemes, an additional constraint purely among the integer variables is also formulated, which can significantly reduce the available combinations of the integer variables (and thus the computational complexity), with the same amount of necessary integer variables. At the end of this chapter, an autonomous vehicle overtaking problem is considered, and it is shown that safe trajectories can be planned by modeling the over-taking procedure with considered hybrid systems.

In Chapter 3, the focus still remains on the local problem of subsystem i, but now with uncertainties. Different classes of uncertainties are considered in this chapter, caused by either modeling errors for local plants or modeling errors for the environment (interaction). For the uncertainty caused by the former, a robust control method is proposed, which ensures that the local control objective can be achieved despite the uncertainties. This method is tailored to the type of hybrid systems in the last chapter, that a robust strategy implies that given hybrid semantics are always followed despite the uncertainties, while the desired control tasks are always achieved. In this approach, a so-called „tube-based" method [86, 110, 67] which was used to handle uncertainties in simpler dynamics, is applied to the considered hybrid dynamics. The other type of uncertainty, the one caused by environment modeling errors, usually arises when the interaction is changing over time and no model is at hand to describe or to predict the change. This is a more realistic setting than the static interaction scheme considered in Chapter 2. For this case, an MPC strategy is developed in order to guarantee recursive feasibility and stability despite the uncertainties. However, these guarantees come with a price of conservativeness with respect to a tightening of the feasible state and input space, i.e., no feasible control action may be found in the worst case. For this problem, a less conservative approach is introduced which uses a penalty term of the cost function to enforce the robustness (instead of tightening the feasible space). Numerical examples confirm that the same robustness can be achieved in most cases by employing this approach (but cannot provide guaranty as the tightening approach). At the end, a

human-robot collaboration problem is considered, where uncertainties are presented for both robot plant and human motion prediction. It is shown that by using the methods proposed in this chapter, a safe and reliable collaboration between robot and human can be guaranteed.

In Chapter 4, the focus is shifted to distributed control problems for CPS. The background of the problems is that, it may not always be possible to cast the interaction among the subsystems into local constraints, and the subsystems may have to decide upon their control actions jointly. This may happen if, e.g., shared resources are considered in CPS and allocation strategies are required to distribute these resources optimally. The narrow passage problem in Fig. 1.2 belongs to one of these problems, where the priority to travel through the passage is a kind of resource to be distributed among the vehicles. As most of the local control problems have been eventually cast into MIP problems in the preceding chapters, this chapter starts with an analysis of the distributed optimization of global MIP problems. Unlike the distributed optimization of convex problems, e.g. linear programs (LP), means to guarantee the convergence of distributed MIP problem towards the optimum are still under study, see [162, 161, 60]. The considered investigation starts from extending recent results on distributed optimization of mixed-integer linear programs (MILP). Here some conservative assumptions required in state-of-the-art approaches are relaxed, and thus enhancing the computational efficiency. Then, a distributed optimization strategy for mixed-integer quadratic programs (MIQP) is introduced, which can handle more general types of problems in the context of optimal control. Various numeric tests are conducted to confirm the efficiency of the developed strategies compared to the centralized one. At the end of this chapter, a trajectory planning problem for autonomous vehicles passing through a narrow passage is considered. The simulation results show that all vehicles can safely travel through the passage (with neglectable performance loss compared to the centralized solution) by employing the proposed distributed strategy.

Finally, it is emphasized that communication problems such as the directed or undirected communication graphs, time-varying graphs, packet losses in transmission channels, or communication delays are not in the focus of this chapter (but belong to part of the future research topics detailed in the conclusion chapter). Instead, the focus is limited to the design of the communication order for CPS, such that the convergence towards the global optimum/sub-optimum can be guaranteed.

Chapter 5 concludes this thesis by listing all main contributions, together with an outlook on future research directions in modeling and control of CPS.

Part II.

Modeling and Control of Single Hybrid Systems

2. Optimization Based Control of Hybrid Systems

This chapter considers an optimal control problem for a single subsystem of CPS. Problems of this kind arise if the interaction with other subsystems can be cast into constraints of the local subsystem, see Fig. 2.1. The design of local controllers therefore has to take into account these constraints together with local constraints when aiming at achieving the local control goal. For example, in autonomous driving (see the example in Fig. 1.2 in the last chapter) the influence of surrounding vehicles can be formulated as forbidden zones of the vehicle, and the local controller should plan its motion by avoiding these zones. However, unlike the optimal control problem for linear dynamics and convex constraints, the optimal control problem for single subsystems of CPS is more challenging with respect to the following aspects:

- Hybrid dynamics are suitable to model the evolution of combined continuous and discrete states (e.g. the continuous states in autonomous driving are the velocity and position of each local vehicle, while the discrete state is the decision whether to overtake the vehicle in front, or not): to decide which type of hybrid system should be employed so that the desired mixed-state evolution can be modeled sufficiently precise, however, is often difficult.
- After a suitable local hybrid dynamics has been identified, the formulation of

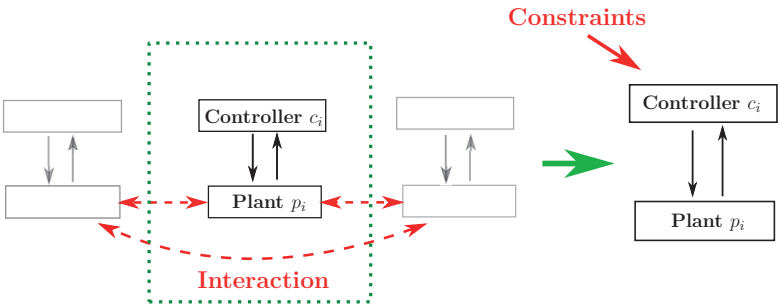

Figure 2.1.: Interaction from other subsystems to local subsystem i is cast into constraints for its local controller.

a related (optimal) control problem may also be challenging with respect to: 1.) the local control task, 2.) the local constraints such as the maximal acceleration of the local vehicle, and 3.) the additional constraints representing the interaction with the environment, such as the forbidden zones to be occupied by other vehicles.

- The obtained optimal control problem may contain both continuous and discrete variables, or logical relations describing the interleaving between continuous and discrete dynamics. The question is, thus, how to properly formulate such mixed dynamics and logical relations in the optimal control problem, so that the problem is accessible to existing solvers.

- Even if the optimal control problem is immediately accessible by a solver, its solution process may also be too time-consuming for online control. Thus, one has to ask if and how the complexity can be reduced, e.g. through a compact formulation of the optimization problem with fewer variables.

Given these challenges, this chapter starts from reviewing some commonly used hybrid systems (for the two cases of continuous-time and discrete-time formulation) in literature. Optimal control strategies for these hybrid systems are also reviewed, such as *Dynamic programming* and *Indirect/Direct methods* for the continuous-time case, and mixed-integer programs for the discrete-time case. Thereafter, departing from the often used PWA systems for modeling of hybrid dynamics, a type of hybrid automata that can cover quite general transition mechanisms of CPS is proposed (which is derived from the *unified hybrid systems* defined in [37]), including autonomous switching, controlled switching, conditional controlled switching, as well as impulsive changes of the continuous state.

Then, for the considered type of hybrid systems in discrete-time, a class of finite-time optimal control problems is obtained and two techniques to compute optimal control strategies are proposed in this chapter. In the case that the flow and reset functions in considered hybrid systems are linear (extensions to nonlinear case are discussed at the end of this chapter), both approaches are capable to translate the hybrid dynamics into a set of linear, mixed-integer constraints, leading to numerical programs which can then be processed by solvers like CPLEX [48]. In addition, exactness of the translation is also guaranteed (and thus the optimality of the solution) for both approaches, while the difference lies in whether the discrete state sequences need to be enumerated in advance. Advantages and disadvantages of the two approaches are then discussed in detail. Finally, a vehicle overtaking problem (similar to the one in Fig. 1.2) is considered, in which safe trajectories of a local autonomous vehicle are found by employing the methods proposed in this chapter. The content of this chapter is based in parts on results previously published in [95, 93, 97, 94].

2.1. Introduction of a General Class of Hybrid Systems

As mentioned in the last chapter, hybrid systems are often classified into switched systems, autonomously switching systems, or the combination hereof based on the transition semantics. While in switched systems the transition of the discrete state is forced by an external discrete input signal [167, 95, 171], the change of the discrete state in autonomously switching systems is only a consequence of the continuous state evolution [23, 89], see Fig. 2.2. A change of the discrete state in both switched and switching systems will lead to another differential or difference equation to be followed thereon by the continuous state. For switching systems, the condition to be satisfied for the continuous state to trigger a discrete state transition is often defined as a *transition guard* [89, 85], i.e. a subset of the continuous state space which must be reached by the continuous state to trigger the transition, see the red line in Fig. 2.2. In contrast, no guard exists in switched systems, as discrete state transitions can be enabled anywhere in the continuous space, provided that an external trigger signal is given. Controlled switching systems are a combination of the two former types, such that a discrete state change is only enabled when a subset part of the continuous state space is reached and a discrete input signal is given in the same time.

Besides the classification above, hybrid systems are further classified according to whether the continuous state changes impulsively at a change of discrete state,

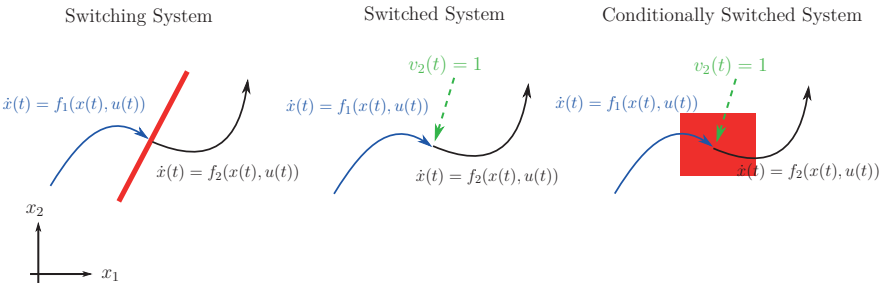

Figure 2.2.: Different transition mechanisms in continuous-time hybrid systems: In the left case, a change of the continuous dynamics is triggered by hitting a hyperplane (or a manifold) marked in red; in the middle, a change is triggered by an external discrete input $v(t) = 1$; in the right case, a transition can only be triggered if the continuous state is located in a set (representing a transition condition) and if the discrete input $v(t) = 1$ is also applied in the same time.

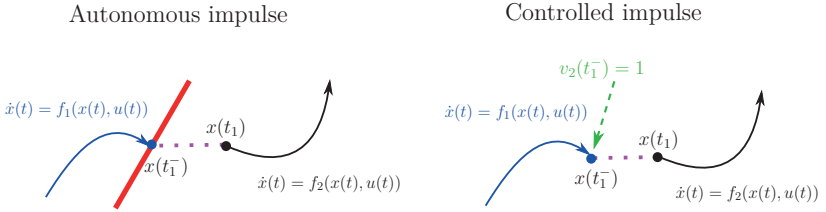

Figure 2.3.: Resets (or jumps) of the continuous state for both autonomous and controlled impulses are illustrated: when a change of a discrete state is triggered at time t_1^-, with $t_1^- = \lim_{t \to t_1, t<t_1} t$, an impulsive change (dashed line) of the continuous state from $x(t_1^-)$ to $x(t_1)$ is also triggered at the same time.

namely, the autonomous impulses for switching systems, and the controlled impulses for switched systems [37], see Fig. 2.3. Impulsive changes of the continuous state, sometimes also called *Resets* or *Jumps*, usually result from an abstraction of the true physical process, especially when the continuous state changes significantly in a negligibly small interval of time.

It should not be hard to notice that modeling by using hybrid systems may lead to a range of possible combinations of the transition mechanisms. Among the possible combinations, piecewise affine (PWA) systems are a relatively popular subclass of hybrid systems, and the optimal control problem for it has been widely considered, e.g. in [22, 20, 152, 17, 31, 33]. In these publications, the continuous state space is typically partitioned into several polyhedral subsets, see Fig. 2.4. Transition guards for PWA systems are defined implicitly in the sense that, whenever the continuous state hits or crosses a hyperplane constituting the boundary of a partition, the discrete state transition occurs as an immediate consequence of the evolution of the continuous state. In this sense, PWA systems also belong to a subclass of switching systems.

However, although PWA systems have attracted significant interest by studying modeling and control of CPS, the way how the discrete states changes in PWA systems is restrictive with respect to the different transition mechanisms introduced above. An example is a model suitable to design an automatic gear-shift for a vehicle, when shifting a gear is permitted in a certain range of engine speed instead of only at the prescribed values of engine speed. This is relevant if the choice of switching speed is still a degree of freedom of the control design.

To overcome these limitations of PWA systems, a general hybrid system HA, which is of the type of hybrid automata [76, 9], encompassing all three transition mechanisms in Fig. 2.2 and two types of impulses in Fig. 2.3 is considered in the sequel. The HA shares a similar structure to the *unified hybrid systems* defined

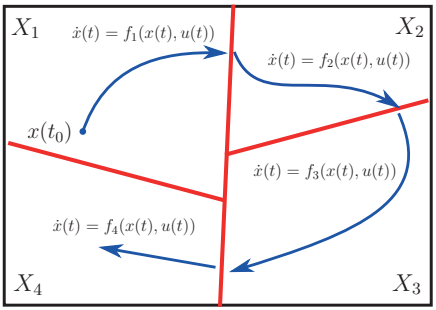

Figure 2.4.: For a PWA system with 4 discrete states obtained by partitioning the state space X into four subspaces X_1, X_2, X_3 and X_4. Starting from the initial continuous state $x(t_0)$ at time t_0, a change of the discrete state as well as the continuous dynamics is a direct consequence by crossing the boundaries of subspaces (marked in red).

in [37], and the optimal control of HA involves the decision on optimal continuous state and controls, discrete state and controls, discrete state sequences and transition times. Before the discussion of HA is started, a brief review of existing optimal control techniques for different classes of hybrid systems is presented for both continuous-time and discrete time cases.

Optimal Control of Continuous-Time Hybrid Systems

Unlike purely continuous systems, the continuous states of hybrid systems evolve according to different differential equations in different discrete states. The overall continuous control inputs are thus a set of continuous trajectories in an infinite dimensional functional space for each discrete state. Techniques to determine the optimal input trajectories have been derived for purely continuous systems, including *Dynamic programming*, *Indirect methods* and *Direct methods*. However, these techniques must be tailored to hybrid systems, in order to take into account the discrete dynamics at the same time.

When *Dynamic programming* is applied to the optimal control of hybrid systems, the key is to find the optimal control law in each discrete state according to the Hamilton-Jacobi-Bellman (HJB) equation, see [38, 37, 74]. The HJB equation is derived from the *Principle of Optimality*, i.e., each subtrajectory (for both continuous and discrete states) of an optimal trajectory must be an optimal trajectory as well. The main advantage of *dynamic programming* is that it provides an optimal feedback control law, thus the optimal control action can be directly computed according to the measured state when applied online. For the *unified hybrid systems*

considered in [37], the optimal control strategy proposed there is also based on *dynamic programming*. However, similar to the case of purely continuous dynamics, a value function satisfying the HJB partial differential equation must be determined in *dynamic programming*. But the computation of the value function is well known to suffer from the „curse of dimensionality" (besides some special cases, such as the linear quadratic optimal control problem, in which the value function can be derived analytically). To determine an approximated value function, a common approach is to sample the continuous state and control spaces, while this limited the application of *dynamic programming* only to problems with small state and input dimension.

Indirect methods start from formulating the necessary optimality conditions of the optimal control problem, see e.g. [154, 130, 41, 129]. This idea is in part similar to *dynamic programming*, but no feedback control law is derived here. Instead, optimal conditions for the controlled states and co-states are formulated, leading to a boundary value problem (BVP) of them. By solving the BVP, the optimal state and co-state trajectory can be determined, and thus the optimal control actions. Note that the formulation of the optimal conditions requires the use of *Pontryagin's Maximum Principle*, sometimes also called the *Minimal principle* (depending on whether the optimization task is to maximize or minimize). The BVP is solved numerically, e.g., by the *Multiple shooting method*, in which the time interval of the BVP is discretized and the Newton method is used to match the boundary conditions. In general, *indirect methods* are able to provide a solution with high accuracy, but suffer a small domain of convergence (as the BVP is possibly described by a set of non-smooth differential equations) and a difficult initialization (a good initialization of the co-state is crucial for the solution quality) [128]. The necessary optimality conditions in literature are often tailored to specific type of hybrid systems, e.g., in [154] a switching system is considered, and in [130, 129] a partitioned state space is required for the hybrid system. For the *unified hybrid systems* in [37], the necessary optimality conditions for a given discrete state sequence are formulated in [126].

Direct methods start from parameterizing the infinite dimensional controls of the original problem, leading to an approximating nonlinear program with finitely many optimization variables, see [158, 160]. A standard step in *direct methods* is to first discretize the whole problem with respect to time, and then optimize the discretized state and control trajectories in a nonlinear program (that is why *direct methods* are often characterized as „first discretize, then optimize", while *indirect methods* are characterized as „first optimize, then discretize"). Since the discretization is made before the optimization, *direct methods* thus share many similarities with methods developed for discrete-time optimal control problems. In addition, as a large variety of optimization methods for nonlinear programs have been well studied in the last decades, see a summary in [26], these optimization methods can eventually be employed to solve the nonlinear programs in *direct methods*, and thus enhances the computational efficiency. By applying *direct methods* to hybrid systems (with nonlinear continuous dynamics), as the discrete states are usually modeled by integer

variables and the transition mechanisms are encoded by mixed-integer constraints, this approach will lead to mixed-integer nonlinear program (MINLP). More details of MINLP will be reviewed in the following section.

Optimal Control of Discrete-Time Hybrid Systems

For discrete-time hybrid systems, a sampling over time leads to slightly different transition mechanisms than in the continuous-time case, see Fig. 2.5 for more details. For optimal control in the case of discrete-time, the method of *dynamic programming* can be further applied (see [95, 94] for example), but the „curse of dimensionality" still exists, and the method is thus only suitable for hybrid systems with low-dimensional state and input space. Most existing work cast the optimal control problem directly into MINLP tasks, and determine the optimal state and control trajectories thereafter. This approach mostly uses a set of mixed-integer algebraic constraints to describe the discrete-valued dynamics and the transition mechanisms. The mixed logical dynamic (MLD) systems tailored to PWA systems (see [20, 19, 31]) also belong to this approach. In general, by collecting all states, controls, as well as additional integer variables into a mixed-integer vector $w \in \mathbb{R}^{n_r} \times \mathbb{Z}^{n_z}$, the following MINLP is obtained:

$$\begin{aligned} &\min_{w} J(w) \\ &s.t.:\ G(w) \leq 0,\ w \in \mathbb{R}^{n_r} \times \mathbb{Z}^{n_z}. \end{aligned} \tag{2.1}$$

Note that here the optimization variable w contains n_r continuous entries and n_z integer components, and the equality constraints are re-written into inequality form. Techniques based on branch-and-bound [88, 112] or branch-and-cut algorithms [111, 149] are standard approaches to solve MINLP problem (2.1). The

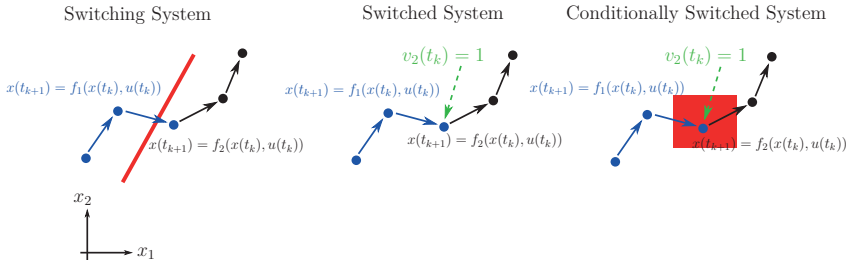

Figure 2.5.: Different transition mechanisms in discrete-time hybrid systems: Compared with the case of continuous-time in Fig. 2.2, difference equations are used to describe the continuous dynamics, and the transition conditions for switching systems are changed to crossing a hyperplane (marked in red) instead of exactly hitting it.

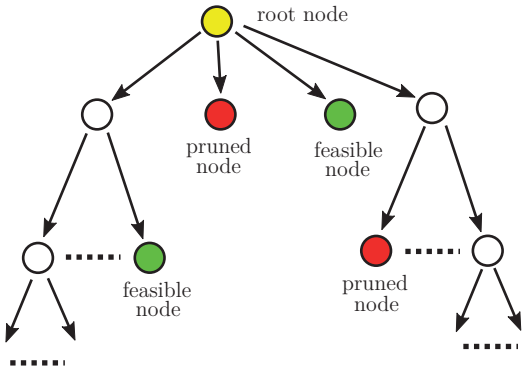

Figure 2.6.: The branch-and-bound search tree starts from a root node $\hat{w}$ (marked in yellow) satisfying $G(\hat{w}) \leq 0, \hat{w} \in \mathbb{R}^{n_r} \times \mathbb{Z}^{n_z}$. The cost $J(\hat{w})$ is then adopted as the incumbent best cost of the problem (2.1) and serves to prune the succeeding nodes in the searching. The branching of the „root node" is realized by branching over an integer variable in $\hat{w}$ and imposing additional constraints to the problem (2.1). A new node is determined by problem (2.1) with the additional constraints. For a new node, if it can lead to a feasible candidate for (2.1), the optimal cost (or an upper bound of the optimal cost) of the candidate is compared with the incumbent best cost, and replaces the latter if it attains a lower value. For the case that the new node cannot lead to a candidate feasible for (2.1), or the lower cost bound of the candidate is even higher than the incumbent best cost, it is pruned from further searching. The search terminates when no node in the tree can lead to a candidate better than the incumbent best one.

tree-search procedure applied in these algorithms relies on the iterative determination of suitable cost bound, based on which sub-trees may not include the optimal solution are eliminated from further exploration, see Fig. 2.6. The efficiency depends critically on the tightness of these bounds, usually obtained by relaxing the integrality constraints (leading to lower cost bounds) or from appropriate heuristics (leading to upper cost bounds).

Nevertheless, by applying some advanced heuristics and when the number of integer variables in w is small, the branch-and-bound method can indeed provide a suitable approximation of the optimal solution within short time. However, when a large number of discrete states is involved or a large time horizon is considered, the exponentially increasing complexity will quickly make branch-and-bound methods impractical. A reason may be that the heuristics are not efficient enough to cope

with a large number of possible value combinations, so that a large amount of nodes has to be explored (e.g., when a node leads only to infeasible candidates of (2.1), or attains a performance worse than the best cost found so far). In this thesis, as the class of discrete-time hybrid systems is considered, most of the optimal control problems are eventually formulated into the form of (2.1) with linear or quadratic cost function $J(w)$, and linear constraints $G(w) \leq 0$. These problems are then solved by using the solver CPLEX, which uses a similar search procedure as in Fig. 2.6, but with many advanced „add-ons" to enhance the performance. This includes questions of smart selection of nodes to be further branched, of backtracking to previous nodes in case of infeasible nodes, the choice of heuristics, etc. These „add-ons", however, can not change the fact that the search procedure in Fig. 2.6 in general takes more and more time if more integer variables are considered. In this case, adopting a more compact formulation of the constraints $G(w) \leq 0$ to reduce the search space (in this chapter), or applying efficient distributed strategies to cast (2.1) into a set of small scale problems being solved in parallel (in Chapter 4), turn out to be better choices to enhance the performance.

In this chapter, the focus is limited to a class of discrete-time hybrid systems HA following the structure of hybrid automata [76, 9]. It will be shown that HA is capable to encompass all aforementioned transition mechanisms and impulsive changes. Then, its optimal control problem over a finite time horizon is considered, as well as the mixed-integer program obtained from the control problem. To solve the mixed-integer program efficiently, especially for a large number of integer variables, a set of tailored constraints involving only integer variables are defined in the program. With the help of these constraints, the number of possible combinations of necessary integer variables can be significantly reduced – thus, only a „slim" tree needs to be explored in branch-and-bound methods and thus enhances the computational efficiency. Finally, although the continuous dynamics in the considered type of HA is limited to the linear case, the idea of introducing pure-integer constraints to enhance the computational efficiency can be well transferred to the case of nonlinear continuous dynamics (see the discussion at the end of this chapter).

A General Class of Discrete-Time Hybrid Systems

The considered general class of discrete-time hybrid systems follows the definition in [93, 97], and is defined by $HA = (T, U, X, Z, I, \mathcal{T}, G, V, r, f)$ with:

- the discrete time-domain $T = \{t_k \mid k \in \mathbb{N} \cup \{0\}, \Delta \in \mathbb{R}^{>0} : t_k := k \cdot \Delta\}$, where k is used in the following to refer to t_k;
- the continuous input space $U \subseteq \mathbb{R}^{n_u}$ with the continuous input signal $u_k \in U$;
- the continuous state space $X \subseteq \mathbb{R}^{n_x}$, on which the continuous state vector x_k is defined;

- the finite set of discrete states $Z = \{z_{(1)}, \cdots, z_{(n_z)}\}$, from which a discrete state variable z_k in time k is selected;
- a set $I = \{I_{(1)}, \ldots, I_{(n_z)}\}$ of invariants, where the invariant of any discrete state $z_{(i)}$ is a polytope $I_{(i)} = \{x \mid n_{I_i} \in \mathbb{N}, C_{(i)} \in \mathbb{R}^{n_{I_i} \times n_x}, d_{(i)} \in \mathbb{R}^{n_{I_i}} : C_{(i)} \cdot x \leq d_{(i)}\}$, $I_{(i)} \subseteq X$;
- the finite set of transitions $\mathcal{T} \subseteq Z \times Z$, in which a transition from $z_{(i)} \in Z$ to $z_{(j)} \in Z$ is denoted by $\tau_{(i,j)} \in \mathcal{T}$;
- the set G of guard sets containing one polytopic set $G_{(i,j)} = \{x \mid C_{(i,j)} \in \mathbb{R}^{n_{G_{(i,j)}} \times n_x}, d_{(i,j)} \in \mathbb{R}^{n_{G_{(i,j)}}}, x \in I_{(i)} : C_{(i,j)} \cdot x \leq d_{(i,j)}\}$ for any transition $\tau_{(i,j)} \in \mathcal{T}$;
- the finite set $V = \{v_{(i,j)} \,|\, v_{(i,j)} \in \{0,1\}, \forall \tau_{(i,j)} \in \mathcal{T}\}$ of discrete input variables, where any element $v_{(i,j),k}$ in V refers to one transition $\tau_{(i,j)} \in \mathcal{T}$ triggered at time k; the variable $v_{(i,j),k}$ is a binary one, encoding that for $v_{(i,j),k} = 1$ the transition $\tau_{(i,j)}$ is triggered if $x_k \in G_{(i,j)}$ applies; while for $v_{(i,j),k} = 0$, the transition cannot occur; in addition, let $v_k \in \{0,1\}^{|V| \times 1}$ represent a vector containing the binary values of all variables $v_{(i,j),k} \in V$, and at any time k at most one entry of $v_k = 1$ is allowed;
- a reset function r: $\mathcal{T} \times X \to X$ which updates the continuous state x_k upon a transition $\tau_{(i,j)} \in \mathcal{T}$ according to $\hat{x}_k = E_{(i,j)} \cdot x_k + e_{(i,j)}$, $E_{(i,j)} : \tau_{(i,j)} \to \mathbb{R}^{n_x \times n_x}$, $e_{(i,j)} : \tau_{(i,j)} \to \mathbb{R}^{n_x \times 1}$. This function models the impulsive change of the continuous state mentioned earlier;
- a flow function $f : X \times U \times Z \to X$ defines the discrete-time continuous dynamics according to $x_{k+1} = A_{(i)} \cdot x_k + B_{(i)} \cdot u_k$ with $z_{(i),k} \in Z$, $x_k \in I_{(i)}$.

The execution of HA over time is defined as follows:

Definition 2.1. ***(Admissible Execution)***
For HA, let a finite time set $T_N = \{0, 1, \ldots, N\}$ and initial states (x_0, z_0) satisfying $z_0 := z_{(s)} \in Z$, $x_0 \in I_{(s)}$ be given. For given input sequences $\phi_u = \{u_0, u_1, \ldots, u_{N-1}\}$ and $\phi_v = \{v_0, v_1, \ldots, v_{N-1}\}$, the pair of state sequences $\phi_x = \{x_0, x_1, \cdots, x_N\}$ and $\phi_z = \{z_0, z_1, \cdots, z_N\}$ is admissible*, if and only if for any $k \in \{0, \ldots, N\}$ the pair (x_{k+1}, z_{k+1}) follows from (x_k, z_k), $x_k \in I_{(i)}$, $z_k := z_{(i)}$ according to the following semantics:*

1.) $x' := A_{(i)} \cdot x_k + B_{(i)} \cdot u_k$,

2.) if $G_{(i,j)} \in G$ exists so that $x' \in G_{(i,j)}$ and if $v_{(i,j),k} = 1$ applies, then $x_{k+1} := E_{(i,j)} \cdot x' + e_{(i,j)} \in I_{(j)}$ and $z_{k+1} := z_{(j)}$; otherwise, $x_{k+1} := x' \in I_{(i)}$, and $z_{k+1} := z_{(i)}$ is assigned.

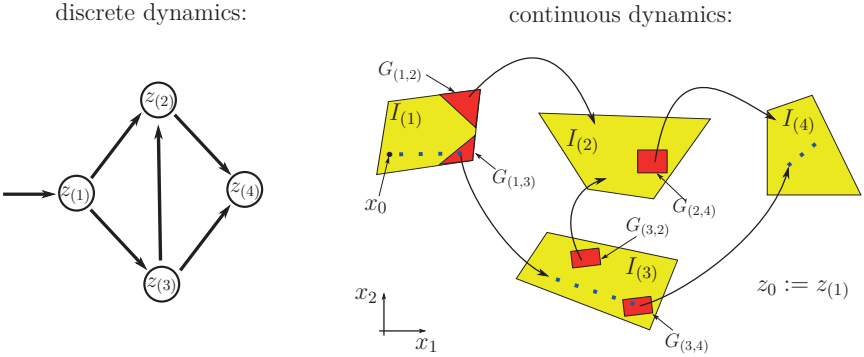

Figure 2.7.: Exemplary HA: the figure on the left demonstrates the transition map of discrete dynamics; the figure on the right depicts the evolution of continuous dynamics, in which the invariant sets are marked in yellow and the guard sets are marked in red, and the blue dashed lines denote a trajectory of continuous states and the black solid lines represent the *resets* when the discrete state changes.

An exemplary HA is demonstrated in Fig. 2.7. In the light of Definition 2.1, one can notice that the three transition mechanisms as well as the two types of impulsive changes introduced in the last section are all encompassed in HA, namely:

- Switching systems: in switching systems, a transition $\tau_{(i,j)}$ is a direct consequence if the continuous state satisfies $x_k \in G_{(i,j)}$ in time k. Thus, by fixing the discrete input $v_{(i,j),k} = 1$ in any time k in HA, a pure switching system is obtained according to the semantics in Definition 2.1. In addition, as a reset function r is also included in HA, an autonomous impulse can thus also be modeled by HA. To represent the subclass PWA systems, one only has to set $G_{(i,j)} := X$, $v_{(i,j),k} = 1$, for all $G_{(i,j)} \in G$ in any time k, as well as choosing an identity function for r for any transition in HA;

- Switched systems: for switched systems, a transition $\tau_{(i,j)}$ can take place in any time k, provided that the continuous state x_k is located in the invariant set $I_{(i)}$ and a discrete input signal $v_{(i,j),k} = 1$ is given. Thus, by setting the guard set $G_{(i,j)}$ equal to the invariant set $I_{(i)}$ for any guard and any discrete state, a switched system is obtained. Similarly to the autonomous impulse, a controlled impulse can thus also be modeled by HA;

- Conditionally switched systems: as both switching and switched systems can be represented by HA, it is obvious that HA can also model this type of hybrid system.

Compared with the commonly used PWA systems, the considered HA clearly provides a higher degree of freedom in modeling the interaction between continuous and discrete dynamics. HA thus appears to be a better option than PWA systems facing the challenges in CPS modeling. In addition, in terms of the interaction structure defined in Fig. 2.1, HA also allows one to model the influence from other subsystems to a change of the local invariants or guards (e.g. their shape or size, and this property is also demonstrated in the case study at the end of this chapter). Moreover, it will be shown in a later chapter, that different class of uncertainties occurring in CPS can also be modeled by using HA.

2.2. Optimal Control of Hybrid Systems with Given Phase Sequences

In this section, the optimal control problem of HA over a finite time horizon is considered. For given hybrid system HA with initial states (x_0, z_0) satisfying $z_0 := z_{(s)} \in Z$, $x_0 \in I_{(s)}$, assume now a set of hybrid goal states (X_g, z_g) which is defined by $z_g \in Z$ and a polyhedral set contained in the invariant of z_g: $X_g = \{x \mid n_{p_g} \in \mathbb{N}, C_g \in \mathbb{R}^{n_{X_g} \times n_x}, d_g \in \mathbb{R}^{n_{X_g}}, x \in I_g : C_g \cdot x \leq d_g\}$. Furthermore, let a state $x_g \in X_g$ be specified (e.g. the volumetric center of X_g) to later define a distance to the goal region in a computationally easy way.

If (x_0, z_0), $z_0 := z_{(s)}$, (X_g, z_g), and T_N are specified, the control objective is to find admissible state sequences ϕ_x and ϕ_z, or corresponding input sequences ϕ_u and ϕ_v respectively, which minimize an appropriate cost functional. Hereto, the following objective function is defined:

$$\mathcal{J}(x_0) = \sum_{k=1}^{N} \{(x_k - x_g)^{\mathrm{T}} Q (x_k - x_g) + u_{k-1}^{\mathrm{T}} R u_{k-1}\} + q_g \cdot N_g \tag{2.2}$$

where Q and R are semi-positive-definite weighting matrices, and $q_g \in \mathbb{R}^{\geq 0}$. The variable $N_g := \min\{k \in \{1, \ldots, N\} \mid x_k \in X_g, z_k = z_g\}$ encodes the first point of time in which the continuous state has reached the goal set. Thus, by only preserving the last term in $\mathcal{J}(x_0)$, one models a time-optimal control goal, so that the goal region X_g should be reached as fast as possible. It must be emphasized that the *switching costs* often considered in literature can also be included into $\mathcal{J}(x_0)$, as well as some other types of control objectives, e.g., see [93]. Now by assuming that the system can be held in the goal set, the overall control problem can be defined as:

Problem 2.1. *For HA initialized to (x_0, z_0), $z_0 := z_{(s)}$, let a time set T_N and a goal (X_g, z_g) be given. Then, determine input sequences ϕ_u^* and ϕ_v^* as the solution of:*

$$\min_{\phi_u, \phi_v} \mathcal{J}(x_0) \tag{2.3}$$

$$
\begin{aligned}
&\textit{s.t.: } \phi_u \textit{ with } u_k \in U, k \in \{0, \ldots, N-1\},\\
&\quad \phi_v \textit{ with } v_{(i,j),k} \in \{0,1\}, k \in \{0, \ldots, N-1\},\\
&\quad \phi_x, \phi_z \textit{ admissible for } HA,\\
&\quad x_N \in X_g, z_N = z_g.
\end{aligned}
$$

Note that the direct solution of this problem is difficult because of the following reasons:

1. The last but one constraint in Problem 2.1 requires the solution to satisfy the semantics in Def. 2.1, and thus the logical conditions of the transition mechanisms have to be converted into a form that is accessible to existing solvers for the optimization problem;

2. With increasing value of the horizon N, the number of possible combinations of ϕ_z and ϕ_v increases exponentially;

3. It is not known a-priori whether a feasible solution (or an admissible execution) exists at all for the selected horizon N, and the optimizer may have to search over many iterations until a first feasible solution (and thus a first finite upper bound on the costs) is obtained, whereas the latter is crucial for reducing the search space of the optimization algorithm.

Given these difficulties, a new approach is proposed which translates Problem 2.1 into a substitute formulation that considers the points mentioned above. The principle is to introduce as few binary variables as necessary to convert the logical conditions in Def. 2.1 exactly into a set of linear constraints for the binary variables (and real variables to encode the continuous part of the dynamics). By this, the combinatorics of ϕ_z and ϕ_v is transformed into a combinatorics of the binary variables. In addition, through a set of linear constraints formulated purely for the binary variables, the relevant combinations for determining the optimal solution can be reduced significantly. All the constraints in Problem 2.1 are equivalently reformulated into a set of linear constraints and can be solved by using existing MIQP solvers, such as CPLEX. The reformulation does not involve any approximation, and the reformulated problem has the same optimal solution as Problem 2.1.

2.2.1. Representation of Admissible Trajectories by Algebraic Programs

This section introduces a particular format to encode Problem 2.1 as algebraic program with binary variables. It is well-known that implications like $(x_k \in I_{(i)}) \Leftrightarrow (b = 1)$ for mapping the invariant set containment of x_k into a binary variable b can be accomplished by rules as those explained in [165] (often referred to as the *Big-M-approach*). Such mechanisms have been re-used in different work on hybrid system optimization, e.g. in [22, 150, 152], but the particular challenge is to use a number

of binary variables and constraints as small as possible on these variables for low computational times. This issue is addressed in the following for Problem 2.1. To facilitate the description and understanding of the procedure, a simplified case is first referred to, where a *phase sequence* is known: let the order of the discrete states Z by which HA passes through be known, but the times in T_N at which the discrete states are left or are reached still have to be determined. Hence, the remaining task is to determine the transition times as well as ϕ_u and ϕ_v such that ϕ_x is led (if possible) through the appropriate series of invariants and guards. Formally, a phase sequence is denoted by $\phi_p = \{p_0, \ldots, p_L\}$, where p_l with $l \in \{0, \ldots, L\}$ is set to the index of the discrete state, which is invariant in the l-th phase (i.e., $\phi_p \subset \phi_z$ is obtained from eliminating consecutive equal elements in ϕ_z).

The phases are now important to identify the number of binary variables required to encode the execution of HA within the optimization problem: consider a phase p_i, as shown in Fig. 2.8, from a hybrid state (x_k, z_k) with $z_k = z_{(i)}$ (reached by a preceding transition) up to the state (x_{k+5}, z_{k+5}) with $z_{k+5} = z_{(j)}$, reached through the transition $\tau_{(i,j)}$. Note that $x' \in G_{(i,j)}$ is an intermediate state, which is immediately transferred into $x_{k+5} := r(\tau_{(i,j)}, x') \in I_{(j)}$ by the transition with reset upon $v_{(i,j),k+4} = 1$, according to the definition of an admissible run above. Two points are obvious from this figure:

1. for any of the states $\{x_k, \ldots, x_{k+4}, x'\}$ the same invariant constraint (element of $I_{(i)}$) applies, i.e. one binary variable per phase is sufficient to express this fact;

2. the intermediate state x' must be associated with an additional binary variable to encode $x' \in G_{(i,j)}$ for p_i.

Since x' must be treated separately, an extended index set for the states is considered here, namely, $\tilde{k} \in T_{\tilde{N}} = \{0, \cdots, N + L\}$, see Fig. 2.9 for a better understanding of such extension mechanisms.

Within this set, the following assignments are provided corresponding to an admissible run of HA:

- the new initial time step $\tilde{k} = 0$ still refers to x_0;

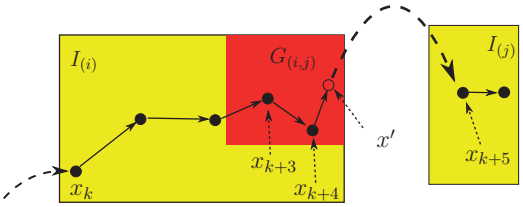

Figure 2.8.: Execution of HA within one phase.

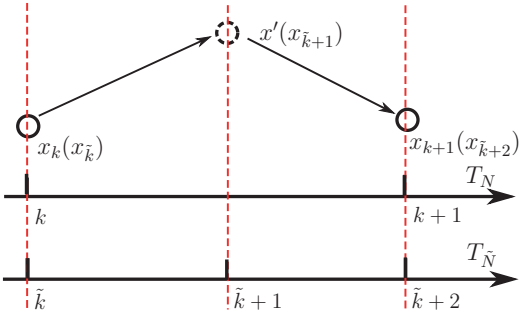

Figure 2.9.: Extension of the time index set T_N to account for the intermediate states x' occurring for transitions.

- L steps from set T_N are additionally assigned to the intermediate states x' (and thus an exit from a discrete state);
- $N+1$ steps (including the initial step) from set T_N belong to the evolution of continuous states in each discrete state;
- the last step in set T_N encodes the entry into X_g.

Next, the constraints on the continuous states $x_{\tilde{k}}$ have to be formulated suitably. Recall that all invariants, guard sets, and X_g are given as polytopic sets. Exemplarily for an invariant set $I_{(i)}$, the efficient algebraic encoding is explained: using the principles proposed in [165], the constraint $C_{(i)} \cdot x_{\tilde{k}} \leq d_{(i)}$ can be modeled equivalently by:

$$C_{(i)} \cdot x_{\tilde{k}} \leq d_{(i)} + b_{(i),\tilde{k}} \cdot M_{(i)} \tag{2.4}$$

if $M_{(i)} \in \mathbb{R}^{n_{p_i} \times 1}$ is a vector of large constants, and $b_{(i),\tilde{k}} \in \{0,1\}$ one binary variable. If $b_{(i),\tilde{k}} = 0$, the invariant constraint is enforced, while $b_{(i),\tilde{k}} = 1$ relaxes the constraint. Likewise, a guard constraint $x_{\tilde{k}} \in G_{(i,j)}$ results in:

$$C_{(i,j)} \cdot x_{\tilde{k}} \leq d_{(i,j)} + b_{(i,j),\tilde{k}} \cdot M_{(i,j)}. \tag{2.5}$$

Consider that two binary variables are required per phase (one for the invariant conditions, and one for the guard condition (or the terminal set, respectively)), a vector of $2(L+1)$ binary variables is introduced, namely:

$$\mathbf{b}_{\tilde{k}} = [b_{(0),\tilde{k}},\, b_{(0,1),\tilde{k}},\, b_{(1),\tilde{k}},\, \ldots,\, b_{(L),\tilde{k}},\, b_{g,\tilde{k}}]^{\mathrm{T}} \tag{2.6}$$

for each $\tilde{k} \in \{0, \cdots, N+L\}$. The last entry represents containment in the goal set X_g. For $\tilde{k} = 0$, the numeric values of this vector are $\mathbf{b}_0 = [0, 1, \ldots, 1]^{\mathrm{T}}$, and for the

transition from phase p_i to p_{i+1}, there exists: (a) $\mathbf{b}_{\tilde{k}} = [1,\dots,1,\underbrace{0}_{2i+1},\underbrace{0}_{2i+2},1,\dots,1]^{\mathrm{T}}$ corresponding to the intermediate state x', and (b) $\mathbf{b}_{\tilde{k}} = [1,\dots,1,\underbrace{0}_{2i+3},1,\dots,1]^{\mathrm{T}}$ for the entry in the next invariant. For $\tilde{k} = N+L$, the vector is: $\mathbf{b}_{N+L} = [1,\dots,1,0,0]^{\mathrm{T}}$, and all of these vectors are collected in a matrix:

$$\mathcal{B}_m = [\mathbf{b}_0, \mathbf{b}_1, \dots \mathbf{b}_{N+L}] = \tag{2.7}$$

$$\left[\begin{array}{ccccccccccc}
\begin{bmatrix}0\\1\\1\\\vdots\\1\\1\\1\end{bmatrix} & \cdots & \underbrace{\begin{bmatrix}0\\1\\1\\\vdots\\1\\1\\1\end{bmatrix}}_{\tilde{k}_0^{out}-1} & \underbrace{\begin{bmatrix}0\\0\\1\\\vdots\\1\\1\\1\end{bmatrix}}_{\tilde{k}_0^{out}} & \underbrace{\begin{bmatrix}1\\1\\0\\1\\\vdots\\1\\1\end{bmatrix}}_{\tilde{k}_1^{in}} & \cdots & \underbrace{\begin{bmatrix}1\\\vdots\\1\\0\\0\\1\\1\end{bmatrix}}_{\tilde{k}_{L-1}^{out}} & \underbrace{\begin{bmatrix}1\\\vdots\\1\\1\\1\\0\\1\end{bmatrix}}_{\tilde{k}_L^{in}} & \cdots & \begin{bmatrix}1\\\vdots\\1\\1\\1\\0\\0\end{bmatrix} & \begin{bmatrix}1\\1\\1\\\vdots\\1\\0\\0\end{bmatrix}
\end{array}\right]$$

The last line refers to the time indexing, where $\tilde{k} = \tilde{k}_0^{out}$ encodes the instance in which the first invariant is left, and $\tilde{k} = \tilde{k}_1^{in}$ encodes the instance in which the second invariant of ϕ_z is reached. The following holds by construction:

Proposition 2.1. *If ϕ_x and ϕ_z determine an admissible run of HA with $z_N = z_g$ and $x_N \in X_g$, then a matrix $\mathcal{B}_m \in \{0,1\}^{(2L+2)\cdot(N+L+1)}$ exists according to the rules* (2.4) *to* (2.7)*, and each column in $\mathcal{B}_m$ uniquely determines which constraints apply to $x_{\tilde{k}}$ for $\tilde{k} \in \{0,\cdots,N+L\}$.* □

This proposition is a direct consequence of using the *Big-M-approach* to reformulate the invariant, guard and terminal constraints in (2.4) – (2.7). Then, by collecting all large constants $M_{(i)}$ and $M_{(i,j)}$ into a vector $\mathcal{M}$ with suitable dimension, and letting all constraints of the form (2.4) and (2.5) be collected in the order of the indexing of $x_{\tilde{k}}$ in:

$$\mathcal{C} \cdot x_{\tilde{k}} \leq \mathcal{D} + diag(\mathcal{B}_m(:,\tilde{k}+1)) \cdot \mathcal{M}, \tag{2.8}$$

the search for an admissible run ϕ_x and ϕ_z thus means to satisfy with (2.8) for all $\tilde{k} \in \{0,\dots,N+L\}$. The number of $(2L+2)\cdot(N+L)$ binary variables (with $b_{0,\tilde{k}}$ being known) in $\mathcal{B}_m$ encode in principle $2^{(2L+2)\cdot(N+L)}$ combinations, which is prohibitively large for an efficient solution of Problem 2.1 with larger N and L values. Nevertheless, the particular structure in (2.7) that $\mathcal{B}_m$ has to satisfy for any admissible execution reduces the number of possible combinations significantly, i.e., from $2^{(2L+2)\cdot(N+L)}$ to $\binom{N+L}{2L}$, and for the case when $N = 10$, $L = 3$, the combinations are reduced from 2^{104} to 1716. The following section proposes a scheme to efficiently exploit this structure in searching for an optimal ϕ_x and ϕ_z.

2.2.2. Formulation of the Optimization Problem

In order to explain how $\mathcal{B}_m$ enables to search only over those value combinations of binary variables that represent admissible runs of HA, the first two rows of the matrix $\mathcal{B}_m$ are focused on first. They represent the values of the binary variables $b_{(0),\tilde{k}}$, $b_{(0,1),\tilde{k}}$ over $\tilde{k} \in \{0, \cdots, N+L\}$, and these variables model that $x_{\tilde{k}}$ is contained in the invariant of the first discrete state (value 0) and, respectively, that the first transition is triggered (again value 0):

$$\begin{bmatrix} \mathcal{B}_m(1,:) \\ \mathcal{B}_m(2,:) \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 & 0 & 1 & 1 & \cdots & 1 \end{bmatrix}. \tag{2.9}$$

Note that the column in which $\mathcal{B}_m(1,:)$ changes from 0 to 1 is not yet determined, as the transition time is still not clear. Nevertheless, the value of $\mathcal{B}_m(1, \tilde{k}+1)$ in $\mathcal{B}_m(1,:)$ can be noticed to depend on an auxiliary vector $\mathbf{d}_{1,\tilde{k}+1}^{\mathrm{T}} = [\mathcal{B}_m(1,\tilde{k}), \mathcal{B}_m(2,\tilde{k})]$ according to:

$$\mathcal{B}_m(1,\tilde{k}+1) = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} \text{ if } \mathbf{d}_{1,\tilde{k}+1}^{\mathrm{T}} = \begin{Bmatrix} [0,1] \\ [0,0] \text{ or } [1,1] \end{Bmatrix}. \tag{2.10}$$

Now, define two arbitrary parameter vectors $\alpha_1 \in \mathbb{R}^{3\times 1}$ and $\beta_1 \in \mathbb{R}^{3\times 1}$ satisfying the following conditions:

$$\begin{aligned} \begin{bmatrix} -\infty \\ 0 \\ 0 \end{bmatrix} < \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \cdot \alpha_1 < \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \\ \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} < \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \cdot \beta_1 < \begin{bmatrix} 1 \\ \infty \\ \infty \end{bmatrix}, \end{aligned} \tag{2.11}$$

where the first two columns of the matrices in front of the vectors α_1 and β_1 encode the possible values of $\mathbf{d}_{1,\tilde{k}+1}^{\mathrm{T}}$ in (2.10). Then the relation (2.10) can be algebraically and equivalently formulated as:

$$\begin{aligned} \mathcal{B}_m(1,\tilde{k}+1) &\geq \alpha_1^{\mathrm{T}}(1:2) \cdot \mathbf{d}_{1,\tilde{k}+1} + \alpha_1(3), \\ \mathcal{B}_m(1,\tilde{k}+1) &\leq \beta_1^{\mathrm{T}}(1:2) \cdot \mathbf{d}_{1,\tilde{k}+1} + \beta_1(3). \end{aligned} \tag{2.12}$$

While this encoding relates to the first phase, the principle can be transferred to the subsequent phases as well. For a phase with index $l \in \{1, \cdots, L-1\}$, the row with index $2l+1$ of $\mathcal{B}_m$ is relevant. It refers to the binary variable $b_{(l),\tilde{k}}$, and the value of $\mathcal{B}_m(2l+1, \tilde{k}+1)$ is written depending on an auxiliary vector $\mathbf{d}_{2l+1,\tilde{k}+1}^{\mathrm{T}} = [\mathcal{B}_m(2l,\tilde{k}), \mathcal{B}_m(2l+1,\tilde{k}), \mathcal{B}_m(2l+2,\tilde{k})]$:

$$\mathcal{B}_m(2l+1,\tilde{k}+1) = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} \qquad \text{if } \mathbf{d}_{2l+1,\tilde{k}+1}^{\mathrm{T}} = \begin{Bmatrix} [0,1,1] \text{ or } [1,0,1] \\ [1,1,1] \text{ or } [1,0,0] \end{Bmatrix}. \tag{2.13}$$

If parameter vectors $\alpha_l \in \mathbb{R}^{4\times 1}$ and $\beta_l \in \mathbb{R}^{4\times 1}$ are defined similarly to (2.11), the assignment (2.13) can be equivalently formulated as:

$$\begin{aligned}\mathcal{B}_m(2l+1,\tilde{k}+1) &\geq \alpha_l^{\mathrm{T}}(1:3)\cdot \mathbf{d}_{2l+1,\tilde{k}+1} + \alpha_l(4),\\ \mathcal{B}_m(2l+1,\tilde{k}+1) &\leq \beta_l^{\mathrm{T}}(1:3)\cdot \mathbf{d}_{2l+1,\tilde{k}+1} + \beta_l(4).\end{aligned} \tag{2.14}$$

With respect to the penultimate row of $\mathcal{B}_m$, which refers to $b_{L,\tilde{k}}$, the value of $\mathcal{B}_m(2L+1,\tilde{k}+1)$ depends likewise on an auxiliary vector $\mathbf{d}_{2L+1,\tilde{k}+1}^{\mathrm{T}} = [\mathcal{B}_m(2L,\tilde{k}), \mathcal{B}_m(2L+1,\tilde{k})]$ with:

$$\mathcal{B}_m(2L+1,\tilde{k}+1) = \begin{Bmatrix}0\\1\end{Bmatrix} \text{ if } \mathbf{d}_{2L+1,\tilde{k}+1}^{\mathrm{T}} = \left\{\begin{matrix}[0,1] \text{ or } [1,0]\\ [1,1]\end{matrix}\right\}. \tag{2.15}$$

Using parameter vectors $\alpha_g \in \mathbb{R}^{3\times 1}$ and $\beta_g \in \mathbb{R}^{3\times 1}$, (2.15) is translated into:

$$\begin{aligned}\mathcal{B}_m(2L+1,\tilde{k}+1) &\geq \alpha_g^{\mathrm{T}}(1:2)\cdot \mathbf{d}_{2L+1,\tilde{k}+1} + \alpha_g(3),\\ \mathcal{B}_m(2L+1,\tilde{k}+1) &\leq \beta_g^{\mathrm{T}}(1:2)\cdot \mathbf{d}_{2L+1,\tilde{k}+1} + \beta_g(3).\end{aligned} \tag{2.16}$$

For any $2l$-th row of $\mathcal{B}_m$ (with $l \in \{1,\cdots,L\}$), which refers to $b_{(l-1,l),\tilde{k}}$, only one entry equals 0 (indicating that the reset is only triggered once), and can be enforced by:

$$\sum_{\tilde{k}=0}^{N+L} \mathcal{B}_m(2l,\tilde{k}+1) = N+L, \quad \forall l \in \{1,\cdots,L\}. \tag{2.17}$$

Finally, for the last row, referring to $b_{g,\tilde{k}}$, only the last entry $\mathcal{B}_m(2L+2,N+L+1)$ is forced to 0, modeling $x_N \in X_g$. This is translated into:

$$\mathcal{B}_m(2L+2,N+L+1) = 0. \tag{2.18}$$

The condition that $x_{\tilde{k}} \in X_g$ if $x_{\hat{k}} \in X_g$ for $\tilde{k} \geq \hat{k}$ is modeled by:

$$\mathcal{B}_m(2L+2,\tilde{k}) \geq \mathcal{B}_m(2L+2,\tilde{k}+1). \tag{2.19}$$

Note that the options considered for $\mathbf{d}_{1,\tilde{k}+1}^{\mathrm{T}}$ in (2.10), for $\mathbf{d}_{2l+1,\tilde{k}+1}^{\mathrm{T}}$ in (2.13), and for $\mathbf{d}_{2L+1,\tilde{k}+1}^{\mathrm{T}}$ in (2.15) are sufficient to encode the part of $\mathcal{B}_m$ which corresponds to the change of phases. Using this fact, and the constructive rules provided above to determine the linear inequalities formulated for elements of $\mathcal{B}_m$, the following fact can be established:

Proposition 2.2. *If a binary matrix $\mathcal{B}_m \in \{0,1\}^{(2L+2)\cdot(N+L+1)}$ with first column $\mathcal{B}_m(:,1) = \mathbf{b}_0$ and last column $\mathcal{B}_m(:,N+L+1) = \mathbf{b}_{N+L}$ satisfies the constraints (2.12), (2.14), and (2.16) to (2.19), then it has the same structure as in (2.7).* □

Proof. Constraint (2.12) enforces the first row of $\mathcal{B}_m$ to satisfy the structure in (2.7); Constraint (2.14) enforces the $(2l+1)$-th row of $\mathcal{B}_m$, for all $l \in \{1, \cdots, L-1\}$, to satisfy the structure in (2.7), while constraint (2.16) is for the $2l$-th row of $\mathcal{B}_m$, for all $l \in \{1, \cdots, L\}$; Constraint (2.19) ensures the last row of $\mathcal{B}_m$ satisfy the desired structure. Finally, all $2L+2$ rows of $\mathcal{B}_m$ are ensured to follow the desired structure in (2.7) and thus finishes the proof. □

Theorem 2.1. *The task of finding an admissible trajectory of HA for given phase sequence ϕ_p is equivalent to finding a matrix $\mathcal{B}_m \in \{0,1\}^{(2L+2)\cdot(N+L+1)}$, with first column $\mathcal{B}_m(:,1) = \mathbf{b}_0$ and last column $\mathcal{B}_m(:,N+L+1) = \mathbf{b}_{N+L}$, satisfying constraints* (2.4)*,* (2.5)*,* (2.12)*,* (2.14)*, and* (2.16) *to* (2.19)*.* □

This theorem is a direct result of Proposition 2.1 and 2.2 and thus the proof is omitted. Now, all constraints introduced for the matrix $\mathcal{B}_m$ can be collected in the set of linear constraints:

$$\mathcal{Q} \cdot [\mathcal{B}_m^\top(:,1), \cdots, \mathcal{B}_m^\top(:,N+L+1)]^\top \leq \mathcal{W}, \tag{2.20}$$

where the matrices $\mathcal{Q}$ and $\mathcal{W}$ depend on the various parameter vectors α and β. The constraints in (2.20) also reduce the value combinations of the respective binary variables in $\mathcal{B}_m$ from $2^{(2L+2)\cdot(N+L)}$ to $\binom{N+L}{2L}$, as the obtained $\mathcal{B}_m$ must follow the structure in (2.7). The search for an admissible run ϕ_x and ϕ_z of HA now means to let $\mathcal{B}_m$ satisfy (2.8) and (2.20). Thus, by introducing auxiliary variables $\xi_{\tilde{k},i}$, $\pi_{\tilde{k},i}$ and $\xi_{\tilde{k},(i,i+1)}$ for all $\tilde{k} \in \{1, \ldots, N+L\}$ (details of such reformulations can be found in [152]), and additional parameters Θ_i^+, Θ_i^-, Θ_u^+, Θ_u^- determined by :

$$\begin{aligned}
\Theta_i^+ &= \left[\max_{x \in I_{(i)}} x_1 \quad \cdots \quad \max_{x \in I_{(i)}} x_{n_x}\right]^{\mathrm{T}}, \\
\Theta_i^- &= \left[\min_{x \in I_{(i)}} x_1 \quad \cdots \quad \min_{x \in I_{(i)}} x_{n_x}\right]^{\mathrm{T}}, \\
\Theta_u^+ &= \left[\max_{u \in U} u_1 \quad \cdots \quad \max_{u \in U} u_{n_u}\right]^{\mathrm{T}}, \\
\Theta_u^- &= \left[\min_{u \in U} u_1 \quad \cdots \quad \min_{u \in U} u_{n_u}\right]^{\mathrm{T}},
\end{aligned} \tag{2.21}$$

as well as parameter vectors $\lambda_x \in R^{n_x}$, $\lambda_u \in R^{n_u}$ to have for all $x \in X$ and $u \in U$:

$$\begin{aligned}
x + \lambda_x &\gg 0^{n_x \times 1}, \quad x - \lambda_x \ll 0^{n_x \times 1}, \\
u + \lambda_u &\gg 0^{n_u \times 1}, \quad u - \lambda_u \ll 0^{n_u \times 1},
\end{aligned} \tag{2.22}$$

the following transformed problem is obtained:

Problem 2.2. *For a given phase sequence ϕ_p, determine input sequences ϕ_u^* and a matrix $\mathcal{B}_m^*$ as solution to:*

$$\min_{\phi_u, \mathcal{B}_m} \sum_{\tilde{k}=0}^{N+L} \{(\hat{x}_{\tilde{k}+1} - x_g)^{\mathrm{T}} Q (\hat{x}_{\tilde{k}+1} - x_g) + u_{\tilde{k}}^{\mathrm{T}} R u_{\tilde{k}}\} + q_g \cdot \sum_{\tilde{k}=0}^{N+L} \mathcal{B}_m(2L+2, \tilde{k}+1) \tag{2.23a}$$

$$s.t.: \mathcal{Q} \cdot [\mathcal{B}_m^{\top}(:,1), \cdots, \mathcal{B}_m^{\top}(:, N+L+1)]^{\top} \leq \mathcal{W}; \tag{2.23b}$$

$$for\ \tilde{k} \in \{1, \ldots, N+L\}:$$

$$\hat{x}_{\tilde{k}} \leq x_{\tilde{k}} + \lambda_x \cdot (L - \sum_{i=1}^{L} \mathcal{B}_m(2i, \tilde{k}+1)), \tag{2.23c}$$

$$\hat{x}_{\tilde{k}} \geq x_{\tilde{k}} - \lambda_x \cdot (L - \sum_{i=1}^{L} \mathcal{B}_m(2i, \tilde{k}+1)), \tag{2.23d}$$

$$\hat{x}_{\tilde{k}} \leq \lambda_x \cdot (\sum_{i=1}^{L} \mathcal{B}_m(2i, \tilde{k}+1) + 1 - L), \tag{2.23e}$$

$$\hat{x}_{\tilde{k}} \geq -\lambda_x \cdot (\sum_{i=1}^{L} \mathcal{B}_m(2i, \tilde{k}+1) + 1 - L); \tag{2.23f}$$

$$x_{\tilde{k}} = \sum_{i=0}^{L} [A_{(i)} \cdot \xi_{\tilde{k},i} + B_{(i)} \cdot \pi_{\tilde{k},i}] + \sum_{i=0}^{L-1} \xi_{\tilde{k},(i,i+1)}; \tag{2.23g}$$

$$\mathcal{C} \cdot x_{\tilde{k}} \leq \mathcal{D} + diag(\mathcal{B}_m(:, \tilde{k}+1)) \cdot \mathcal{M}, \;\; u_{\tilde{k}-1} \in U; \tag{2.23h}$$

$$for\ i \in \{0, \cdots, L-1\}:$$

$$\xi_{\tilde{k},i} \leq \Theta_i^{+} \cdot (\mathcal{B}_m(2i+2, \tilde{k}) - \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23i}$$

$$\xi_{\tilde{k},i} \geq \Theta_i^{-} \cdot (\mathcal{B}_m(2i+2, \tilde{k}) - \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23j}$$

$$\xi_{\tilde{k},i} \leq x_{\tilde{k}-1} + \lambda_x \cdot (1 - \mathcal{B}_m(2i+2, \tilde{k}) + \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23k}$$

$$\xi_{\tilde{k},i} \geq x_{\tilde{k}-1} - \lambda_x \cdot (1 - \mathcal{B}_m(2i+2, \tilde{k}) + \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23l}$$

$$\pi_{\tilde{k},i} \leq \Theta_u^{+} \cdot (\mathcal{B}_m(2i+2, \tilde{k}) - \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23m}$$

$$\pi_{\tilde{k},i} \geq \Theta_u^{-} \cdot (\mathcal{B}_m(2i+2, \tilde{k}) - \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23n}$$

$$\pi_{\tilde{k},i} \leq u_{\tilde{k}-1} + \lambda_u \cdot (1 - \mathcal{B}_m(2i+2, \tilde{k}) + \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23o}$$

$$\pi_{\tilde{k},i} \geq u_{\tilde{k}-1} - \lambda_u \cdot (1 - \mathcal{B}_m(2i+2, \tilde{k}) + \mathcal{B}_m(2i+1, \tilde{k})), \tag{2.23p}$$

$$\xi_{\tilde{k},(i,i+1)} \leq \Theta_{i+1}^{+} \cdot (1 - \mathcal{B}_m(2i+2, \tilde{k})), \tag{2.23q}$$

$$\xi_{\tilde{k},(i,i+1)} \geq \Theta_{i+1}^{-} \cdot (1 - \mathcal{B}_m(2i+2, \tilde{k})), \tag{2.23r}$$

$$\xi_{\tilde{k},(i,i+1)} \leq E_{(i,i+1)} \cdot x_{\tilde{k}-1} + e_{(i,i+1)} + \lambda_x \cdot \mathcal{B}_m(2i+2, \tilde{k}), \tag{2.23s}$$

$$\xi_{\tilde{k},(i,i+1)} \geq E_{(i,i+1)} \cdot x_{\tilde{k}-1} + e_{(i,i+1)} - \lambda_x \cdot \mathcal{B}_m(2i+2, \tilde{k}). \tag{2.23t}$$

The cost function (2.23a) is an equivalent reformulation of the one in Problem 2.1, where the sum in the last term counts the total number of steps in which $x_{\tilde{k}}$ is not in X_g. The constraints (2.23c) to (2.23f) ensure that the costs induced by the

intermediate states x' are not recorded in the cost function. The conditions (2.23b) and (2.23h) force the resulting trajectory ϕ_x to comply to ϕ_p. The equations and inequalities (2.23i) to (2.23t) refer to reformulate the hybrid dynamics into linear ones by using auxiliary variables. Note that Problem 2.2 can be eventually reformulated into the form of (2.1) at the beginning of this chapter. In addition, as all constraints in Problem 2.2 are linear, together with a quadratic cost function, the optimization problem represents an MIQP problem and thus can be solved by existing solvers. After Problem 2.2 has been solved, the obtained $\mathcal{B}_m^*$ can straightforwardly determine the ϕ_z^* and ϕ_v^* according to the zero entries in $\mathcal{B}_m^*$. Furthermore, since (2.23b) admits all possible values of $\mathcal{B}_m$ corresponding to the structure in (2.7) and since no approximation is involved, the following applies:

Corollary 2.1. *If no feasible solution exists to Problem 2.2, then there exists no* admissible *trajectory corresponding to the given phase sequence ϕ_p.* □

Thus, Problem 2.2 can be used to verify the existence of an *admissible* trajectory satisfying Problem 2.1 for the considered ϕ_p.

Theorem 2.2. *If the solution of Problem 2 returns a feasible solution ϕ_u^* and $\mathcal{B}_m^*$, then it represents the optimal solution of Problem 2.1 for the given phase sequence ϕ_p.* □

This result follows from the relation between Problem 2.2 and 2.1 for the given ϕ_p as established by Proposition 2.1 and 2.2, and from the fact that the branch-and-bound algorithms for MIQP problems are capable to terminate with the optimal solution, if the search tree is fully explored [26]. If now Problem 2.1 is addressed without restriction to certain single ϕ_p, the solution is obtained by solving one instance of Problem 2.2 for any possible phase sequence connecting z_0 with z_g. If the number of possible phase sequences connecting the initial discrete state z_0 and the target state z_g is not very large, the search can be carried out by full enumeration.

2.2.3. Numeric Examples

To illustrate the procedure, the following example considers an HA with $x \in \mathbb{R}^3$ and 5 discrete states $Z = \{z_{(0)}, z_{(1)}, z_{(2)}, z_{(3)}, z_{(g)}\}$. The invariant sets of these states are marked by yellow regions and the guard sets by orange regions in Fig. 2.10 and the following figures. The continuous dynamics, reset functions, and input constraints are parametrized suitably (but not shown here for brevity), and the set of transitions follows from the adjacency of the invariant sets.

The initial state is $x_0 = [12, -7, 0]^{\mathrm{T}} \in I_{(0)}$, and the terminal state is set to $x_g = [-2, -12, -2]^{\mathrm{T}} \in I_{(g)}$. The terminal region X_g is marked as a green region in the figures, and N is first selected to be 15, which leads to a number of 102 binary variables to be employed in Problem 2.2. Three different phase sequences

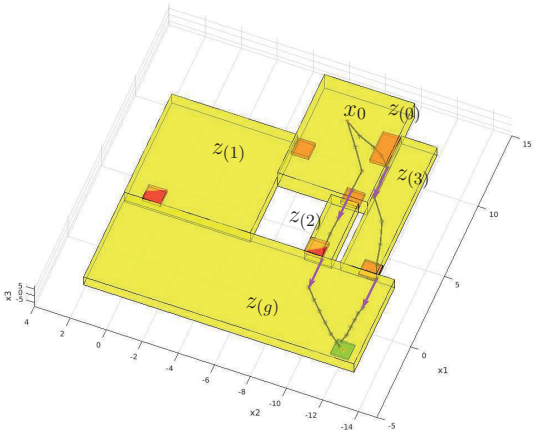

Figure 2.10.: Optimal trajectories for the two phase sequences $\phi_p = \{z_{(0)}, z_{(2)}, z_{(g)}\}$ and $\phi_p = \{z_{(0)}, z_{(3)}, z_{(g)}\}$, where the trajectories in magenta are resets of transitions.

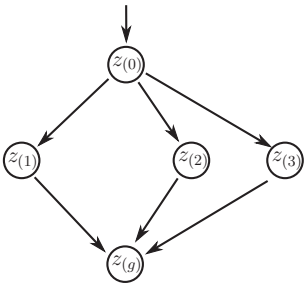

Figure 2.11.: Transitions map of the HA in Fig. 2.10.

are possible as shown in Fig. 2.11. Nevertheless, only for $\phi_p = \{z_{(0)}, z_{(2)}, z_{(g)}\}$ and $\phi_p = \{z_{(0)}, z_{(3)}, z_{(g)}\}$ optimal admissible trajectories are found with selected horizon $N = 15$, leading to costs of 3135.18 and 3429.26, and requiring computation times of 0.08 *sec* and 0.09 *sec* on a 3.4GHz processor using Matlab 2015a and the solver CPLEX. Through constraint (2.20), the relevant combinations of the binary variables are reduced from 2^{102} to $\binom{17}{4} = 2380$ according to previous discussion, and

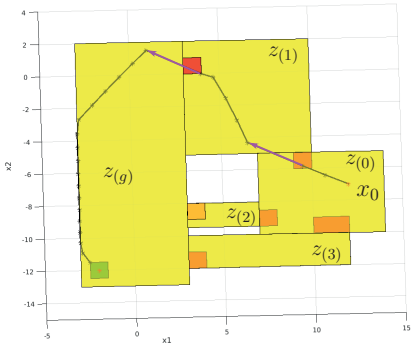

Figure 2.12.: Optimal trajectory for $\phi_p = \{z_{(0)}, z_{(1)}, z_{(g)}\}$ with $N = 25$.

the time to verify the infeasibility of $\phi_p = \{z_{(0)}, z_{(1)}, z_{(g)}\}$ for $N = 15$ is about 0.01 *sec*. If, for the latter ϕ_p, the time horizon is increased to $N = 25$, then the admissible trajectory shown in Fig. 2.12 is obtained with optimal cost of 6160.51 computed in 0.72 *sec*. A further test on a new HA with unique $\phi_p = \{z_{(0)}, z_{(1)}, z_{(2)}, z_{(g)}\}$, i.e., a longer phase sequence than in the last test, and a horizon $N = 24$, the optimal trajectory (determined in 1.06 *sec*) is illustrated in Fig. 2.13.

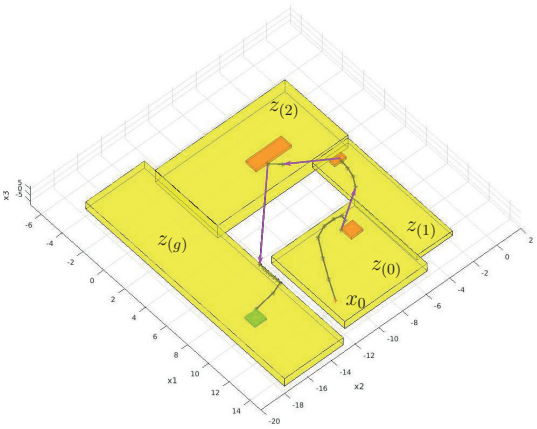

Figure 2.13.: Optimal trajectory for $\phi_p = \{z_{(0)}, z_{(1)}, z_{(2)}, z_{(g)}\}$ with $N = 24$.

2.3. Controller Synthesis without Enumerating Phase Sequences

In the last section, the solution of Problem 2.1 has been realized by first enumerating all available phase sequences in a given HA, and then formulating and solving Problem 2.2 for each phase sequence and comparing their costs. This scheme, however, may not be suitable for HA with larger sets of discrete states and transitions, as the number of available phase sequences increases exponentially (in the worst case) when more discrete states and transitions are included in HA. This section, in contrast, proposes a solution of Problem 2.1 by avoiding the enumeration procedure, and shows that the globally optimal solution can be determined by solving only one MIQP problem. This scheme is especially beneficial for some simpler tasks, such as checking the feasibility of a state-to-set transfer problem with a given finite horizon. In addition, a constraint purely among the binary variables is also formulated in the MIQP problem, in order to reduce the solution complexity.

The key idea of the approach, which was first proposed in the publication [97], is still to encode the logic conditions contained in Def. 2.1, namely:

1. starting from state x_k, an intermediate state x' is first reached according to $x' := A_{(i)} \cdot x_k + B_{(i)} \cdot u_k$;
2. the reset function r is triggered and the state x_{k+1} is reached according to $x_{k+1} := E_{(i,j)} \cdot x' + e_{(i,j)}$, if a transition is triggered,

and to cast them into a set of linear constraints using binary variables. Note that the extension of the index set of the time horizon is still necessary here. But as the phase sequence ϕ_p is not enumerated and thus unknown here, the extent of this extension also belongs to the variables to be optimized. Accordingly, the extended index set is defined as $T_{\tilde{N}} = \{0, 1, \ldots, N + |\phi_p| - 1\}$, and for any step $\tilde{k} \in T_{\tilde{N}}$, the semantics in Def. 2.1 is equivalently reformulated into the following:

$$
\begin{aligned}
&\textit{If } x_{\tilde{k}} \in I_{(i)}, x_{\tilde{k}} \notin G_{(i,j)}, \forall G_{(i,j)} \in G, \textit{ then } x_{\tilde{k}+1} := A_{(i)} x_{\tilde{k}} + B_{(i)} u_{\tilde{k}} \in I_{(i)},\ z_{\tilde{k}+1} := z_{(i)};\\
&\textit{If } x_{\tilde{k}} \in G_{(i,j)} \textit{ and } v_{(i,j),\tilde{k}} = 0, \textit{ then } x_{\tilde{k}+1} := A_{(i)} x_{\tilde{k}} + B_{(i)} u_{\tilde{k}} \in I_{(i)},\ z_{\tilde{k}+1} := z_{(i)};\\
&\textit{If } x_{\tilde{k}} \in G_{(i,j)} \textit{ and } v_{(i,j),\tilde{k}} = 1, \textit{ then } x_{\tilde{k}+1} := E_{(i,j)} x_{\tilde{k}} + e_{(i,j)} \in I_{(j)},\ z_{\tilde{k}+1} := z_{(j)}.
\end{aligned}
\tag{2.24}
$$

For convenience of description, the index extension is first bound to $T_{\tilde{N}} = \{0, 1, \ldots, N + L_{max}\}$ conservatively, where L_{max} is an upper bound of the times in which a transition τ may be triggered along an admissible trajectory. It will be shown that the feasibility and optimality of Problem 2.1 will not be affected by such a conservative extension.

After the time horizon is extended, by introducing one binary variable for each invariant set, guard set, and the terminal set for any step $\tilde{k} \in T_{\tilde{N}}$ as in the last section, a number of $(|Z| + |G| + 1) \cdot (N + L_{max} + 1)$ binary variables are defined and are collected in a set $\mathcal{B}$. Meanwhile, the following constraints are obtained for all $\tilde{k} \in \{0, \cdots, N + L_{max}\}$:

$$\begin{aligned} &C_{(i)} \cdot x_{\tilde{k}} \leq d_{(i)} + b_{(i),\tilde{k}} \cdot M_{(i)},\ \forall z_{(i)} \in Z; \\ &C_{(i,j)} \cdot x_{\tilde{k}} \leq d_{(i,j)} + b_{(i,j),\tilde{k}} \cdot M_{(i,j)},\ \forall G_{(i,j)} \in G; \\ &C_g \cdot x_{\tilde{k}} \leq d_g + b_{g,\tilde{k}} \cdot M_g. \end{aligned} \tag{2.25}$$

The value of the binary variables for $\tilde{k} = 0$ is known according to the initial hybrid states (as in the last section). But unlike the scheme in the last section where the constraint (2.25) is only used to encode the logic conditions for a single phase sequence, here it will be applied to directly encode the transition rule in (2.24), and thus the original semantics in Def. 2.1. Following this line, the transition rule in (2.24) will first be decomposed into the parts of the discrete and the continuous dynamics in the coming section. Then, a type of encoding techniques based on (2.25) will be introduced for each part to cast them into linear constraints.

2.3.1. Invariant Sets

When abstracting from the continuous dynamics in (2.24) for a moment, the following logical relations are obtained:

$$\textit{If } x_{\tilde{k}} \in I_{(i)},\ x_{\tilde{k}} \notin G_{(i,j)},\ \forall G_{(i,j)} \in G,\ \textit{ then } x_{\tilde{k}+1} \in I_{(i)} \textit{ must apply;} \tag{2.26}$$

$$\textit{If } x_{\tilde{k}} \in I_{(i)},\ x_{\tilde{k}} \in G_{(i,j)} \textit{ and } v_{(i,j),\tilde{k}} = 0,\ \textit{ then } x_{\tilde{k}+1} \in I_{(i)} \textit{ must apply;} \tag{2.27}$$

$$\textit{If } x_{\tilde{k}} \in I_{(i)},\ x_{\tilde{k}} \in G_{(i,j)} \textit{ and } v_{(i,j),\tilde{k}} = 1,\ \textit{ then } x_{\tilde{k}+1} \in I_{(j)} \textit{ must apply.} \tag{2.28}$$

As an auxiliary means, let a set of possible discrete successor states of a discrete state $z_{(i)} \in Z$ be defined as $Z^{out}_{(i)}$, where $z_{(j)} \in Z$, $z_{(j)} \neq z_{(i)}$ is included in $Z^{out}_{(i)}$ exactly if a transition $\tau_{(i,j)} \in \mathcal{T}$ exists. Now, to encode the logical relations in (2.26) – (2.28) in terms of binary variables, the following constraints together with the constraint (2.25) are introduced for $\tilde{k}$:

$$\sum_{i=1}^{|Z|} b_{(i),\tilde{k}} = |Z| - 1. \tag{2.29}$$

$$\sum_{l=1}^{|Z^{out}_{(i)}|} b_{(i,l),\tilde{k}} \geq |Z^{out}_{(i)}| - 1,\ \sum_{G_{(i,j)} \in G}^{|G|} b_{(i,j),\tilde{k}} \geq |G| - 1. \tag{2.30}$$

$$\textit{If } b_{(i),\tilde{k}} = 0,\ b_{(i,j),\tilde{k}} = 1,\ \sum_{l=1}^{|Z^{out}_{(i)}|} b_{(i,l),\tilde{k}} = |Z^{out}_{(i)}|,\ \textit{then } b_{(i),\tilde{k}+1} = 0 \textit{ must apply.} \tag{2.31}$$

$$\textit{If } b_{(i),\tilde{k}} = 0,\ b_{(i,j),\tilde{k}} = 0,\ \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} = |Z_{(i)}^{out}| - 1, \textit{ then } b_{(j),\tilde{k}+1} = 0 \textit{ must apply.} \tag{2.32}$$

Note that for given continuous state $x_{\tilde{k}}$ and $x_{\tilde{k}+1}$, as well as discrete input $v_{(i,j),\tilde{k}}$, if the logical relations in (2.25) and (2.29) – (2.32) are satisfied, then the continuous states and discrete input must also satisfy the constraints (2.26) – (2.28):

- For the first relation (2.26), if $x_{\tilde{k}}$ satisfies the condition $x_{\tilde{k}} \in I_{(i)}$, $x_{\tilde{k}} \notin G_{(i,j)}$ for all $G_{(i,j)} \in G$, then there must exist $b_{(i),\tilde{k}} = 0$ and $b_{(i,j),\tilde{k}} = 1$, $\forall G_{(i,j)} \in G$ according to constraints (2.25) and (2.29). Then, based on constraint (2.31) one knows that if $b_{(i),\tilde{k}} = 0$ and $b_{(i,j),\tilde{k}} = 1$ are satisfied, as well as $\sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} = |Z_{(i)}^{out}|$ (since $b_{(i,j),\tilde{k}} = 1$, $\forall G_{(i,j)} \in G$), there must exist $b_{(i),\tilde{k}+1} = 0$, which implies $x_{\tilde{k}+1} \in I_{(i)}$ according to (2.25), i.e., the logical relation in (2.26) is satisfied;

- For relations (2.27) and (2.28), if $x_{\tilde{k}}$ satisfies the condition $x_{\tilde{k}} \in I_{(i)}$ and $x_{\tilde{k}} \in G_{(i,j)}$, then there must exist $b_{(i),\tilde{k}} = 0$ and $b_{(i,j),\tilde{k}} \in \{0, 1\}$ in the light of constraint (2.25). Thus, if the state $x_{\tilde{k}}$ is inside of the guard set $G_{(i,j)}$, the relation: If $b_{(i,j),\tilde{k}} = 0$, then $\sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} = |Z_{(i)}^{out}| - 1$, must apply according to constraint (2.30). Now, based on (2.31), if $b_{(i,j),\tilde{k}} = 1$ and $\sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} = |Z_{(i)}^{out}|$, there must exist $b_{(i),\tilde{k}+1} = 0$, which means $x_{\tilde{k}+1} \in I_{(i)}$ due to (2.25). Thus, the relation in (2.27) is satisfied. For the case that $b_{(i,j),\tilde{k}} = 0$ and $\sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} = |Z_{(i)}^{out}| - 1$, there must exist $b_{(j),\tilde{k}+1} = 0$ according to (2.32), which means $x_{\tilde{k}+1} \in I_{(j)}$ according to (2.25), i.e., the relation in (2.28) is also satisfied.

Following the description above, it is not hard to notice that the selection of the discrete input $v_{(i,j),\tilde{k}}$ in (2.27) and (2.28) is cast into the value selection of the binary variable $b_{(i,j),\tilde{k}}$ in constraints (2.29) – (2.32) and (2.25). However, besides constraint (2.29) and (2.30) are formulated in linear form, the other two constraints (2.31) and (2.32) do still contain logical expressions. Thus the following steps aim at casting the (2.31) and (2.32) into linear form as well.

First note that both (2.31) and (2.32) are actually representing a mapping from the vector $\left[b_{(i),\tilde{k}} \quad b_{(i,j),\tilde{k}} \quad \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}}\right]$ in step $\tilde{k}$ to the vector $\left[b_{(i),\tilde{k}+1} \quad b_{(j),\tilde{k}+1}\right]$ in the succeeding step $\tilde{k}+1$. This mapping is summarized in Table 2.1. The binary choice of $\sum_{l=1}^{|Z_{[i]}^{out}|} b_{(i,l),\tilde{k}}$ in this table, together with constraint (2.29), take four value

Table 2.1.: Relevant cases of the vector $[b_{(i),\tilde{k}}\ b_{(i,j),\tilde{k}}\ \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}}]$.

$b_{(i),\tilde{k}}$	$b_{(i,j),\tilde{k}}$	$\sum_{l=1}^{\lvert Z_{(i)}^{out}\rvert} b_{(i,l),\tilde{k}}$	implies	$b_{(i),\tilde{k}+1}$	$b_{(j),\tilde{k}+1}$
0	1	$\lvert Z_{(i)}^{out}\rvert$	$\rightarrow$	0	1
0	0	$\lvert Z_{(i)}^{out}\rvert - 1$	$\rightarrow$	1	0
0	1	$\lvert Z_{(i)}^{out}\rvert - 1$	$\rightarrow$	1	1
1	1	$\lvert Z_{(i)}^{out}\rvert$	$\rightarrow$	$\{0,1\}$	$\{0,1\}$

combinations of the vector $\begin{bmatrix} b_{(i),\tilde{k}} & b_{(i,j),\tilde{k}} & \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} \end{bmatrix}$ into account. The remaining combinations are all not related or not feasible due to (2.25), (2.29) and (2.30). It has to be emphasized that for the combination $\begin{bmatrix} 1 & 1 & |Z_{(i)}^{out}| \end{bmatrix}$ in Table 2.1, it only implies that the state $x_{\tilde{k}}$ is not in the set $I_{(i)}$. Whether $x_{\tilde{k}+1} \in I_{(i)}$ or $x_{\tilde{k}+1} \in I_{(j)}$ applies, still has to be considered.

Next, the mappings from $\begin{bmatrix} b_{(i),\tilde{k}} & b_{(i,j),\tilde{k}} & \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} \end{bmatrix}$ to $\begin{bmatrix} b_{(i),\tilde{k}+1} & b_{(j),\tilde{k}+1} \end{bmatrix}$ in Table 2.1 are cast into a set of linear constraints by using a similar method introduced in Sec. 2.2.2: a set of arbitrary parameter vectors $\alpha_{i,1} \in \mathbb{R}^{4\times 1}$, $\beta_{i,1} \in \mathbb{R}^{4\times 1}$, and $\alpha_{i,2} \in \mathbb{R}^{4\times 1}$, $\beta_{i,2} \in \mathbb{R}^{4\times 1}$ are first introduced, which satisfy the following four conditions:

$$
\begin{aligned}
\begin{bmatrix} -\infty \\ 0 \\ 0 \\ -\infty \end{bmatrix} < \begin{bmatrix} 0 & 1 & |Z_{(i)}^{out}| & 1 \\ 0 & 0 & |Z_{(i)}^{out}| - 1 & 1 \\ 0 & 1 & |Z_{(i)}^{out}| - 1 & 1 \\ 1 & 1 & |Z_{(i)}^{out}| & 1 \end{bmatrix} \cdot \alpha_{i,1} < \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}; \\
\begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} < \begin{bmatrix} 0 & 1 & |Z_{(i)}^{out}| & 1 \\ 0 & 0 & |Z_{(i)}^{out}| - 1 & 1 \\ 0 & 1 & |Z_{(i)}^{out}| - 1 & 1 \\ 1 & 1 & |Z_{(i)}^{out}| & 1 \end{bmatrix} \cdot \beta_{i,1} < \begin{bmatrix} 1 \\ \infty \\ \infty \\ \infty \end{bmatrix}; \\
\begin{bmatrix} 0 \\ -\infty \\ 0 \\ -\infty \end{bmatrix} < \begin{bmatrix} 0 & 1 & |Z_{(i)}^{out}| & 1 \\ 0 & 0 & |Z_{(i)}^{out}| - 1 & 1 \\ 0 & 1 & |Z_{(i)}^{out}| - 1 & 1 \\ 1 & 1 & |Z_{(i)}^{out}| & 1 \end{bmatrix} \cdot \alpha_{i,2} < \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}; \\
\begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} < \begin{bmatrix} 0 & 1 & |Z_{(i)}^{out}| & 1 \\ 0 & 0 & |Z_{(i)}^{out}| - 1 & 1 \\ 0 & 1 & |Z_{(i)}^{out}| - 1 & 1 \\ 1 & 1 & |Z_{(i)}^{out}| & 1 \end{bmatrix} \cdot \beta_{i,2} < \begin{bmatrix} \infty \\ 1 \\ \infty \\ \infty \end{bmatrix}.
\end{aligned} \tag{2.33}
$$

Here, the first three columns in the matrix before the parameter vectors encode

all value combinations of the vector $[b_{(i),\tilde{k}}\ b_{(i,j),\tilde{k}}\ \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}}]$ in Table 2.1. With the help of $\alpha_{i,1}$, $\beta_{i,1}$, $\alpha_{i,2}$ and $\beta_{i,2}$, the mappings in Table 2.1 can be algebraically and equivalently formulated as the following linear constraints:

$$\begin{aligned}
b_{(i),\tilde{k}+1} &\geq \alpha_{i,1}^{\mathrm{T}}(1:4)\cdot\left[b_{(i),\tilde{k}}\quad b_{(i,j),\tilde{k}}\quad \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}}\quad 1\right]^{\mathrm{T}};\\
b_{(i),\tilde{k}+1} &\leq \beta_{i,1}^{\mathrm{T}}(1:4)\cdot\left[b_{(i),\tilde{k}}\quad b_{(i,j),\tilde{k}}\quad \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}}\quad 1\right]^{\mathrm{T}};\\
b_{(j),\tilde{k}+1} &\geq \alpha_{i,2}^{\mathrm{T}}(1:4)\cdot\left[b_{(i),\tilde{k}}\quad b_{(i,j),\tilde{k}}\quad \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}}\quad 1\right]^{\mathrm{T}};\\
b_{(j),\tilde{k}+1} &\leq \beta_{i,2}^{\mathrm{T}}(1:4)\cdot\left[b_{(i),\tilde{k}}\quad b_{(i,j),\tilde{k}}\quad \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}}\quad 1\right]^{\mathrm{T}}.
\end{aligned} \tag{2.34}$$

Finally, as the mappings in Table 2.1 also represent the logical relations in (2.31) and (2.32), the latter relations are now successfully cast into linear constraints (2.34). These linear constraints, together with constraint (2.29) and (2.30), are thus sufficient to represent the logical constraints (2.26) – (2.28), i.e., the logic behind the discrete state transition in (2.24).

2.3.2. Continuous Dynamics

For the logical relations in (2.24), by only considering the continuous dynamics to be followed in step $\tilde{k}$, the following relations are established:

$$\begin{aligned}
&\textit{If } x_{\tilde{k}} \in I_{(i)}, x_{\tilde{k}} \notin G_{(i,j)}, \forall G_{(i,j)} \in G, \textit{ then } x_{\tilde{k}+1} := A_{(i)}x_{\tilde{k}} + B_{(i)}u_{\tilde{k}};\\
&\textit{If } x_{\tilde{k}} \in G_{(i,j)} \textit{ and } v_{(i,j),\tilde{k}} = 0, \textit{ then } x_{\tilde{k}+1} := A_{(i)}x_{\tilde{k}} + B_{(i)}u_{\tilde{k}};\\
&\textit{If } x_{\tilde{k}} \in G_{(i,j)} \textit{ and } v_{(i,j),\tilde{k}} = 1, \textit{ then } x_{\tilde{k}+1} := E_{(i,j)}x_{\tilde{k}} + e_{(i,j)}.
\end{aligned} \tag{2.35}$$

This together with the constraint (2.25), (2.29), and (2.34) can be equivalently represented by:

$$\begin{aligned}
&\textit{If } b_{(i),\tilde{k}} = 0,\ b_{(i,j),\tilde{k}} = 1, \textit{ and } \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} = |Z_{(i)}^{out}|, \textit{ then } x_{\tilde{k}+1} := A_{(i)}x_{\tilde{k}} + B_{(i)}u_{\tilde{k}};\\
&\textit{If } b_{(i),\tilde{k}} = 0,\ b_{(i,j),\tilde{k}} = 0, \textit{ and } \sum_{l=1}^{|Z_{(i)}^{out}|} b_{(i,l),\tilde{k}} = |Z_{(i)}^{out}| - 1, \textit{ then } x_{\tilde{k}+1} := E_{(i,j)}x_{\tilde{k}} + e_{(i,j)}.
\end{aligned} \tag{2.36}$$

Now by introducing auxiliary variables $\xi_{(i),\tilde{k}}$, $\pi_{(i),\tilde{k}}$, and $\eta_{(i,l),\tilde{k}}$ similar to those in Problem 2.2, as well as vectors $\Theta_{(i)}^{+}$, $\Theta_{(i)}^{-}$, Θ_{u}^{+}, Θ_{u}^{-} and λ_x, λ_u determined according

to (2.21) and (2.22), the continuous dynamics in step $\tilde{k}$ in (2.24) can be reformulated by using the following linear constraints:

$$x_{\tilde{k}+1} := A_{(i)}\xi_{(i),\tilde{k}} + B_{(i)}\pi_{(i),\tilde{k}} + \sum_{l=1}^{|Z_{(i)}^{out}|} \eta_{(i,l),\tilde{k}}, \tag{2.37}$$

where for all $z_{(l)} \in Z_{(i)}^{out}$:

$$\Theta_{(i)}^{-} \cdot (b_{(i,l),\tilde{k}} - b_{(i),\tilde{k}}) \leq \xi_{(i),\tilde{k}} \leq \Theta_{(i)}^{+} \cdot (b_{(i,l),\tilde{k}} - b_{(i),\tilde{k}}), \tag{2.38a}$$
$$x_{\tilde{k}} - \lambda_x \cdot (1 - b_{(i,l),\tilde{k}} + b_{(i),\tilde{k}}) \leq \xi_{(i),\tilde{k}} \leq x_{\tilde{k}} + \lambda_x \cdot (1 - b_{(i,l),\tilde{k}} + b_{(i),\tilde{k}}), \tag{2.38b}$$

$$\Theta_{u}^{-} \cdot (b_{(i,l),\tilde{k}} - b_{(i),\tilde{k}}) \leq \pi_{(i),\tilde{k}} \leq \Theta_{u}^{+} \cdot (b_{(i,l),\tilde{k}} - b_{(i),\tilde{k}}), \tag{2.38c}$$
$$u_{\tilde{k}} - \lambda_u \cdot (1 - b_{(i,l),\tilde{k}} + b_{(i),\tilde{k}}) \leq \pi_{(i),\tilde{k}} \leq u_{\tilde{k}} + \lambda_u \cdot (1 - b_{(i,l),\tilde{k}} + b_{(i),\tilde{k}}), \tag{2.38d}$$

$$\lambda_x \cdot (b_{(i,l),\tilde{k}} - 1) \leq \eta_{(i,l),\tilde{k}} \leq \lambda_x \cdot (1 - b_{(i,l),\tilde{k}}), \tag{2.38e}$$
$$E_{(i,l)}x_{\tilde{k}} + e_{(i,l)} - \lambda_x \cdot b_{(i,l),\tilde{k}} \leq \eta_{(i,l),\tilde{k}} \leq E_{(i,l)}x_{\tilde{k}} + e_{(i,l)} + \lambda_x \cdot b_{(i,l),\tilde{k}}. \tag{2.38f}$$

Now, through the derivations above, both the discrete state and continuous state transitions in (2.24) are reformulated into a set of linear constraints (2.25), (2.29), (2.30), (2.34), (2.37) and (2.38a)-(2.38f). This means that the admissibility of ϕ_x, ϕ_z in Problem 2.1 can be ensured through these linear constraints. Note for the remaining constraints contained in Problem 2.1, $u_k \in U$ is already in linear form and $v_{(i,j),k} \in \{0,1\}$ has already been ensured through constraint (2.34). The only constraint that has not yet been considered is the terminal constraint $x_N \in X_g, z_N = z_g$, which will be the focus of the coming part.

2.3.3. Encoding the Terminal Constraint

The requirement that, once $x_{\tilde{k}} \in X_g$ applies, then $x_{\hat{k}} \in X_g$ for $\tilde{k} \geq \hat{k}$ can be equivalently modeled by:

$$b_{g,\tilde{k}} \geq b_{g,\tilde{k}+1}. \tag{2.39}$$

This constraint implies that once $b_{g,\tilde{k}} = 0$, then $b_{g,\tilde{k}+1} = 0$ must apply, i.e., enforcing $x_{\tilde{k}+1} \in X_g$ according to (2.25). For the terminal constraint $x_N \in X_g$, recall that it corresponds to the constraint $x_{N+L_{max}} \in X_g$ for the extended time set $T_{\tilde{N}}$, if a number of L_{max} transitions were indeed triggered.

In the general case, the terminal constraint $x_N \in X_g$ is equal to $x_{N+\rho} \in X_g$, where ρ denotes the amount of additional steps caused by transitions. However, as mentioned earlier the exact number of necessary additional steps is only known after solving Problem 2.1, since it is an outcome of the optimization. Nevertheless,

one can still compute the number ρ by:

$$\rho = (N + L_{max} + 1) \cdot |G| - \sum_{\tilde{k}=0}^{N+L_{max}} \sum_{G_{(i,j)} \in G} b_{\tilde{k},(i,j)}. \tag{2.40}$$

This is because, if the constraints (2.25), (2.29), (2.30) and (2.34) hold, then the transition from $z_{(i)}$ to $z_{(j)}$ can only be triggered if the binary variable $b_{\tilde{k},(i,j)}$ is 0 for all $\tilde{k} \in \{0, \cdots, N + L_{max}\}$. As a result, the right hand side of (2.40) is actually recording how many $b_{\tilde{k},(i,j)}$ are equal to 0 among all guard sets, which forces the outcome of the right hand side of (2.40) to be equal to the number of the transitions being executed.

Accordingly, together with the constraint (2.39), the terminal constraint $x_N \in X_g$ can be equivalently reformulated into the following linear form:

$$\sum_{\tilde{k}=0}^{N+L_{max}} b_{g,\tilde{k}} \leq N + \rho. \tag{2.41}$$

It implicitly requires $b_{g,N+\rho} = 0$, which enforces $x_{N+\rho} \in X_g$ according to (2.25). Now, all the constraints in Problem 2.1 are reformulated into linear form, leading to the following equivalent problem:

Problem 2.3. *For HA initialized to (x_0, z_0), $z_0 := z_{(s)}$, let a time set T_N and a goal (X_g, z_g) be given. Then, determine continuous input sequences ϕ_u^* and the set of binary variable [1] $\mathcal{B}$, as the solution of:*

$$\min_{\phi_u, \mathcal{B}} \sum_{\tilde{k}=1}^{N+L_{max}} \{(\hat{x}_{\tilde{k}} - x_g)^{\mathrm{T}} Q (\hat{x}_{\tilde{k}} - x_g) + u_{\tilde{k}-1}^{\mathrm{T}} R u_{\tilde{k}-1}\} + q_g \cdot (\sum_{\tilde{k}=1}^{N+L_{max}} b_{g,\tilde{k}} - \rho)$$

$$s.t.: \ \textit{Constraints}\,(2.25),\ (2.39) - (2.41),\ \textit{and} \tag{2.42a}$$

$$\textit{for } \tilde{k} \in \{0, \ldots, N + L_{max} - 1\}:$$

$$u_{\tilde{k}} \in U, \tag{2.42b}$$

$$x_{\tilde{k}+1} := \sum_{i=1}^{|Z|} \{A_{(i)} \cdot \xi_{(i),\tilde{k}} + B_{(i)} \cdot \pi_{(i),\tilde{k}}\} + \sum_{i=1}^{|Z|} \sum_{j=1}^{|Z_{(i)}^{out}|} \eta_{(i,j),\tilde{k}} \tag{2.42c}$$

$$x_{\tilde{k}} - \lambda_x \cdot (|G| - \sum_{G_{(i,j)} \in G} b_{(i,j),\tilde{k}}) \leq \hat{x}_{\tilde{k}} \leq x_{\tilde{k}} + \lambda_x \cdot (|G| - \sum_{G_{(i,j)} \in G} b_{(i,j),\tilde{k}}), \tag{2.42d}$$

$$-\lambda_x \cdot (\sum_{G_{(i,j)} \in G} b_{(i,j),\tilde{k}} + 1 - |G|) \leq \hat{x}_{\tilde{k}} \leq \lambda_x \cdot (\sum_{G_{(i,j)} \in G} b_{(i,j),\tilde{k}} + 1 - |G|), \tag{2.42e}$$

$$\textit{Constraints}\,(2.29), (2.30), (2.34), (2.38a) - (2.38f),\ \forall z_{(i)} \in Z,\ \forall z_{(j)} \in Z_{(i)}^{out}. \tag{2.42f}$$

The state evolution (2.42c) is an extension of (2.37) by taking all invariant sets and guard sets into account. The constraints (2.42d) to (2.42e) ensure that the

[1] Note that through constraint (2.34), the optimization of the sequence ϕ_v has been cast into the optimization of binary variables $b_{(i,j),\tilde{k}}$ for all $G_{(i,j)} \in G$ and for all $\tilde{k} \in \{0, \cdots, N + L_{max} - 1\}$.

costs induced by the intermediate states x' of the transition are not recorded in the cost function. The cost term N_g in the original cost function (2.2) is equivalently cast into $\sum_{\tilde{k}=1}^{N+L_{max}} b_{g,\tilde{k}} - \rho$, which counts the number of steps that the state x is not in the terminal set X_g. Note that the Problem 2.3 is once more in the form of (2.1), i.e., it represents an MIQP problem with linear constraints, and can thus be solved by CPLEX.

Furthermore, as all the constraints and the cost function defined in Problem 2.3 are obtained through equivalent reformulation of the constraints in Problem 2.1, both problems share exactly the same feasibility and optimality. This property enables one to use the set of constraints defined in Problem 2.3 to specify the feasibility of Problem 2.1 (a verification task rather than optimization). Moreover, the constraint (2.34) only involves the binary variables and can thus reduce the set of possible value combinations, i.e., reduce the overall complexity (same reason as for constraint (2.20) in the last section). At last, compared with the last approach where enumeration is required, the new approach can realize the optimization and verification task in one program. In addition, when new transitions are included or certain transitions are prohibited in given HA, or certain discrete states must be visited in the obtained trajectory, one only has to adapt the corresponding constraints in this approach, instead of enumerating all discrete state sequences once more (this property is illustrated in the following example).

2.3.4. Numeric Examples

The HA considered here for numeric illustration is similar to the one in Sec. 2.2.3, but contains more transitions and thus makes the enumeration of all phase sequences difficult. Again, the invariant sets are marked in yellow, the guard sets in orange and the target set in green, see Fig. 2.15. The transition map of the considered HA with a number $n_z = 7$ of discrete states is shown in Fig. 2.14. The weighting factors, the continuous dynamics in each invariant set, the reset function of each guard set, and the input set are parametrized suitably.

In the first test, the initial state is selected to be $x_0 = [15, -6, 0]^{\mathrm{T}} \in I_{(1)}$ and the terminal state $x_g = [-4, -10, 0]^{\mathrm{T}} \in I_{(3)}$. The considered horizon is $N = 20$, L_{max} is chosen to be 5, so that a number of $(7+17+1)\cdot(20+5+1) = 650$ binary variables is needed. By solving Problem 2.3 for the given setting, the optimal trajectory is obtained in 18.77 *sec* with a cost of 1385.40, as shown in Fig. 2.15. There, the optimal continuous state sequence starts from $z_{(1)}$, enters into guard set $G_{(1,3)}$, and then reaches $z_{(3)}$. If one only seeks for a feasible trajectory instead of the optimal one (a verification task), the considered program terminates within only 0.13 *sec*.

In the second test, an additional requirement is considered, that the discrete state $z_{(5)}$ must once be visited. This requirement can be equivalently formulated as constraint $\sum_{\tilde{k}=0}^{N+L_{max}} b_{(5),\tilde{k}} \leq N + L_{max}$ in Problem 2.3, which implies that the

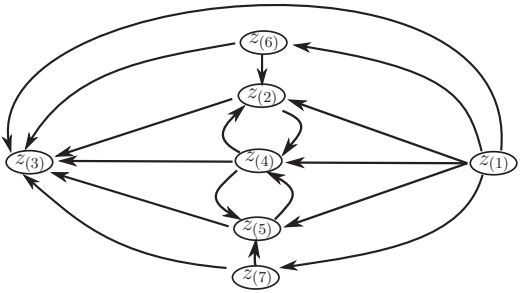

Figure 2.14.: Transition map of the considered HA.

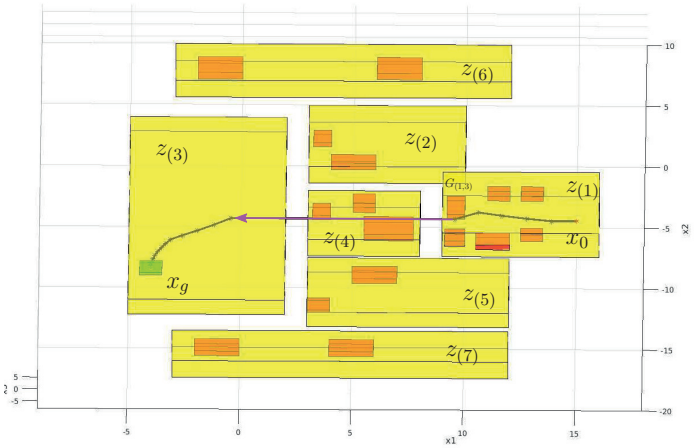

Figure 2.15.: Optimal trajectory of the first test, where the reset of the continuous states by transition $z_{(1)} \rightarrow z_{(3)}$ is marked in magenta.

binary variable $b_{(5),\tilde{k}}$ must at least once be zero over the horizon. The new optimal trajectory is obtained in 23.75 *sec* with a cost of 1496.21 (while the first feasible trajectory in found in only 0.63 *sec*), and is shown in Fig. 2.16.

For the alternative requirement that the discrete state $z_{(7)}$ must be visited, the infeasibility of the task is verified within 0.35 *sec*. The tests above shows the high flexibility with which Problem 2.3 can be adapted when the control task changes, as well as efficiency with respect to computation time. All the tests were carried out on a 3.4GHZ processor using Matlab 2015a with the solver CPLEX.

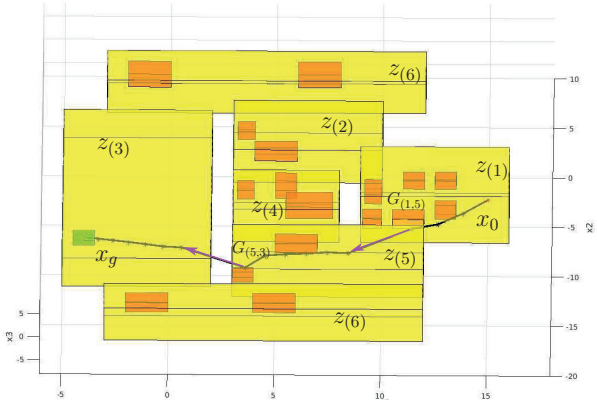

Figure 2.16.: Optimal trajectory with visiting $z_{(5)}$ once.

2.4. Case Study: An Overtaking Problem for an Autonomous Vehicle

The case study of automated driving is an example of a CPS being composed of several subsystems (representing the automated vehicles) which evolve over time in interacting manner, see Fig. 2.17. The number of vehicles involved in a scenario determines the information exchanged through the communication network and the number of restrictions to be considered for the driving plans of each vehicle.

For each local vehicle in the network, such as s_1 marked in Fig. 2.17, its basic objective is to guarantee that the local maneuvers are safe with respect to avoiding inter-vehicle collision. The maneuvers of s_1 are thus determined by its local dynamics and the information sent from other vehicles, which contains, e.g., the region of

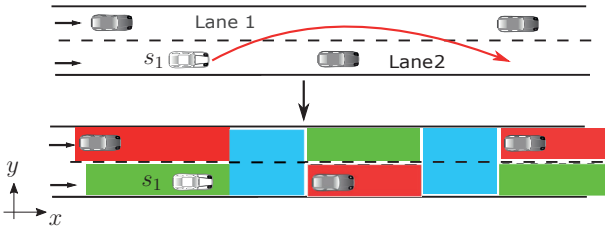

Figure 2.17.: Vehicle overtaking scenario on a straight road for an automated car.

possible positions to be occupied over a future time span (marked in red). Based on this information, the controller of s_1 can identify the regions that it can freely move (marked in green and blue) over the future time span. If, in addition, the controller of s_1 intends to overtake the vehicle in front, a set of new regions on the road are assigned (marked in blue), within which the lane-change maneuver is allowed to be executed safely. The goal of this study is to plan a trajectory for s_1 by constructing a hybrid system HA, such that the overtaking task can be accomplished, while all safety issues are taken into account.

Model of the Vehicle Dynamics

For the considered trajectory planning problem, the selected vehicle model directly affects the quality of the obtained trajectory. For reducing the complexity of planning, a double integrator model is often used [54, 53], since only the position and velocity of the vehicle are relevant. If, however, a precise model of the vehicle was used in planning, e.g. a high-dimensional nonlinear model consisting of the vehicle position, velocity, acceleration, yaw angle, yaw rate, steering angle [135], the complexity of the planning would increase significantly.

As a compromise between accuracy and complexity, the vehicle dynamics is here approximated by a set of linear differential equations, each of which models a typical behavior of the vehicle during the overtaking. In detail, the state of the vehicle contains longitudinal position $p_x(t)$ and velocity $v_x(t)$, and lateral position $p_y(t)$ and velocity $v_y(t)$, satisfying the differential equation:

$$\begin{bmatrix} \dot{p}_x(t) \\ \dot{p}_y(t) \\ \dot{v}_x(t) \\ \dot{v}_y(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_x(t) \\ p_y(t) \\ v_x(t) \\ v_y(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ cos(\theta(t)) \\ sin(\theta(t)) \end{bmatrix} u(t), \tag{2.43}$$

where $\theta(t)$ is the angle of the vehicle to the longitudinal road axis and $u(t)$ denotes the acceleration along the driving direction (local coordinate). Both $\theta(t)$ and $u(t)$ are the inputs to be selected in (2.43), and it is assumed that $\theta(t)$ can only take value from the finite set $\{-\frac{\pi}{9}, 0, \frac{\pi}{9}\}$, each representing a typical angle during the overtaking. The change of $\theta(t)$ from one value to another in the finite set can be modeled by using reset functions. The other constraints are chosen suitably.

Trajectory Planning with HA

As different θ-values lead to different linearized dynamics in (2.43), a hybrid system HA is constructed to model the hybrid behavior of s_1 during the overtaking scenario (also as an approximation of the original nonlinear dynamics (2.43)). In detail, a number of 5 discrete states $Z = \{z_{(0)}, z_{(1)}, z_{(2)}, z_{(3)}, z_{(4)}\}$ are assigned to encode the green (straight drive) and blue (lane change allowed) regions (see Fig. 2.18) and the following hybrid dynamics is applied:

- for $z_{(0)}$, $z_{(2)}$ and $z_{(4)}$, angle $\theta(t) = 0$ applies in (2.43);
- for $z_{(1)}$, $\theta(t) = \frac{\pi}{9}$ applies;
- for $z_{(3)}$, $\theta(t) = -\frac{\pi}{9}$ applies.

A set of guards $G = \{G_{(0,1)}, G_{(1,2)}, G_{(2,3)}, G_{(3,4)}\}$, $G_{(j,j+1)} \subseteq I_{z_{(j)}}$, $\forall j \in \{0,1,2,3\}$ are also assigned through a partition of the road, representing the transition condition of the discrete states, see Fig. 2.18. A set of discrete inputs $v_{(j,j+1)}$, $j \in \{0,1,2,3\}$ are also applied together with the guards, in order to decide the time of the transition. Clearly, for a more precise approximation of the nonlinear dynamics (2.43), one can take more available θ-values into account. But this will also lead to an increase of discrete states and transitions in the obtained HA, and thus complicates the corresponding control problems.

After the HA is determined and by selecting a suitable objective function, e.g., the time-optimal one, the overtaking problem is cast into a trajectory planning problem connecting $z_{(0)}$ and $z_{(4)}$, see Fig. 2.18. To achieve the overtaking task, an optimization problem in the form of Problem 2.2 is formulated (as there is only one possible phase sequence in the HA) and solved by using CPLEX. With a sampling time of $0.5sec$ and a planning horizon of 15 sec, the trajectory in Fig. 2.19 is found after 0.28 sec, as well as the velocity profile plotted in Fig. 2.20. Note that the impulsive change of the velocities in step 8 and 15 are due to an impulse change (jump) of the angle $\theta(t)$ when discrete state changes. This issue of non-smoothness

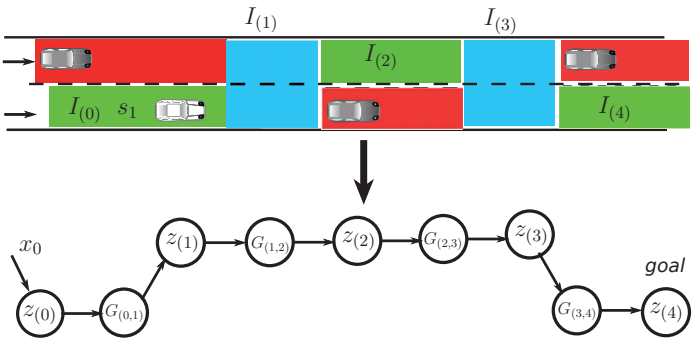

Figure 2.18.: The vehicle overtaking problem of s_1 is cast into an optimal control problem of a hybrid system HA: before the overtaking procedure starts, each invariant I and guard G of the HA is obtained by partitioning the road, and the partition is based on the local status of s_1 (including position, speed and acceleration) and the status of the vehicles around (through communication or observation).

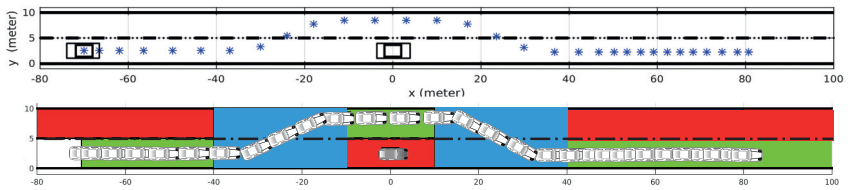

Figure 2.19.: Trajectory obtained by solving Problem 2.2.

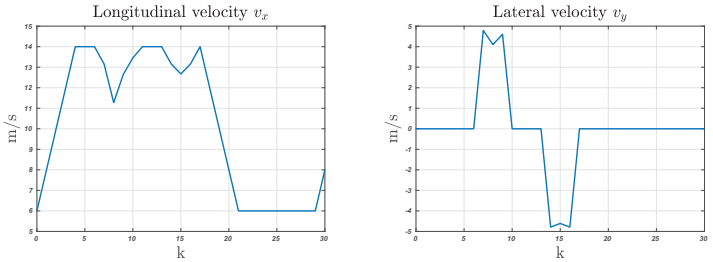

Figure 2.20.: Longitudinal and lateral velocity of s_1 during the overtaking.

may not be desired in practical realization of the plan, but can be overcome by providing an additional (subordinated) controller to smooth the sharp changes.

2.5. Summary and Discussion

In this chapter, a class of hybrid systems that can describe a large variety of transition mechanisms has been proposed to model the local dynamics of CPS. Compared to commonly considered hybrid systems (such as PWA systems), more interleaving schemes between the discrete and continuous dynamics can be modeled in the proposed HA. The enhanced modeling ability, however, also makes the optimal control problem more complicated, primarily due to the extended degree of freedom by describing the hybrid dynamics including: 1.) the discrete state path; 2.) the allocation of time points to the discrete states; 3.) the evolution of the continuous state in each discrete state; 4.) the continuous/discrete inputs. Efficient methods for the solution are thus required to meet the real-time requirement when controlling CPS online.

With this task in mind, a method requiring that the possible phase sequences of HA are enumerated (before the optimization is started) has first been proposed in Sec. 2.2. For each phase sequence, the semantics of the corresponding admissible

trajectories are cast into a tailored set of linear constraints, reducing the value combinations of binary variables required to formulate the transition dynamics. The significant reduction of the number of value combinations also reduces the search space of the underlying MIP, and thus increases the computational efficiency. The procedure does not involve approximation and thus ensures that the globally optimal solution is found.

The requirement that all phase sequences must be enumerated, however, may not be efficient, since the enumeration procedure may become intractable for a large number of discrete states/transitions of HA. To avoid the enumeration process, a new method has been proposed in Sec. 2.3, which determines the optimal control actions through only one numerical program. The basic idea is to directly cast the semantics of HA into linear constraints, instead of the semantics for each phase sequence, and it enables efficient solution as demonstrated for simulation examples.

A limiting assumption of the HA in this chapter is that the flow functions are assumed to be linear (affine). For a more general case in which nonlinear functions arise, the results published recently in [101] can enable the two solution schemes above be further applied. In a nutshell, the nonlinear flow function $f_{(i)}$ for each discrete state $z_{(i)} \in Z$ can first be approximated by a piecewise affine system, using the technique called *hybridization* [13, 50]. The *hybridization* also partitions each invariant set $I_{(i)}$ into a finite number of sub-invariants, so that a linear flow function is obtained for each sub-invariant. As a result, the original HA with nonlinear flow functions can be approximated by a new HA, in which only linear flow functions are followed, but with linearization error and more invariants (discrete states) than before. The two solution schemes can then be applied to solve the optimal control problem of the approximated HA, which indirectly solves the original problem. For the linearization error caused by approximation, a set of advanced methods are introduced in [101], so that the feasibility of the original problem can be ensured, while the performance loss is also bounded.

Nevertheless, the setting at the beginning of the chapter that the interaction among the subsystems of CPS is static and can be cast into deterministic constraints (e.g. deterministic invariants and guards) of a local subsystem may seem unrealistic especially for online control. This implies that uncertainties should be taken into account during the controller synthesis of HA, which will be the main topic of the following chapter.

3. Robust Control of Hybrid Systems with Uncertain Dynamics and Environments

In this chapter, different classes of uncertainties encountered in CPS will be taken into account, including additive disturbances and parametric uncertainties affecting the local hybrid dynamics, and time-varying state constraints (with unknown but bounded change) caused by the interaction among the subsystems, see Fig. 3.1. For the additive disturbances and parametric uncertainties, they may be obtained by unpredictable disturbances or the use of an approximated model, e.g., the use of a double integrator model to describe the nonlinear vehicle dynamics. The unknown change of state constraints, on the other hand, is primarily due to incomplete knowledge of the behavior of the environment around the local subsystem, e.g., the vehicle to be overtaken may suddenly start to accelerate.

Starting from the uncertainties affecting local plants, a robust control strategy is proposed first in this chapter, ensuring that the given control objective can always be achieved despite the uncertainties. This is realized by constructing a nominal HA (without uncertain terms affecting the flow and reset functions) as in the last chapter, by using a „robust invariant tube". Then, the robust control problem of the uncertain HA is cast into the optimal control problem of the nominal one.

Next, the local controller synthesis problem with a time-varying environment is

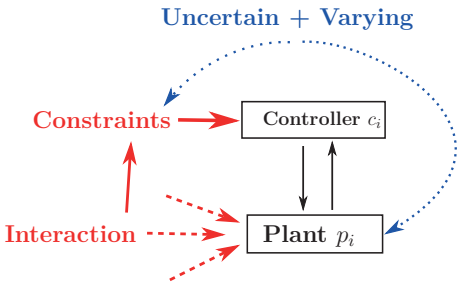

Figure 3.1.: Uncertainty may arise in both the local plant and the environment around.

considered. In this case, the MPC strategy is applied to control the local subsystem, and important properties such as recursive feasibility and stability have to be (and are) investigated in this chapter, especially for two types of changes of the environment:

- If no detailed model is available to describe the change of the environment precisely, the maximal change of the environment within one time step may still be known to be bounded, and it may be possible to conservatively estimate the bound a priori.

- If a dynamic model of the change of the environment exists but is subject to uncertainties, the model can be used to specify the change of the environment conservatively.

For the two cases, means to guarantee the above named properties of MPC are presented. However, if the over-approximation of the uncertain changes is large, quite conservative control strategies may result, and no feasible control action may be found in the worst case. To overcome this issue, a method using a penalty term (added to the cost function) to preserve the desired properties is proposed, and no conservativeness with respect to reducing feasible solution space is introduced in this approach.

Eventually, a human-robot collaboration example is considered, in which the robot dynamics is affected by additive disturbances, and the prediction of human motion also contains uncertainties. By applying the methods proposed in this chapter simultaneously, it is shown that the desired task of the robot can be accomplished while the safety of the humans is ensured. This chapter is based on results published partly already in [100, 98].

3.1. Robust Point-to-Set Control with Uncertain Dynamics

An important extension to the HA introduced in the last chapter is that the continuous as well as discrete dynamics (more precisely the reset functions) are subject to uncertainties. So far, robust control of hybrid systems with focus on optimization-based approaches, has only considered simpler system classes: In [79], PWA systems with additive disturbances were under study, and robust controllers were obtained by backward computation of robust controllable sets. The work in [91] extended the idea of backward computation for the case of parametric model uncertainties. For the same class of PWA systems, the authors in [115] and [143] aim at setting up min-max problems, or to employ multi-parametric linear programming (see [32]). Specific to the considered class of hybrid systems, the work in [113] proposes a robust switching law for the case when no continuous input is applied.

In this section, however, both parametric uncertainties and additive disturbances affecting the continuous dynamics are considered, as well as uncertain reset functions assigned to transitions. For the task of computing robust control trajectories, ideas that were used for simpler systems in [86, 110, 67, 136] were employed and extended, namely to construct reachable tubes (robust invariant sets) around nominal trajectories. Through a reduction of the size (or sometimes called as tightening) of the invariants and guard sets of the hybrid systems by use of such tubes, and by optimizing the nominal trajectories for a obtained nominal hybrid system with tightened invariants and guards, it is shown that the optimal point-to set control tasks can be solved reliably despite the presence of the uncertainties.

The class of uncertain hybrid systems $HA^u = (T, U, X, Z, I, \mathcal{T}, G, V, r^u, f^u)$, which has been introduced in [100], is extended compared to the one in Sec. 2.1, which contains:

- the discrete time domain $T = \{t_k \mid k \in \mathbb{N} \cup \{0\}, \Delta \in \mathbb{R}^{>0} : t_k := k \cdot \Delta\}$;
- the continuous inputs $u \in U \subseteq \mathbb{R}^{n_u}$,
- the continuous states $x \in X \subseteq \mathbb{R}^{n_x}$;
- the finite set of discrete states $Z = \{z_{(1)}, \cdots, z_{(n_z)}\}$;
- a set $I = \{I_{(1)}, \ldots, I_{(n_z)}\}$ of invariants $I_{(i)}$;
- the finite set of transitions $\mathcal{T} \subseteq Z \times Z$, where the transition from $z_{(i)}$ to $z_{(j)}$ is denoted by $\tau_{(i,j)} \in \mathcal{T}$;
- the set G of guard sets, of which any element $G_{(i,j)}$ is assigned to $\tau_{(i,j)} \in \mathcal{T}$;
- the finite set V of discrete inputs, where any element $v_{(i,j)} \in \{0, 1\}$ in V refers to one transition $\tau_{(i,j)} \in \mathcal{T}$;
- an uncertain reset function $r^u : \mathcal{T} \times X \to X$ to randomly update the continuous states x upon a transition $\tau_{(i,j)} \in \mathcal{T}$ according to the following scheme:

$$\hat{x} = F_{(i,j)} \cdot x + e_{(i,j)} \tag{3.1}$$

with a matrix $F_{(i,j)}$ and a vector $e_{(i,j)}$; $F_{(i,j)}$ is randomly taken from a polytope $\mathcal{F}_{(i,j)}$ defined by:

$$\mathcal{F}_{(i,j)} = \left\{ F_{(i,j)} = \sum_{l=1}^{\rho_{(i,j)}} \gamma_l \cdot \mathcal{F}_{(i,j)}^{(l)}, \ \sum_{l=1}^{\rho_{(i,j)}} \gamma_l^{\geq 0} = 1 \right\} \tag{3.2}$$

with $\mathcal{F}_{(i,j)}^{(l)}$ denoting the vertices of $\mathcal{F}_{(i,j)}$, and $\rho_{(i,j)}$ is the number of these vertices, and $\gamma_l \in [0, 1]$. The vector $e_{(i,j)}$ is randomly taken from a polytopic set $\mathcal{E}_{(i,j)}$;

- an uncertain flow function $f^u : X \times U \times Z \to X$ defining the discrete-time continuous-valued dynamics to:

$$x_{k+1} = A_{(i)} \cdot x_k + B_{(i)} \cdot u_k + w_{(i),k} \tag{3.3}$$

with $x_{k+1} := x(t_{k+1})$ and $z_{(i)} \in Z$. The matrices $A_{(i)}$ and $B_{(i)}$ are randomly taken from polytopic sets $\mathcal{A}_{(i)}$ and $\mathcal{B}_{(i)}$ with numbers of vertices given by $\rho_{(i,A)}$ and $\rho_{(i,B)}$:

$$\begin{aligned} \mathcal{A}_{(i)} &= \left\{ A_{(i)} = \sum_{l=1}^{\rho_{(i,A)}} \gamma_l \cdot \mathcal{A}_{(i)}^{(l)}, \sum_{l=1}^{\rho_{(i,A)}} \gamma_l = 1 \right\}, \\ \mathcal{B}_{(i)} &= \left\{ B_{(i)} = \sum_{l=1}^{\rho_{(i,B)}} \gamma_l \cdot \mathcal{B}_{(i)}^{(l)}, \sum_{l=1}^{\rho_{(i,B)}} \gamma_l = 1 \right\}, \end{aligned} \tag{3.4}$$

and with coefficients $\gamma_l \in [0, 1]$. The additive disturbances $w_{(i),k}$ are taken from a set $\mathcal{W}_{(i)}$ containing the origin.

The set of admissible executions of the model HA^u considering the uncertain components of the continuous dynamics and reset functions is defined as follows:

Definition 3.1. *(**Admissible execution of the uncertain model** HA^u) Let a finite time domain $T_N = \{0, 1, \ldots, N\} \subset T$ and an initial hybrid state (x_0, z_0) with $z_0 := z_{(s)} \in Z$, $x_0 \in I_{(s)}$ be given. Then, for given input sequences $\phi_u = \{u_0, u_1, \cdots, u_{N-1}\}$ and $\phi_v = \{v_0, v_1, \cdots, v_{N-1}\}$, an admissible execution is a pair of state sequences $\phi_x = \{x_0, x_1, \cdots, x_N\}$ and $\phi_z = \{z_0, z_1, \cdots, z_N\}$ complying to the following rules: for $k \in \{0, \ldots, N-1\}$, $x_k \in \phi_x$, and $z_k = z_{(i)} \in \phi_z$, the successor states $x_{k+1} \in \phi_x$ and $z_{k+1} \in \phi_z$ must satisfy:*

- *if $v_k = 0$, then there must exist $A_{(i)} \in \mathcal{A}_{(i)}$, $B_{(i)} \in \mathcal{B}_{(i)}$, and $w_{(i),k} \in \mathcal{W}_{(i)}$ to obtain $x_{k+1} \in I_{(i)}$ according to* (3.3) *and $z_{k+1} = z_{(i)}$;*
- *if $v_k = v_{(i,j)} = 1$ and if $A_{(i)} \in \mathcal{A}_{(i)}$, $B_{(i)} \in \mathcal{B}_{(i)}$, and $w_{(i),k} \in \mathcal{W}_{(i)}$ exist to obtain an intermediate state $x' = A_{(i)} \cdot x_k + B_{(i)} \cdot u_k + w_{(i),k}$, $x' \in G_{(i,j)}$ according to* (3.3)*, then there must exist $F_{(i,j)} \in \mathcal{F}_{(i,j)}$ and $e_{(i,j)} \in \mathcal{E}_{(i,j)}$ to have $x_{k+1} = F_{i,j} \cdot x' + e_{i,j} \in I_{(j)}$ according to* (3.1)*, and $z_{k+1} = z_{(j)}$.*

Obviously, for given inputs in time k, any value of the uncertain components $A_{(i)}$, $B_{(i)}$, $w_{(i)}$, $F_{(i,j)}$, and $e_{(i,j)}$ may contribute to and determine the hybrid successor state, as long as the necessary containment in the invariants and guard sets are observed – thus, any controller designed for a model of type HA^u must consider the complete set of possible executions according to Def. 3.1.

Turning to the control task to be addressed within this section, now consider the optimal transfer of HA^u from an initial hybrid state (x_0, z_0) into a set of goal states while taking the uncertainties into account. For the hybrid goal states, assume the pair (X_g, z_g) with terminal discrete state $z_g \in Z$ and terminal continuous goal set

$X_g \subseteq I_g$, as well as a continuous state $x_g \in X$ denote the volumetric center of X_g. Furthermore, let a cost functional $\mathcal{J}(x_0)$ be specified to quantify the performance of transferring the system into the goal within a finite time domain of T_N.

Definition 3.2. *(**Point-to-set control task for** HA^u) For an initial hybrid state (x_0, z_0) of HA^u, a time domain T_N, as well as a set of goal states (X_g, z_g), find the pair of sequences of continuous inputs $\phi_u = \{u_0, u_1, \cdots, u_{N-1}\}$ and discrete inputs $\phi_v = \{v_0, v_1, \cdots, v_{N-1}\}$ so that:*

- *the resulting pair of state sequences $\phi_x = \{x_0, x_1, \cdots, x_N\}$ and ϕ_z=$\{z_0, z_1, \cdots, z_N\}$ satisfy Def. 3.1,*
- *the terminal states satisfy $x_N \in X_g$ and $z_N = z_g$,*
- *and the cost functional $\mathcal{J}(x_0)$ is minimized.*

Note that an equivalent problem for the case without uncertainties was already addressed in the last chapter, proposing a particular structure to take care of the theoretically exponential increase of the number of possible ϕ_z and ϕ_v over N. For the variant of hybrid systems with uncertainties considered in this section, the additional challenge is to guarantee that the selected pair of ϕ_u and ϕ_v realizes the path into (X_g, z_g) for all possible realizations of the uncertainties. To succeed in this task, the principles inspired by the so-called *tube-based predictive control* for continuous-valued dynamics [86, 110] are employed. Along this line, it is shown that, based on tubes of reachable sets, the control problem in Def. 3.2 can be cast into a problem for a nominal model $\overline{HA}$ with deterministic flow and reset functions, to which the method in the last chapter can be applied. In addition, as has been shown in the last chapter, by either enumerating the phase sequences in advance, or not doing so, the additional time steps caused by the intermediate states can always be properly encoded in the numerical program without affecting optimality. Thus, to avoid redundancy, these additional time steps will not be considered in this section.

3.1.1. Reachability Tubes to Handle Uncertain Transitions

In order to explain the principle of using reachability tubes to robustly control HA^u, this section focuses first on a single transition as part of the sequence ϕ_z solving the problem in Def. 3.2. More precisely, an arbitrary transition $\tau_{(i,j)} \in \mathcal{T}$ is considered, and it is shown how to ensure the semantics in Def. 3.1 in terms of: 1.) the continuous state evolving inside of $I_{(i)}$ of discrete state $z_{(i)}$ (called the *pre-transition phase*), 2.) transitioning from $z_{(i)}$ to $z_{(j)}$, and 3.) further evolving inside of $I_{(j)}$ of discrete state $z_{(i)}$ (the *post-transition phase*). If this procedure is later applied to all transitions in $\mathcal{T}$, this will lead to a substitute hybrid system $\overline{HA}$, in which both flow and reset functions are without any uncertain term.

Pre-Transition Phase

First, it is assumed that the polytopic sets leading to uncertainties in both the reset function and the flow function, can be decomposed into two parts, namely a nominal part (indicated by $\bar{\bullet}$), and a disturbance set containing the origin:

$$\begin{aligned} &\mathcal{F}_{(i,j)} = \bar{F}_{(i,j)} \oplus \mathbb{F}_{(i,j)},\ \mathcal{E}_{(i,j)} = \bar{e}_{(i,j)} \oplus \mathbb{E}_{(i,j)}; \\ &\mathcal{A}_{(i)} = \bar{A}_{(i)} \oplus \mathbb{A}_{(i)},\ \mathcal{B}_{(i)} = \bar{B}_{(i)} \oplus \mathbb{B}_{(i)}. \end{aligned} \tag{3.5}$$

Here, the symbol $\oplus$ denotes the Minkowski addition (or *set addition* according to [110]), and the nominal part $\bar{\bullet}$ can be determined by, e.g., the arithmetic mean of all vertices of the corresponding polytope. Note that the sets $\mathcal{W}_{(i)}$ already contain the origin by definition and need not to be decomposed. Next, a nominal flow function as well as a nominal reset function is defined (considering the deterministic part in f^u and r^u only) to obtain the nominal continuous state and input:

$$\bar{x}_{k+1} := \bar{A}_{(i)} \cdot \bar{x}_k + \bar{B}_{(i)} \cdot \bar{u}_k; \tag{3.6}$$

$$\dot{\bar{x}} = \bar{F}_{(i,j)} \cdot \bar{x} + \bar{e}_{(i,j)}. \tag{3.7}$$

Now assume for step k that the state x_k from (3.3) and the nominal state $\bar{x}_k$ from (3.6) are located inside of the invariant $I_{(i)}$. Then, by applying a continuous input u_k and a nominal $\bar{u}_k$ in (3.3) and (3.6) respectively, the difference between x_{k+1} and $\bar{x}_{k+1}$ can be determined according to:

$$x_{k+1} - \bar{x}_{k+1} = \bar{A}_{(i)}(x_k - \bar{x}_k) + \bar{B}_{(i)}(u_k - \bar{u}_k) + w_{(i),k} + \Delta_{(A,i),k} \cdot x_k + \Delta_{(B,i),k} \cdot u_k \tag{3.8}$$

with $\Delta_{(A,i),k} \in \mathbb{A}_{(i)}$ and $\Delta_{(B,i),k} \in \mathbb{B}_{(i)}$.

Suppose further that a closed-loop feedback controller with matrix $K_i \in \mathbb{R}^{n_u \times n_x}$ is defined such that $\bar{A}_{K,(i)} := \bar{A}_{(i)} + \bar{B}_{(i)} \cdot K_i$ is stable. If x_k is measurable and u_k chosen to:

$$u_k = \bar{u}_k + K_i \cdot (x_k - \bar{x}_k), \tag{3.9}$$

then the difference between x_{k+1} and $\bar{x}_{k+1}$ in (3.8) can be written as:

$$x_{k+1} - \bar{x}_{k+1} = \bar{A}_{K,(i)}(x_k - \bar{x}_k) + w_{(i),k} + \Delta_{(A,i),k} \cdot x_k + \Delta_{(B,i),k} \cdot u_k. \tag{3.10}$$

Furthermore, as $x_k \in I_{(i)}$ and $\Delta_{(A,i),k} \in \mathbb{A}_{(i)}$, and both $I_{(i)}$ and $\mathbb{A}_{(i)}$ are polytopic, the following applies according to [39]:

$$\Delta_{(A,i),k} \cdot x_k \in Conv\left(\left\{\mathbb{A}_{(i)}^{(l)} \cdot I_{(i)}^{(q)} \mid l \in \{1,\dots,\rho_{(\mathbb{A}_{(i)})}\}\ ,\ q \in \{1,\dots,\rho_{(I_{(i)})}\}\right\}\right). \tag{3.11}$$

Here, l and q are the indices running over the $\rho_{(\mathbb{A}_{(i)})}$ vertices, or respectively $\rho_{(I_{(i)})}$ vertices of the respective polytopes, and $Conv$ is the function to determine the

convex hull over the combinations of vertices. Similarly, the value of $\Delta_{(B,i),k} \cdot u_k$ in (3.10) can be bounded by:

$$\Delta_{(B,i),k} \cdot u_k \in Conv\left(\left\{\mathbb{B}_{(i)}^{(l)} \cdot U^{(q)} \mid l \in \{1,\ldots,\rho_{(\mathbb{B}_{(i)})}\}\ ,\ q \in \{1,\ldots,\rho_{(U)}\}\right\}\right). \quad (3.12)$$

For abbreviation, the terms $Conv(\mathbb{A}_{(i)}I_{(i)})$ and $Conv(\mathbb{B}_{(i)}U)$ are used to denote the convex set on the right hand side of (3.11), and (3.12) respectively. With it, a disturbance invariant set $\mathcal{D}_i$ can be determined according to (3.10) satisfying:

$$\bar{A}_{K,(i)}\mathcal{D}_i \oplus (W_{(i)} \oplus Conv(\mathbb{A}_{(i)}I_{(i)}) \oplus Conv(\mathbb{B}_{(i)}U)) \subseteq \mathcal{D}_i. \quad (3.13)$$

The set $\mathcal{D}_i$ exists, since $\bar{A}_{K,(i)}$ is stable and contains the origin according to [110], and for the computation of $\mathcal{D}_i$, a series of methods are introduced in [78]. The relation (3.13) means that, if $x_k - \bar{x}_k \in \mathcal{D}_i$ applies, then $x_{k+1} - \bar{x}_{k+1} \in \mathcal{D}_i$ also follows from using the control law (3.9) despite all the uncertainties in the flow function, and the following holds:

Lemma 3.1. *If for two given states x_k and $\bar{x}_k \in I_{(i)}$ applies that $x_k \in \bar{x}_k \oplus \mathcal{D}_i$, then using the control law* (3.9) *implies that the relation $x_{k+1} \in \bar{x}_{k+1} \oplus \mathcal{D}_i$ holds for all $\Delta_{(A,i),k} \in \mathbb{A}_{(i)}$, $\Delta_{(B,i),k} \in \mathbb{B}_{(i)}$ and $w_{(i),k} \in \mathbb{W}_{(i)}$.* □

Note that this lemma is a direct result of (3.10) and (3.13) and the proof is thus omitted. By use of Lemma 3.1 and when denoting the Pontryagin difference (or *set subtraction* according to [110]) by $\ominus$, the following fact can also be established:

Proposition 3.1. *If the nominal state satisfies $\bar{x}_k, \bar{x}_{k+1} \in I_{(i)} \ominus \mathcal{D}_i$, and $x_k \in \bar{x}_k \oplus \mathcal{D}_i$, and if the nominal input satisfies $\bar{u}_k \in U \ominus K_i\mathcal{D}_i$, then there always exists $u_k := \bar{u}_k + K_i(x_k - \bar{x}_k) \in U$ such that $x_{k+1} \in I_{(i)}$ for all $\Delta_{(A,i),k} \in \mathbb{A}_{(i)}$, $\Delta_{(B,i),k} \in \mathbb{B}_{(i)}$ and $w_{(i),k} \in \mathbb{W}_{(i)}$.* □

The proof of Proposition 3.1 follows the lines in [110] and is briefly sketched in the following:

Proof. First of all, according to the property of operation $\oplus$ and $\ominus$ introduced in [6], one knows that the relation $(I_{(i)} \ominus \mathcal{D}_i) \oplus \mathcal{D}_i \subseteq I_{(i)}$ always hold. This also implies that, if the nominal input satisfies $\bar{u}_k \in U \ominus K_i\mathcal{D}_i$, then the relation $\bar{u}_k \oplus K_i\mathcal{D}_i \subseteq (U \ominus K_i\mathcal{D}_i) \oplus K_i\mathcal{D}_i \subseteq U$ must apply. Now, as relation $x_k - \bar{x}_k \in \mathcal{D}_i$ applies, and the input satisfies $u_k = \bar{u}_k + K_i(x_k - \bar{x}_k) \in \bar{u}_k \oplus K_i\mathcal{D}_i \subseteq U$ according to the control law (3.9), the relation $x_{k+1} \in \bar{x}_{k+1} \oplus \mathcal{D}_i$ thus must hold despite all uncertainties according to Lemma 3.1. Then, as relation $\bar{x}_{k+1} \in I_{(i)} \ominus \mathcal{D}_i$ applies according to the given setting, one can ensure that $\bar{x}_{k+1} \oplus \mathcal{D}_i \subseteq (I_{(i)} \ominus \mathcal{D}_i) \oplus \mathcal{D}_i \subseteq I_{(i)}$ always holds. Thus, the desired relation $x_{k+1} \in \bar{x}_{k+1} \oplus \mathcal{D}_i \subseteq I_{(i)}$ must apply. □

This proposition indicates that, as long as the evolution of the nominal $\bar{x}_k$ lies inside of $I_{(i)} \ominus \mathcal{D}_i$ (i.e. inside of a tightened invariant), and if the nominal continuous

input $\bar{u}_k$ is selected from set $U \ominus K_i\mathcal{D}_i$ (i.e. from a tightened input set), then one can always find $u_k \in U$ to obtain the next continuous state x_{k+1} inside of $I_{(i)}$ despite all the uncertainties. An exemplary evolution is illustrated in Fig. 3.2. In addition, as all the sets in (3.13) are polytopic, the computation of $\mathcal{D}_i$ can be assumed to be tractable according to [28]. Furthermore, it is desired that $\mathcal{D}_i$ is as small as possible, in order to reduce the scale of the tightening, and thus reduce the conservatism. With respect to the semantics of HA^u in Def. 3.1, the state x can be kept inside of $I_{(i)}$ during the evolution in $z_{(i)}$, by forcing the nominal state $\bar{x}$ to be inside of $I_{(i)} \ominus \mathcal{D}_i$.

The Transition Phase

Next, if the transition $\tau_{(i,j)}$ is triggered in step k, it must hold that $v_{(i,j),k} = 1$ and the following two conditions must be satisfied:

1. An intermediate state x' obtained from (3.3) must lie inside of $G_{(i,j)}$;
2. The state x_{k+1} resulting from (3.1) must be inside of $I_{(j)}$.

For the first condition, it is known from Proposition 3.1 that by applying the control law (3.9) the state x_k lies inside of a tube $\mathcal{D}_i$ around the nominal state $\bar{x}_k$ during evolution in $I_{(i)}$. As the guard $G_{(i,j)}$ is fully contained in $I_{(i)}$, the tube around the nominal state must also exist when $\bar{x}_k$ evolves inside of $G_{(i,j)}$. The Proposition 3.1 can thus be extended to:

Lemma 3.2. *By applying* (3.9)*, if* $\bar{x}_k \in G_{(i,j)} \ominus \mathcal{D}_i$ *is satisfied, then* $x_k \in G_{(i,j)}$ *holds.* □

Proof. According to Lemma 3.1, the relation $x_k \in \bar{x}_k \oplus \mathcal{D}_i$ holds when $\bar{x}_k \in I_{(i)} \ominus \mathcal{D}_i$.

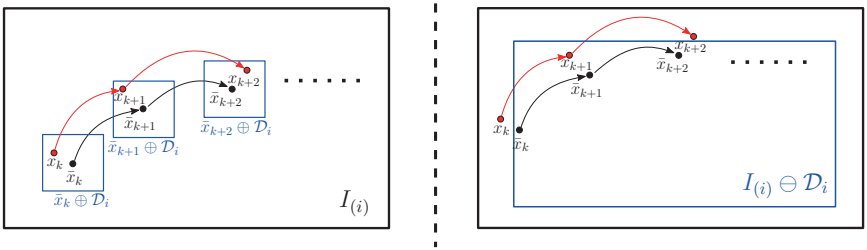

Figure 3.2.: By applying (3.9), the difference between x and $\bar{x}$ is recursively lying inside of $\mathcal{D}_i$ (denoted by the blue polytope in the left figure) despite all the uncertainties; Thus, a robust evolution of x_k in invariant $I_{(i)}$ is guaranteed when $\bar{x}_k \in I_{(i)} \ominus \mathcal{D}_i$ is satisfied, see the figure on the right.

Thus, as $\bar{x}_k \in G_{(i,j)} \ominus \mathcal{D}_i$, and $\{G_{(i,j)} \ominus \mathcal{D}_i\} \subseteq \{I_{(i)} \ominus \mathcal{D}_i\}$, the relation $x_k \in G_{(i,j)}$ must apply. □

The lemma implies that $x' \in G_{(i,j)}$ exists despite the uncertainties, if a nominal intermediate state $\bar{x}'$, as obtained from (3.7), is inside of $G_{(i,j)} \ominus \mathcal{D}_i$. Thus, the first condition is guaranteed to be satisfied by restricting the position of the nominal intermediate state $\bar{x}'$, see Fig. 3.3.

Next, due to the uncertainties on the reset function (3.1), a difference between the states $\bar{x}_{k+1}$ and x_{k+1} may be obtained according to:

$$x_{k+1} - \bar{x}_{k+1} = \bar{F}_{(i,j)}(x' - \bar{x}') + \Delta_{F_{(i,j)}} \cdot x' + \Delta_{e_{(i,j)}}, \tag{3.14}$$

where $\Delta_{F_{(i,j)}} \in \mathbb{F}_{(i,j)}$ and $\Delta_{e_{(i,j)}} \in \mathbb{E}_{(i,j)}$. By Lemma 3.2, the difference between x' and $\bar{x}'$ is bounded to $\mathcal{D}_i$. As x' must be inside of $G_{(i,j)}$, the value of $\Delta_{F_{(i,j)}} \cdot x'$ must be bounded by $Conv(\mathbb{F}_{(i,j)} G_{(i,j)})$. Hence, the difference of $\bar{x}_{k+1}$ and x_{k+1} in (3.14) is bounded by:

$$x_{k+1} - \bar{x}_{k+1} \in \bar{F}_{(i,j)} \mathcal{D}_i \oplus Conv(\mathbb{F}_{(i,j)} G_{(i,j)}) \oplus \mathbb{E}_{(i,j)}. \tag{3.15}$$

With the help of this bound, Lemma 3.2 can be extended to:

Lemma 3.3. *If the nominal state after the transition $\tau_{(i,j)}$ satisfies $\bar{x}_{k+1} \in I_{(j)} \ominus (\bar{F}_{(i,j)} \mathcal{D}_i \oplus Conv(\mathbb{F}_{(i,j)} G_{(i,j)}) \oplus \mathbb{E}_{(i,j)})$, then the relation $x_{k+1} \in I_{(j)}$ holds for the state obtained from $\tau_{(i,j)}$ despite all the uncertainties in* (3.1). □

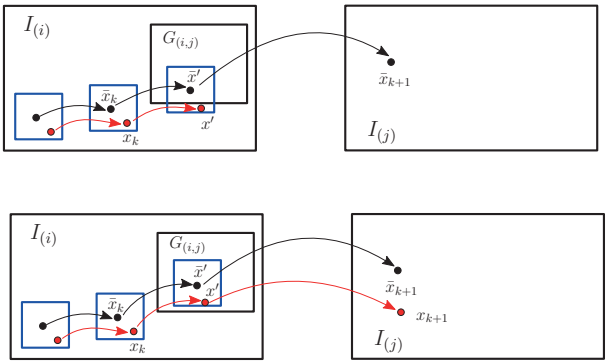

Figure 3.3.: For the case that $\bar{x}' \oplus \mathcal{D}_i$ is not fully contained in $G_{(i,j)}$ (upper part), the transition $\tau_{(i,j)}$ may not be correctly triggered due to the deviation between $\bar{x}'$ and x'; If $\bar{x}' \oplus \mathcal{D}_i \subseteq G_{(i,j)}$ (lower part), then the transition $\tau_{(i,j)}$ is guaranteed to be triggered as required by the system semantics. The blue polytope represents the disturbance invariant set $\mathcal{D}_i$.

The proof of Lemma 3.3 is similar to that of Lemma 3.2. According to Lemma 3.3, the second condition listed at the beginning of this section is satisfied by restricting the position of the nominal state $\bar{x}_{k+1}$.

The Post-Transition Phase

After reaching the discrete state $z_{(j)}$, the further evolution inside of $I_{(j)}$ has to be ensured. Obviously, this can again be achieved by applying a similar control law as in (3.9), considering a disturbance invariant set $\mathcal{D}_j$ for $z_{(j)}$. However, such a recursive robustness guaranty relies on the condition that the difference of the states x_{k+1} and $\bar{x}_{k+1}$ is in $\mathcal{D}_j$ according to Lemma 3.1. But according to (3.15), the difference is known to be inside of $\bar{F}_{(i,j)}\mathcal{D}_i \oplus Conv(\mathbb{F}_{(i,j)}G_{(i,j)}) \oplus \mathbb{E}_{(i,j)}$, which is not depending on $\mathcal{D}_j$, see Fig. 3.4. To avoid that robustness is lost at this point, the following criterion is formulated:

Definition 3.3. *(**Robust Transition**) Any transition $\tau_{(i,j)} \in \mathcal{T}$ of HA^u may only be executed, if the condition:*

$$\bar{F}_{(i,j)}\mathcal{D}_i \oplus Conv(\mathbb{F}_{(i,j)}G_{(i,j)}) \oplus \mathbb{E}_{(i,j)} \subseteq \mathcal{D}_j \tag{3.16}$$

is satisfied, otherwise it is prohibited, i.e., transition $\tau_{(i,j)}$ is removed from set $\mathcal{T}$.

Lemma 3.4. *Let the transition $\tau_{(i,j)}$ satisfy* (3.16) *and let the control law* (3.9) *be applied in both $z_{(i)}$ and $z_{(j)}$. If $x - \bar{x}$ is then bounded to $\mathcal{D}_i$ before the transition $\tau_{(i,j)}$, then $x - \bar{x}$ is also bounded to the set $\mathcal{D}_j$ after the transition $\tau_{(i,j)}$.* □

Proof. If $x_k - \bar{x}_k \in \mathcal{D}_i$ applies before the transition $\tau_{(i,j)}$, then $x_{k+1} - \bar{x}_{k+1} \in \bar{F}_{(i,j)}\mathcal{D}_i \oplus Conv(\mathbb{F}_{(i,j)}G_{(i,j)}) \oplus \mathbb{E}_{(i,j)} \subseteq \mathcal{D}_j$ holds true after the transition according to (3.15) and (3.16). Then, using Lemma 3.1, it follows that the difference will be recursively bounded to $\mathcal{D}_j$ by applying the control law (3.9) in the further steps. □

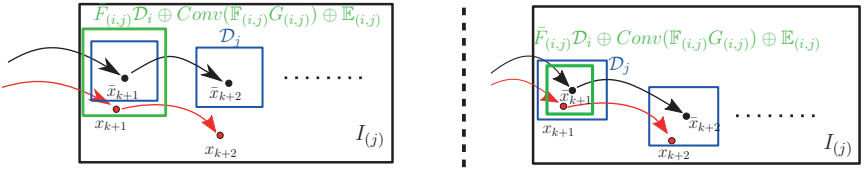

Figure 3.4.: For the case $x_{k+1} \notin \bar{x}_{k+1} \oplus \mathcal{D}_j$ due to (3.15) (the set described by the right-hand side of (3.15) is marked in green), the further evolution of the real continuous state in $z_{(j)}$ may not be bounded by the $\mathcal{D}_j$ around the nominal state (left part); for $x_{k+1} \in \bar{x}_{k+1} \oplus \mathcal{D}_j$ (right part), such a bound applies. The blue polytope denotes the disturbance invariant set $\mathcal{D}_j$.

Eventually at this stage, the sequence of system evolution in $z_{(i)}$, of the transition $\tau_{(i,j)}$, and of the further evolution in $z_{(j)}$ was successfully cast into a set of constraints on the nominal state and inputs as well as the condition (3.16). In the next section, a substitute hybrid system $\overline{HA}$ with nominal dynamics is constructed based on these constraints for all transitions in $\mathcal{T}$. It is then shown that the control task for HA^u in Def. 3.2 can be transformed into a corresponding problem for $\overline{HA}$.

3.1.2. Hybrid Systems with Nominal Dynamics

In this section, a nominal hybrid system $\overline{HA} = (T, \bar{U}, X, Z, \bar{I}, \bar{\mathcal{T}}, \bar{G}, \bar{V}, \bar{r}, \bar{f})$ based on the uncertain one HA^u is constructed. The sets T, X and Z are identical to those in HA^u, while the other components of $\overline{HA}$ are determined according to the following rules:

- a set $\bar{U} = \{\bar{U}_{(1)}, \ldots, \bar{U}_{(n_z)}\}$ of continuous input sets, where for any $z_{(i)} \in Z$, the continuous input set is $\bar{U}_{(i)} := U \ominus K_i \mathcal{D}_i$;
- a set $\bar{I} = \{\bar{I}_{(1)}, \ldots, \bar{I}_{(n_z)}\}$ of invariants, where the invariant of any discrete state $z_{(i)} \in Z$ is obtained to $\bar{I}_{(i)} := I_{(i)} \ominus \mathcal{D}_i$;
- the finite set of transitions $\bar{\mathcal{T}}$, obtained from deleting those of $\mathcal{T}$, which do not satisfy (3.16);
- the set $\bar{G}$ of guard sets contains one polytopic set $\bar{G}_{(i,j)} := G_{(i,j)} \ominus \mathcal{D}_i$ assigned to any transition $\bar{\tau}_{(i,j)} \in \bar{\mathcal{T}}$;
- the finite set $\bar{V}$ of discrete input variables, where any element $\bar{v}_{(i,j)} \in \{0, 1\}$ in $\bar{V}$ refers to one transition $\bar{\tau}_{(i,j)} \in \bar{\mathcal{T}}$;
- a deterministic (nominal) reset function $\bar{r}$, which updates the continuous state $\bar{x}'$ according to (3.7);
- a deterministic (nominal) flow function $\bar{f}$ defines the continuous dynamics according to (3.6).

Note that all the uncertain components of HA^u are excluded here, and an admissible execution of the nominal $\overline{HA}$ is defined as:

Definition 3.4. *(**Admissible Execution of** $\overline{HA}$) For $\overline{HA}$, let a finite time set $T_N = \{0, 1, \ldots, N\} \subset T$ and an initial hybrid state $(\bar{x}_0, z_0)$ satisfying $z_0 := z_{(s)} \in Z$, $\bar{x}_0 \in \bar{I}_{(s)}$ be given. For selected input sequences $\phi_{\bar{u}} = \{\bar{u}_0, \bar{u}_1, \ldots, \bar{u}_{N-1}\}$ and $\phi_{\bar{v}} = \{\bar{v}_0, \bar{v}_1, \ldots, \bar{v}_{N-1}\}$, the pair of state sequences $\phi_{\bar{x}} = \{\bar{x}_0, \bar{x}_1, \cdots, \bar{x}_N\}$ and $\phi_z = \{z_0, z_1, \cdots, z_N\}$ is* admissible*, if and only if for any $k \in \{0, \ldots, N\}$ the pair $(\bar{x}_{k+1}, z_{k+1})$ follows from $(\bar{x}_k, z_k)$, $\bar{x}_k \in \bar{I}_{(i)}$, $z_k := z_{(i)}$ according to the following semantics:*

1.) $\bar{x}' := \bar{A}_{(i)} \cdot \bar{x}_k + \bar{B}_{(i)} \cdot \bar{u}_k$,

2.) if $\bar{G}_{(i,j)} \in \bar{G}$ exists so that $\bar{x}' \in \bar{G}_{(i,j)}$ and if $\bar{v}_{(i,j),k} = 1$ applies, then $\bar{x}_{k+1} := \bar{F}_{(i,j)} \cdot \bar{x}' + \bar{e}_{(i,j)} \in \bar{I}_{(j)}$ and $z_{k+1} := z_{(j)}$; otherwise, $\bar{x}_{k+1} := \bar{x}' \in \bar{I}_{(i)}$ and $z_{k+1} := z_{(i)}$ is assigned.

Then, by defining the new hybrid goal set $(\bar{X}_g, z_g)$, where $\bar{X}_g := X_g \ominus \mathcal{D}_g$ (while assuming x_g is still the volumetric center of $\bar{X}_g$), as well as assigning $\bar{x}_0 := x_0$, the following cost functional $\mathcal{J}(x_0)$ is selected:

$$\mathcal{J}(x_0) = \sum_{k=1}^{N} \{(\bar{x}_k - x_g)^{\mathrm{T}} Q (\bar{x}_k - x_g) + \bar{u}_{k-1}^{\mathrm{T}} R \bar{u}_{k-1}\} + q_g \cdot N_g, \tag{3.17}$$

where Q and R are semi-positive-definite weighting matrices, and $q_g \in \mathbb{R}^{\geq 0}$. The variable $N_g := \min\{k \in \{1, \ldots, N\} \mid \bar{x}_k \in \bar{X}_g, z_k = z_g\}$ still encodes the first point of time at which the continuous state has reached the goal set. The control problem of the nominal $\overline{HA}$ can then be defined as:

Definition 3.5. *For $\overline{HA}$ initialized to $(\bar{x}_0, z_0)$, $z_0 := z_{(s)}$, let a time set $T_N = \{0, 1, \ldots, N\}$ and a goal $(\bar{X}_g, z_g)$ be given. Then, determine input sequences $\phi_{\bar{u}}^*$ and $\phi_{\bar{v}}^*$ as the solution of:*

$$\begin{aligned}
&\min_{\phi_{\bar{u}}, \phi_{\bar{v}}} \mathcal{J}(x_0) \\
&\textit{s.t.: for all } k \in \{0, \ldots, N-1\}: \\
&\quad \phi_{\bar{u}} \textit{ with } \bar{u}_k \in \bar{U}_{(i)}, \textit{ when } \bar{x}_k \in \bar{I}_{(i)}, \ \forall\, z_{(i)} \in Z, \\
&\quad \phi_{\bar{v}} \textit{ with } \bar{v}_{(i,j),k} \in \{0, 1\}; \\
&\quad \phi_{\bar{x}}, \phi_z \textit{ admissible for } \overline{HA}; \\
&\quad \bar{x}_N \in \bar{X}_g, z_N = z_g.
\end{aligned}$$

If now the parameters of the initial and goal sets as well as that of the cost functional in Def. 3.5 are chosen identical to that of Def. 3.2, a solution of the previous one can be referred to the latter problem, see below. In addition, the problem of Def. 3.5 is of the same type as the one in Problem 2.1 in the last chapter, which implies it can be solved by using the same methods proposed there, i.e., by solving an MIQP problem in the form of (2.1). At last, the following conclusion is established:

Lemma 3.5. *If, for the same parameterization of Def. 3.2 and Def. 3.5, a feasible solution is obtained for the problem in Def. 3.5, then this solution also provides a feasible solution to the control task in Def. 3.2.* □

Proof. For the initial continuous state, the difference $\bar{x}_0 - x_0 = 0 \in \mathcal{D}_s$ applies (up to this point, the Lemma above also applies when $\bar{x}_0 \neq x_0$, as long as $\bar{x}_0 - x_0 \in \mathcal{D}_s$ is satisfied). The optimized inputs $\bar{u}_k^*$ in ϕ_u^* obtained from the solution of the problem in Def. 3.5 are all selected from the tightened input sets $\bar{U}$. The optimal

continuous states $\bar{x}_k^*$ in ϕ_x^* are all located in the tightened invariant sets $\bar{I}$, as well as the transitions to be executed in ϕ_z^* are all satisfying the condition (3.16). Thus, according to Proposition 3.1 as well as Lemma 3.2, 3.3 and 3.4, the satisfaction of the semantics in Def. 3.1 is always ensured during the evolution in each z_k^*, as well as for each transition in ϕ_z^* when using the through control law (3.9). In addition, for the last state $\bar{x}_N^*$ in $\phi_{\bar{x}}^*$, as $x_N - \bar{x}_N^* \in \mathcal{D}_g$ applies according to Lemma 3.1, and $\bar{x}_N^* \in X_g \ominus \mathcal{D}_g$ applies according the last constraint in the problem in Def. 3.5, it is ensured that $x_N \in X_g$. □

Accordingly, by controlling the original HA^u over the time domain T_N, one only has to: 1.) solve the problem in Def. 3.5; 2.) assign $v_{(i,j),k} := \bar{v}_{(i,j),k}^*$ and calculate u_k according to (3.9) in each step k. Then the goal states (X_g, z_g) are ensured to be reached at the end of the horizon, while satisfying the semantics in Def. 3.1.

3.1.3. Numerical Examples

The uncertain HA^u considered here for illustration of the procedure consists of four discrete states with a set of possible transitions, as shown in Fig. 3.5. The invariants of the model are marked in red, the guards are in green, and the selected terminal set $X_g \subseteq I_{(2)}$ is marked in yellow. The uncertain flow function f^u and reset function r^u of the states and transitions are parametrized with different bounds of uncertainties. The initial state is $x_0 = [10, 4.5]^{\mathrm{T}} \in I_{(1)}$ and a horizon of $N = 20$ is available to realize the transition from the initial state into the goal set. The disturbance invariant set $\mathcal{D}_i$ should be chosen as small as possible in order to reduce the conservativeness by constructing the nominal $\overline{HA}$, i.e., compare the size of each invariant and guard in Fig. 3.5 and 3.6. Thus, by requiring $\mathcal{D}_i$ to be polytopic, the smallest $\mathcal{D}_i$ is obtained by using the minimum time controller K_i according to [86], and is computed with the help of the Matlab Invariant Set Toolbox [78]. When

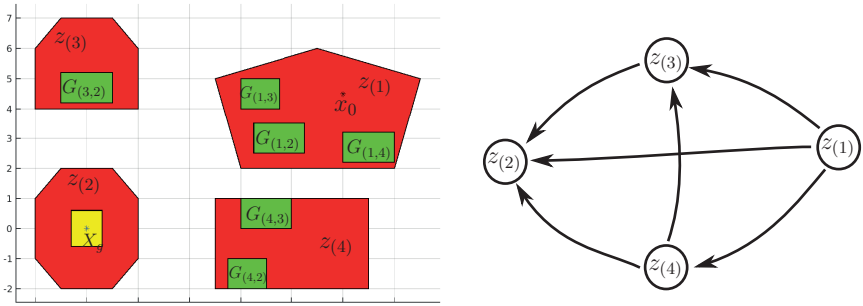

Figure 3.5.: Relevant sets and the transitions of the uncertain model HA^u.

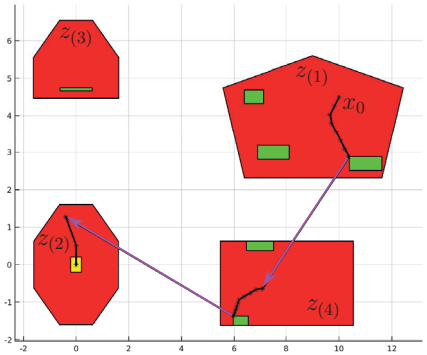

Figure 3.6.: The nominal $\overline{HA}$ obtained by tightening each invariant set, guard set, and the terminal set of HA^u; The trajectory in black is the optimal nominal state sequence of the problem in Def. 3.5, and the set of magenta lines denote the resets of the continuous states for the transitions.

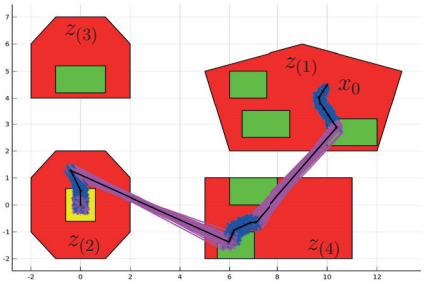

Figure 3.7.: The set of blue trajectories are the state sequences under different realizations of the uncertainties. The set of magenta points contained in the terminal set X_g are the last states of the trajectories. The black trajectory is the solution of the problem in Def. 3.5.

constructing the nominal $\overline{HA}$, each invariant and guard set, the terminal set as well as the continuous input set are tightened according to the disturbance invariant set, see Fig. 3.6. By evaluating the condition (3.16) for each transition in HA, only the transition $\tau_{(1,2)}$ fails to satisfy it, and is thus prohibited. Thereafter, by solving the problem in Def. 3.5 for the resulting nominal $\overline{HA}$, the optimal trajectory is found by first transferring from $z_{(1)}$ to $z_{(4)}$, and then to $z_{(2)}$, see Fig. 3.6. The time required for the computation is 0.66 *sec* on a 3.4GHz processor using Matlab 2015a.

Based on the solution of the problem in Def. 3.5, the original uncertain system HA^u is controlled by the control law (3.9) in each step, and for 50 different realizations of the uncertainties, the trajectories shown in Fig. 3.7 are obtained. It can be seen in this figure that all continuous states along these trajectories evolve inside of the invariant, and that the transitions are correctly triggered in the guard set, despite the uncertainties. In addition, all 50 trajectories reach the terminal set at the end of the horizon.

3.2. Model Predictive Control with Time-Varying Environments

Until now, this thesis has focused throughout on finite-horizon control problems, in which the input signals over a predefined planning horizon N are optimized once and in open-loop. There, once the optimal inputs are determined, they will be applied to the plant for the whole horizon without any re-optimization in the intermediate steps. Model Predictive Control (MPC) is a strategy similar to the finite-horizon one, but it is an online strategy that solves the finite-horizon control problem repeatedly and in closed-loop. In each discrete time step of the online process, MPC determines the optimal inputs over a finite and moving horizon in the future based on the new measurement of the system, but only applies the first step of the optimized inputs to the plant. This receding horizon scheme, in general, ensures that the input to be applied is always optimal in terms of the measured information and the limited horizon in each time instance, thus it enhances the control performance especially when time-varying components are present.

In most discussions of MPC, however, only the uncertainty of the dynamics is taken into account but not of the environment around the plant. For the case that the state constraints representing the environment remain invariant over time, it is well known under which conditions the important properties of recursive feasibility, stability, and robustness are obtained, see e.g. [109, 140, 69, 86]. Nevertheless, in recent years domains such as autonomous driving or human-robot collaboration have led to an increased interest in applying MPC also to settings, in which the constraints imposed on the system state vary over time. This arises in autonomous driving, e.g., when the controlled vehicle has to determine its path within the complement of the space occupied by other traffic participants [54, 56], or in human-robot cooperation, e.g., when the robot controller has to ensure that a robotic manipulator circumvents the regions momentarily blocked by a human operator [131, 83].

When MPC is used in these application cases, the starting point is that, in any step of the receding horizon scheme, the state constraints of the system to be controlled must also be predicted over the future time horizon. These constraints can be obtained by reachable set computations for the environment, e.g., by encoding the regions of a street topology that are potentially occupied by another car. These

reachable set computations can be executed either offline or online. The state constraints to be considered for the system to be controlled can then be determined as convex subsets of the complement of the reachable sets of all relevant entities of the environment. The requirement of convexity for the state constraints is straightforwardly used to simplify the computation of control strategies in real-time (and in response to changes of the environment).

When the MPC strategy is adapted in any time instant of a discrete-time scheme to varying constraints, obviously the question arises, whether it is possible to always find a feasible solution for the control problem. Phrased differently, it has to be determined which changes of the environment are permissible to ensure the existence of a feasible control strategy, especially when the change of the environment cannot be exactly predicted in advance. The corresponding property is known as *recursive feasibility* in predictive control [104, 29, 105], and it is one important aspect of this section. The second property to be investigated is that of *stability*, thus the question of whether the system (subject to the constraints) is certainly driven into a goal set (or towards a reference state) by the predictive controller [109, 72, 90].

While for time-invariant constraints, recursive feasibility and stability have been studied for different settings and definitions, only very little work addresses these properties for time-varying constraints. Accordingly, the conditions for ensuring these important properties of MPC subject to time-varying state constraints will be provided in this section. The discussion starts from brief review on how to ensure recursive feasibility and stability of MPC when the state constraints are time-invariant. Then, the discussion is extended to the cases when: (a) there is no suitable model available to describe the change of the environment precisely, but the maximal change of the environment within one sampling time is known to be bounded (e.g., from the maximum acceleration of a human operating close to a robot to be controlled); (b) a dynamic model of the change of the environment exists, but it is subject to uncertainties for which bounds can be conservatively specified. In both cases, the change of the environment is to be understood as being small in between two successive sampling times, and the bounds of the change can be obtained from reachability computations. Finally, it will be shown that under moderate and realistic assumptions, recursive feasibility and asymptotic stability of MPC can be preserved for the two cases.

3.2.1. Review: Time-Invariant State Constraints

This section starts from general, nonlinear dynamics of the system to be controlled, and the properties of the MPC controller addressed here are also valid for the case of hybrid dynamics. Consider the following nonlinear discrete-time difference equations describing the nonlinear dynamics:

$$x_{k+1} = f(x_k, u_k), \tag{3.18}$$

with state vector $x_k \in \mathbb{R}^{n_x}$ and input vector $u_k \in \mathbb{R}^{n_u}$. In each time step with index $k \in \mathbb{N} \cup \{0\}$, the system evolution is subject to convex state constraints $X = \{x \,|\, C \in \mathbb{R}^{n_c \times n_x}, d \in \mathbb{R}^{n_c}, x \in \mathbb{R}^{n_x} : C \cdot x \leq d\}$. The input vector u_k is bounded to a time-invariant set $U \in \mathbb{R}^{n_u}$.

Before indeed considering time-varying state constraints, i.e., X is varying over time, a review of asymptotic stability and recursive feasibility of standard MPC problems with time-invariant constraints is first provided. Let

$$\phi_{u,k} = \{u_{k|k}, u_{k+1|k}, \ldots, u_{k+H-1|k}\} \tag{3.19}$$

$$\phi_{x,k} = \{x_{k+1|k}, x_{k+2|k}, \ldots, x_{k+H|k}\} \tag{3.20}$$

denote the prediction of the input and state sequences for time index k over a prediction horizon of H steps, and assume the state constraints X remains unchanged for all $k \in \mathbb{N} \cup \{0\}$. To model state-dependent, input-dependent, and terminal costs, a standard quadratic form of the cost functional $\mathcal{J}(x_k)$ is applied:

$$\mathcal{J}(x_k) = \sum_{j=0}^{H-1} \underbrace{(x_{k+j|k}^{\mathrm{T}} Q x_{k+j|k} + u_{k+j|k}^{\mathrm{T}} R u_{k+j|k})}_{\text{step cost } L(x_{k+j|k}, u_{k+j|k})} + \underbrace{x_{k+H|k}^{\mathrm{T}} Q_g x_{k+H|k}}_{\text{terminal cost } F(x_{k+H|k})}, \tag{3.21}$$

in which Q, R, and Q_g are chosen as positive-definite weighting matrices. Furthermore, let a terminal set $X_g \subseteq X$ be selected. The problem to be solved in step k can then be defined as:

Problem 3.1.

$$\min_{\phi_{u,k}} \mathcal{J}(x_k)$$

$$\text{s.t.: } u_{k+j|k} \in U,\ j \in \{0, \ldots, H-1\}; \tag{3.22a}$$

$$x_{k+j|k} \in X,\ j \in \{1, \ldots, H-1\}; \tag{3.22b}$$

$$x_{k+H|k} \in X_g. \tag{3.22c}$$

When using the standard receding-horizon scheme of MPC, only the first step input signal $u_{k|k}$ of the solution $\phi_{u,k}^*$ of Problem 3.1 is applied in time k, then the next state x_{k+1} is measured, and the solution of Problem 3.1 is repeated for the updated data in $k+1$. However, the feasibility of Problem 3.1 in time $k+1$ must also be guaranteed, and this refers to the concept of recursive feasibility of the MPC strategy (similarly defined in [104]):

Definition 3.6. *(Recursive Feasibility) Given a compact set $\mathcal{F}$ of possible initialization x_0 of the system* (3.18)*, the MPC controller established by solving Problem 3.1 in any step k is recursively feasible if and only if for any $x_0 \in \mathcal{F}$, a feasible solution to Problem 3.1 for $k = 0$ implies the existence of a feasible solution to the problem for any $k \in \{1, 2, \cdots\}$.*

Next, the asymptotic stabilization of the system (3.18) by the MPC controller obtained from solving Problem 3.1 is defined similarly to [140]:

Definition 3.7. *(Asymptotic Stability) If there exists a function $V : \mathbb{R}^{n_x} \to \mathbb{R}$ with $V(0) = 0$ on X, such that for all $k \in \{0, 1, 2, \cdots\}$, the system (3.18) under control of the solution to Problem 3.1 satisfies $V(x_{k+1}) < V(x_k)$, then the controlled system is asymptotically stabilized to the origin.*

The function $V(x)$ here is a measure for the distance of the state x to the origin. As discussed in [109], the asymptotic stability according to Def. 3.7 can be ensured by imposing additional assumptions on the terminal set X_g, namely:

Assumption 3.1. *The terminal set $X_g \subseteq X$ is closed, and $0 \in X_g$ applies.*

Assumption 3.2. *A terminal controller $\kappa_g \in \mathbb{R}^{n_u \times n_x}$ exists so that $\kappa_g \cdot x \in U$ for all $x \in X_g$, and $f(x, \kappa_g \cdot x) \in X_g$ for all $x \in X_g$, i.e., X_g is a control invariant set of the system.*

Assumption 3.3. *The condition $F(f(x, \kappa_g \cdot x)) - F(x) + L(x, \kappa_g \cdot x) \leq 0$ with L according to (3.21) applies for all $x \in X_g$.*

Then, the following lemma applies:

Lemma 3.6. *If the Assumptions 3.1, 3.2, and 3.3 hold, then the solution to Problem 3.1 in any step k leads to a state x_{k+1}, for which Problem 3.1 again leads to a feasible solution, and the controlled system is asymptotically stabilized over k.* □

Proof. To start with recursive feasibility, assume first that the state sequence $\phi^*_{x,k} = \{x^*_{k+1|k}, x^*_{k+2|k}, \ldots, x^*_{k+H|k}\}$ is the optimal solution to Problem 3.1 in time k. Since $x^*_{k+H|k} \in X_g$ according to (3.22c) applies, there must exist a new state $x_{k+H+1|k} = f(x^*_{k+H|k}, \kappa_g \cdot x^*_{k+H|k}) \in X_g$ and $\kappa_g \cdot x^*_{k+H|k} \in U$ according to Assumption 3.2.

Furthermore, each intermediate state in the sequence $\phi^*_{x,k}$ satisfies $x^*_{k+j|k} \in X$, $\forall j \in \{1, \cdots, H\}$. Thus after moving to state $x^*_{k+1|k}$ in step $k+1$, a new candidate state sequence:

$$\phi^{cd}_{x,k+1} = \{x^*_{k+2|k}, \ldots, x^*_{k+H|k}, x_{k+H+1|k}\}$$

does also satisfy all state constraints of Problem 3.1 in $k+1$, and recursive feasibility according to Def. 3.6 follows from induction.

As for asymptotic stability, the state sequence $\phi^{cd}_{x,k+1}$ in step $k+1$ leads to costs $\mathcal{J}^{cd}(x_{k+1})$. This value constitutes an upper bound of the optimal cost: $\mathcal{J}^{cd}(x_{k+1}) \geq \mathcal{J}^*(x_{k+1})$. Furthermore, the cost difference between $\mathcal{J}^{cd}(x_{k+1})$ and $\mathcal{J}^*(x_k)$ can be calculated from:

$$\begin{aligned} &\mathcal{J}^{cd}(x_{k+1}) - \mathcal{J}^*(x_k) \\ &= F(f(x^*_{k+H|k}, \kappa_g \cdot x^*_{k+H|k})) - F(x^*_{k+H|k}) + L(x^*_{k+H|k}, \kappa_g \cdot x^*_{k+H|k}) - L(x_k, u^*_{k|k}). \end{aligned} \tag{3.23}$$

According to Assumption 3.3, the sum of the first three terms on the right-hand side of (3.23) is non-positive, thus $\mathcal{J}^{cd}(x_{k+1}) - \mathcal{J}^*(x_k) \leq -L(x_k, u^*_{k|k})$, implying also: $\mathcal{J}^*(x_{k+1}) - \mathcal{J}^*(x_k) \leq -L(x_k, u^*_{k|k})$. As the step cost L defined in (3.21) is always strictly positive outside of the origin, $\mathcal{J}^*$ decreases monotonically. If $\mathcal{J}^*(x)$ is taken as the function $V(x)$ according to Def. 3.7, then asymptotic stability of the controlled system according to Def. 3.7 is obtained[1]. □

3.2.2. Time-Varying Constraints

After review of the time-invariant case, the focus is shifted to the case of time-varying state constraints. The state constraint now takes a form of $X_k = \{x \,|\, C \in \mathbb{R}^{n_c \times n_x}, d_k \in \mathbb{R}^{n_c}, x \in \mathbb{R}^{n_x} : C \cdot x \leq d_k\}$ in each step k, indicating that its value is varying over time k. In addition, the settings that only vector d_k is indexed with k implies that only the positions of bounding hyperplanes of the polytope X_k are changing, while the orientation of the hyperplane remains unchanged.

Obviously, a new problem arises by applying MPC under time-varying state constraints X_k: the exact changes of the constraints can typically not be predicted, but the properties such as recursive feasibility and asymptotic stability of MPC must be further ensured. Regarding this problem, two different scenarios are considered in which the state constraints are not precisely known, and sufficient conditions are proposed to ensure the satisfaction of the desired properties of MPC.

Case One: Bounded Changes of the State Constraints

First, consider the case that no explicit model to predict the change of the constraints between two subsequent steps is available, but only an upper bound of the change. As indicated before, assume that the state constraint changes only with respect to the right hand sides of the inequalities, e.g., from $X_k = \{x \,|\, C \cdot x \leq d_k\}$ to $X_{k+1} = \{x \,|\, C \cdot x \leq d_{k+1}\}$, while the matrix C remains unchanged. This is useful if a translation from X_k to X_{k+1} is sufficient to model the available subset of the state-space, e.g., if an obstacle to the change of state x_k moves, and the constraint X_k is adapted by changing the vector d_k accordingly (without changing the orientation of X_k).

Let the maximal change of any component of the vector d_{k+1} compared to d_k be bounded by:

$$|d_{k+1}(i) - d_k(i)| \leq w_{i,max}, \;\; w_{i,max} \in \mathbb{R}^{\geq 0}, \tag{3.24}$$

for all $i \in \{1, \cdots, n_c\}$. The value of $w_{i,max}$ can be obtained, e.g., by evaluating the physical limits of the entity which constitutes the changing environment (e.g.

[1] The optimized cost function $\mathcal{J}^*(x)$, instead of the original cost function $\mathcal{J}(x)$, is taken as the function $V(x)$ in Def. 3.7.

the maximal acceleration of a vehicle, interacting with an autonomous car to be controlled).

Based on this information, one can obtain a conservative estimation of the change of the environment over the horizon by using the prediction:

$$X_{k+j|k} = \{x \,|\, C \cdot x \leq d_{k+j|k},\ d_{k+j|k} := d_k - j \cdot d_{max}\},\ \forall\, j \in \{1, \cdots, H\}, \quad (3.25)$$

with $d_{max} = [w_{1,max}, \cdots, w_{n_c,max}]^{\mathrm{T}} \in \mathbb{R}^{n_c}$. The set $X_{k+j|k}$ represents a conservative estimation (obtained in time k) of the future measured constraint X_{k+j} being indeed available for trajectory planning. Also define the set:

$$\phi_X = \{X_{k+1|k}, X_{k+2|k}, \ldots, X_{k+H|k}\} \quad (3.26)$$

of conservatively predicted state constraints, see also Fig. 3.8 for an illustration.

Assumption 3.4. *The set $X_{k+H|k}$ is not empty and contains the terminal set X_g, $0 \in X_g$, for all $k \in \{0, 1, 2, \cdots\}$.*

An interpretation of this assumption is that the environment does not block the path from the set $X_{k+0|k}$ into the terminal set X_g. Note that as long as the terminal set X_g is contained in $X_{k+H|k}$ for all $k \in \{0, 1, 2, \cdots\}$, then it will remain to be a control invariant set of the system by employing the terminal controller κ_g, despite the change of the state constraints [28]. In other words, if the Assumption 3.4 holds, then one does not have to re-determine the terminal set X_g with respect to the change of the environment.

Lemma 3.7. *If the condition $x_{k+j} \in X_{k+j|k}$ is satisfied for all $j \in \{1, \cdots, H\}$, then $x_{k+j} \in X_{k+j}$ applies, too.* □

Proof. According to (3.24), the relation $|d_{k+j}(i) - d_k(i)| \leq j \cdot w_{i,max}$ applies for all $j \in \{1, \cdots, H\}$, and for all $i \in \{1, \cdots, n_c\}$. This, implies also the relation $|d_{k+j} - d_k| \leq j \cdot d_{max}$ according to the definition of d_{max}. Thus, $d_k - j \cdot d_{max} \leq d_{k+j}$ holds true and implies $X_{k+j|k} \subseteq X_{k+j}$, i.e., the predicted set is always contained in the measured one. Accordingly, $x_{k+j} \in X_{k+j|k}$ implies that $x_{k+j} \in X_{k+j}$. □

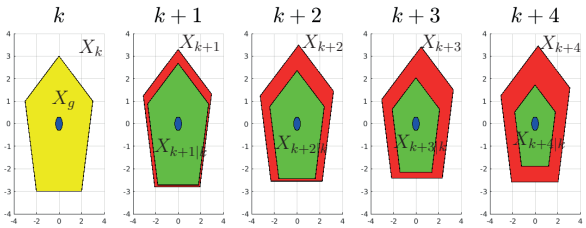

Figure 3.8.: Conservative inner approximation of X_{k+j} through $X_{k+j|k}$ for all $j \in \{1, \cdots, 4\}$ (yellow: X_k, red: X_{k+j}, green: $X_{k+j|k}$, blue: X_g).

Lemma 3.8. *Given $j \in \{1, \cdots, H\}$, then for any $j_1 \in \mathbb{N} \cup \{0\}$ with $0 \leq j_1 \leq j$, it applies that $X_{k+j|k} \subseteq X_{k+j|k+j_1}$.* □

Proof. According to (3.25), for j and j_1 with $0 \leq j_1 \leq j \leq H$, the predicted constraints $X_{k+j|k}$ and $X_{k+j|k+j_1}$ take the form of:

$$X_{k+j|k} = \{x \,|\, C \cdot x \leq d_k - j \cdot d_{max}\},$$
$$X_{k+j|k+j_1} = \{x \,|\, C \cdot x \leq d_{k+j_1} - (j - j_1) \cdot d_{max}\}.$$

Then, according to (3.24), the relation $d_{k+j_1} \geq d_k - j_1 \cdot d_{max}$ implies that:

$$d_{k+j_1} - (j - j_1) \cdot d_{max} \geq d_k - j_1 \cdot d_{max} - (j - j_1) \cdot d_{max} = d_k - j \cdot d_{max}, \quad (3.27)$$

i.e. the right hand side of the inequality for $X_{k+j|k}$ is not larger than that for $X_{k+j|k+j_1}$. Furthermore, as the matrix C in both constraints $X_{k+j|k}$ and $X_{k+j|k+j_1}$ are the same, the relation $X_{k+j|k} \subseteq X_{k+j|k+j_1}$ applies. □

Note the Lemmas 3.7 and 3.8 together establish the following facts:

- The constraint $X_{k+j|k}$ is a conservative (inner) estimation of the true set X_{k+j} by taking all possible realizations of the changes of the environment into account.
- The relation $X_{k+j+1|k} \subseteq X_{k+j|k}$ applies according to (3.25), meaning that the estimation is increasingly more conservative over j.
- The estimation of the true constraint X_{k+j} based on set X_{k+j_1} with $j_1 \leq j$, is less conservative than based on set X_k, according to Lemma 3.8.

Now, the following optimization problem is defined for step k with use of the predicted state constraints:

Problem 3.2.

$$\min_{\phi_{u,k}} \mathcal{J}(x_k)$$
$$s.t.: u_{k+j|k} \in U,\, j \in \{0, \ldots, H-1\}; \quad (3.28a)$$
$$x_{k+j|k} \in X_{k+j|k},\, j \in \{1, \ldots, H-1\}; \quad (3.28b)$$
$$x_{k+H|k} \in X_g. \quad (3.28c)$$

Lemma 3.9. *If the Assumptions 3.2, 3.3, and 3.4 hold for any $k \in \{0, 1, 2, \cdots\}$, then the solution to Problem 3.2 will establish recursive feasibility and the system* (3.18) *is asymptotically stabilized into the origin.* □

Proof. First, since $x_{k+1|k} \in X_{k+1|k}$ is ensured by constraint (3.28b) in Problem 3.2, and $X_{k+1|k} \subseteq X_{k+1}$ applies according to Lemma 3.7, the state x_{k+1} resulting from solving Problem 3.2 is guaranteed to be contained in constraint X_{k+1} despite the uncertainties. Then, consider the state constraints in Problem 3.2 for step $k+1$, which take a form of:

$$x_{k+1+(j)|k+1} \in X_{k+1+(j)|k+1},\, j \in \{1, \ldots, H-1\}; \tag{3.29a}$$

$$x_{k+1+(H)|k+1} \in X_g. \tag{3.29b}$$

Similar to the proof of Lemma 3.6, let the state sequence $\phi^*_{x,k} = \{x^*_{k+1|k}, x^*_{k+2|k}, \ldots, x^*_{k+H|k}\}$ denote the optimal solution of Problem 3.2 in k, and a candidate state sequence

$$\phi^{cd}_{x,k+1} = \{x^*_{k+2|k}, \ldots, x^*_{k+H|k}, x_{k+H+1|k}\}$$

can be obtained based on $\phi^*_{x,k}$, with $x_{k+H+1|k} = f(x^*_{k+H|k}, \kappa_g \cdot x^*_{k+H|k}) \in X_g$.

Note that for the first $H-2$ states in the candidate sequence $\phi^{cd}_{x,k+1}$, it applies that $x^*_{k+j+1|k} \in X_{k+j+1|k}$ for all $j \in \{1, \cdots, H-2\}$ according to constraint (3.28b). Based on Lemma 3.8, the relation $x^*_{k+j+1|k} \in X_{k+j+1|k} \subseteq X_{k+1+(j)|k+1}$ thus holds for all $j \in \{1, \cdots, H-2\}$.

Furthermore, the penultimate state $x^*_{k+H|k}$ in $\phi^{cd}_{x,k+1}$ is contained in X_g according to constraint (3.28c). Then, since $X_g \subseteq X_{k+H|k} \subseteq X_{k+1+(H-1)|k+1}$ applies according to Assumption 3.4 and Lemma 3.8, state $x^*_{k+H|k}$ is also contained in $X_{k+1+(H-1)|k+1}$.

Finally, the last state $x_{k+H+1|k}$ in $\phi^{cd}_{x,k+1}$ satisfies $x_{k+H+1|k} \in X_g$ according to Assumption 3.2. Thus, the candidate sequence $\phi^{cd}_{x,k+1}$ exists and satisfies all the constraints of Problem 3.2 in step $k+1$. Since in addition the corresponding input sequence satisfies the input constraint according to Assumption 3.2 and (3.28a), recursive feasibility of the MPC strategy according to Def. 3.7 is guaranteed.

The proof of *asymptotic stability* follows a path similar as in the time-invariant case, since the relation $\mathcal{J}^{cd}(x_{k+1}) - \mathcal{J}^*(x_k) \leq -L(x_k, u^*_{k|k})$ still applies, ensuring $\mathcal{J}^*(x_{k+1}) - \mathcal{J}^*(x_k) \leq -L(x_k, u^*_{k|k})$. Thus, the monotonic decrease of the cost of the function $\mathcal{J}^*$ applies, leading to asymptotic stability of the controlled system according to Def. 3.7. □

Case Two: Modeled Constraint Variation with Uncertainties

In contrast to the previous case, in which the constrained sets are shrinking in all directions over the prediction, here the changes of the constrained state set are modeled in a way that a translation towards the goal is possible, i.e., a more general type of uncertain prediction of the constrained sets is considered.

Now, assume that a model $\mathcal{M}$ for the prediction of the change of the state constraints exists, but it contains uncertainties, which may accumulate over the steps of the prediction horizon. Assume further that in the current step k, the

state constraints still have the form $X_k = \{x \,|\, C \cdot x \leq d_k\}$, but the predictions $\hat{X}_{k+j|k} = \{x \,|\, \hat{d}_{k+j|k} \in \mathbb{R}^{n_c} : C \cdot x \leq \hat{d}_{k+j|k}\}$, for $j \in \{1, \ldots, H\}$ are iteratively computed according to the model $\mathcal{M}$:

$$\hat{d}_{k+j+1|k} := \mathcal{M}(\hat{d}_{k+j|k}),\ \forall\, j \in \{0, \ldots, H-1\}, \tag{3.30}$$

where $\hat{d}_{k+0|k} := d_k$, and $\mathcal{M} : R^{n_c} \to R^{n_c}$ denotes a model for the variation of the vector $\hat{d}_{k+j|k}$. This prediction scheme implies that the state constraint predicted in the next step is derived from the one of the previous step. In addition, the model $\mathcal{M}$ requires that, for two different state constraints (in step k) with a specific bound on their difference, the predicted constraints for the succeeding step (as obtained from the model $\mathcal{M}$) is limited by the same bound:

Assumption 3.5. *For any two vectors d_k^a, $d_k^b \in R^{n_c}$ with $d_k^a \neq d_k^b$, if $|d_k^a(i) - d_k^b(i)| \leq \gamma_i$, $\gamma_i \geq 0$ applies for all $i \in \{1, \cdots, n_c\}$, then $|\mathcal{M}(d_k^a)(i) - \mathcal{M}(d_k^b)(i)| \leq \gamma_i$ also applies.*

After the prediction model is introduced, the prediction error by using this model is considered next. Unlike in the last section where the maximal change of the state constraints is considered, here the maximal prediction error by using model $\mathcal{M}$ is taken into account. It is assumed that the deviation between the predicted constraint $\hat{X}_{k+j|k}$ and the measured constraint $X_{k+j} = \{x \,|\, C \cdot x \leq d_{k+j}\}$ accumulates over the prediction step $k + j$, and satisfies the following property:

Assumption 3.6. *For the predicted constraint $\hat{X}_{k+j|k} = \{x \,|\, C \cdot x \leq \hat{d}_{k+j|k}\}$ and the measured one $X_{k+j} = \{x \,|\, C \cdot x \leq d_{k+j}\}$, each component of the vector $\hat{d}_{k+j|k}$ satisfies*

$$|\hat{d}_{k+j|k}(i) - d_{k+j}(i)| \leq j \cdot \hat{w}_{i,max},\ \hat{w}_{i,max} \in \mathbb{R}^{\geq 0}, \tag{3.31}$$

for all $i \in \{1, \cdots, n_c\}$.

This implies that the uncertainty over the prediction may linearly increase over j. Similarly to (3.24) in the last section, the requirement (3.31) is reasonable, since the upper bound of $\hat{w}_{i,max}$ can be determined offline from experiments.

To consider the maximally possible uncertainty of the predicted constraint $\hat{X}_{k+j|k}$, a tightened constraint $\tilde{X}_{k+j|k} = \{x \,|\, \tilde{d}_{k+j|k} \in \mathbb{R}^{n_c} : C \cdot x \leq \tilde{d}_{k+j|k}\}$ is determined according to:

$$\tilde{d}_{k+j|k} := \hat{d}_{k+j|k} - j \cdot \hat{d}_{max}, \tag{3.32}$$

with vector $\hat{d}_{max} = [\hat{w}_{1,max}, \cdots, \hat{w}_{n_c,max}]^{\mathrm{T}} \in \mathbb{R}^{n_c}$ (see Fig. 3.9).

Assumption 3.7. *Let the set $\tilde{X}_{k+j|k}$ be non-empty for all $k \in \{0, 1, 2, \cdots\}$ and $j \in \{1, \ldots, H\}$, and the terminal set X_g be included in $\tilde{X}_{k+H|k}$, and $0 \in X_g$.*

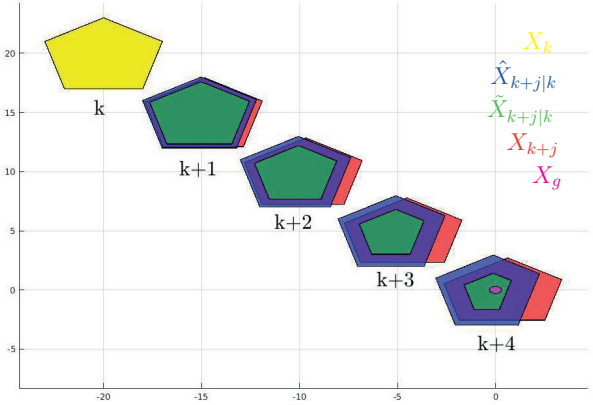

Figure 3.9.: Tightening the predicted set $\hat{X}_{k+j|k}$ according to the knowledge of $\hat{d}_{max}$ for all $j \in \{1, \cdots, 4\}$ (yellow: present X_k, blue: predicted $\hat{X}_{k+j|k}$, green: tightened $\tilde{X}_{k+j|k}$, red: measured X_{k+j}, magenta: terminal X_g).

Similar to Assumption 3.4, if the Assumption 3.7 holds, then the terminal set X_g has not to be redetermined with respect to the change of the environment[2].

Lemma 3.10. *Use the satisfaction of Assumption 3.6 as additional condition, and if $\tilde{X}_{k+j|k}$ is obtained from (3.32), where $\hat{d}_{k+j|k}$ follows from model $\mathcal{M}$ according to (3.30), and if $x_{k+j} \in \tilde{X}_{k+j|k}$ applies for all $j \in \{1, \cdots, H\}$, then $x_{k+j} \in X_{k+j}$ applies too.* □

Proof. According to (3.31) and the definition of $\hat{d}_{max}$, the relation $\hat{d}_{k+j|k} - d_{k+j} \leq j \cdot \hat{d}_{max}$ applies. Furthermore, since $\tilde{d}_{k+j|k} = \hat{d}_{k+j|k} - j \cdot \hat{d}_{max}$ holds, the relation $\tilde{d}_{k+j|k} \leq d_{k+j}$ applies, implying $\tilde{X}_{k+j|k} \subseteq X_{k+j}$. Accordingly, for all $x_{k+j} \in \tilde{X}_{k+j|k}$, relation $x_{k+j} \in X_{k+j}$ must hold, too. □

Lemma 3.11. *Given the situation in Lemma 3.10 and $j \in \{1, \cdots, H\}$, then for any $j_1 \in \mathbb{N} \cup \{0\}$ with $0 \leq j_1 \leq j$, it applies that $\tilde{X}_{k+j|k} \subseteq \tilde{X}_{k+j|k+j_1}$.* □

Proof. According to (3.31), it applies for the constraint X_{k+j_1} with $0 \leq j_1 \leq j \leq H$, that:

$$|\hat{d}_{k+j_1|k} - d_{k+j_1}| \leq j_1 \cdot \hat{d}_{max}.$$

[2]For the case that the terminal set X_g must be newly determined due to a change of the environment, the work in [106] introduced a method to homotheticly change the terminal region in order to preserve the desired invariant property.

Then, for the prediction of the constraint X_{k+j} by starting once from the constraint $\hat{X}_{k+j_1|k}$, and once starting from the constraint X_{k+j_1}, their difference is bounded by:

$$|\hat{d}_{k+j|k} - \hat{d}_{k+j|k+j_1}| \leq j_1 \cdot \hat{d}_{max}$$

according to Assumption 3.5. Now, according to (3.32), substituting $\hat{d}_{k+j|k}$ by $\tilde{d}_{k+j|k} + j \cdot \hat{d}_{max}$, and substituting $\hat{d}_{k+j|k+j_1}$ by $\tilde{d}_{k+j|k+j_1} + (j - j_1) \cdot \hat{d}_{max}$ leads to the relation:

$$(\tilde{d}_{k+j|k} + j \cdot \hat{d}_{max}) - (\tilde{d}_{k+j|k+j_1} + (j - j_1) \cdot \hat{d}_{max}) \leq j_1 \cdot \hat{d}_{max},$$

and thus to:

$$\tilde{d}_{k+j|k} - \tilde{d}_{k+j|k+j_1} \leq 0.$$

With $\tilde{d}_{k+j|k} \leq \tilde{d}_{k+j|k+j_1}$, and given that C in the constraints $\tilde{X}_{k+j|k}$ and $\tilde{X}_{k+j|k+j_1}$ are the same, the relation $\tilde{X}_{k+j|k} \subseteq \tilde{X}_{k+j|k+j_1}$ holds. □

Now, the following substitute optimization problem can be defined for step k:

Problem 3.3.

$$\begin{aligned} &\min_{\phi_{u,k}} \mathcal{J}(x_k) \\ &s.t.: u_{k+j|k} \in U,\, j \in \{0, \ldots, H-1\}; && \text{(3.33a)} \\ &\quad x_{k+j|k} \in \tilde{X}_{k+j|k},\, j \in \{1, \ldots, H-1\}; && \text{(3.33b)} \\ &\quad x_{k+H|k} \in X_g. && \text{(3.33c)} \end{aligned}$$

Lemma 3.12. *If the Assumptions 3.2, 3.3, 3.5, 3.6, and 3.7 hold, then the solution of Problem 3.3 in step k will lead to a state x_{k+1} which also satisfies the constraint X_{k+1} in step $k+1$. Furthermore, this solution implies that Problem 3.3 again has a feasible solution in step $k+1$, and the controlled system is asymptotically stabilized to the origin.* □

Proof. Following the reasoning in the proof of Lemma 3.9, as $x_{k+1|k} \in \tilde{X}_{k+1|k}$ is ensured by the constraint (3.33b) in Problem 3.3, and since $\tilde{X}_{k+1|k} \subseteq X_{k+1}$ applies according to Lemma 3.10, the state x_{k+1} resulting from the solution of Problem 3.3 in step k is guaranteed to be contained in X_{k+1}. To obtain recursive feasibility, consider the state constraints of Problem 3.3 in step $k+1$:

$$\begin{aligned} &x_{k+1+(j)|k+1} \in \tilde{X}_{k+1+(j)|k+1}, j \in \{1, \ldots, H-1\}; && \text{(3.34a)} \\ &x_{k+1+(H)|k+1} \in X_g. && \text{(3.34b)} \end{aligned}$$

If the state sequence $\phi^*_{x,k} = \{x^*_{k+1|k}, x^*_{k+2|k}, \dots, x^*_{k+H|k}\}$ is the optimal solution to Problem 3.3, let a candidate state sequence $\phi^{cd}_{x,k+1} = \{x^*_{k+2|k}, \dots, x^*_{k+H|k}, x_{k+H+1|k}\}$ be obtained from $\phi^*_{x,k}$ with $x_{k+H+1|k} = f(x^*_{k+H|k}, \kappa_g \cdot x^*_{k+H|k}) \in X_g$.

For the first $H-2$ states in $\phi^{cd}_{x,k+1}$, the inclusion $x^*_{k+j+1|k} \in \tilde{X}_{k+j+1|k}$ applies for all $j \in \{1, \cdots, H-2\}$ according to constraint (3.33b). Based on Lemma 3.11, $\tilde{X}_{k+j+1|k} \subseteq \tilde{X}_{k+1+(j)|k+1}$ holds for all $j \in \{1, \cdots, H-2\}$, and thus $x^*_{k+j+1|k} \in \tilde{X}_{k+j+1|k} \subseteq \tilde{X}_{k+1+(j)|k+1}$, for all $j \in \{1, \cdots, H-2\}$. Since the penultimate state $x^*_{k+H|k}$ in $\phi^{cd}_{x,k+1}$ is contained in X_g given (3.33c), and since $X_g \subseteq \tilde{X}_{k+H|k} \subseteq \tilde{X}_{k+1+(H-1)|k+1}$ applies according to Assumption 3.7 and Lemma 3.11, it also applies that $x^*_{k+H|k} \in \tilde{X}_{k+1+(H-1)|k+1}$. Furthermore, the last state $x_{k+H+1|k}$ in $\phi^{cd}_{x,k+1}$ satisfies $x_{k+H+1|k} \in X_g$ according to Assumption 3.2.

Hence, the sequence $\phi^{cd}_{x,k+1}$ satisfies all state constraints of Problem 3.3. In addition, the input sequence leading to $\phi^{cd}_{x,k+1}$ must satisfy the input constraints according to Assumption 3.2 and (3.33a), thus recursive feasibility of the MPC strategy according to Def. 3.6 is guaranteed.

The proof of asymptotic stability is the same as in Lemma 3.9, where the recursive feasibility implies the monotonic costs decrease of the function according to Def. 3.7. The single steps are not repeated here in detail. □

3.2.3. Discussion

Determination of w_{max} and $\hat{w}_{max}$

As mentioned earlier, the value of w_{max} can be obtained through evaluating the physical limits of the entity which constitutes the changing environment. For the determination of $\hat{w}_{max}$, which represents the maximal prediction uncertainty that may occur in a single time step, the following applies: In general, the model $\mathcal{M}$ allows to account for a more detailed representation of the change of the environment than in the first case. For example, modeling or identifying the motion of another subsystem evolving in the same space may lead to $\mathcal{M}$. If this model is evaluated by reachability analysis, see e.g. [151, 30, 7], the region of the $\mathbb{R}^{n_x}$ predicted to be occupied by the environment over the prediction horizon can be determined. The complement of this space can then be used to construct the time-varying convex state constraints used within the MPC procedure (see the case study later), and thus also to obtain the bounds w_{max}, or $\hat{w}_{max}$ respectively.

In addition, the assumption that the uncertainty w only affects the vector d in the set $X = \{x \,|\, C \cdot x \leq d\}$, is only an example to simplify the discussion. For a general form of the set X, or the uncertainties w also affect matrix C, the Minkowski computations [68, 64] can be employed to the results obtained in this section.

Requirement on the Terminal Set

Apart from common requirements on the terminal set X_g as described in Assumptions 3.1, 3.2, and 3.3 for time-invariant constraints, Assumption 3.4 brings in the condition that the change of the state constraints is bounded. Furthermore, Assumption 3.7 demands that the maximal uncertainty of the prediction model is bounded. Quite obviously, the latter one is less strict than the previous one (Assumption 3.4), since X_g only has to be contained in $\tilde{X}_{k+H|k}$ for all $k \in \{0, 1, 2, \cdots\}$ according to Assumption 3.7, whereas it has to be contained in $X_{k+j|k}$ for all $k \in \{0, 1, 2, \cdots\}$ and for all $j \in \{1, \cdots, H\}$ according to Assumption 3.4 (due to $X_{k+j+1|k} \subset X_{k+j|k}$ in (3.25)). In this respect, although the true change of the environment cannot be exactly predicted and a model $\mathcal{M}$ is needed, it is preferable to employ a prediction model rather than only using a plain bound on the change of the environment. This also allows to guarantee recursive feasibility and asymptotic stability under less strict requirements on X_g.

3.3. The Penalty-Term Approach Controlling HA with Time-Varying Environments

When the tightening method in the last section is applied to the predictive control of HA in a time-varying environment, recursive feasibility and stability of MPC can be further guaranteed in a similar way. However, as a change of the environment usually yields a change of the invariant and guard sets for HA, tightening these sets may introduce large conservativeness, as the tightening is based on the maximally possible change over all prediction steps. This would cause the optimization problem 3.2 and 3.3 to become infeasible in the worst case.

Regarding this problem, a penalty term method is proposed in this section, which intends to ensure recursive feasibility and stability of MPC through worsening the costs, instead of tightening the feasible space. This property of the penalty term method, thus enables it be employed as a complementary strategy to the tightening one in case of infeasible problems.

3.3.1. The Online Control Problem

For a time-varying hybrid system $HA_k = (T, U, X, Z, I_k, \mathcal{T}, G_k, V, r, f)$, which shares equal components with the time-invariant one in Sec. 2.1, except the following elements that change over time:

- in the set $I_k = \{I_{(1),k}, \ldots, I_{(n_z),k}\}$, the invariant of each discrete state $z_{(i)}$ in step k takes the form: $I_{(i),k} = \{x \mid n_{I_i} \in \mathbb{N}, C_{(i)} \in \mathbb{R}^{n_{I_i} \times n_x}, d_{(i),k} \in \mathbb{R}^{n_{I_i}}, x \in X : C_{(i)} \cdot x \leq d_{(i),k}\}$;

- in the set G_k, the guard for any transition $\tau_{(i,l)} \in \mathcal{T}$ in step k takes the form: $G_{(i,l),k} = \{x \mid C_{(i,l)} \in \mathbb{R}^{n_{G_{(i,l)}} \times n_x}, d_{(i,l),k} \in \mathbb{R}^{n_{G_{(i,l)}}}, x \in I_{(i),k} : \ C_{(i,l)} \cdot x \leq d_{(i,l),k}\}$.

Note that for both $I_{(i),k}$ and $G_{(i,l),k}$, a change of the environment around HA_k is assumed to only lead to a change of the right hand sides of the inequalities, i.e., only the boundaries of the polytope are changing. This setting coincides with the one in the last section to provide a good comparison. In addition, an upper bound of the change is also assumed to exist, namely:

$$||d_{(i),k+1} - d_{(i),k}||_\infty \leq w_{i,max}, \ w_{i,max} \in \mathbb{R}^{\geq 0}, \quad \forall z_{(i)} \in Z, \tag{3.35}$$

and

$$||d_{(i,l),k+1} - d_{(i,l),k}||_\infty \leq w_{(i,l),max}, \ w_{(i,l),max} \in \mathbb{R}^{\geq 0}, \quad \forall \tau_{(i,l)} \in \mathcal{T}. \tag{3.36}$$

Then, for a given initial state (x_0, z_0), a goal state (x_g, z_g), and a terminal set $X_g \subseteq I_g$, $x_g \in X_g$ (X_g is a control invariant set by applying a terminal controller κ_g, and satisfies Assumptions 3.1, 3.2, 3.3 and 3.4), the following MPC problem is solved in each step k:

Problem 3.4. *For HA_k and a hybrid state (x_k, z_k) measured in present step k, determine the continuous input sequence $\phi^*_{u,k} = \{u^*_{k|k}, u^*_{k+1|k}, \ldots, u^*_{k+H-1|k}\}$ and discrete input sequence $\phi^*_{v,k} = \{v^*_{k|k}, v^*_{k+1|k}, \ldots, v^*_{k+H-1|k}\}$ as the solution of:*

$$\begin{aligned} &\min_{\phi_{u,k}, \phi_{v,k}} \ \mathcal{J}(x_k) \\ &s.t.: \ \phi_{u,k} \text{ with } u_{k+j|k} \in U, j \in \{0, \ldots, H-1\}, \\ &\quad \phi_{v,k} \text{ with } v_{(i,l),k+j|k} \in \{0,1\}, j \in \{0, \ldots, H-1\}, \\ &\quad \phi_{x,k}, \phi_{z,k} \text{ admissible for } HA^k, \\ &\quad x_{k+H|k} \in X_g, z_{k+H|k} = z_g. \end{aligned} \tag{3.37}$$

Similar to Problem 2.1, this problem can also be reformulated into the form of (2.42) and solved by a MIP solver. Then, after the optimal solution $(\phi^*_{u,k}, \phi^*_{v,k})$ is found and the first step input $(u^*_{k|k}, v^*_{k|k})$ is applied, a new hybrid system HA_{k+1} is detected in step $k+1$. Note that if $HA_{k+1} = HA_k$ applies, the Problem 3.4 in step $k+1$ can easily be proven to be feasible through the following feasible candidates according to Lemma 3.6:

$$\begin{aligned} \phi^{cd}_{x,k+1} &= \{x^*_{k+2|k}, \ldots, x^*_{k+H|k}, (A_g + B_g \cdot \kappa_g) \cdot x^*_{k+H|k}\}, \\ \phi^{cd}_{u,k+1} &= \{u^*_{k+1|k}, \ldots, u^*_{k+H-1|k}, \kappa_g \cdot x^*_{k+H|k}\}, \\ \phi^{cd}_{v,k+1} &= \{v^*_{k+1|k}, \ldots, v^*_{k+H-1|k}, 0\}, \end{aligned} \tag{3.38}$$

where the last zero in $\phi^{cd}_{v,k+1}$ means that no discrete input is applied in step $k+H$. With the help of these feasible candidates, recursive feasibility of MPC is demonstrated. However, if $HA_{k+1} \neq HA_k$ due to a change of invariants or guards, the candidates in (3.38) may lose their feasibility in step $k+1$, since:

1. the continuous states in $\phi^{cd}_{x,k+1}$ may be located outside of the new invariants in HA_{k+1};

2. the transitions triggered by $\phi^{cd}_{v,k+1}$ may not be executable due to the new guards in HA_{k+1}.

As a result, the recursive feasibility of MPC is no longer guaranteed. With respect to this problem, the tightening method in Sec. 3.2.2 can be employed to recover the recursive feasibility. This method, in general, will first tighten each invariant and guard of HA_k according to $w_{i,max}$ and $w_{(i,l),max}$, and then let Problem 3.4 in step k be subject to HA_k with tightened sets. However, the tightening may cause some invariants and guards to become empty with the increase of the horizon H (especially the guards, since they are usually subsets of the invariants and are thus smaller). In the worst case, all transitions in HA_k are prohibited as all guards get empty, i.e., Problem 3.4 turns out to be infeasible and the control objective can not be achieved.

3.3.2. Planning with Penalty Term

In order to reduce the conservativeness caused by the tightening, an alternative method by using penalty terms is introduced in this section. The idea starts from the observation that for the candidate trajectory $\phi^{cd}_{x,k+1}$ in (3.38), if the continuous states evolve near to a boundary of I_k, or if the intermediate state x' triggering a transition is located near to a boundary of G_k, even a small change from HA_k to HA_{k+1} may render $\phi^{cd}_{x,k+1}$ to lose its feasibility, see the example in Fig. 3.10.

To maintain the feasibility of $\phi^{cd}_{x,k+1}$ in HA_{k+1} and thus to ensure recursive feasibility of the MPC strategy, $\phi^{cd}_{x,k+1}$ should be kept away from any boundary of I_k or G_k (since which boundary will change is unknown in advance). This requirement implies that the following two tasks must be additionally taken into account in Problem 3.4:

- If $x_{k+j|k}$ evolves inside of $I_{(i),k}$, then the minimal distance between $x_{k+j|k}$ and any boundary of $I_{(i),k}$ should be maximized;

- If x' triggers a transition $\tau_{(i,l)} \in \mathcal{T}$, then the minimal distance between x' and any boundary of guard $G_{(i,l),k}$ should be maximized.

These two additional goals are formulated into a penalty term of the cost function $\mathcal{J}(x_k)$ afterwards.

Remark 3.1. *Besides maximizing the minimal distance to any boundary of the invariant and guard sets, an alternative approach is to minimize the distance to the center of each set by planning $\phi^{cd}_{x,k+1}$. However, the latter choice cannot exclude the case that all states of $\phi^{cd}_{x,k+1}$ are located around the center, except one state is located near to the boundary – the feasibility of $\phi^{cd}_{x,k+1}$ may be lost due to this single state.*

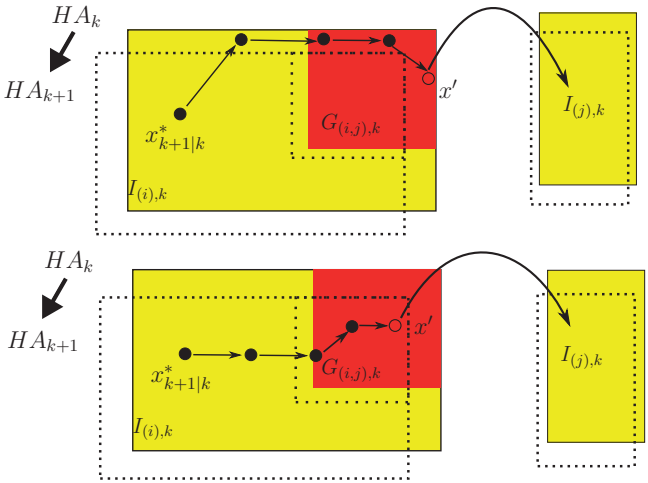

Figure 3.10.: If the hybrid system changes from HA_k (in solid line) to HA_{k+1} (in dashed line), the candidate trajectory in the upper part loses its feasibility in HA_{k+1}, since it evolves near to the boundary of $I_{(i),k}$ and the intermediate state x' is located near to the boundary of $G_{(i,j),k}$; For the same change from HA_k to HA_{k+1}, the candidate trajectory in the lower part can retain its feasibility, as it evolves away from the critical boundaries of HA_k.

Thus, the first choice is preferred as it considers the location of the most critical state in $\phi^{cd}_{x,k+1}$.

First of all, by using $C_{(i)}(h,:)$ to denote the h-th row of matrix $C_{(i)}$, and $d_{(i),k}(h)$ to denote the h-th element of vector $d_{(i),k}$, the distance $\mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))}$ between any state $x_{k+j|k} \in I_{(i),k}$ and any facet $h \in \{1, \cdots, n_{I_i}\}$ of set $I_{(i),k}$ can be computed by:

$$\mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))} = \sqrt{\frac{(C_{(i)}(h,:) \cdot x_{k+j|k} - d_{(i),k}(h))^2}{C_{(i)}(h,:) \cdot C^{\mathrm{T}}_{(i)}(h,:)}}. \tag{3.39}$$

See also the example of $\mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))}$ in Fig. 3.11. Then, as $C_{(i)}(h,:) \cdot x_{k+j|k} - d_{(i),k}(h) \leq 0$ applies if $x_{k+j|k} \in I_{(i),k}$, the distance in (3.39) is further reformulated into:

$$\mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))} = \frac{d_{(i),k}(h) - C_{(i)}(h,:) \cdot x_{k+j|k}}{\sqrt{C_{(i)}(h,:) \cdot C^{\mathrm{T}}_{(i)}(h,:)}}. \tag{3.40}$$

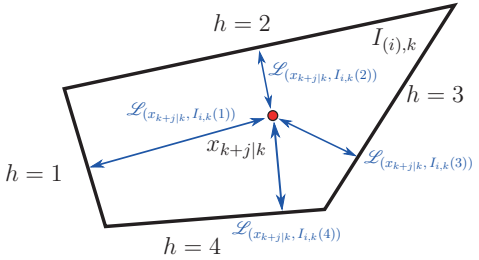

Figure 3.11.: The distance between $x_{k+j|k}$ and the four boundaries of $I_{(i),k}$ can be calculated by using (3.39).

By computing (3.40) for all facets of $I_{(i),k}$, the minimal distance between $x_{k+j|k}$ and any facet of $I_{(i),k}$ can be determined from:

$$\min_{h\in\{1,\cdots,n_{I_i}\}} \mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))}. \tag{3.41}$$

This implies that, if $x_{k+j|k}$ is a variable to be chosen, the value maximizing the minimal distance to any facet of $I_{(i),k}$ can be determined by:

$$\begin{aligned} &\max_{x_{k+j|k}\in I_{(i),k}} \alpha_{I_{(i),k+j|k}} \\ &\text{s.t.: } \alpha_{I_{(i),k+j|k}} \leq \mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))},\ \forall h \in \{1,\cdots,n_{I_i}\}. \end{aligned} \tag{3.42}$$

Here, $\alpha_{I_{(i),k+j|k}}$ is only a slack variable for the maximization. Similarly, for the intermediate state x' in guard $G_{(i,l),k}$, the value maximizing the minimal distance to any facet of $G_{(i,l),k}$ can be determined by:

$$\begin{aligned} &\max_{x'\in G_{(i,l),k}} \alpha_{G_{(i,l),k+j|k}} \\ &\text{s.t.: } \alpha_{G_{(i,l),k+j|k}} \leq \mathscr{L}_{(x',\, G_{(i,l),k}(h))},\ \forall h \in \{1,\cdots,n_{G_{(i,l)}}\}, \end{aligned} \tag{3.43}$$

where

$$\mathscr{L}_{(x',\, G_{(i,l),k}(h))} = \frac{d_{(i,l),k}(h) - C_{(i,l)}(h,:)\cdot x'}{\sqrt{C_{(i,l)}(h,:)\cdot C^{\mathrm{T}}_{(i,l)}(h,:)}}. \tag{3.44}$$

Now, the two tasks above are cast into the numerical programs (3.42) and (3.43), which can be embedded into Problem 3.4. However, the computation in (3.42) and (3.43) require that $x_{k+j|k}$ is contained in a certain invariant $I_{(i),k}$, or x' is contained in a certain guard $G_{(i,l),k}$. This requirement, in general, cannot be guaranteed to be satisfied, since the phase sequence ϕ_p in Problem 3.4 is not known a priori. As only the set of invariants and guards, which are used by $\phi^*_{x,k}$, are relevant to the two

tasks, the binary variables in (2.42) are applied here once more, in order to encode the relevant sets.

The encoding process starts by assigning the binary variable $b_{(i),k+j|k}$ to invariant $I_{(i),k}$ in each prediction step $k+j$ according to (2.25). Then, the maximization task in (3.42) can be cast into:

$$\max \alpha_{I_{(i),k+j|k}} \tag{3.45}$$

$$\text{s.t.: } \alpha_{I_{(i),k+j|k}} \leq \mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))} + b_{(i),k+j|k} \cdot M,\ \forall h \in \{1, \cdots, n_{I_i}\}; \tag{3.46}$$

$$(b_{(i),k+j|k} - 1) \cdot M \leq \alpha_{I_{(i),k+j|k}} \leq (1 - b_{(i),k+j|k}) \cdot M; \tag{3.47}$$

$$C_{(i)} \cdot x_{k+j|k} \leq d_{(i),k} + b_{(i),k+j|k} \cdot M_i, \tag{3.48}$$

where M and M_i are factor or vector of large constants. In (3.45), if $b_{(i),k} = 0$ applies, the relation $x_{k+j|k} \in I_{(i),k}$ is enforced according to (3.48), and the remaining part of the problem turns out to be identical to (3.42). In other words, the problem (3.45) ensures that if $x_{k+j|k} \in I_{(i),k}$ applies, then the value of $\alpha_{I_{(i),k+j|k}}$ will be maximized exactly as in (3.42). For the other case of $b_{(i),k} = 1$, the constraint (3.48) is relaxed (which implies $x_{k+j|k} \notin I_{(i),k}$ according to the constraints in (2.42)), and the constraint (3.46) and (3.47) turn out to be:

$$\alpha_{I_{(i),k+j|k}} \leq \mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))} + M,\ \ \forall h \in \{1, \cdots, n_{I_i}\};$$
$$0 \leq \alpha_{I_{(i),k+j|k}} \leq 0.$$

The value of $\alpha_{I_{(i),k+j|k}}$ is thus fixed to zero through these constraints. This means that if $x_{k+j|k} \notin I_{(i),k}$, then the value of $\alpha_{I_{(i),k+j|k}}$ will constantly be equal to zero and not be maximized in (3.45).

Therefore, although the phase sequence ϕ_p in Problem 3.4 is not known in advance, the application of (3.45) ensures that only the distance to the relevant invariants is maximized. For the guard $G_{(i,l),k}$, the problem (3.43) can be modified in the same way by using a binary variable $b_{(i,l),k+j|k}$. Finally, the following MPC problem is proposed regarding the two additional tasks:

Problem 3.5. *For HA_k and a hybrid state (x_k, z_k), determine input sequences $\phi^*_{u,k} = \{u^*_{k|k}, u^*_{k+1|k}, \ldots, u^*_{k+H-1|k}\}$ and $\phi^*_{v,k} = \{v^*_{k|k}, v^*_{k+1|k}, \ldots, v^*_{k+H-1|k}\}$ as the solution of:*

$$\min_{\phi_{u,k}, \phi_{v,k}} \mathcal{J}(x_k) \underbrace{-\beta \cdot \sum_{j=1}^{H} \Big(\sum_{z_{(i)} \in Z} \alpha_{I_{(i),k+j|k}} + \sum_{\tau_{(i,l)} \in \mathcal{T}} \alpha_{G_{(i,l),k+j|k}} \Big)}_{\textit{penalty term}} \tag{3.49}$$

$$\textit{s.t.: Constraints in } (3.37); \tag{3.50}$$

$$\textit{For all } j \in \{1, \cdots, H\} \textit{ and } z_{(i)} \in Z:$$
$$\alpha_{I_{(i),k+j|k}} \leq \mathscr{L}_{(x_{k+j|k},\, I_{(i),k}(h))} + b_{(i),k+j|k} \cdot M,\ \forall h \in \{1, \cdots, n_{I_i}\}; \tag{3.51}$$

$$(b_{(i),k+j|k} - 1) \cdot M \leq \alpha_{I_{(i),k+j|k}} \leq (1 - b_{(i),k+j|k}) \cdot M; \tag{3.52}$$
$$C_{(i)} \cdot x_{k+j|k} \leq d_{(i),k} + b_{(i),k+j|k} \cdot M_i; \tag{3.53}$$

For all $j \in \{1, \cdots, H\}$ *and* $\tau_{(i,l)} \in \mathcal{T}$:
$$\alpha_{G_{(i,l),k+j|k}} \leq \mathscr{L}_{(x', G_{(i,l),k}(h))} + b_{(i,l),k+j|k} \cdot M, \, \forall h \in \{1, \cdots, n_{G_{(i,l)}}\}; \tag{3.54}$$
$$(b_{(i,l),k+j|k} - 1) \cdot M \leq \alpha_{G_{(i,l),k+j|k}} \leq (1 - b_{(i,l),k+j|k}) \cdot M; \tag{3.55}$$
$$C_{(i,l)} \cdot x_{k+j|k} \leq d_{(i,l),k} + b_{(i,l)k+j|k} \cdot M_{(i,l)}. \tag{3.56}$$

Here $\beta \in \mathbb{R}^{\geq 0}$ is the weighting factor of the two additional tasks. Compared to Problem 3.4, the original constraints in (3.37), including the original invariants and guards, must be further satisfied in the new problem, while the penalty term added here forces the obtained trajectory away from the boundaries in HA_k, in order to enhance the robustness of the candidates in (3.38). In addition, as the constraints (3.51) – (3.56) are only applied to encode the relevant invariants and guards, they do not reduce the original feasible space in Problem 3.4 (but the penalty term may worsen the control performance in unfavorable cases).

At last, it must be emphasized that the solution of Problem 3.5 can only enhance the possibility that the candidates in (3.38) remain feasible in step $k+1$, instead of providing a guaranty as when using the tightening method. This is because the extent of the uncertainties $w_{i,max}$ and $w_{(i,l),max}$ is not taken into account in this method, i.e., although the obtained trajectory is away from any boundary, it may still not be sufficient to preserve the recursive feasibility when the encountered uncertainty is large. In addition, as the value of weighting factor β affects how much effort is spent to enhance the robustness, its selection turns out to be crucial with respect to the balance between robustness guaranty and performance loss. Accordingly, this penalty term method should be employed as a complementary strategy to the tightening one, when the latter indeed leads to infeasible problems.

3.3.3. Numerical Examples

The time-varying hybrid system considered here consists of three discrete states with only one phase sequence connecting the initial and target discrete states, as shown in Fig. 3.12. The flow function f and the reset function r of the states and transitions are parametrized suitably. The continuous state in initial step is $x_0 = [12, -8.5]^{\mathrm{T}} \in I_{(0)}$ and a prediction horizon of $H = 12$ is chosen. The goal set X_g (marked in green) is a control invariant set satisfying Assumptions 3.1, 3.2, 3.3 and 3.4. The maximal change of the invariants and guards between two consecutive steps is selected to: $w_{i,max} = 0.3$, $\forall z_{(i)} \in Z$ and $w_{(i,l),max} = 0.2$, $\forall \tau_{(i,l)} \in \mathcal{T}$.

When the original Problem 3.4 is solved for initial step $k = 0$, the transitions in the obtained trajectory (as shown in Fig. 3.12) are triggered near to the boundaries of the guards. As a result, the candidates in (3.38) may lose their feasibility, if the guards do slightly change in the succeeding step. If the tightening method in Sec.

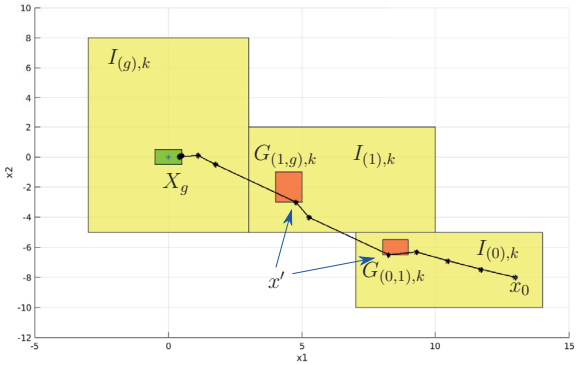

Figure 3.12.: Shown are hybrid system HA_k for $k = 0$ and the trajectory obtained from solving Problem 3.4 **without** penalty term.

3.2.2 is applied, however, the guards turn out to be empty and no feasible solution exists in the corresponding problem. In contrast, when the penalty-term method is

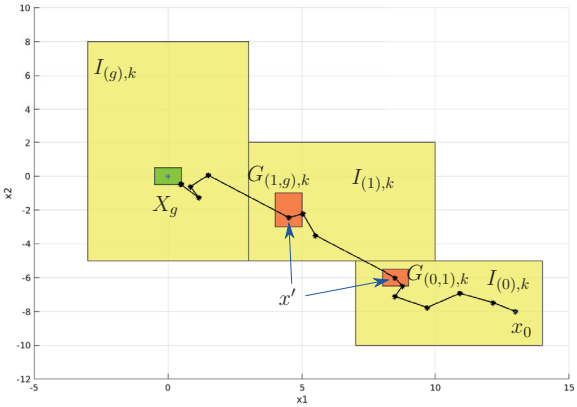

Figure 3.13.: The trajectory obtained from solving Problem 3.5 **with** penalty term: the evolution of the continuous state is no longer straight to the goal set (compared to the case without the penalty term).

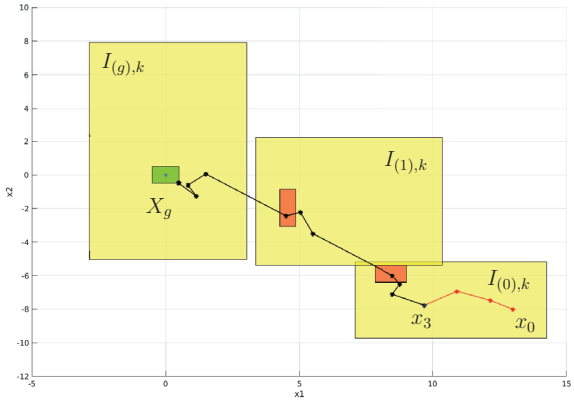

Figure 3.14.: At step $k = 3$: note the change of HA_k compared to step $k = 0$, and both coming transitions can be correctly triggered despite the change.

applied, Problem 3.5 turns out to be feasible as it shares the same feasible space as the original Problem 3.4. Transitions in the obtained trajectory are triggered away from the boundary (but the control performance are worsen as one can notice from the evolution of the trajectory), as shown in Fig. 3.13. Many numeric tests have shown that this trajectory possesses a higher robustness upon guard change. In some tests, this trajectory can even remain feasible until the terminal set is reached in the last step of the horizon, while the HA_k changes in every intermediate step $k \in \{1, \cdots, H\}$. See the selected points of time in Fig. 3.14 − Fig. 3.16, in which the trajectory $\phi^*_{x,k=0}$ obtained from Problem 3.5 is tested for all H steps. By marking the parts of $\phi^*_{x,k=0}$ that have been reached in red, one can notice that the part not yet reached can always keep being feasible despite the change of HA_k in each step.

3.4. Case Study: Robot Control with Modeling Error and Uncertain Human Movement

In this section, the MPC strategy is applied to control a robot manipulator (RM) cooperating with humans. Especially, it is assumed that the robot dynamics is affected by additive disturbances, while the prediction of human motion contains uncertainty as well. This section will show that by using the tube-based method (developed in Sec. 3.1) to deal with the former disturbances, and the tightening

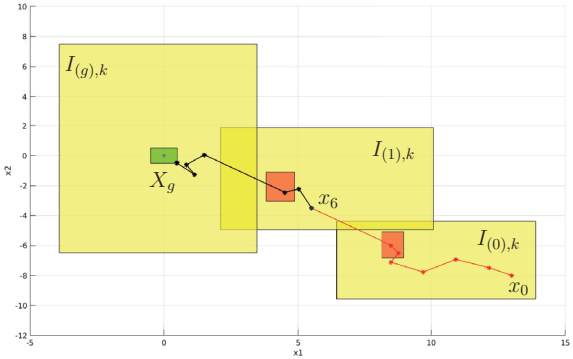

Figure 3.15.: At step $k = 6$: the last transition can still be correctly triggered despite changes in the preceding 6 steps.

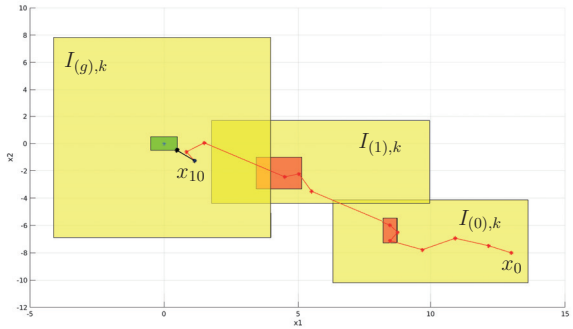

Figure 3.16.: At step $k = 10$.

method (developed in Sec. 3.2.2) for the latter uncertainty, the safety of the human can be guaranteed while the RM can accomplish its given task.

Traditionally, the workspace of the RM and humans are strictly separated for safety reasons, see the left part in Fig. 3.17. Then, the RM is allowed to share some joint workspace with humans to enhance work efficiency, but must stop when a potential collision is detected, see the middle part in Fig. 3.17. As a sudden stop may delay the whole manufacturing process (especially when several RMs are working in the

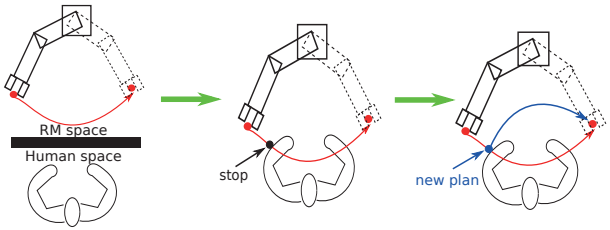

Figure 3.17.: Development of human-robot collaboration.

same product line), the MPC strategy has been observed to be preferable in recent years (see [82, 81]), so that the RM can quickly re-plan its movement (or trajectory) when a potential collision is detected but continues its operation, see the right part in Fig. 3.17.

The application of MPC, however, requires an exact dynamic model of the RM, which is often not provided by the manufacturer. But according to different control modes of the RM (provided by the manufacturer), namely:

- generate desired actuator torque;
- rotate to desired joint rotation;
- move to desired End-Effector (EE) position,

a substitute model can be derived for the MPC use. The model applied here is based on the last control mode, where:

$$\begin{bmatrix} p^E_{x,k+1} \\ p^E_{y,k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{A} \cdot \begin{bmatrix} p^E_{x,k} \\ p^E_{y,k} \end{bmatrix} + \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B} \cdot \begin{bmatrix} u_{x,k} \\ u_{y,k} \end{bmatrix} + \begin{bmatrix} w_{x,k} \\ w_{y,k} \end{bmatrix}. \tag{3.57}$$

In this discrete-time model, vector $\begin{bmatrix} p^E_{x,k} & p^E_{y,k} \end{bmatrix}^{\mathrm{T}} \in P = [-180, 180] \times [-180, 0]$ (with units mm) contains the longitudinal and lateral position of the EE. Input vector $\begin{bmatrix} u_{x,k} & u_{y,k} \end{bmatrix}^{\mathrm{T}} \in U = [-20, 20] \times [-20, 20]$ represents the distance that the EE can move per sampling time (70 ms) and per direction. In addition, a disturbance term $\begin{bmatrix} w_{x,k} & w_{y,k} \end{bmatrix}^{\mathrm{T}} \in \mathcal{W} = [-6, 6] \times [-6, 6]$ (i.e. a maximal error of 6 mm per sampling time in each direction) is also considered in this model. For the motion of each joint of the RM, they are not considered here for simplification, i.e., only the position of the EE is relevant.

To deal with the disturbance term in (3.57), a closed-loop controller $K \in \mathbb{R}^{2\times 2}$ is applied in MPC, which can bound the effect of the disturbance according to Sec. 3.1. With the help of K, a disturbance invariant set $\mathcal{D} \subseteq \mathbb{R}^2$ can be determined,

satisfying $(A + BK)\mathcal{D} \oplus W \subseteq \mathcal{D}$. The controller K ensures that the measured position $\begin{bmatrix} p^E_{x,k+j} & p^E_{y,k+j} \end{bmatrix}^{\mathrm{T}}$ in step $k+j$ will always be located in the tube $\mathcal{D}$ around the predicted nominal position $\begin{bmatrix} \overline{p}^E_{x,k+j|k} & \overline{p}^E_{y,k+j|k} \end{bmatrix}^{\mathrm{T}}$, i.e.:

$$\begin{bmatrix} p^E_{x,k+j} \\ p^E_{y,k+j} \end{bmatrix} \in \begin{bmatrix} \overline{p}^E_{x,k+j|k} \\ \overline{p}^E_{y,k+j|k} \end{bmatrix} \oplus \mathcal{D}, \ \forall j \in \{1, \cdots, H\}. \tag{3.58}$$

The nominal position is determined by the nominal dynamics:

$$\begin{bmatrix} \overline{p}^E_{x,k+1} \\ \overline{p}^E_{y,k+1} \end{bmatrix} = A \cdot \begin{bmatrix} \overline{p}^E_{x,k} \\ \overline{p}^E_{y,k} \end{bmatrix} + B \cdot \begin{bmatrix} \overline{u}_{x,k} \\ \overline{u}_{y,k} \end{bmatrix}, \tag{3.59}$$

with a tightened nominal input space $\bar{U} = U \ominus K\mathcal{D} = [-14, 14] \times [-14, 14]$, and a tightened nominal state space $\bar{P} = P \ominus \mathcal{D} = [-174, 174] \times [-174, -6]$. This implies that when the nominal dynamics (3.59) is used in MPC, the EE always locates in a tube $\mathcal{D}$ around the nominal trajectory despite the disturbances in each step.

Besides the additive disturbance, the error generated in human-motion prediction should also be taken into account. Note that the human arm position is measured by a camera system in each sampling time, e.g., Optitrack Prime 13w Cameras, see http://optitrack.com/products/prime-13w/specs.html. Based on the measured positions, the movement of the human arm is predicted according to the model (3.60). The predicted position is then over-approximated into a rectangular set P^{rec} (marked in red in Fig. 3.18), and the EE must avoid these rectangular sets when approaching to its goal position.

However, an exact prediction of the human movement is impossible – the predicted positions can deviate from the measured ones, see Fig. 3.18. Nevertheless, consider the scenario that the human arm is moving up to the left, while the pose is assumed to be constant during the movement. Thus, the width and rotation of the rectangle are assumed to be constant over time. The moving distance per sampling time is assumed to vary between $v_{min} = 5mm$ and $v_{max} = 10mm$ along each direction. In each step k, the upper-left corner $\begin{bmatrix} p^{arm}_{x,k} & p^{arm}_{y,k} \end{bmatrix}^{\mathrm{T}}$ of the rectangle P^{rec}_k is measured by the camera system (according to the assumption that the pose is constant, by knowing the upper-left corner is enough to determine the rectangle P^{rec}_k). Then, the following model $\mathcal{M}_{arm}$ is applied to predict the movement of the human arm:

$$\begin{bmatrix} \hat{p}^{arm}_{x,k+j|k} \\ \hat{p}^{arm}_{y,k+j|k} \end{bmatrix} = \begin{bmatrix} p^{arm}_{x,k} \\ p^{arm}_{y,k} \end{bmatrix} + j \cdot \begin{bmatrix} \frac{v_{min}+v_{max}}{2} \\ \frac{v_{min}+v_{max}}{2} \end{bmatrix}, \ \forall j \in \{1, \cdots, H\}, \tag{3.60}$$

where $\begin{bmatrix} \hat{p}^{arm}_{x,k+j|k} & \hat{p}^{arm}_{y,k+j|k} \end{bmatrix}^{\mathrm{T}}$ denotes the predicted position in step $k+j$. Obviously, this is a very simple model in which the moving distance of the human arm (along

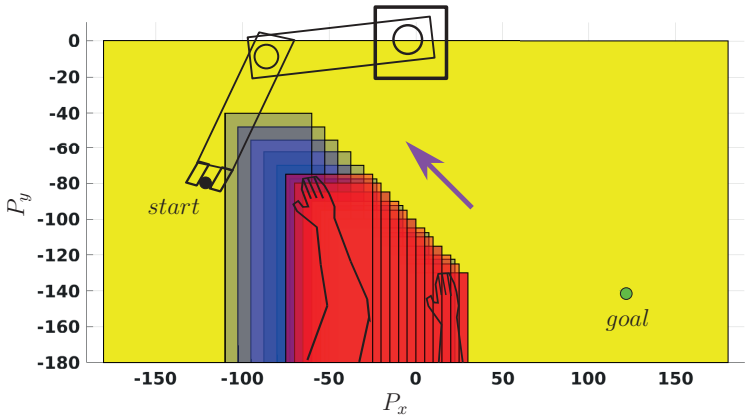

Figure 3.18.: The blue rectangles are the predicted positions over the future 12 steps. The red rectangles are the positions measured in each step.

each direction) in each step is assumed to be constant, and equals $\frac{v_{min}+v_{max}}{2}$. By employing this model, the maximal error generated in a single prediction step is bounded by:

$$\begin{bmatrix} |\hat{p}^{arm}_{x,k+j|k} - p^{arm}_{x,k+j|k}| \\ |\hat{p}^{arm}_{y,k+j|k} - p^{arm}_{y,k+j|k}| \end{bmatrix} \leq j \cdot \begin{bmatrix} \frac{v_{min}+v_{max}}{2} \\ \frac{v_{min}+v_{max}}{2} \end{bmatrix}, \ \forall j \in \{1, \cdots, H\}. \tag{3.61}$$

In addition, the following relation can also be derived according to (3.60):

$$\begin{bmatrix} |\hat{p}^{arm}_{x,k+j|k} - \hat{p}^{arm}_{x,k+j|k+1}| \\ |\hat{p}^{arm}_{y,k+j|k} - \hat{p}^{arm}_{y,k+j|k+1}| \end{bmatrix} \leq \begin{bmatrix} \frac{v_{min}+v_{max}}{2} \\ \frac{v_{min}+v_{max}}{2} \end{bmatrix}, \ \forall j \in \{2, \cdots, H\}. \tag{3.62}$$

Based on (3.61) and (3.62), one can notice that the model $\mathcal{M}_{arm}$ satisfies the Assumption 3.5 in Sec. 3.2.2, i.e., the following facts can be established:

- If the predicted rectangle $\hat{P}^{rec}_{k+j|k}$ determined by $\begin{bmatrix} \hat{p}^{arm}_{x,k+j|k} & \hat{p}^{arm}_{y,k+j|k} \end{bmatrix}^{\mathrm{T}}$, is enlarged by the maximal prediction error $j \cdot \begin{bmatrix} \frac{v_{min}+v_{max}}{2} & \frac{v_{min}+v_{max}}{2} \end{bmatrix}^{\mathrm{T}}$, then the enlarged rectangle $\tilde{P}^{rec}_{k+j|k}$ will always contain the measured rectangle in step $k+j$ according to Lemma 3.10.
- As long as the EE does not encounter into $\tilde{P}^{rec}_{k+j|k}$, or the following safety condition is satisfied:

$$\begin{bmatrix} p^{E}_{x,k+j} \\ p^{E}_{y,k+j} \end{bmatrix} \notin \tilde{P}^{rec}_{k+j|k}, \ \forall j \in \{1, \cdots, H\}, \tag{3.63}$$

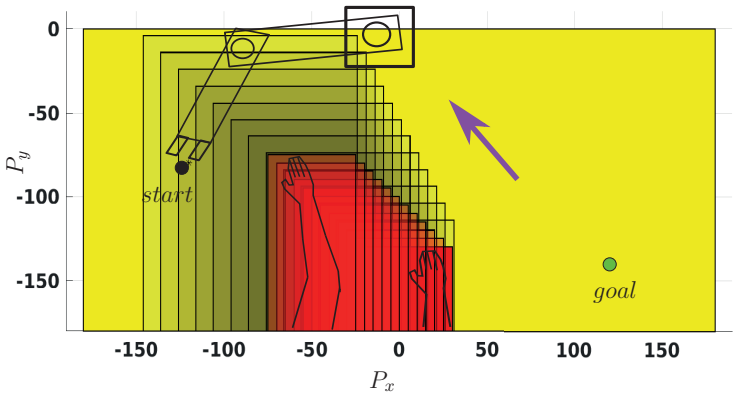

Figure 3.19.: Rectangles in black are the set $\tilde{P}^{rec}_{k+j|k} \oplus \mathcal{D}$, $\forall j \in \{1, \ldots, H\}$.

the collision with the human is guaranteed to be avoided.

Nevertheless, as the nominal dynamics (3.59) is applied to plan the movement of EE, the safety condition (3.63) is extended to the following by taking into account the additive disturbances (see the example in Fig. 3.19):

$$\begin{bmatrix} \overline{p}^E_{x,k+j|k} \\ \overline{p}^E_{y,k+j|k} \end{bmatrix} \notin \{\tilde{P}^{rec}_{k+j|k} \oplus \mathcal{D}\}, \ \forall j \in \{1, \cdots, H\}. \tag{3.64}$$

Finally, the following MPC problem is solved to plan the nominal trajectory online:

Problem 3.6. *In each step k, if the position $\begin{bmatrix} p^E_{x,k} & p^E_{y,k} \end{bmatrix}^{\mathrm{T}}$ of the EE and the position $\begin{bmatrix} p^{arm}_{x,k} & p^{arm}_{y,k} \end{bmatrix}^{\mathrm{T}}$ (and thus P^{rec}_k) of the human are measured, determine the enlarged set $\tilde{P}^{rec}_{k+j|k}$, $\forall j \in \{1, \cdots, H\}$ according to the maximal prediction error, and solve the following problem:*

$$\begin{aligned}
&\min_{\phi_{\overline{u},k}} \mathcal{J}(\begin{bmatrix} \overline{p}^E_{x,k} & \overline{p}^E_{y,k} \end{bmatrix}^{\mathrm{T}}) \\
&\text{s.t.: nominal dynamics (3.59)}, \ \forall j \in \{0, \ldots, H-1\}; \\
&\overline{u}_{k+j|k} \in \overline{U}, j \in \{0, \ldots, H-1\}; \\
&\begin{bmatrix} \overline{p}^E_{x,k+j|k} & \overline{p}^E_{y,k+j|k} \end{bmatrix}^{\mathrm{T}} \in \bar{P}, \forall j \in \{1, \ldots, H\}; \\
&\begin{bmatrix} \overline{p}^E_{x,k+j|k} & \overline{p}^E_{y,k+j|k} \end{bmatrix}^{\mathrm{T}} \notin \{\tilde{P}^{rec}_{k+j|k} \oplus \mathcal{D}\}, \forall j \in \{1, \ldots, H\}; \\
&\begin{bmatrix} \overline{p}^E_{x,k+H|k} & \overline{p}^E_{y,k+H|k} \end{bmatrix}^{\mathrm{T}} \in P_g.
\end{aligned}$$

Here P_g denotes a terminal set containing the goal position of the EE. Then, by adopting the input $u_k := \overline{u}^*_{k|k} + K(\begin{bmatrix} p^E_{x,k} & p^E_{y,k} \end{bmatrix}^{\mathrm{T}} - \begin{bmatrix} \overline{p}^E_{x,k} & \overline{p}^E_{y,k} \end{bmatrix}^{\mathrm{T}})$ in each online step k, the recursive feasibility and stability of the MPC are ensured according to Lemma 3.12, i.e., Problem 3.6 is always feasible and the goal set P_g is guaranteed to be reached if the used assumptions hold. The results illustrated in Fig. 3.20 – Fig. 3.24 show the outcome of the solution of Problem 3.6 in different steps.

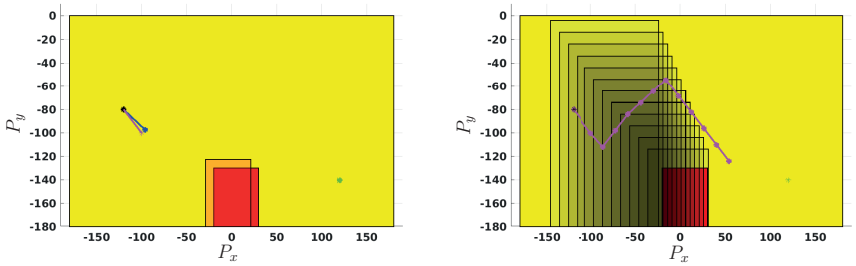

Figure 3.20.: The magenta trajectory on the right is determined by solving the Problem 3.6 in step $k = 0$. But the desired position in step $k = 1$ by executing the first step of the optimized input cannot be reached due to the disturbance, see the blue part on the left, which denotes the position of EE measured in $k = 1$. The dark red rectangles in both figures are the human position measured in step $k = 0$, while the transparent red rectangle on the left is the human position measured in step $k = 1$.

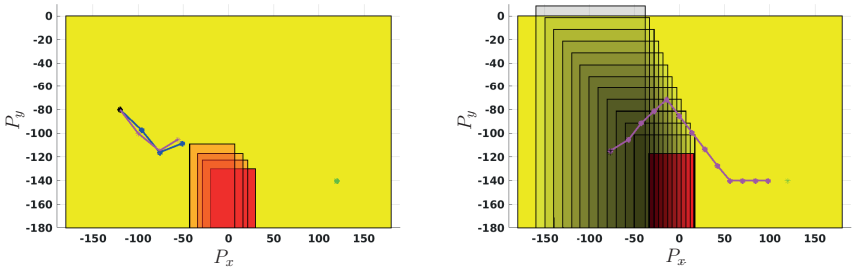

Figure 3.21.: Results in step $k = 3$.

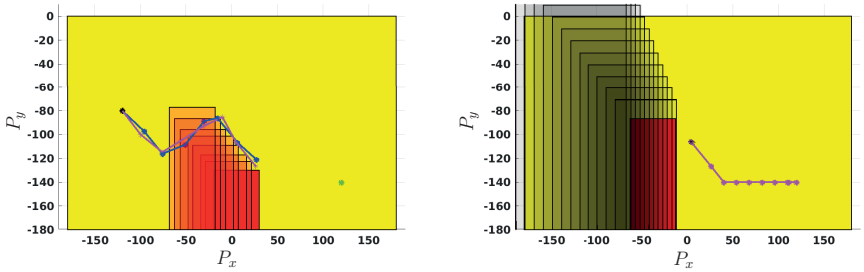

Figure 3.22.: Results in step $k = 7$.

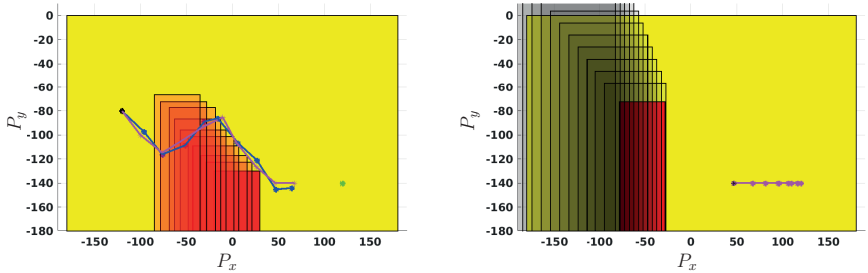

Figure 3.23.: Results in step $k = 9$.

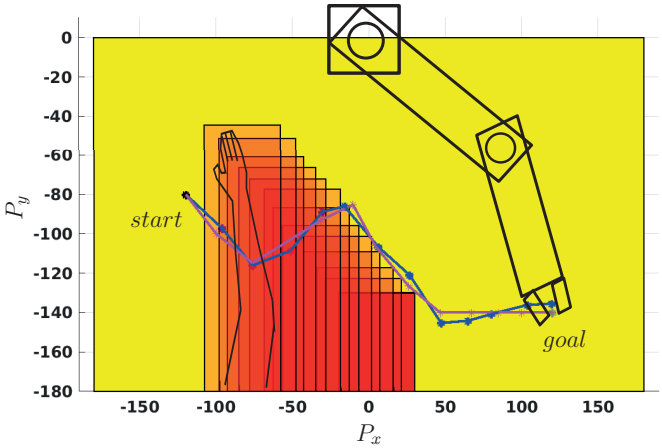

Figure 3.24.: The goal position of the EE is reached after 12 steps and the safety of the human is ensured in all intermediate steps.

3.5. Summary and Discussion

This chapter starts with the problem that uncertainties may act on local plants, extends the principle of tube-based control which was established for purely continuous dynamics, to hybrid system HA^u with guarded transitions and mixed inputs. As the main result, point-to-set transitions for this class of hybrid systems can be computed by the proposed techniques in optimized fashion, while reaching the goal set is guaranteed despite uncertainties of the flow and reset functions. This idea can also be extended to robust control of HA^u with nonlinear flow functions according to the discussion in the last chapter: first of all, a set of PWA systems are determined to approximate the nonlinear dynamics in each invariant, by which the invariants are partitioned into small sub-spaces, and the linearization error is cast into additive disturbances of the PWA dynamics [101]. Then, the partitioned state spaces are tightened according to both original disturbances and the disturbances caused by the linearization error. A nominal hybrid system can then be constructed as in Sec. 3.1.2, and by planning with the nominal hybrid system, the feasibility and the ability to attain the goal for the original uncertain and nonlinear hybrid system are guaranteed.

For the online control problem of HA (or any general nonlinear systems) by using MPC, the conditions on ensuring recursive feasibility and asymptotic stability of MPC strategies are addressed in this chapter. Especially when the state constraints representing the influence of other subsystems are with unknown changes over time, two different change mechanisms of the state constraints are considered, and in both cases, recursive feasibility and asymptotic stability can be and have been shown. With respect to the problem that the conservative tightening of the state space may lead to infeasible problems, a penalty term method is also proposed in this chapter. It has been shown that this method does not affect the feasible space of the original problem and often results in satisfiable outcome. However, this approach can only enhance the possibility of MPC being recursive feasible, instead of providing a guaranty compared to the tightening method. Thus, it is worth to investigate in how far the penalty approach can replace the tightening one, as well as to investigate the use of a time-varying penalty term to reduce the performance loss.

Until now, the interaction among the subsystems in CPS has been cast into local constraints in the last two chapters, either with or without uncertainties. In many cases, however, such interaction is not quantifiable and cannot be formulated as local constraints. In this case, the use of coupling constraints is preferred to model the interaction, i.e., the local solution of each subsystem should jointly satisfy the coupling constraints. To ensure the satisfaction of the coupling constraints, the centralized optimization over all subsystems is the most promising way but will often be impossible due to complexity. Therefore, newly developed distributed strategies are proposed in the coming chapter, which can significantly reduce the overall complexity, while the satisfaction of the coupling constraints remains ensured.

Part III.

Distributed Control of Networked Hybrid Systems with Coupling Constraints

4. Distributed Control of CPS with Coupling Constraints

In contrast to the coupling scheme discussed in the previous chapters, the subsystems in this chapter are assumed to interact according to joint constraints (or coupling constraints). These joint constraints must be satisfied by each local controller simultaneously, see Fig. 4.1. For instance, consider a coordinated multi-vehicle driving scenario as sketched in Fig. 4.2, in which a building site narrows the road so that only one vehicle can pass through at each time. In this case, the limited width of the road can be regarded as a coupling constraint among the vehicles (subsystems), and the vehicles should coordinate their driving plans (i.e., adjust their local control strategies) by traveling through the narrow passage without conflicts.

A particular difficulty of controlling CPS with coupling constraints is to provide methods for control and optimization which scale with the system size, possibly measured in the number of relevant subsystems. In order to ensure a safe, reliable, and highly performant operation of the distributed systems, local control strate-

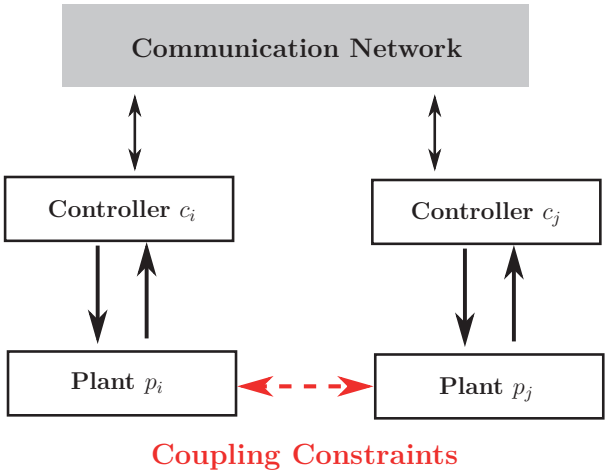

Figure 4.1.: Interleaving structure of the subsystems in CPS.

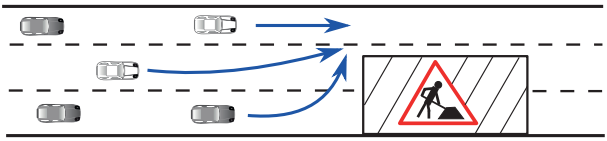

Figure 4.2.: Example of a distributed system with coupling constraints induced by limited resource: the local controllers of autonomous vehicles on a three-lane road have to observe a narrow passage at a building site, requiring to coordinate their driving plans.

gies for each subsystem as well as mechanisms for coordinating or adapting the interaction of subsystems must be devised. These schemes have to account for (a) heterogenous dynamics, objectives, and constraints, and (b) limited computational power of the local controllers to determine (optimal) control inputs. The optimal and distributed control of this type of systems is an important problem class facing these difficulties, typically too complicated to be solved in a centralized fashion. Most investigations on distributed control focus on the cases when

- the subsystems are coupled by common cost, and the overall cost function is the sum of each local costs depending on common optimization variables, see [35, 116, 164, 117, 121, 168, 124];
- the subsystems are coupled by common constraints (the case in this chapter), and each subsystem has to optimize local variables to fulfill the local constraints, while the global cost is minimized and the coupling constraints are satisfied, see [59, 61, 47, 132, 127, 172].

For the first case, two main approaches have been developed to solve the problem in a distributed form, namely, the consensus-based primal methods [124, 116], and the Lagrangian-based dual methods [35, 159]. The main challenge of this problem is to realize a faster convergence to the optimal common variables, while the reduction of computational complexity does not belong to the major concerns. This is because all subsystems share the same optimization variables in this problem, and an increase of subsystems does not typically lead to an increase of optimization complexity, but to communication complexity. For the second case above, the distributed solution is more necessary and often preferred with respect to optimization complexity reduction, since an increase of subsystems will raise the overall number of optimization variables and thus increases the overall complexity. In most of the existing work for the second case, the centralized problem is assumed to be convex, and methods to ensure the feasibility and optimality of the distributed solution are studied. In particular, they aim at decomposing the centralized problem into a set of small-scale sub-problems to reduce the problem size, and at solving the resulting

subproblems using sequential or parallel schemes. It is worth to mention that sometimes a common cost problem of the first case can also be equivalently reformulated into a coupled constraint problem. This reformulation is usually realized by setting up the dual problem of the original centralized problem [124]. In this respect, an efficient distributed solution of the second case can also utilize the first case.

For the distributed control problems in CPS, especially if hybrid dynamics is used to model the subsystems, or if the local state space is not convex, a mixed-integer programming (MIP) problem in the form of (2.1) may have to be solved, e.g. see Problem 2.2, 2.3 and 3.5 in the previous chapters, or the problems in [55, 96]. As the MIP problems are known to be non-convex, the solution suffers from non-polynomial worst-case runtime [155]. Typical solution techniques for MIP problems are the branch-and-bound [88, 112] and branch-and-cut algorithm [111, 149], which have been reviewed at the beginning of Chapter 2. Accordingly, compared with convex centralized problems, the investigation on efficient distributed solution of MIP problems, while satisfying both local and coupling constraints, is becoming even more urgent for CPS with respect to control complexity reduction.

In front of this challenge, accordingly, a set of efficient distributed solution schemes is proposed for MIP problems in this chapter: first, a particular focus is put on the case in which the cost function is linear and an MILP problem results. For this case, a recently proposed method based on the Shapley-Folkman-Starr theorem [147, 46, 16, 148], is extended by relaxing some conservative assumptions required in state-of-the-art work, and thus the computational efficiency is enhanced. Then, a distributed solution scheme for the case of quadratic cost functions, i.e., MIQP problems, is proposed. This scheme partly exploits the idea of the linear case, while feasibility as well as continuous cost reduction over the iterations are guaranteed for both cases. Lastly, one instance of the proposed method is applied to solve the vehicle coordination problem from Fig. 4.2. The content of this chapter is mainly based on ideas which have been published already in [96, 99, 103, 102].

4.1. Efficient Distributed Solution for MILP Problems

Before the discussion on MIP problems starts, a brief review on existing distributed solution techniques for convex problems is provided. First of all, consider the following local problem of subsystem i for all $i \in N = \{1, \cdots, n_s\}$:

$$\begin{aligned} &\min_{x_i} J_i(x_i) \\ &\text{s.t.: } x_i \in X_i \subset \mathbb{R}^{r_i}, \end{aligned} \tag{4.1}$$

which is similar to the problem (2.1) but without integrality constraints. The symbol $J_i(x_i)$ denotes local costs, and $x_i \in \mathbb{R}^{r_i}$ denotes the local optimization variables, consisting of local input and states in each discrete time of the horizon (according to diverse problems appeared in previous chapters). The local constrained set X_i comprises the local input and state constraints as well as the dynamics of the subsystem.

Note that X_i represents a polytopic set when both constraints and dynamics are linear.

Next, consider a centralized problem defined over all subsystems, which takes the form of:

$$\min_{x_1,\cdots,x_{ns}} \sum_{i\in N} J_i(x_i) \tag{4.2}$$

$$\text{s.t.:} \sum_{i\in N} A_i x_i \leq b \tag{4.3}$$

$$x_i \in X_i \subset \mathbb{R}^{r_i},\ i \in N. \tag{4.4}$$

Compared with problem (4.1), the new constraint (4.3) represents a coupling constraint (linear) among the subsystems. For the centralized problem (4.2), if each local J_i is a convex function, and as each local feasible set X_i is polytopic, the entire problem turns out to be convex. In this case, a common approach to avoid the centralized solution of (4.2) for a large number of subsystems is to set up the dual problem of (4.2), see [127, 59]. In the first step of this approach (which is also called the Lagrangian-based dual method), the coupling constraint (4.3) is dualized by a vector of multipliers $\lambda \geq 0$, and the dimension of λ equals to the dimension of the vector b. Then, the following dual problem is obtained:

$$\max_{\lambda\geq 0} -\lambda^T \cdot b + \sum_{i\in N} \min_{x_i\in X_i} (J_i(x_i) + \lambda^T \cdot A_i x_i). \tag{4.5}$$

The coupling constraint (4.3) is transferred into an objective function of (4.5), and for any fixed value of $\lambda := \hat{\lambda}$, the minimization task of all x_i, $i \in N$, in problem (4.5) becomes decomposable, and a set of local problems:

$$\min_{x_i} J_i(x_i) + \hat{\lambda}^T \cdot A_i x_i \tag{4.6}$$

$$\text{s.t.:}\ x_i \in X_i$$

of reduced size is obtained for any subsystem $i \in N$. Then, starting from an initial pair of $\lambda^{[0]}$ and $x_i^{[\lambda^{[0]}]}$, $i \in N$, the following procedure is successively executed in each iteration ρ:

- Solve the local problems (4.6) in parallel for all i based on the present value of $\lambda^{[\rho]}$, and obtain $x_i^{[\lambda^{[\rho+1]}]}$, $i \in N$, for the coming iteration $\rho + 1$;
- Update the value of $\lambda^{[\rho]}$ to $\lambda^{[\rho+1]}$ according to $x_i^{[\lambda^{[\rho+1]}]}$, $i \in N$, and a cost-ascent direction of the dual function (4.5).

This approach provides a guaranty on both primal and dual optimality[1] of the problem (4.2), while the solution of the intermediate problem (4.6) is carried out in

[1]The dual optimum can be obtained in the terminal iteration of this approach, while the primal optimum must be recovered in addition by adopting some further steps, e.g., by evaluating the KKT conditions of (4.2).

a distributed form, see [36, 27]. Afterwards, this distributed strategy has also been further explored with respect to the aspects of (a) how to speed up the convergence rate [118, 120, 61], (b) whether a central coordinator is necessary to distribute the information during the solution [168, 141], and (c) which information should be communicated in case of privacy issues [59, 144].

However, for MIP problems in the form of (4.2), i.e. if a certain part of x_i is integer valued, the problem (4.2) turns out to be non-convex, leading to a duality gap. In this case, the Lagrangian-based dual method may cause the distributed solution to be non-optimal, or even infeasible. This is due to the effect that the primal optimum and the dual optimum may no longer be a saddle-point of the problem (4.5), if the integrality constraints imposed on each local variables x_i in (4.2) (see [36] for more detailed reasoning). As a result, the primal „solution" recovered from the dual optimum may not coincide with the true primal optimum, and thus may not satisfy the local and coupling constraints.

Nonetheless, recent studies have revealed that, if the local cost functions in (4.2) are linear, leading to an MILP problem, and if the number of subsystems is large, the duality gap vanishes under certain conditions [162]. Following this line and by making further use of the Lagrangian dual methods, one can indeed determine an optimal (or close-to-optimal solution) in a distributed way. In detail, the considered MILP problem takes the form:

$$\min_{x_1,\cdots,x_{n_s}} \sum_{i\in N} c_i^{\mathrm{T}} x_i \tag{4.7}$$

$$\text{s.t.:} \sum_{i\in N} A_i x_i \leq b; \tag{4.8}$$

$$x_i \in X_i \subset \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\ i \in N. \tag{4.9}$$

Similar to (2.1), the states and controls in each time step of the horizon, as well as the additional integer and auxiliary variables (used for encoding the hybrid dynamics) of local subsystem i, are collected into a mixed-integer vector x_i. For the optimal control problems of the local subsystems in Problem 2.2, 2.3 and 3.5, the system dynamics and constraints can eventually be reformulated into linear form, and are collected into local constraint (4.9), where X_i is a mixed-integer polyhedral set:

$$X_i = \{x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i} | D_i x_i \leq d_i\}. \tag{4.10}$$

For the local variable x_i, the r_i and z_i in (4.9) denote the number of continuous and discrete components, which depend on the state and input dimension of the local HA, the number of invariants and guards, the auxiliary variables applied for encoding, and the time horizon. For constraint (4.8), $A_i \in \mathbb{R}^{m\times(r_i+z_i)}$ and $b \in \mathbb{R}^{m\times 1}$ denote matrices of appropriate dimension, and it models the coupling among the subsystems. Note that the formulation in (4.8) can represent a large variety of coupling schemes, e.g., coupled states, inputs, or a mixture hereof, as x_i contains all local variables to be optimized. The number m of the coupling constraints will

play an important role in the later description. The local cost function is in linear form, while the global cost function $\sum_{i\in N} c_i^{\mathrm{T}} x_i$ is the sum of each local costs. Moreover, $J(\mathbf{x}) := \sum_{i\in N} c_i^{\mathrm{T}} x_i$ is used to refer to the global costs of problem (4.7), as well as $\mathbf{x}^* = [x_1^*; \cdots; x_{n_s}^*]$ to denote the optimal solution of (4.7), and $J(\mathbf{x}^*)$ for the optimal global cost.

Strategies for solving the problem (4.7) in distributed form have been considered in a few publications before, see e.g. [162, 60, 43, 161, 44]. Most of this work, which is briefly summarized below, is based on the Shapley-Folkman-Starr theorem:

Theorem 4.1. *(Shapley-Folkman-Starr theorem [147]) For MILP problem* (4.7)*, define a set:*

$$S_i := \left\{ s_i \in \mathbb{R}^{m+1} \,\middle|\, s_i = \begin{bmatrix} c_i^{\mathrm{T}} x_i \\ A_i x_i \end{bmatrix}, x_i \in X_i \right\}$$

for all subsystems $i \in \{1, \cdots, n_s\}$*, and a set*

$$S := \left\{ s \in \mathbb{R}^{m+1} \,\middle|\, s = s_1 + \cdots + s_{n_s},\, s_i \in S_i,\, i \in \{1, \cdots, n_s\} \right\}.$$

If the relation $n_s > m + 1$ *is satisfied, then for every vector* $\hat{s} \in Conv(S)$*, there always exists a local vector* $\hat{s}_i \in Conv(S_i)$ *for all* $i \in \{1, \cdots, n_s\}$*, while* $\hat{s}_i \notin S_i$ *for at most* $m + 1$ *subsystems, such that the relation*

$$\hat{s} = \hat{s}_1 + \cdots + \hat{s}_{n_s}$$

holds.

The proof of this theorem can be found in [25] and [162]. This theory establishes the fact that, although each local set S_i is not convex due to the integrality constraints on x_i, one can still find a combination of $\hat{s}_1, \cdots, \hat{s}_{n_s}$, where most $\hat{s}_i$ are selected from the non-convex set S_i, so that the relation $\hat{s} = \hat{s}_1 + \cdots + \hat{s}_{n_s}$ applies for any $\hat{s}$ selected from $Conv(S)$. According to this theory, let the mixed-integer set X_i in (4.7) be replaced by a convex set $Conv(X_i)$, denoting the convex hull of all points in X_i. Then, by dropping the integrality constraint of all variables in x_i, i.e. $x_i \in \mathbb{R}^{r_i+z_i}$, the MILP problem (4.7) is cast into a Linear Programming (LP) problem:

$$\begin{aligned} &\min_{x_1,\cdots,x_{n_s}} \sum_{i\in N} c_i^{\mathrm{T}} x_i \\ \text{s.t.:}\ &\sum_{i\in N} A_i x_i \le b; \\ &x_i \in Conv(X_i),\, x_i \in \mathbb{R}^{r_i+z_i},\, i \in N. \end{aligned} \tag{4.11}$$

It is remarkable that $Conv(X_i)$ does not coincide with the set $\mathcal{R}(X_i) = \{x_i \in \mathbb{R}^{r_i+z_i} | D_i x_i \le d_i\}$, where the latter is obtained by relaxing the integrality constraints

to intervals in X_i, and is more commonly used to determine a lower cost bound of MIP problems. In fact, both sets $Conv(X_i)$ and $\mathcal{R}(X_i)$ contain the original mixed-integer set X_i, whereas the former is harder to be computed but is tighter than the latter, see [66] for the reasoning, as well as the following example used in [157]: Given a mixed-integer set:

$$X = \{x_1 \in \mathbb{Z}, x_2 \in \mathbb{R} \mid \begin{bmatrix} 1 & 10 & -5 & -9 \\ 10 & -2 & -8 & -3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 18 & 10 & 4 & 18 \end{bmatrix}^{\mathrm{T}}\}, \tag{4.12}$$

the size of the sets X, $Conv(X)$ and $\mathcal{R}(X)$ are compared in Fig. 4.3. Note that the relation $X \subseteq Conv(X) \subseteq \mathcal{R}(X)$ applies.

Furthermore, it is well known that, if the LP problem (4.11) has a unique optimal solution, then it must be attained in a vertex of the feasible set. By using $\bar{\mathbf{x}}^* = [\bar{x}_1^*, \cdots, \bar{x}_{ns}^*]$ to denote the unique optimal solution of (4.11), the following relation applies according to the Shapley-Folkman-Starr theorem [162]:

Theorem 4.2. *If the LP problem* (4.11) *has a unique optimal solution* $\bar{\mathbf{x}}^*$*, then a partitioning* $N = N_1 \cup N_2$, $|N_1| \geq n_s - m - 1$*, of the subsystems can be determined, so that the local solution* $\bar{x}_i^*$ *in* $\bar{\mathbf{x}}^*$ *is attained at the vertices of* $Conv(X_i)$ *for all* $i \in N_1$*. In addition, as all vertices of* $Conv(X_i)$ *are also located in the mixed-integer set* X_i*, the local solution* $\bar{x}_i^*$ *thus also satisfies* X_i*,* $\forall i \in N_1$*, although subject to a real-valued set in* (4.11)*.*

The proof of this theorem was provided in [162] and is briefly sketched here:

Proof. Due to the uniqueness assumption of (4.11) one knows that, for the optimal solution $\bar{\mathbf{x}}^*$ of (4.11), there exists no other candidate $\bar{\mathbf{x}}^{can} = [\bar{x}_1^{can}, \cdots, \bar{x}_{n_s}^{can}]$, $\bar{x}_i^{can} \in Conv(X_i)$, $i \in N$, to fulfil the relation

$$\sum_{i \in N} c_i^{\mathrm{T}} \bar{x}_i^* = \sum_{i \in N} c_i^{\mathrm{T}} \bar{x}_i^{can}, \quad \sum_{i \in N} A_i \bar{x}_i^* = \sum_{i \in N} A_i \bar{x}_i^{can}.$$

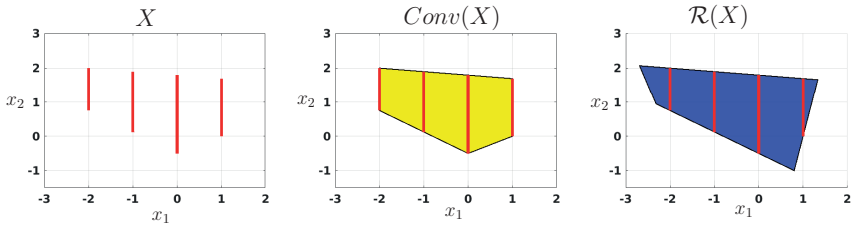

Figure 4.3.: Compare the size of set X (in red) defined in (4.12), and the corresponding set $Conv(X)$ (in yellow) and set $\mathcal{R}(X)$ (in blue).

Otherwise, the candidate $\bar{\mathbf{x}}^{can}$ will also be an optimal solution of (4.11), as it attains the same global costs as $\bar{\mathbf{x}}^*$ and also satisfies both local and coupling constraints in (4.11). Then, regarding the set S_i, $\forall i \in N$, and the set S defined in Theorem 4.1, the above fact implies that for a vector s^* determined by

$$s^* := s_1^* + \cdots + s_{n_s}^*,$$

where

$$s_i^* = \begin{bmatrix} c_i^{\mathrm{T}} \bar{x}_i^* \\ A_i \bar{x}_i^* \end{bmatrix},\ s_i^* \in Conv(S_i) = \left\{ s_i \in \mathbb{R}^{m+1} \,\middle|\, s_i = \begin{bmatrix} c_i^{\mathrm{T}} x_i \\ A_i x_i \end{bmatrix}, x_i \in Conv(X_i) \right\},\ i \in N,$$

there exists no other candidate $s_i^{can} \in Conv(S_i)$, $\forall i \in N$, to fulfil the relation:

$$s^* = s_i^{can} + \cdots + s_{n_s}^{can}.$$

Then, as the relation

$$s^* = \sum_{i \in N} s_i^* \in Conv(S)$$

applies since each s_i^* is selected from $Conv(S_i)$, one knows that there must exist a local vector $\hat{s}_i \in Conv(S_i)$ for all $i \in N$, while $\hat{s}_i \notin S_i$ for at most $m+1$ subsystems. Thus, the relation:

$$s^* = \hat{s}_1 + \cdots + \hat{s}_{n_s}$$

holds according to Theorem 4.1. However, as $s_1^*, \cdots, s_{n_s}^*$ is the unique candidate to fulfil this equation, there must exist $\hat{s}_i = s_i^*$ for all $i \in N$. This immediately implies that for $s_1^*, \cdots, s_{n_s}^*$, at most $m+1$ subsystems do not satisfy the constraint $s_i^* \in S_i$. Then, as the mapping from $\bar{x}_i^*$ to s_i^* is linear, there must also exist at most $m+1$ subsystems in $\bar{\mathbf{x}}^*$, for which the local solution $\bar{x}_i^*$ does not belong to the mixed-integer set X_i. □

In case that a large number of subsystems are involved in (4.7), but only a few coupling constraints (4.8) need to be considered, i.e. $n_s \gg m$, the above theorem implies that the majority of the subsystems can already determine their local feasible solutions of (4.7) through the solution of the LP problem (4.11). Most importantly, the solution of (4.11) can be carried out in a distributed way due to its convexity (and in the same form as problem (4.2)).

Following this line, the authors in [162] introduced a method to determine a feasible candidate of (4.7), by which the LP problem (4.11) is first solved in a distributed way, and then the local feasibility of the remaining $|N_2|$ subsystems is recovered. However, this method faces the following drawbacks, limiting its application:

1. the condition $A_i x_i \geq 0$, $\forall x_i \in X_i$, $\forall i \in N$ is assumed in problem (4.7), in order to guarantee the success of the recovering process for the $|N_2|$ subsystems;

2. the convergence rate towards $\bar{x}_i^*$ by employing the Lagrangian-based dual method is extremely slow (mainly due to the use of a sub-gradient direction[2] of the dual function and the vanishing step size in each iteration [12]).

Facing these drawbacks, a new relation between (4.7) and (4.11) was established in [161] to guide the distributed solution of (4.7):

Theorem 4.3. *If the LP problem* (4.11) *and its dual problem*

$$\bar{\lambda}^* := \arg\max_{\lambda \geq 0} -\lambda^T \cdot b + \sum_{i \in N} \min_{x_i \in Conv(X_i)} (c_i^{\mathrm{T}} + \lambda^T \cdot A_i) x_i \tag{4.13}$$

obtained by dualizing the coupling constraints (4.8)*, have a unique optimal solution* $\bar{\mathbf{x}}^*$ *and* $\bar{\lambda}^*$*, then the local solutions* x_i^d *obtained by solving the following local problems*

$$x_i^d := \arg\min_{x_i \in X_i} (c_i^{\mathrm{T}} + \bar{\lambda}^{*,T} \cdot A_i) x_i \tag{4.14}$$

for all n_s *subsystems, differ for at most* $m+1$ *subsystems from the optimal solution* $\bar{\mathbf{x}}^*$ *of problem* (4.11).

This enables each subsystem i to first solve the dual problem (4.13) in a distributed way, until the dual optimum $\bar{\lambda}^*$ is obtained. Then, the problem (4.14) is solved locally by each subsystem according to $\bar{\lambda}^*$. The resulting local solution x_i^d from (4.14) thus satisfies X_i for all $i \in N$, but may eventually violate the coupling constraint (4.8). Accordingly, the authors in [161] tightened the coupling constraint (4.8) in advance, i.e., each dimension j of the right-hand side of inequality (4.8) is reduced by:

$$(m+1) \cdot \max_{i \in N} (\max_{x_i \in X_i} A_i(j,:) x_i - \min_{x_i \in X_i} A_i(j,:) x_i), \tag{4.15}$$

where $A_i(j,:)$ denotes the j-th row of matrix A_i. Then, the corresponding LP problem (4.11) and dual problem (4.13) with the tightened coupling constraints is constructed. Thereafter, the dual optimum of the new dual problem (4.13) is determined, as well as the x_i^d from (4.14) based on the new dual optimum. This approach enables each x_i^d from (4.14) only to violate the tightened coupling constraint, but not the original one – thus, they constitute a feasible candidate of (4.7). Compared with the last approach, the assumption on $A_i x_i \geq 0$, $\forall x_i \in X_i$, $\forall i \in N$ is relaxed here, but the following significant drawbacks still exist:

1. the convergence rate towards the dual optimum $\bar{\lambda}^*$ by using the Lagrangian-based dual method is as slow as the one towards $\bar{x}_i^*$;

[2]The sub-gradient is used to determine a cost-ascent direction of the multiplier λ in each iteration. This is because the dual function (4.5) is always concave with respect to λ, but may not always be differentiable, i.e., no gradient can be determined. In this case, the sub-gradient is adopted to update λ, but its selection may not always be unique in each iteration.

2. the new LP problem (4.11) must be feasible after tightening the coupling constraints (4.8). However, the tightening scheme in (4.15) is quite conservative, resulting in a substitute coupling constraint which is hard to satisfy, i.e., leading to an infeasible problem in the worst case.

Still based on the Shapley-Folkman-Starr theorem, the work in [60] proposed an improved Lagrangian-based dual method, aiming at reducing the conservativeness caused by the tightening. In [43, 44], the authors employed the primal decomposition method for the solution of (4.11) instead of the dual one.

Obviously, the common drawback of the work above is the slow convergence rate towards either $\bar{x}_i^*$ or $\bar{\lambda}^*$ when employing the Lagrangian-based dual method. In addition, requirements such as $A_i x_i \geq 0$, $\forall x_i \in X_i$, $\forall i \in N$, or the feasibility of (4.11) with tightened coupling constraints further limit the applicability of these methods. Thus, this section proposes a novel distributed solution scheme aiming at overcoming these issues. The main idea of the proposed method is still based on the Shapley-Folkman-Starr theorem – but unlike the existing methods, in which the theorem is employed to directly determine a feasible candidate of (4.7), here it is applied to successively generate and improve the feasible candidates of (4.7), until the global optimum (or at least a sub-optimum) is found. By using the proposed method, the conservative assumptions mentioned above are relaxed, while the computations can also be accelerated.

4.1.1. Improvement by the Lagrangian-Based Dual Method

The step of finding a first feasible candidate of a MIP problem usually constitutes the first phase of solution in most of the existing solvers, typically incurring high costs, as reviewed at the beginning of Chapter 2. Only after a certain number of iterations, established search strategies like the branch-and-bound or branch-and-cut algorithms, the quality of the candidate improves. Specifically for MILP problems according to (4.7), many tests and applications have shown that – even if the problem is large – the determination of the first feasible candidate is typically much faster than the process of converging to the optimal one. As an example, for the electric vehicle charging problem considered in [161, 60], the determination of the global optimal solution needs over 6 hours, but the first feasible candidate was found in less than one second.

Thus, by assuming that a first feasible candidate of (4.7) is on hand and denoted by $\mathbf{x}^f = [x_1^f; \cdots; x_{n_s}^f]$, the main task of the algorithm to be proposed for distributed solution is to improve $\mathbf{x}^f$ until the global optimum (or a value close to it) is found. Recall that, if the standard branch-and-bound method is applied to improve $\mathbf{x}^f$, it usually keeps branching on the integer variables (leading to nodes). This process, however, requires heuristics for the priority of the nodes to be considered first for branching. If the heuristics is not efficient, this causes a large amount of meaningless computation, e.g., when the new nodes are infeasible, or worse than $\mathbf{x}^f$. It

also requires to store a large amount of data. Here instead, search directions for continuous cost improvement in any iteration are proposed.

The considered method first starts with determining a new coupling constraint of (4.7) based on $\mathbf{x}^f$, and taking the form:

$$\sum_{i\in N} A_i x_i \leq b^f, \text{ with } b^f = \sum_{i\in N} A_i x_i^f. \tag{4.16}$$

Since $\mathbf{x}^f$ is a feasible candidate of (4.7), the vector b^f must satisfy:

$$b^f \leq b. \tag{4.17}$$

Obviously, this implies that (4.16) constitutes a tighter coupling constraint compared to the original one in (4.8). Now, by replacing (4.8) with (4.16) in problem (4.7), the following MILP problem is obtained:

$$\begin{aligned} &\min_{x_1,\cdots,x_{n_s}} \sum_{i\in N} c_i^{\mathrm{T}} x_i \\ \text{s.t.:}\ &\sum_{i\in N} A_i x_i \leq b^f; \\ &x_i \in X_i,\ x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\ i \in N. \end{aligned} \tag{4.18}$$

A similar MILP problem with tightened coupling constraint was also considered in [161], but there, the existence of a feasible solution of the tightened problem was only assumed, whereas the feasibility of (4.18) here always holds due to $\mathbf{x}^f$. Now, by replacing the mixed-integer set X_i in (4.18) with the convexified set $Conv(X_i)$, an LP problem is obtained:

$$\begin{aligned} &\min_{x_1,\cdots,x_{n_s}} \sum_{i\in N} c_i^{\mathrm{T}} x_i \\ \text{s.t.:}\ &\sum_{i\in N} A_i x_i \leq b^f; \\ &x_i \in Conv(X_i),\ x_i \in \mathbb{R}^{r_i+z_i},\ i \in N. \end{aligned} \tag{4.19}$$

Its dual problem obtained from dualizing the coupling constraint $\sum_{i\in N} A_i x_i \leq b^f$ has the form:

$$\sup_{\lambda\geq 0} -\lambda^T b^f + \sum_{i\in N} \min_{x_i\in Conv(X_i)} (c_i^{\mathrm{T}} + \lambda^T \cdot A_i) x_i \tag{4.20}$$

referring to (4.13). Here, $\bar{\mathbf{x}}^{*,f} = [\bar{x}_1^{*,f}; \cdots; \bar{x}_{n_s}^{*,f}]$ and $\bar{\lambda}^{*,f}$ denote the optimal solution of (4.19) and (4.20).

Assumption 4.1. *Both problems* (4.19) *and* (4.20) *have unique optimal solutions* $\bar{\mathbf{x}}^{*,f}$ *and* $\bar{\lambda}^{*,f}$.

Note that this assumption is typically not conservative, since one can avoid non-unique cases by introducing small perturbations to the cost or constraints in (4.19), as indicated in [161].

For the set of MILP and LP problems introduced above, let the optimal costs be compared here: By using $J(\mathbf{x}^*)$, $J(\bar{\mathbf{x}}^*)$, $J(\mathbf{x}^{*,f})$ and $J(\bar{\mathbf{x}}^{*,f})$ to represent the optimal costs of the problems (4.7), (4.11), (4.18), and (4.19) respectively, as well as $J(\mathbf{x}^f)$ for the global costs of $\mathbf{x}^f$, the following relations can be established:

Lemma 4.1. *For a given $\mathbf{x}^f$, it applies that:*

$$J(\bar{\mathbf{x}}^*) \leq J(\mathbf{x}^*) \leq J(\mathbf{x}^{*,f}) \leq J(\mathbf{x}^f), \tag{4.21}$$

$$J(\bar{\mathbf{x}}^*) \leq J(\bar{\mathbf{x}}^{*,f}) \leq J(\mathbf{x}^{*,f}). \tag{4.22}$$

Proof. The first inequality (from left to right) $J(\bar{\mathbf{x}}^*) \leq J(\mathbf{x}^*)$ in (4.21) follows from the relaxed integrality constraint in (4.11) than in (4.7); the second inequality $J(\mathbf{x}^*) \leq J(\mathbf{x}^{*,f})$ in (4.21) follows from the tighter coupling constraint in (4.18) than in (4.7); the last inequality $J(\mathbf{x}^{*,f}) \leq J(\mathbf{x}^f)$ in (4.21) follows from the fact that $\mathbf{x}^f$ represents a feasible candidate of (4.18) only, but not necessarily the optimal one; the first inequality $J(\bar{\mathbf{x}}^*) \leq J(\bar{\mathbf{x}}^{*,f})$ in (4.22) is implied by the tighter coupling constraint in (4.19) than in (4.11); the second inequality $J(\bar{\mathbf{x}}^{*,f}) \leq J(\mathbf{x}^{*,f})$ in (4.22) results from the relaxed integrality constraint in (4.19) compared to (4.18). □

For the objective of improving $\mathbf{x}^f$, the relations listed in Lemma 4.1 point to a useful direction: it can be noticed that $J(\mathbf{x}^*)$ and $J(\bar{\mathbf{x}}^{*,f})$ are both bounded by $J(\bar{\mathbf{x}}^*)$ from below, and bounded by $J(\mathbf{x}^{*,f})$ from above. This fact implies that, although the relations between $J(\mathbf{x}^*)$ and $J(\bar{\mathbf{x}}^{*,f})$ are still unknown, their costs are bounded by the same range and both lead to global costs lower than $J(\mathbf{x}^f)$ (as $J(\mathbf{x}^{*,f}) \leq J(\mathbf{x}^f)$ applies). Then, since:

- $\mathbf{x}^*$ is the global optimum of (4.7), which defines the best candidate one can find for $\mathbf{x}^f$;
- and $\bar{\mathbf{x}}^{*,f}$ is the global optimum of the LP problem (4.19), which may not satisfy the local constraint X_i (and thus may not be feasible for (4.7)),

it is straightforward to assume that a feasible candidate of (4.7) being located close to $\bar{\mathbf{x}}^{*,f}$, will attain similar global costs as $J(\bar{\mathbf{x}}^{*,f})$ and $J(\mathbf{x}^*)$. This feasible candidate, accordingly, can be regarded as better than $\mathbf{x}^f$, and it constitutes a better candidate for (4.7). To compute this feasible candidate, however, one must first determine the value of $\bar{\mathbf{x}}^{*,f}$. Note that, as $\bar{\mathbf{x}}^{*,f}$ represents the optimal solution of the LP problem (4.19) and this problem also shares a similar structure as the one in (4.2), the Lagrangian-based dual method is thus employed to compute $\bar{\mathbf{x}}^{*,f}$ in a distributed fashion, see Algorithm 4.1.

Note that by temporarily neglecting the lines 7 to 12 in Algorithm 4.1, this would represent a standard Lagrangian-based dual method for the problems in the form of (4.2). In this algorithm, ρ is the iteration counter and ρ^{max} represents the maximal number of iterations allowed to be executed (usually a large number to ensure the convergence to $\bar{\mathbf{x}}^{*,f}$). The computation in line 4 updates the local candidate $x_i^{[\lambda^{[\rho]}]}$ based on the present multiplier $\lambda^{[\rho]}$ in iteration ρ (referring to (4.6)), while the index $[\lambda^{[\rho]}]$ in $x_i^{[\lambda^{[\rho]}]}$ is used to clarify that this solution to the sub-problem in line 4 is specific for the present value of $\lambda^{[\rho]}$. The computation in line 5 is an averaging over the $x_i^{[\lambda^{[\rho]}]}$ obtained in the previous iterations, in order to recover the primal optimal solution $\bar{\mathbf{x}}^{*,f}$, see [59] for more details of this recovering process. The computations in the lines 4 and 5 are carried out in a distributed form. Then, $\gamma^{[\rho]}$ in line 13

1: **Initialization:** $\rho = 1$, $\lambda^{[\rho]} = 0$, $\mathbf{x}^f$, $Flag = 0$;
2: **while** $\rho \leq \rho^{max}$ and $Flag = 0$ **do**
3: **for** all subsystem $i \in N$ **do**
4: $x_i^{[\lambda^{[\rho]}]} := \arg\min_{x_i \in X_i}(c_i^{\mathrm{T}} + \lambda^{[\rho],T} A_i) x_i$;
5: $\overline{x}_i^{[\rho]} := \frac{1}{\rho} \sum_{j=1}^{\rho} x_i^{[\lambda^{[j]}]}$;
6: **end for**
7: Decompose $x_i^{[\lambda^{[\rho]}]}$ into $[x_{i,r}^{[\lambda^{[\rho]}]}; x_{i,t}^{[\lambda^{[\rho]}]}]$, $\forall i \in N$, and solve the LP problem:

$$\min_{x_{1,r},\cdots,x_{n_s,r}} \sum_{i \in N} c_i^{\mathrm{T}} \cdot [x_{i,r}; \mathbf{0}^{z_i \times 1}] \tag{4.23}$$

$$\text{s.t.:} \sum_{i \in N} A_{i,r} \cdot x_{i,r} \leq b - \sum_{i \in N} A_{i,t} \cdot x_{i,t}^{[\lambda^{[\rho]}]}; \tag{4.24}$$

$$D_{i,r} \cdot x_{i,r} \leq d_i - D_{i,t} \cdot x_{i,t}^{[\lambda^{[\rho]}]},\ x_{i,r} \in \mathbb{R}^{r_i},\ i \in N. \tag{4.25}$$

8: **if** (4.23) – (4.25) is feasible and the optimized solution $x_{i,r}^*$, $i \in N$ satisfies that $\sum_{i \in N} c_i^{\mathrm{T}} [x_{i,r}^*; x_{i,t}^{[\lambda^{[\rho]}]}] < J(\mathbf{x}^f)$ **then**
9: $Flag = 1$;
10: **else**
11: $Flag = 0$;
12: **end if**
13: $\gamma^{[\rho]} := \sum_{i \in N} A_i x_i^{[\lambda^{[\rho]}]} - b^f$;
14: $\lambda^{[\rho+1]} := P_+(\lambda^{[\rho]} + s^{[\rho]} \gamma^{[\rho]})$;
15: $\rho := \rho + 1$;
16: **end while**

Algorithm 4.1.: Distributed computation of $\bar{\mathbf{x}}^{*,f}$

determines a sub-gradient of the dual function in (4.20), which is used in line 14 to update the multiplier $\lambda^{[\rho]}$. The symbol $s^{[\rho]}$ in line 14 is the step length chosen to update the multiplier $\lambda^{[\rho]}$ in each iteration ρ, and the operation P_+ denotes the projection onto the positive sub-space of $\mathbb{R}^m$.

In general, this algorithm leads to $\bar{x}_i^{*,f}$ for $\rho \to \infty$, as long as the following conditions for the step length $s^{[\rho]}$ are satisfied (see [12] for the proof):

$$s^{[\rho]} \to 0,\ \sum_{\rho=1}^{\infty} s^{[\rho]} = \infty,\ \sum_{\rho=1}^{\infty} (s^{[\rho]})^2 < \infty. \tag{4.26}$$

A simple choice of $s^{[\rho]}$ satisfying these conditions is $s^{[\rho]} = \frac{1}{\rho}$, but the vanishing step size in general, would lead to an extremely slow convergence rate. Finally, it is remarked that the constraint $x_i \in X_i$ in line 4 differs from the original constraint $x_i \in Conv(X_i)$ in (4.19) and (4.20), since the computation of the convexified set $Conv(X_i)$ is hard, especially for a large dimension. The authors in [161] suggest to use column generation techniques, see [18, 51], to construct approximations of $Conv(X_i)$. Here instead, as the term to be minimized in line 4 is linear, i.e., there must exist an optimal $x_i^{[\lambda^{[\rho]}]}$ located in the vertices of $Conv(X_i)$, thus also in X_i. Accordingly, the constraint $x_i \in X_i$ is adopted in line 4, since the outcome will not be affected according to Assumption 4.1. In other words, a small-scale MILP problem with local variable only is solved in line 4, instead of an LP problem requiring the knowledge of $Conv(X_i)$.

Clearly, without the steps in lines 7 to 12, Algorithm 4.1 would terminate after averaging to $\bar{\mathbf{x}}^{*,f}$ in line 5. At this stage, an improvement of $\mathbf{x}^f$ can be determined based on $\bar{\mathbf{x}}^{*,f}$. However, since the convergence towards $\bar{\mathbf{x}}^{*,f}$ usually requires many iterations due to the vanishing step size in (4.26) (and thus large computation times), the computations in lines 7 to 12 are carried out in addition, to reduce the number of necessary iterations: Any set of local variables $x_i \in X_i$ can always be decomposed into the real-valued part $x_{i,r} \in \mathbb{R}^{r_i}$ and the integer part $x_{i,t} \in \mathbb{Z}^{z_i}$. Similarly, the matrices A_i and D_i can also be decomposed into $A_i = [A_{i,r},\ A_{i,t}]$ and $D_i = [D_{i,r},\ D_{i,t}]$, so that $A_i x_i = [A_{i,r},\ A_{i,t}] \cdot [x_{i,r};\ x_{i,t}]$ and $D_i x_i = [D_{i,r},\ D_{i,t}] \cdot [x_{i,r};\ x_{i,t}]$ hold. With this scheme for the $x_i^{[\lambda^{[\rho]}]}$ obtained in line 4 in iteration ρ, the newly assigned problem (4.23) in line 7 fixes the integer part $x_{i,t}^{[\lambda^{[\rho]}]}$, and leaves the real part $x_{i,r}^{[\lambda^{[\rho]}]}$ to be newly selected. This aims at achieving the following goals through the solution of (4.23):

- reducing the global costs attained by $x_i^{[\lambda^{[\rho]}]}$, $i \in N$, through the variation of their real-valued parts;
- as $x_i^{[\lambda^{[\rho]}]} \in X_i$ applies according to line 4, the constraint (4.25) aims at preserving the local feasibility in problem (4.7) during the variation of the real variables;

- the $x_i^{[\lambda^{[\rho]}]}$ obtained in iteration ρ may, in general, violate the dualized coupling constraints (4.16) (see [161] for the reasoning, and also see the second plot in Fig. 4.4 which demonstrates the maximal violation to (4.16) in each iteration). However, as the original coupling constraints (4.8) determine a larger feasible space than (4.16) according to (4.17), the $x_i^{[\lambda^{[\rho]}]}$ may have satisfied (4.8) even if (4.16) is not satisfied (see the first plot in Fig. 4.4, recording the maximal violation to (4.8)). Thus, as the real-valued part of $x_i^{[\lambda^{[\rho]}]}$ is allowed to be newly selected in (4.23), the constraint (4.24) aims at ensuring that the feasibility of (4.8) is eventually recovered after the optimization.

Let $x_{i,r}^*$, $i \in N$, denote the optimized solution of (4.23). If problem (4.23) is feasible in iteration ρ, then a new feasible candidate $\mathbf{x}^{new} = [x_1^{new}; \cdots ; x_{n_s}^{new}]$ of (4.7) is found, in which $x_i^{new} := [x_{i,r}^*; x_{i,t}^{[\lambda^{[\rho]}]}]$, $i \in N$. If the global costs of the new candidate

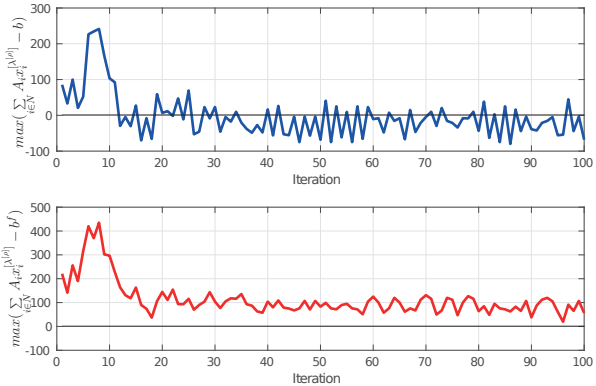

Figure 4.4.: Outcome of Algorithm 4.1 for randomly generated instances of problem (4.7): the upper plot shows the maximal violation of the original coupling constraint (4.8) by $x_i^{[\lambda^{[\rho]}]}$, $i \in N$, over the iterations and values below zero indicate that no violation is observed. The term $max(\sum_{i \in N} A_i x_i^{[\lambda^{[\rho]}]} - b)$ records the largest entry in vector $\sum_{i \in N} A_i x_i^{[\lambda^{[\rho]}]} - b$. The lower plot shows, to the contrary, the maximal violation of the tightened coupling constraint (4.16) over the iterations. Notice that $x_i^{[\lambda^{[\rho]}]}$, $i \in N$, does not satisfy the tightened coupling constraint (4.16) for the first 100 iterations by applying Algorithm 4.1, but has already satisfied the original one (4.8) for $\rho = 11$, i.e., the tightened constraint has to be assessed as overly conservative.

$J(\mathbf{x}^{new})$ are smaller than $J(\mathbf{x}^f)$, the $\mathbf{x}^{new}$ is found to be a candidate of (4.7) better than $\mathbf{x}^f$, and Algorithm 4.1 terminates (but the whole improvement process does not terminate yet, see the overall Algorithm 4.2 in the coming sections). Thus, the objective to improve $\mathbf{x}^f$ is realized before $\bar{\mathbf{x}}^{*,f}$ has converged by the additional solution of (4.23), leading to a significant reduction of the computation time, as will be shown later.

It is remarked that the satisfaction of the conditions in line 8 of Algorithm 4.1 is not guaranteed in general. However, the probability that the conditions are satisfied rises with increasing ρ: since the multiplier $\lambda^{[\rho]}$ converges towards its optimal value $\bar{\lambda}^{*,f}$ for $\rho \to \infty$, the primal variable $x_i^{[\lambda^{[\rho]}]}$, $i \in N$, will eventually differ in at most $m+1$ subsystems from $\bar{\mathbf{x}}^{*,f}$ according to Theorem 4.3. In addition, as $\bar{\mathbf{x}}^{*,f}$ also satisfies the tightened coupling constraint (4.16) and leads to low global costs according to Lemma 4.1, the $x_i^{[\lambda^{[\rho]}]}$, $i \in N$, thus also tend to satisfy the tightened coupling constraint (4.16) (and thus the original one (4.8)) and attain low global costs for $\rho \to \infty$.

At last, the solution of the LP problem (4.23) can be carried out in a centralized fashion (e.g. by a central coordinator), since the required computational effort is negligible compared to the MILP problem (4.7), given that only $\sum_{i \in N} r_i$ real variables are involved.

4.1.2. Further Improvement of the Solution Candidate

As mentioned in the last section, the conditions in line 8 of Algorithm 4.1 may never be satisfied throughout the iterations. In this case, no feasible candidate better than $\mathbf{x}^f$ is found by the algorithm, and it terminates after $\bar{\mathbf{x}}^{*,f}$ is converged. Then, a set of $|N_1| \geq n_s - m - 1$ subsystems can be determined according to Theorem 4.2, so that for $i \in N_1$ the local solution $\bar{x}_i^{*,f}$ (as contained in $\bar{\mathbf{x}}^{*,f}$) also satisfies X_i. At this stage, in order to determine a feasible candidate for the solution of (4.7) being close to $\bar{\mathbf{x}}^{*,f}$ (thus better than $\mathbf{x}^f$), the following problem is set up for the remaining $|N_2| \leq m+1$ subsystems:

$$\min_{x_i, \forall i \in N_2} \sum_{i \in N_2} c_i^{\mathrm{T}} x_i \tag{4.27}$$

$$\text{s.t.:} \sum_{i \in N_2} A_i x_i \leq b - \sum_{i \in N_1} A_i \bar{x}_i^{*,f}; \tag{4.28}$$

$$x_i \in X_i,\ x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\ i \in N_2. \tag{4.29}$$

The solution of (4.27) aims at recovering the local feasibility $x_i \in X_i$ of the remaining $|N_2|$ subsystems (through constraint (4.29)), while the feasibility of the original coupling constraint (4.8) is maintained (through constraint (4.28)). Note that a new feasible candidate of (4.7) can be determined if problem (4.27) has a feasible solution. In this case, let the optimized solution of (4.27) be denoted by $x_i^{*,rec}$, $i \in N_2$, and a new candidate of (4.7) be denoted by $\mathbf{x}^{new} = [x_1^{new}; \cdots; x_{n_s}^{new}]$, where

$x_i^{new} := \bar{x}_i^{*,f}$, $\forall i \in N_1$ and $x_i^{new} := x_i^{*,rec}$, $\forall i \in N_2$. Then, if $J(\mathbf{x}^{new})$ attains lower global costs than $J(\mathbf{x}^f)$, the $\mathbf{x}^{new}$ constitutes a candidate of (4.7) better than $\mathbf{x}^f$.

However, the feasibility of problem (4.27) can not be guaranteed in general either, in most cases due to the violation of (4.28). But the following facts indicate that the existence of a feasible solution to (4.27) is likely:

1. Due to $|N_2| \ll |N_1|$ (since $m \ll n_s$), only a small fraction of the n_s subsystems need to re-select their local share of $\bar{\mathbf{x}}^{*,f}$. This implies that the left-hand side of the coupling constraint (4.8), i.e., $\sum_{i \in N} A_i x_i = \sum_{i \in N_1} A_i x_i + \sum_{i \in N_2} A_i x_i$, will not deviate much from $\sum_{i \in N} A_i \bar{x}_i^{*,f}$, after the $|N_2|$ subsystems have re-selected their local candidates in (4.27);

2. For the new local candidates of the $|N_2|$ subsystems, the left-hand side of the coupling constraint (4.8) only has to be smaller than b instead of b_f in (4.28) (since $\sum_{i \in N} A_i \bar{x}_i^{*,f} \leq b_f \leq b$ applies according to (4.19)).

Anyhow, if (4.27) has no feasible solution, or $J(\mathbf{x}^{new}) < J(\mathbf{x}^f)$ fails to hold, no improvement can be obtained for $\mathbf{x}^f$ by the proposed method. Then, the maximal difference between $J(\mathbf{x}^f)$ and the globally optimal costs $J(\mathbf{x}^*)$ of (4.7) can be assessed by the following criteria:

Lemma 4.2. *For given $\mathbf{x}^f$, the difference between $J(\mathbf{x}^f)$ and $J(\mathbf{x}^*)$ is bounded by:*

$$J(\mathbf{x}^f) - J(\mathbf{x}^*) \leq J(\mathbf{x}^f) - J(\bar{\mathbf{x}}^{*,f}) + \bar{\lambda}^{*,f,T}(b - b_f). \tag{4.30}$$

Proof. According to the relations listed in Lemma 4.1 the following applies:

$$J(\mathbf{x}^f) - J(\mathbf{x}^*) \leq J(\mathbf{x}^f) - J(\bar{\mathbf{x}}^*) = J(\mathbf{x}^f) - J(\bar{\mathbf{x}}^{*,f}) + J(\bar{\mathbf{x}}^{*,f}) - J(\bar{\mathbf{x}}^*). \tag{4.31}$$

As the LP problem (4.19) is transformed into (4.11) by perturbing the dualized constraints according to (4.17), the difference between their optimal costs are thus bounded by:

$$J(\bar{\mathbf{x}}^{*,f}) - J(\bar{\mathbf{x}}^*) \leq \bar{\lambda}^{*,f,T}(b - b_f). \tag{4.32}$$

See Chapter 5.6 in [36] for a detailed explanation of this inequality. By substituting inequality (4.32) into (4.31), the relation (4.30) is obtained. □

As $\bar{\lambda}^{*,f}$ and $\bar{\mathbf{x}}^{*,f}$ have been both determined through Algorithm 4.1, the value of the right-hand side of (4.30) can be directly calculated, which gives an upper bound of the performance loss of $\mathbf{x}^f$ compared to the global optimum $\mathbf{x}^*$.

4.1.3. The Overall Solution-Improvement Procedure

As the last two paragraphs have explained, a better candidate $\mathbf{x}^{new}$ can be obtained by different mechanisms, and Algorithm 4.2 shows how these mechanisms are combined to an overall procedure. This procedure does not require conservative assumptions as established in previous work, i.e. the applicability of the proposed method is significantly enhanced. Note that, in the best case for the computation time, $\mathbf{x}^{new}$ is found when the conditions in line 8 of Algorithm 4.1 are satisfied for the first time. Then, Algorithm 4.1 does not have to be executed until the primal/dual optimum is reached. If these conditions are not satisfied in any iteration of Algorithm 4.1, $\mathbf{x}^{new}$ is determined by solving (4.27). Only if no feasible solution exists for (4.27) either, or the relation $J(\mathbf{x}^{new}) < J(\mathbf{x}^f)$ fails to hold, no candidate better than $\mathbf{x}^f$ is found, and the performance loss of $\mathbf{x}^f$ compared to $\mathbf{x}^*$ is checked by use of (4.30).

Finally, if a better candidate $\mathbf{x}^{new}$ is found, the value of $\mathbf{x}^f$ is then set equal to $\mathbf{x}^{new}$, and the whole computation procedure, starting from line 2 in Algorithm 4.2, is carried out once more. This scheme enables further improvement of the solution candidate, as the properties established in Lemma 4.1 are valid for any candidate

1: **Initialization:** $\mathbf{x}^f$, $flag = 0$;
2: **while** $flag = 0$ **do**
3: determine coupling constraint (4.16) with $\mathbf{x}^f$
4: formulate (4.18), (4.19) and (4.20) with (4.16)
5: run Algorithm 4.1 and:
6: **if** the conditions in line 8 of Algorithm 4.1 are satisfied before ρ^{max} is reached **then**
7: a better candidate $\mathbf{x}^{new}$ is found and set:

$$\mathbf{x}^f := \mathbf{x}^{new}$$

8: **else**
9: solve (4.27)
10: **if** a feasible candidate $\mathbf{x}^{new}$ exists for (4.27) and if it satisfies: $J(\mathbf{x}^{new}) < J(\mathbf{x}^f)$ **then**
11: set: $\mathbf{x}^f := \mathbf{x}^{new}$
12: **else**
13: $flag = 1$ and an upper bound of the performance loss can be evaluated according to (4.30)
14: **end if**
15: **end if**
16: **end while**

Algorithm 4.2.: The overall improvement procedure.

$\mathbf{x}^f$, no matter how near it is located to the global optimum $\mathbf{x}^*$. The whole procedure stops if no improvement is found after one repetition in Algorithm 4.2, or an upper bound on the computation time is reached.

4.1.4. Numerical Examples

In this section, the proposed distributed solution was tested for various MILP problems (4.7) of different size. The local cost function c_i, local constraints X_i, and the coupling constraints (4.8) are randomly generated in each test, while instances without feasible solutions are discarded. In addition, no assumption on $A_i x_i \geq 0$, $\forall x_i \in X_i$, $\forall i \in N$ is imposed, as done in [162], nor is any feasibility assumption used for the tightened problem, as in [161].

In the first test instance, a number of $n_s = 40$ subsystems was considered, each with $z_i = r_i = 15$ integer and real variables. The number of coupling constraints was $m = 5$. For comparison purposes, the centralized solution of this problem, which involves in total 600 integer variables, was found in 336 seconds by using the solver CPLEX [48] on a 3.4GHZ processor, and the optimal cost was $J(\mathbf{x}^*) = -2.17 \cdot 10^5$. However, the first feasible candidate was determined already after only 3.34 seconds, but with a cost of $J(\mathbf{x}^f) = -211.92$. By employing the proposed distributed solution to improve $\mathbf{x}^f$, a better candidate $\mathbf{x}^{new,1}$ was found after only 6 iterations in Algorithm 4.1 (0.35 seconds), with a cost of $J(\mathbf{x}^{new,1}) = -2.14 \cdot 10^5$, i.e. a performance loss of only 1.13% compared to $J(\mathbf{x}^*)$. Then, starting from $\mathbf{x}^{new,1}$ and executing a further iteration of Algorithm 4.2, an even better candidate $\mathbf{x}^{new,2}$ with $J(\mathbf{x}^{new,2}) = -2.16 \cdot 10^5$ was found after only 5 iterations in Algorithm 4.1 (which is embedded in Algorithm 4.2), taking 0.48 seconds. In this iteration, the performance loss was further reduced to 0.46%. Thereafter, no further improvement could be made.

For a larger problem instance with $n_s = 80$ subsystems, each with $z_i = r_i = 25$ integer and real variables (leading to overall 2000 integer variables in the centralized problem), and with $m = 8$ coupling constraints, the global optimum could not be found by centralized solution within one hour using CPLEX, but it only took 77 seconds to find the first feasible candidate $\mathbf{x}^f$ with $J(\mathbf{x}^f) = -2.32 \cdot 10^3$. The proposed method then generated a better candidate $\mathbf{x}^{new,1}$ with $J(\mathbf{x}^{new,1}) = -6.04 \cdot 10^5$ after 6 iterations (6.27 seconds) in Algorithm 4.1. By employing a further iteration of Algorithm 4.2 for $\mathbf{x}^{new,1}$, an even better candidate $\mathbf{x}^{new,2}$ was found after 7 iterations in Algorithm 4.1 (in 9.06 seconds) and with $J(\mathbf{x}^{new,2}) = -6.07 \cdot 10^5$. Further improvements were not found, but the difference between $J(\mathbf{x}^{new,2})$ and $J(\mathbf{x}^*)$ was bounded by:

$$J(\mathbf{x}^{new,2}) - J(\mathbf{x}^*) \leq 0.034 \cdot 10^5 \tag{4.33}$$

according to (4.30). Thus, although it is hard to compute the optimal cost $J(\mathbf{x}^*)$ due to the high computational complexity, one can still ensure at most 0.55% performance loss for $J(\mathbf{x}^{new,2})$ through (4.33).

Table 4.1.: Numerical experiments for different MILP problems with T indicating the time required for the solution of $\mathbf{x}^f$, $\mathbf{x}^{new,1}$ and $\mathbf{x}^*$, while $J(\mathbf{x}^*) = -$ indicates that the global optimum was not found within given time limit.

N, z_i/r_i	m	$T_{\mathbf{x}^f}$	$J(\mathbf{x}^f)$	$T_{\mathbf{x}^{new,1}}$	$J(\mathbf{x}^{new,1})$	$T_{\mathbf{x}^*}$	$J(\mathbf{x}^*)$
10, 10	3	$0.20sec$	335	$0.11sec$	$-3.27 \cdot 10^4$	$12.23sec$	$-3.40 \cdot 10^4$
10, 40	3	$4.99sec$	$2.89 \cdot 10^3$	$2.37sec$	$-1.05 \cdot 10^5$	$> 20min$	$-$
20, 10	4	$0.53sec$	-680	$0.18sec$	$-8.29 \cdot 10^4$	$3.59sec$	$-8.29 \cdot 10^4$
20, 30	4	$4.43sec$	$-2.04 \cdot 10^3$	$0.76sec$	$-1.37 \cdot 10^5$	$> 20min$	$-$
40, 10	6	$1.28sec$	$-1.65 \cdot 10^3$	$0.39sec$	$-1.67 \cdot 10^5$	$67.97sec$	$-1.68 \cdot 10^5$
40, 20	8	$11.58sec$	$4.66 \cdot 10^3$	$1.98sec$	$-2.53 \cdot 10^5$	$> 20min$	$-$
80, 5	5	$0.62sec$	$5.93 \cdot 10^3$	$0.18sec$	$-1.93 \cdot 10^5$	$3.47sec$	$-2.01 \cdot 10^5$
80, 8	7	$1.36sec$	$-1.80 \cdot 10^3$	$0.59sec$	$-2.50 \cdot 10^5$	$416sec$	$-2.59 \cdot 10^5$
160, 4	8	$0.60sec$	$3.00 \cdot 10^3$	$0.10sec$	$-2.28 \cdot 10^5$	$21.40sec$	$-3.23 \cdot 10^5$
160, 20	8	$65sec$	394	$2.52sec$	$-6.00 \cdot 10^5$	$> 20min$	$-$
200, 5	9	$2.70sec$	$-5.58 \cdot 10^3$	$0.22sec$	$-3.21 \cdot 10^5$	$72sec$	$-5.03 \cdot 10^5$
300, 20	15	$231sec$	$-1.85 \cdot 10^3$	$18sec$	$-8.19 \cdot 10^5$	$> 20min$	$-$
500, 10	10	$23sec$	$9.26 \cdot 10^3$	$26sec$	$-2.01 \cdot 10^6$	$> 20min$	$-$

In the third test instance, a number of $n_s = 200$ subsystems was considered, and $z_i = r_i = 10$, $m = 12$. The global optimum could again not be found by centralized computation within one hour, but the first feasible candidate $\mathbf{x}^f$ with $J(\mathbf{x}^f) = -1.69 \cdot 10^4$ was obtained in only 7.23 seconds. The proposed method produced a better candidate $\mathbf{x}^{new,1}$ with $J(\mathbf{x}^{new,1}) = -7.57 \cdot 10^5$ after 2.09 seconds. No better candidate was found afterwards, and the maximal performance loss compared to $J(\mathbf{x}^*)$ was bounded by 3.56% according to (4.30).

The tests above have shown that the proposed method in all cases achieves drastic improvement of $\mathbf{x}^f$ within a very short computation time (in particular with the

first iteration of Algorithm 4.2). The obtained candidates attain global costs that are only slightly worse than the global optima. Even for the case that the global optimum cannot be determined in centralized fashion due to the high complexity of (4.7), the bound defined in (4.30) still enables one to evaluate the obtained candidate. A set of additional tests is listed in Table 4.1.

4.2. Efficient Distributed Solution for MIQP Problems

In the last section, the focus was put on the distributed solution of MILP problems. However, a quadratic cost function is more often applied in optimal control problems, especially in Model Predictive Control (MPC) problems. Thus, an MIQP problem arises with a form of:

$$\begin{aligned} & \min_{x_1,\cdots,x_{n_s}} \sum_{i\in N} x_i^{\mathrm{T}} Q_i x_i + c_i^{\mathrm{T}} x_i \\ \text{s.t.:} \quad & \sum_{i\in N} A_i x_i \leq b; \\ & x_i \in X_i, \quad x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i}, \quad i \in N. \end{aligned} \tag{4.34}$$

Recall that Problems 2.2, 2.3 and 3.5, which appeared in previous chapters, are all with quadratic cost functions. Here it is assumed that the matrix Q_i being positive-definite and diagonal, while the other notation shares the same meaning as in (4.7). For this problem, however, it has to be noticed that the distributed solution of the MILP problem (4.7) cannot be applied here. This is because Theorem 4.2 does not hold if the cost function is quadratic: Note for the proof of Theorem 4.2, the following two sets:

$$S_i := \left\{ s_i \in \mathbb{R}^{m+1} \,\middle|\, s_i = \begin{bmatrix} J_i(x_i) \\ A_i x_i \end{bmatrix},\ x_i \in X_i \right\} \tag{4.35}$$

and:

$$\tilde{S}_i := \left\{ s_i \in \mathbb{R}^{m+1} \,\middle|\, s_i = \begin{bmatrix} J_i(x_i) \\ A_i x_i \end{bmatrix},\ x_i \in Conv(X_i) \right\} \tag{4.36}$$

are required to satisfy the relation:

$$Conv(S_i) = \tilde{S}_i. \tag{4.37}$$

Here, the vector $\begin{bmatrix} J_i(x_i) & A_i x_i \end{bmatrix}^{\mathrm{T}}$ is usually referred to as the „cost-constraint" pair according to [24]. Apparently, when the cost function is linear, i.e., $J_i(x_i) = c_i^{\mathrm{T}} x_i$, the relation in (4.37) is satisfied due to the linear mapping from x_i to $s_i = \begin{bmatrix} c_i^{\mathrm{T}} x_i \\ A_i x_i \end{bmatrix}$.

However, if the cost function $J_i(x_i)$ is quadratic, as the one in (4.34), the relation in (4.37) is no longer guaranteed to hold. As a result, Theorem 4.2 does not apply to the MIQP problem (4.34) in general.

Notice that not many studies on the distributed solution of MIQP problems can be found in the literature, although this kind of problem appears in many domains: In [71], a distributed control strategy for coupled PWA systems was proposed, but the quality of the obtained local optima (i.e. its deviation from the global optimum) is not discussed there. In [63], a distributed computation process was proposed, in which a lower and upper value bound of the MIQP problem were determined, but feasibility was not thoroughly investigated there. In addition, the integer variables there are also limited to binary ones. In [34], a distributed MPC algorithm for networked hybrid systems was proposed, which indirectly solved an MIQP problem. However, the method is limited to a specific class of hybrid systems, leading to a special structure of the MIQP problem and allowing only for a small number of subsystems. In the more recent work [155], the alternating direction method of multipliers (ADMM [35]), which is typically applied to the distributed solution of convex optimization problems, is embedded into the distributed solution process of MIQP problems. However, only an approximated solution (without feasibility guarantee) can be obtained and heuristic steps are required in this work. In [153], the authors considered the distributed solution of MIP problems with convex cost function. There, the subsystems are coupled by a common cost function (leading to a mixed-integer consensus problem), and a method called the „projected subgradient algorithm" [4] for convex programs is adopted to the mixed-integer case.

Given the fact that the distributed solution of the MILP problem (4.7) cannot be extended to the MIQP problem (4.34) (despite the similar structure), a novel multi-stage distributed solution is proposed here for the latter problem. The solution process again starts from a feasible candidate $\mathbf{x}^f = [x_1^f; \cdots; x_{n_s}^f]$ (Note that the feasible candidate of (4.7) is also feasible for (4.34), since both problems share the same solution space). Different methods are then proposed to improve $\mathbf{x}^f$ successively. In particular, the improvement of $\mathbf{x}^f$ is carried out in a distributed form and in different stages, using the following statements on the optimal condition of problem (4.34):

- **Condition 1**: if $\mathbf{x}^f$ is the global optimum of (4.34), then no subsystem can unilaterally reduce its local costs without changing the solution of the other subsystems;
- **Condition 2**: if $\mathbf{x}^f$ is the global optimum of (4.34), then by fixing the integer part of $\mathbf{x}^f$ and re-optimizing the real part, the global costs of $\mathbf{x}^f$ cannot be reduced;
- **Condition 3**: if $\mathbf{x}^f$ is the global optimum of (4.34), then no such increment vector $\Delta\mathbf{x}$ exists, that $\mathbf{x}^f + \Delta\mathbf{x}$ is feasible for (4.34) and leads to lower global

costs than $\mathbf{x}^f$.

Using these statements, the distributed solution of (4.34) is realized by employing three different stages to improve $\mathbf{x}^f$, and moves from one stage to the next if the associated terminal condition is satisfied. The feasibility of the candidates generated over the stages is ensured in this method, as well as the continuous reduction of the global costs. For the candidate obtained in the last stage, the solution process will start from stage one once more, in order to realize further improvement (similar to the scheme in Algorithm 4.2). The whole process terminates when no better candidate is found in any stage, leading often to only moderate optimality gaps.

4.2.1. The Three-Stage Distributed Solution

Similar to the MILP problem (4.7), it is further assumed that the relation $m \ll n_s$ applies in (4.34), i.e., the number of subsystems is much larger than the one of the coupling constraints. In addition, $J_i(x_i) = x_i^{\mathrm{T}} Q_i x_i + c_i^{\mathrm{T}} x_i$ is used to denote the local costs of subsystem i and $J(\mathbf{x}) = \sum_{i \in N} J_i(x_i)$ for the global costs of the global candidate $\mathbf{x}$. The optimal solution of (4.34) is denoted by $\mathbf{x}^* = [x_1^*; \cdots; x_{n_s}^*]$ together with the optimal costs $J(\mathbf{x}^*)$.

Stage One: Update of the Local Variables in $\mathbf{x}^f$

For a given feasible $\mathbf{x}^f$ of (4.34), if this candidate still deviates from the global optimum $\mathbf{x}^*$, then some subsystems in N may[3] be able to reduce their local costs according to **Condition 1**. This is realized by fixing the candidate of other subsystems in $\mathbf{x}^f$, and re-select a local candidate by solving the following problem:

$$\begin{aligned} & \min_{x_i} J_i(x_i) \\ \text{s.t.:} \quad & A_i x_i \leq b - \sum_{j \in N^{/i}} A_j x_j^f; \\ & x_i \in X_i, \quad x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i}. \end{aligned} \tag{4.38}$$

Here, the symbol $N^{/i}$ denotes the set of subsystems in N without i. Note that, as only local variables are involved in this problem, leading to a small-scale MIQP, its solution is assumed to be tractable. In addition, the following properties of (4.38) can be observed:

- The feasibility of (4.38) is always maintained.
- The optimal solution $\hat{x}_i$ of (4.38) satisfies $J_i(\hat{x}_i) \leq J_i(x_i^f)$.

[3] As **Condition 1** only states a necessary optimal condition of (4.34), there may exist the case that it is satisfied by some candidates other than the optimum.

- By replacing the previous local candidate x_i^f with the optimized one $\hat{x}_i$ in $\mathbf{x}^f$, and keeping the local candidate of other subsystems unchanged, the updated global candidate $\hat{\mathbf{x}}^f$ is still feasible for (4.34) and satisfies $J(\hat{\mathbf{x}}^f) \leq J(\mathbf{x}^f)$.

The first property is based on the fact that x_i^f itself is a feasible candidate of (4.38). The second property follows from the fact that x_i^f represents a feasible candidate of (4.38), but not necessarily the optimal one. The third property is a direct consequence of the first two properties, since the local costs of any other subsystem in $N^{/i}$ are not changed.

Based on these properties, Algorithm 4.3 is proposed to improve the global candidate of (4.34) iteratively from $\mathbf{x}^f$, using an iteration counter ρ. In this algorithm, the solution of the local problems (4.38) are carried out in parallel in each iteration. In addition, the update scheme of $\mathbf{x}^{[\rho]}$, in which only the subsystem with the largest cost reduction is updated, guarantees the feasibility of $\mathbf{x}^{[\rho]}$ over the iterations, as well as the continuous reduction of the global costs. The algorithm terminates if no subsystem is able to reduce its local costs in an iteration, i.e., if **Condition 1** is satisfied. The global candidate of (4.34) obtained in the terminal iteration is then denoted by $\mathbf{x}^{S1} = [x_1^{S1}; \cdots; x_{n_s}^{S1}]$.

However, although the feasibility is ensured in each iteration of Algorithm 4.3, the update scheme, that only one subsystem is permitted to improve its local solution,

1: **Initialization:** $\rho = 0$, feasible $\mathbf{x}^f$ (known to all subsystems), $\mathbf{x}^{[\rho]} := \mathbf{x}^f$;
2: **while** exists such a subsystem $i \in N$ that $J_i(\hat{x}_i^{[\rho]}) < J_i(x_i^{[\rho]})$ **do**
3: **for** all subsystem $i \in N$ **do**
4: solve

$$\begin{aligned} & \hat{x}_i^{[\rho]} := \arg\min_{x_i} J_i(x_i) \\ \text{s.t.:} \quad & A_i x_i \leq b - \sum_{j \in N^{/i}} A_j x_j^{[\rho]}; \\ & x_i \in X_i, \quad x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i}. \end{aligned}$$

5: broadcast optimized $\hat{x}_i^{[\rho]}$ as well as local cost reduction $J_i(\hat{x}_i^{[\rho]}) - J_i(x_i^{[\rho]})$ to all subsystems;
6: update the local solution for subsystem $\tilde{i}$ with the largest local cost reduction in $\mathbf{x}^{[\rho]}$, and keep the solution of other subsystems unchanged;
7: **end for**
8: $\mathbf{x}^{[\rho+1]} := \mathbf{x}^{[\rho]}$;
9: $\rho := \rho + 1$;
10: **end while**

Algorithm 4.3.: The distributed improvement process in stage one without central coordinator $\mathcal{C}$.

1: **Initialization:** $\rho = 0$, feasible $\mathbf{x}^f$ (known to all subsystems and the central coordinator $\mathcal{C}$), $\mathbf{x}^{[\rho]} := \mathbf{x}^f$;
2: **while** exists such a subsystem $i \in N$ that $J_i(\hat{x}_i^{[\rho]}) < J_i(x_i^{[\rho]})$ **do**
3: **for** all subsystem $i \in N$ **do**
4: solve

$$\begin{aligned} & \hat{x}_i^{[\rho]} := \arg\min_{x_i} J_i(x_i) \\ \text{s.t.:} \quad & A_i x_i \leq b - \sum_{j \in N/i} A_j x_j^{[\rho]}; \\ & x_i \in X_i, \quad x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i}. \end{aligned}$$

5: send optimized $\hat{x}_i^{[\rho]}$ as well as local cost reduction $J_i(\hat{x}_i^{[\rho]}) - J_i(x_i^{[\rho]})$ to $\mathcal{C}$;
6: **end for**
7: introduce a vector of binary variables $\mathbf{r} \in \{0,1\}^{n_s \times 1}$, and let $\mathcal{C}$ solve the following problem:

$$\begin{aligned} & \min_{\mathbf{r}} \sum_{i=1}^{n_s} \mathbf{r}(i) \cdot (J_i(\hat{x}_i^{[\rho]}) - J_i(x_i^{[\rho]})) \\ \text{s.t.:} \quad & \mathbf{r} \in \{0,1\}^{n_s \times 1}, \\ & \sum_{i \in N} A_i(x_i^{[\rho]} + \mathbf{r}(i) \cdot (\hat{x}_i^{[\rho]} - x_i^{[\rho]})) \leq b \end{aligned} \tag{4.39}$$

8: if $\mathbf{r}^*$ denotes the solution of (4.39), then replace $x_i^{[\rho]}$ with $\hat{x}_i^{[\rho]}$ in $\mathbf{x}^{[\rho]}$ for all subsystems $i \in \{1, \cdots, n_s\}$ with $\mathbf{r}^*(i) = 1$;
9: $\mathbf{x}^{[\rho+1]} := \mathbf{x}^{[\rho]}$;
10: $\rho := \rho + 1$;
11: **end while**

Algorithm 4.4.: The distributed improvement process in stage one with central coordinator $\mathcal{C}$.

may lead to only minor improvements of the global candidate in each iteration, i.e., a slow convergence towards termination. Thus, in order to achieve faster convergence, an alternative algorithm with a central coordinator $\mathcal{C}$ is proposed, which decides upon a subset of subsystems (larger than one) for improving a part of the global candidate in each iteration, see Algorithm 4.4.

In Algorithm 4.4, the local problems (4.38) are first solved in parallel for all subsystems in each iteration. Then, the subsystems send their optimized solution to the central coordinator $\mathcal{C}$ and let $\mathcal{C}$ decide who should update the local solution in $\mathbf{x}^{[\rho]}$ (major difference compared to Algorithm 4.3). The new optimization problem (4.39) solved by $\mathcal{C}$ allows for a larger improvement comparing to Algorithm 4.3: for

$\mathbf{r}(i) = 1$, i.e., the i-th element of $\mathbf{r}$ equals to one, the relation:

$$x_i^{[\rho]} + \mathbf{r}(i) \cdot (\hat{x}_i^{[\rho]} - x_i^{[\rho]}) = \hat{x}_i^{[\rho]}$$

applies, as well as:

$$\mathbf{r}(i) \cdot (J_i(\hat{x}_i^{[\rho]}) - J_i(x_i^{[\rho]})) = J_i(\hat{x}_i^{[\rho]}) - J_i(x_i^{[\rho]}).$$

This implies that the optimized solution of (4.38) is adopted for the global candidate $\mathbf{x}^{[\rho]}$, and the adoption also enables a reduction of the global cost. Otherwise, if $\mathbf{r}(i) = 0$, the relation $x_i^{[\rho]} + \mathbf{r}(i) \cdot (\hat{x}_i^{[\rho]} - x_i^{[\rho]}) = x_i^{[\rho]}$ applies, together with $\mathbf{r}(i) \cdot (J_i(\hat{x}_i^{[\rho]}) - J_i(x_i^{[\rho]})) = 0$. In this case, the local subsystem i must preserve its previous local solution $x_i^{[\rho]}$ in $\mathbf{x}^{[\rho]}$.

Accordingly, the solution of (4.39) aims at finding the best combination of previous and currently optimized solutions over the subsystems, i.e., more than one subsystem is allowed to update its local solution within one iteration. As a result, the global cost $J(\mathbf{x}^{[\rho]})$ is reduced maximally with respect to the outcome of the local problems (4.38), while satisfying the coupling constraints. Note that only a number of n_s binary variables (but no real-valued decision variable) is involved in (4.39), i.e., the problem constitutes an Integer Linear Programming (ILP) problem, which can be solved in very short time for the size typically obtained for the instances under study. Moreover, the feasibility of (4.39) can also be guaranteed, since the choice $\mathbf{r} = 0^{n_s \times 1}$ always establishes a feasible solution. Finally, it must be emphasized that both Algorithm 4.3) and 4.4) terminate when **Condition 1** is satisfied, and the global candidate of (4.34) obtained in the terminal iteration of Algorithm 4.4) is further denoted by $\mathbf{x}^{S1} = [x_1^{S1}; \cdots; x_{n_s}^{S1}]$. Which algorithm should be used during application depends on whether a central coordinator can be provided to the distributed system.

Stage Two: Update of the Real Variables in x^{S1}

Note that **Condition 1** does not coincide with **Condition 2** in general: the former establishes a property of the local variables in the global optimum $\mathbf{x}^*$, whereas the latter describes a property of the global continuous variables in $\mathbf{x}^*$. This difference indicates that the candidate $\mathbf{x}^{S1}$, satisfying **Condition 1**, may fail to satisfy **Condition 2** in case $\mathbf{x}^{S1} \neq \mathbf{x}^*$. Accordingly, a method to decompose the real and integer part of the candidate $\mathbf{x}^{S1}$, which has been similarly applied in Algorithm 4.1, is adopted here to improve the $\mathbf{x}^{S1}$.

First of all, the local candidate x_i^{S1} in $\mathbf{x}^{S1}$, $i \in N$, is decomposed into the real part $x_{i,r}^{S1} \in \mathbb{R}^{r_i}$ and the integer part $x_{i,t}^{S1} \in \mathbb{Z}^{z_i}$, and the matrices A_i and D_i are also decomposed into $A_i = [A_{i,r},\, A_{i,t}]$ and $D_i = [D_{i,r},\, D_{i,t}]$. Then, by fixing the integer part $x_{i,t}^{S1}$, $i \in N$, and re-optimizing the real part in $\mathbf{x}^{S1}$, a global candidate with lower costs may be obtained as long as $\mathbf{x}^{S1} \neq \mathbf{x}^*$. This re-optimization procedure

is realized by solving a QP problem:

$$\min_{x_{1,r},\cdots,x_{n_s,r}} \sum_{i\in N} J_i([x_{i,r},\, x_{i,t}^{S1}]) \tag{4.40}$$

$$\text{s.t.:} \quad \sum_{i\in N} A_{i,r}\cdot x_{i,r} \le b - \sum_{i\in N} A_{i,t}\cdot x_{i,t}^{S1}; \tag{4.41}$$

$$D_{i,r}\cdot x_{i,r} \le d_i - D_{i,t}\cdot x_{i,t}^{S1}, \quad x_{i,r}\in\mathbb{R}^{r_i},\, i\in N. \tag{4.42}$$

Note that the feasibility of (4.40) always holds, since $x_{i,r} := x_{i,r}^{S1}$, $i \in N$, is a feasible one. For the optimized solution $[x_{1,r}^{new},\cdots,x_{n_s,r}^{new}]$ of (4.40), the relation

$$\sum_{i\in N} J_i([x_{i,r}^{new};\, x_{i,t}^{S1}]) \le J(\mathbf{x}^{S1})$$

also applies according to **Condition 2**, which indicates that an improvement of candidate $\mathbf{x}^{S1}$ is achieved. Most importantly, as (4.40) represents a convex QP problem in the form of (4.2), its solution can be carried out in a distributed way by using the Lagrangian-based dual method. Alternatively, as (4.40) represents a QP problem only and the required computational effort is far less than the original MIQP problem (4.34), one can also choose to solve (4.40) in a centralized fashion if a central coordinator is provided. After the optimal solution $x_{i,r}^{new}$, $i \in N$, of (4.40) is obtained, a better global candidate $\mathbf{x}^{S2} = [x_1^{S2};\cdots;x_{n_s}^{S2}]$ is found, where $x_i^{S2} = [x_{i,r}^{new};\, x_{i,t}^{S1}]$, $i \in N$.

Stage Three: Improve $\mathbf{x}^{S2}$ Through an MILP Problem

In the previous stages, the local mixed-integer variables of any subsystem are first optimized in stage one, and the global real variables are further optimized in stage two. For the $\mathbf{x}^{S2}$ obtained at the end of stage two, however, the global mixed-integer variables have not yet been optimized simultaneously in any previous stage – this may cause $\mathbf{x}^{S2}$ to still deviate from the global optimum $\mathbf{x}^*$. Accordingly, an MILP problem based on **Condition 3** is constructed in this stage, in order to improve all variables in $\mathbf{x}^{S2}$ simultaneously. This MILP problem takes a similar form as (4.7), so that the distributed solution developed for the latter problem in Section 4.1 can be applied to accelerate the computation in this stage.

First of all, the global cost function in (4.34) is rewritten into: $J(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{C}^{\mathrm{T}}\mathbf{x}$, where $\mathbf{Q} = \mathbf{Diag}(Q_1,\cdots,Q_{n_s})$ and $\mathbf{C} = [c_1;\cdots;c_{n_s}]$. Then, the increment vector $\Delta\mathbf{x}^*$, which denotes the difference between $\mathbf{x}^*$ and $\mathbf{x}^{S2}$, can be determined by:

$$\Delta\mathbf{x}^* := \underset{\Delta\mathbf{x}=[\Delta x_1;\cdots;\Delta x_{n_s}]}{\arg\min} J(\mathbf{x}^{S2}+\Delta\mathbf{x}) - J(\mathbf{x}^{S2}) \tag{4.43}$$

$$\text{s.t.:} \quad \sum_{i\in N} A_i\Delta x_i \le b - \sum_{i\in N} A_i x_i^{S2}; \tag{4.44}$$

$$D_i\Delta x_i \le d_i - D_i x_i^{S2},\ \Delta x_i\in\mathbb{R}^{r_i}\times\mathbb{Z}^{z_i},\ i\in N. \tag{4.45}$$

Note that (4.43) still represents an MIQP problem, since its cost function can be reformulated into: $\Delta\mathbf{x}^{\mathrm{T}}\mathbf{Q}\Delta\mathbf{x}+\mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x}$ with $\mathbf{f}(\mathbf{x}^{S2}) := 2\mathbf{x}^{S2\mathrm{T}}\mathbf{Q}+\mathbf{C}^{\mathrm{T}}$. In addition, the number of variables involved in (4.43) is also the same as in the original problem (4.34), i.e., the two problems share the same complexity class. This makes the direct computation of $\Delta\mathbf{x}^*$ impossible, and an approximation strategy is used in the sequel to solve (4.43).

In the first step of the approximation, only the linear term $\mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x}$ is left to be minimized in (4.43). This is because the condition:

$$\Delta\mathbf{x}^{\mathrm{T}}\mathbf{Q}\Delta\mathbf{x} + \mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x} \leq 0 \tag{4.46}$$

must be satisfied for $\Delta\mathbf{x}^*$, as $J(\mathbf{x}^{S2}+\Delta\mathbf{x})$ must attain lower or the same global costs than or as $J(\mathbf{x}^{S2})$. Then, as the matrix $\mathbf{Q}$ is positive-definite, the linear term $\mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x}$ in (4.46) thus must be negative and dominate the quadratic term, in order to fulfill this inequality. This explains why only minimizing the linear term $\mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x}$ is a meaningful approach.

However, by only taking the linear term into account in (4.43), the obtained $\Delta\mathbf{x}$ minimizing $\mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x}$ may eventually fail to satisfy the inequality in (4.46) (and thus fails to reduce the global costs). Hence, an additional constraint $\Delta\mathbf{x} \in \varepsilon^{S2}_{\Delta\mathbf{x}}$ has to be added to the problem in (4.43), where $\varepsilon^{S2}_{\Delta\mathbf{x}}$ denotes a mixed-integer ellipsoidal set of $\Delta\mathbf{x}$:

$$\begin{aligned}\varepsilon^{S2}_{\Delta\mathbf{x}} = \{&\Delta x_i \in \mathbb{R}^{r_i}\times\mathbb{Z}^{z_i},\, i\in N| \\ &(\Delta\mathbf{x}+\frac{1}{2}\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1})^{\mathrm{T}}\frac{4\mathbf{Q}}{\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}\mathbf{f}(\mathbf{x}^{S2})^{\mathrm{T}}}(\Delta\mathbf{x}+\frac{1}{2}\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}) \leq 1\}.\end{aligned} \tag{4.47}$$

Note that this ellipsoidal set describes exactly the same feasible space as in (4.46), and thus for any $\Delta\mathbf{x}$ selected from set $\varepsilon^{S2}_{\Delta\mathbf{x}}$, the inequality in (4.46) is always satisfied. Now, by only minimizing the linear term $\mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x}$ and taking the additional constraint $\Delta\mathbf{x} \in \varepsilon^{S2}_{\Delta\mathbf{x}}$ into account, the problem (4.43) is cast into:

$$\Delta\tilde{\mathbf{x}} := \underset{\Delta\mathbf{x}=[\Delta x_1;\cdots;\Delta x_{n_s}]}{\arg\min} \mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x} \tag{4.48}$$

$$\text{s.t.:} \quad \sum_{i\in N} A_i\Delta x_i \leq b - \sum_{i\in N} A_i x_i^{S2}; \tag{4.49}$$

$$D_i\Delta x_i \leq d_i - D_i x_i^{S2},\ \Delta x_i \in \mathbb{R}^{r_i}\times\mathbb{Z}^{z_i},\ i\in N; \tag{4.50}$$

$$\Delta\mathbf{x} \in \varepsilon^{S2}_{\Delta\mathbf{x}}. \tag{4.51}$$

For the new optimal solution $\Delta\tilde{\mathbf{x}}$ resulting from problem (4.48), which is an approximation of $\Delta\mathbf{x}^*$, the feasibility of the new global candidate $\mathbf{x}^{S2}+\Delta\tilde{\mathbf{x}}$ is ensured by (4.49) and (4.50), as well as the global costs reduction by (4.51).

However, although the cost function in (4.48) is linear, its solution process is still quite complicated because of the quadratic constraint (4.51). An intuitive approach

to address this problem is to approximate (4.51) by a linear one. Following this line, one can use a mixed-integer polyhedral set $P^{S2}_{\Delta\mathbf{x}}$, where:

$$P^{S2}_{\Delta\mathbf{x}} = \{\Delta x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\, i \in N |\, D^{in}\Delta\mathbf{x} \leq d^{in}\}$$

inner-approximates the ellipsoidal set $\varepsilon^{S2}_{\Delta\mathbf{x}}$. By temporarily neglecting the explicit value of D^{in} and d^{in} in $P^{S2}_{\Delta\mathbf{x}}$, but simply assuming that $P^{S2}_{\Delta\mathbf{x}} \subseteq \varepsilon^{S2}_{\Delta\mathbf{x}}$ applies, an MILP problem in a similar form to (4.7) can be obtained to approximate problem (4.48):

$$\Delta\tilde{\mathbf{x}} := \underset{\Delta\mathbf{x}=[\Delta x_1;\cdots;\Delta x_{n_s}]}{\arg\min} \mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x} \tag{4.52}$$

$$\text{s.t.:} \quad \sum_{i\in N} A_i\Delta x_i \leq b - \sum_{i\in N} A_i x_i^{S2}; \tag{4.53}$$

$$D_i\Delta x_i \leq d_i - D_i x_i^{S2},\ \Delta x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\ i \in N; \tag{4.54}$$

$$\Delta\mathbf{x} \in P^{S2}_{\Delta\mathbf{x}}. \tag{4.55}$$

Here, (4.53) and (4.55) are the coupling constraints and (4.54) is the local one. According to the discussion in the last section, one knows that for MILP problems in this structure, the Shapley-Folkman-Starr theorem can be applied to guide the distributed solution. This theorem, however, also requires the number of coupling constraints in the MILP problem be much smaller than the subsystems. Back to the problem (4.52), the number of coupling constraints is $m + \tau$, where m is the dimension of vector b (given, known as $m \ll n_s$) and τ is the dimension of vector d^{in}. For the value of τ, if $P^{S2}_{\Delta\mathbf{x}}$ is a full-dimensional polytope, then it has at least $\sum_{i\in N}(r_i+z_i+1)$ facets, which means $\tau > \sum_{i\in N}(r_i+z_i)$ must apply. Hence, the number of coupling constraints in (4.52) is much larger than n_s, and this causes the distributed solution based on the Shapley-Folkman-Starr theorem be not applicable here.

Nevertheless, the number of coupling constraints in problem (4.52) can indeed be reduced if a certain polyhedral set is adopted to inner-approximate $\varepsilon^{S2}_{\Delta\mathbf{x}}$. In detail, the new polyhedral set $P^{S2,rec}_{\Delta\mathbf{x}}$ is defined by:

$$P^{S2,rec}_{\Delta\mathbf{x}} := \{\Delta x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\, i \in N | \tag{4.56}$$
$$\min\{0, \mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}(j))\} \leq \Delta x(j) \leq \max\{0, \mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}(j)\},\, j \in \{1,\cdots,\sum_{i\in N} r_i + z_i\}\},$$

which is a mixed-integer hyperrectangular set with points $\mathbf{0}$ and $-\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}$ as a pair of vertices located opposite to each other. In addition, these two vertices also locate on the hull of $\varepsilon^{S2}_{\Delta\mathbf{x}}$.

Proposition 4.1. *The hyperrectangular set $P^{S2,rec}_{\Delta\mathbf{x}}$ is contained in $\varepsilon^{S2}_{\Delta\mathbf{x}}$.*

Proof. According to the definition of $P^{S2,rec}_{\Delta\mathbf{x}}$, a diagonal matrix

$$\mathcal{A} = \mathbf{Diag}(\alpha(1),\cdots,\alpha(\sum_{i\in N} r_i + z_i)),\ \alpha(j) \in [0,1],\ \forall j \in \{1,\cdots,\sum_{i\in N} r_i + z_i\}$$

always exists for any $\Delta\mathbf{x} \in P_{\Delta\mathbf{x}}^{S2,rec}$, so that the equation

$$\Delta\mathbf{x} = \mathcal{A} \cdot (-\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1})$$

applies. By substituting this equation into the left hand side of the inequality in (4.47), one writes:

$$h := (\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1})^{\mathrm{T}} \frac{(\mathbf{Diag}(\frac{1}{2}\cdots\frac{1}{2}) - \mathcal{A}) \cdot 4\mathbf{Q} \cdot (\mathbf{Diag}(\frac{1}{2}\cdots\frac{1}{2}) - \mathcal{A})}{\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}\mathbf{f}(\mathbf{x}^{S2})^{\mathrm{T}}} (\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}). \tag{4.57}$$

Then, as $\alpha_j \in [0,1]$ holds for all $j \in \{1, \cdots, \sum_{i\in N} r_i + z_i\}$ in matrix $\mathcal{A}$, the following relation thus applies:

$$\left|\mathbf{Diag}(\frac{1}{2}\cdots\frac{1}{2}) - \mathcal{A}\right| \leq \mathbf{Diag}(\frac{1}{2}\cdots\frac{1}{2}), \tag{4.58}$$

which enables one to determine an upper value bound of (4.57) by:

$$h \leq \frac{(\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1})^{\mathrm{T}}\mathbf{Q}(\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1})}{\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}\mathbf{f}(\mathbf{x}^{S2})^{\mathrm{T}}} = 1. \tag{4.59}$$

Thus, for any $\Delta\mathbf{x}$ selected from $P_{\Delta\mathbf{x}}^{S2,rec}$, it must also be located in $\varepsilon_{\Delta\mathbf{x}}^{S2}$. □

In Fig. 4.5, an exemplary ellipsoidal set $\varepsilon_{\Delta\mathbf{x}}^{S2}$ and the associated hyperrectangular set $P_{\Delta\mathbf{x}}^{S2,rec}$ are illustrated.

Remark 4.1. *Note that the size of $P_{\Delta\mathbf{x}}^{S2,rec}$ may be smaller than the other polyhedral sets used to inner-approximate $\varepsilon_{\Delta\mathbf{x}}^{S2}$. This shortage, as will be shown later, may only lead to a minor improvement of candidate $\mathbf{x}^{S2}$ in the later program, but will not cause infeasible problems.*

By replacing $P_{\Delta\mathbf{x}}^{S2}$ with $P_{\Delta\mathbf{x}}^{S2,rec}$ in (4.52), the new problem is obtained:

$$\Delta\tilde{\mathbf{x}} := \underset{\Delta\mathbf{x}=[\Delta x_1;\cdots;\Delta x_{n_s}]}{\arg\min} \mathbf{f}(\mathbf{x}^{S2})\Delta\mathbf{x} \tag{4.60}$$

$$\text{s.t.:} \quad \sum_{i\in N} A_i\Delta x_i \leq b - \sum_{i\in N} A_i x_i^{S2}; \tag{4.61}$$

$$D_i\Delta x_i \leq d_i - D_i x_i^{S2},\ \Delta x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\ i \in N; \tag{4.62}$$

$$\Delta\mathbf{x} \in P_{\Delta\mathbf{x}}^{S2,rec}. \tag{4.63}$$

For the new constrained set $P_{\Delta\mathbf{x}}^{S2,rec}$, apart from being contained in $\varepsilon_{\Delta\mathbf{x}}^{S2}$, the normal vector of each facet j of $P_{\Delta\mathbf{x}}^{S2,rec}$ is also orthogonal to any axis besides j of the coordinate system. This fact implies that the constraint (4.63) constitutes local constraints, since each dimension of $P_{\Delta\mathbf{x}}^{S2,rec}$ is bounded separately. As a result, only

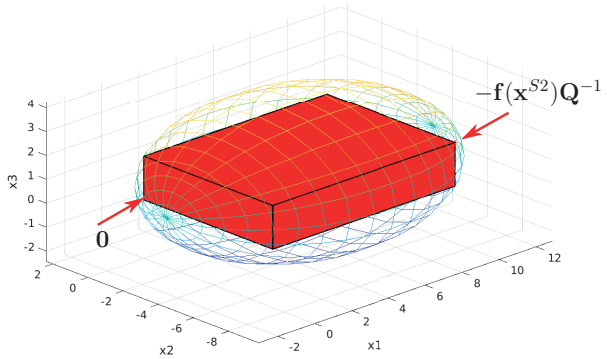

Figure 4.5.: Hyperrectangular set $P_{\Delta\mathbf{x}}^{S2,rec}$ (marked in red) is contained in $\varepsilon_{\Delta\mathbf{x}}^{S2}$, and the normal vector of each facet $j \in \{1, \cdots, \sum_{i \in N} r_i + z_i\}$ points into direction of one axis of the coordinate system.

m coupling constraints arise in (4.60), which enables one to employ the distributed solution based on the Shapley-Folkman-Starr theorem. Note that the feasibility of this problem is guaranteed as $\Delta\mathbf{x} = \mathbf{0}$ is a feasible option, and the global cost reduction for $\mathbf{x}^{S2} + \Delta\tilde{\mathbf{x}}$ is also ensured (if $\mathbf{x}^{S2}$ does not equal the global optimum) according to constraint (4.63). For the distributed solution of the MILP problem (4.60), one can either adopt the method introduced in [162, 60, 43, 161, 44], or the one in Algorithm 4.2.

Finally, a new global candidate $\mathbf{x}^{S3} := \mathbf{x}^{S2} + \Delta\hat{\mathbf{x}}$ is determined through the solution[4] of (4.60). For the new candidate $\mathbf{x}^{S3}$, however, it may fail to satisfy the **Conditions 1** and **2** due to the increment $\Delta\hat{\mathbf{x}}$. This fact enables one to re-use these two conditions to further improve $\mathbf{x}^{S3}$. The overall improvement process starting from $\mathbf{x}^{f}$ is summarized in Algorithm 4.5, and this algorithm terminates when no further improvement is found in any stage.

The Application Order of the Different Stages

Since all three stages in Algorithm 4.5 are capable to improve the candidate of (4.34), their application order becomes crucial for a fast convergence of the algorithm. However, the NP-hard nature of MIP problems makes it hard to discuss, in

[4]By applying the distributed solution based on the Shapley-Folkman-Starr theorem, one may only obtain a sub-optimal solution of (4.60) according to [162, 161], but with feasibility and bounded performance loss guaranty. This is why the notion $\Delta\hat{\mathbf{x}}$ is adopted to denote the outcome of (4.60), instead of $\Delta\tilde{\mathbf{x}}$.

1: **Initialization:** $\mathbf{x}^f$, $q = 0$, $\mathbf{x}^{[q]} := \mathbf{x}^f$;
2: **while** $J(\mathbf{x}^{[q-1]}) - J(\mathbf{x}^{[q]}) > 0$ **do**
3: Stage one:
4: run Algorithm 4.3 or 4.4, and obtain $\mathbf{x}^{S1,[q]}$;
5: Stage two:
6: solve the QP problem (4.40) (distributed or centralized), and obtain $\mathbf{x}^{S2,[q]}$;
7: Stage three:
8: solve the MILP problem (4.60) (distributed), and obtain $\mathbf{x}^{S3,[q]}$;
9: set $\mathbf{x}^{[q+1]} := \mathbf{x}^{S3,[q]}$;
10: $q := q + 1$;
11: **end while**

Algorithm 4.5.: The overall algorithm for MIQP problem (4.34).

general, the convergence rate analytically.

Empirically, the computation in stage one leads to a faster convergence if the coupling strength is low, i.e., if only a few coupling constraints are active. Then, the coupling constraint in problem (4.38) will not cause a significant reduction of the local feasible space X_i in problem (4.38). Accordingly, each local subsystem only has to slightly deviate from its local optimum (by minimizing J_i and subject to the local constraint), in order to meet the coupling constraint. As a result, the subsystems can quickly approach their local optimum, and thus quickly reduce the global costs.

The computation in stage two can realize a faster convergence if the integer variables in (4.40) play a minor role (for both cost function and constraints) compared to the continuous variables. In this case, no matter which value the integer variables in problem (4.40) take, the optimal costs of (4.40) will not differ much from each other. In other words, the original MIQP problem (4.34) is very similar to a QP problem, since the continuous variables are most relevant for both the feasibility and optimality in the former problem. Accordingly, by minimizing the continuous variables first, a faster convergence would be achieved in this case.

To start by solving problem (4.60) in stage three is not recommended. This is because, on the one hand, the original space $\varepsilon_{\Delta\mathbf{x}}^{S2}$ may be considerably reduced by using the inner-approximation $P_{\Delta\mathbf{x}}^{S2,rec}$. On the other hand, the distributed solution based on the Shapley-Folkman-Starr theorem may also only lead to a sub-optimum of the problem instead of the global optimum. All these facts imply that only a minor improvement may be achieved in stage three, and thus leads to slow convergence. Nevertheless, the solution of (4.60) provides the only chance to update all variables simultaneously in Algorithm 4.5. This fact is especially useful if a „bug" candidate satisfying both **Conditions 1** and **2** is found. In this case, although the global costs may not be significantly reduced by solving this problem, a slight improvement may help one to get rid of this candidate, and thus, let the first two

conditions be applied hereafter to realize further improvements.

4.2.2. Evaluation of the Performance

For the outcome of Algorithm 4.5, denoted by $\mathbf{x}^L$, the performance loss compared to $\mathbf{x}^*$ is evaluated in this section. As it is impossible to directly compute the value of $J(\mathbf{x}^*)$ due to the high complexity (which was the motivation for the distributed scheme), an upper bound of $J(\mathbf{x}^L) - J(\mathbf{x}^*)$ is derived here.

First, as the problem (4.60) is always feasible, the termination of Algorithm 4.5 thus implies that $\Delta\mathbf{x} = \mathbf{0}$ must be its only feasible solution. Otherwise, a candidate better than $\mathbf{x}^L$ would be found. The following fact can thus be established:

Corollary 4.1. *For a given $\mathbf{x}^L$, there exists no $\Delta\mathbf{x} \in P_{\Delta\mathbf{x}}^{L,rec}$, $\Delta\mathbf{x} \neq \mathbf{0}$, so that $\mathbf{x}^L + \Delta\mathbf{x}$ is feasible for problem* (4.34).

Then, for a selected value $\beta \in \mathbb{R}^{\leq 0}$, if a $\Delta\mathbf{x}$ enables a cost reduction $J(\mathbf{x}^L + \Delta\mathbf{x}) - J(\mathbf{x}^L) \leq \beta$, it must satisfy:

$$\Delta\mathbf{x}^{\mathrm{T}}\mathbf{Q}\Delta\mathbf{x} + \mathbf{f}(\mathbf{x}^L)\Delta\mathbf{x} \leq \beta, \tag{4.64}$$

where $\mathbf{f}(\mathbf{x}^L) = 2{\mathbf{x}^L}^{\mathrm{T}}\mathbf{Q} + \mathbf{C}^{\mathrm{T}}$, according to (4.46). Similar to (4.47), this condition also determines a mixed-integer ellipsoidal set $\varepsilon_{\Delta\mathbf{x}}^{L,\beta}$ of $\Delta\mathbf{x}$:

$$\begin{aligned}\varepsilon_{\Delta\mathbf{x}}^{L,\beta} = \{&\Delta x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i},\, i \in N| \\ &(\Delta\mathbf{x} + \frac{1}{2}\mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1})^{\mathrm{T}}\frac{4\mathbf{Q}}{4\beta + \mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1}\mathbf{f}(\mathbf{x}^L)^{\mathrm{T}}}(\Delta\mathbf{x} + \frac{1}{2}\mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1}) \leq 1\}.\end{aligned} \tag{4.65}$$

The length of each axis of $\varepsilon_{\Delta\mathbf{x}}^{L,\beta}$ is determined by β, and by varying the value of β, a new ellipsoidal $\varepsilon_{\Delta\mathbf{x}}^{L,\beta}$, being contained in the hyperrectangular set $P_{\Delta\mathbf{x}}^{L,rec}$, can be determined. For all $\varepsilon_{\Delta\mathbf{x}}^{L,\beta}$ contained in $P_{\Delta\mathbf{x}}^{L,rec}$, the one with the largest size, i.e., the largest feasible value of β is given by:

$$\beta^* = \min_{j \in \{1,\cdots,\sum_{i\in N} r_i + z_i\}} \frac{((\mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1})(j))^2\mathbf{Q}(j,j) - \mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1}\mathbf{f}(\mathbf{x}^L)^{\mathrm{T}}}{4}. \tag{4.66}$$

The value of β^* is determined according to the fact that the sets $\varepsilon_{\Delta\mathbf{x}}^{L,\beta}$ and $P_{\Delta\mathbf{x}}^{L,rec}$ share the same center $-\frac{1}{2}\mathbf{f}(\mathbf{x}^{S2})\mathbf{Q}^{-1}$ and rotation (since matrix $\mathbf{Q}$ is diagonal). Thus, the relation

$$\varepsilon_{\Delta\mathbf{x}}^{L,\beta} \subseteq P_{\Delta\mathbf{x}}^{L,rec}$$

applies, as long as each axis of $\varepsilon_{\Delta\mathbf{x}}^{L,\beta}$ is shorter than the edge of $P_{\Delta\mathbf{x}}^{L,rec}$. By formulating this requirement into the following constraint of β using (4.65):

$$2 \cdot (\frac{4\mathbf{Q}(j,j)}{4\beta + \mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1}\mathbf{f}(\mathbf{x}^L)^{\mathrm{T}}})^{-\frac{1}{2}} \leq |(\mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1})(j)|,\ \forall j \in \{1,\cdots,\sum_{i\in N} r_i + z_i\}, \tag{4.67}$$

which is equivalent to:

$$\beta \leq \frac{((\mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1})(j))^2\mathbf{Q}(j,j) - \mathbf{f}(\mathbf{x}^L)\mathbf{Q}^{-1}\mathbf{f}(\mathbf{x}^L)^{\mathrm{T}}}{4}, \forall j \in \{1, \cdots, \sum_{i \in N} r_i + z_i\}. \quad (4.68)$$

The largest β^* supporting the relation $\varepsilon_{\Delta\mathbf{x}}^{L,\beta} \subseteq P_{\Delta\mathbf{x}}^{L,rec}$ can thus be determined by (4.66), also see the example in Fig. 4.6.

Now, based on the relation $\varepsilon_{\Delta\mathbf{x}}^{L,\beta^*} \subseteq P_{\Delta\mathbf{x}}^{L,rec}$, the Corollary (4.1) can be extended to:

Corollary 4.2. *For a given* $\mathbf{x}^L$*, there exists no* $\Delta\mathbf{x} \in \varepsilon_{\Delta\mathbf{x}}^{L,\beta^*}$, $\Delta\mathbf{x} \neq \mathbf{0}$*, so that* $\mathbf{x}^L + \Delta\mathbf{x}$ *is feasible for problem* (4.34)*.*

This corollary implies that, in case $\mathbf{x}^* \neq \mathbf{x}^L$, the difference vector $\Delta\mathbf{x}^* = \mathbf{x}^* - \mathbf{x}^L$ must be located somewhere outside of $\varepsilon_{\Delta\mathbf{x}}^{L,\beta^*}$. Otherwise, Corollary 4.2 would be violated. Thus, as $\varepsilon_{\Delta\mathbf{x}}^{L,\beta^*}$ contains the set of $\Delta\mathbf{x}$ for which $J(\mathbf{x}^L + \Delta\mathbf{x}) - J(\mathbf{x}^L) \leq \beta^*$ applies, the following relation must hold:

$$J(\mathbf{x}^L + (\mathbf{x}^* - \mathbf{x}^L)) - J(\mathbf{x}^L) > \beta^* \implies J(\mathbf{x}^L) - J(\mathbf{x}^*) < -\beta^* \quad (4.69)$$

Hence, the performance loss of $\mathbf{x}^L$ is bounded by $-\beta^*$.

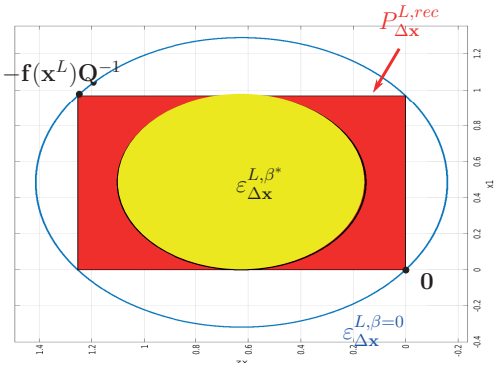

Figure 4.6.: The ellipsoidal set $\varepsilon_{\Delta\mathbf{x}}^{L,\beta=0}$ is marked in blue and contains the set of $\Delta\mathbf{x}$ for which $J(\mathbf{x}^L + \Delta\mathbf{x}) - J(\mathbf{x}^L) \leq 0$ applies; the rectangular set $P_{\Delta\mathbf{x}}^{L,rec}$ is an inner-approximation of $\varepsilon_{\Delta\mathbf{x}}^{L,\beta=0}$ determined by (4.56); the ellipsoidal set $\varepsilon_{\Delta\mathbf{x}}^{L,\beta^*}$ for β^* determined by (4.66) is marked in yellow and contains the set of $\Delta\mathbf{x}$ for which $J(\mathbf{x}^L + \Delta\mathbf{x}) - J(\mathbf{x}^L) \leq \beta^*$ applies.

4.2.3. Numerical Examples

In this section, the proposed distributed solution was tested for various MIQP problems of different sizes. The Q_i and c_i in the local cost functions, the local constraints X_i, and the coupling constraints were randomly generated in each test, while the infeasible instances were discarded.

In the first test, a number of $n_s = 20$ subsystems was considered, each with $z_i = r_i = 3$ integer and real variables. The number of coupling constraints was chosen to $m = 6$. The centralized version of this problem for comparison purposes, involves in total 60 integer variables and 60 real variables. Its solution was found after 35 minutes with the optimal cost $J(\mathbf{x}^*) = 2.42 \cdot 10^3$ on a 3.4GHZ processor using CPLEX. However, the first feasible candidate $\mathbf{x}^{[0]} := \mathbf{x}^f$ was determined after only 0.03 seconds, but with a cost of $J(\mathbf{x}^{[0]}) = 7.57 \cdot 10^3$, i.e., almost three times higher than the optimal one, see Fig. 4.7.

By employing Algorithm 4.5, the computation in stage one terminated after only 1.20 seconds, and the global cost was reduced to $J(\mathbf{x}^{S1,[0]}) = 2.83 \cdot 10^3$. Then, the computation in stage two was finished after 0.07 seconds, through which the global cost was reduced to $J(\mathbf{x}^{S2,[0]}) = 2.76 \cdot 10^3$. Thereafter, the global cost was further reduced to $J(\mathbf{x}^{S3,[0]}) = 2.57 \cdot 10^3$ in stage three within 0.16 seconds. Then, starting from stage one once more with $\mathbf{x}^{S3,[0]}$, the global cost was reduced to $J(\mathbf{x}^{S1,[1]}) = 2.42 \cdot 10^3$ in 0.86 seconds, i.e., the global optimum was found and the whole algorithm terminated after totally 2.37 seconds, see Fig. 4.7.

For a set of additional tests for more complicated MIQP problems (centralized solution times longer than 20 minutes), the results are listed in Table 4.2.

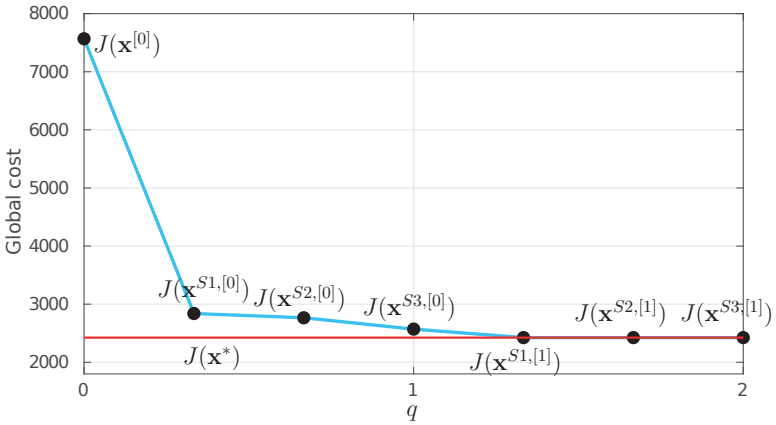

Figure 4.7.: The continuous reduction of global costs by employing Algorithm 4.5.

Table 4.2.: Numerical experiments for different MIQP problems with T indicating the time required for the solution of $\mathbf{x}^f$ and $\mathbf{x}^L$.

N	z_i, r_i	m	$T_{\mathbf{x}^f}$	$J(\mathbf{x}^f)$	$T_{\mathbf{x}^L}$	$J(\mathbf{x}^L)$
50	6	4	0.74 *sec*	$4.48 \cdot 10^5$	9.34 *sec*	$0.20 \cdot 10^5$
50	12	4	2.19 *sec*	$4.45 \cdot 10^4$	4.02 *sec*	$0.44 \cdot 10^5$
80	4	10	0.26 *sec*	$2.51 \cdot 10^4$	5.99 *sec*	$0.47 \cdot 10^4$
80	12	2	3.67 *sec*	$7.03 \cdot 10^4$	17.43 *sec*	$0.67 \cdot 10^4$
120	4	10	0.57 *sec*	$3.89 \cdot 10^4$	6.91 *sec*	$1.02 \cdot 10^4$
120	8	6	1.78 *sec*	$4.86 \cdot 10^4$	7.34 *sec*	$0.88 \cdot 10^4$
200	2	3	0.05 *sec*	$1.17 \cdot 10^5$	22.09 *sec*	$0.40 \cdot 10^5$
200	10	14	4.93 *sec*	$1.48 \cdot 10^5$	16.60 *sec*	$0.15 \cdot 10^5$
500	8	12	6.21 *sec*	$3.75 \cdot 10^5$	16.41 *sec*	$0.42 \cdot 10^5$

4.3. Case Study: Distributed Optimization for a Narrow Passage Problem

In this section, a discrete time, point-to-set trajectory planning problem involving $n_s = 7$ autonomously driving vehicles is considered with quadratic cost function J_i of each vehicle. Note that the number of coupling constraints in this example is larger than the number of vehicles, thus only the Algorithm 4.3 and 4.4 are applied here (these two algorithms are not limited to $n_s \gg m$). Nevertheless, a satisfiable suboptimal solution can already be obtained within very short time by applying these algorithms.

For each vehicle i, the permitted state space is non-convex and consists of the six rectangles (regions) ① – ⑥ shown in Fig. 4.8. Note that this non-convex state space equals to the union of the six rectangles (convex). Thus, according to [96], by introducing one binary variable for each region in each discrete time k, the non-convex state space can be reformulated into mixed-integer constraints for each vehicle. For any vehicle i, the initial state is randomly assigned within the region ① (see Fig. 4.9), and the target state is chosen to be located inside of region ⑥. All vehicles are assumed to share a common target region X_g (located in region ⑥, marked in green).

For simplicity, an identical double-integrator model is chosen here for the linear discrete-time dynamics of each vehicle, but with different input sets and weighting matrices in the local quadratic cost function. The coupling constraints are induced by the limited width of the regions ②, ④, and ⑤ together with the following constraining assumptions:

- in region ②, at most two vehicles are allowed to be present for any time k;
- in region ④ and ⑤, only one vehicle is allowed to be present for any k.

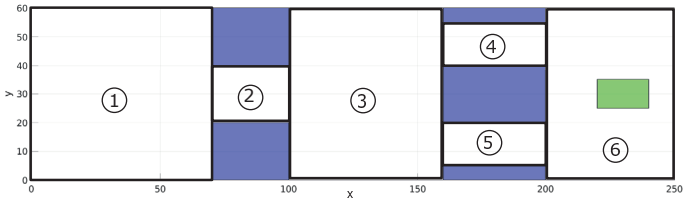

Figure 4.8.: The non-convex state space consisting of 6 polytopes.

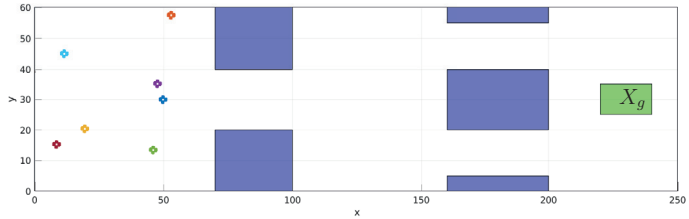

Figure 4.9.: State initialization for all vehicles within region ①.

For a chosen horizon $H = 40$, the control goal is to compute a trajectory for each vehicle which transfers it from the initial state to the target region X_g, while taking the local constraints and the coupling constraints into account, and while minimizing the global cost $J = \sum_{i=1}^{n_s} J_i$. Note that the centralized problem (an MIQP problem due to the binary variables and the quadratic local cost functions) can be eventually formulated in the form of (4.34).

By using binary variables to encode the non-convex state space, a number of $7(n_s) \times 40(H) \times 6(regions) = 1680$ binary variables are needed for the centralized problem (4.34). The centralized solution of this problem, which is still possible for comparison purposes, requires a computation time of 432.18 *sec* by using the solver CPLEX on a 3.4GHZ processor, and the optimal cost is $J(\mathbf{x}^*) = 2.445 \cdot 10^6$. In contrast, the first feasible candidate $\mathbf{x}^f$ of (4.34) is determined after only 1.12 *sec*, but with a cost of $J(\mathbf{x}^f) = 6.553 \cdot 10^6$ (i.e., almost three times higher than $J(\mathbf{x}^*)$).

Now, starting from the first feasible candidate $\mathbf{x}^f$, the distributed solution in Algorithm 4.4 (in which a central coordinator is required) is employed. The termination of this algorithm occurs after just 5 iterations, leading to cost upon termination of $J(\mathbf{x}^L) = 2.574 \cdot 10^6$, i.e., a performance loss of only 5.27% compared to $J(\mathbf{x}^*)$ is attained. The computation time for each iteration is shown in Fig. 4.10 and varies between 0.1 *sec* and 0.3 *sec*. The total computation time for the 5 iterations is 0.86 *sec*, and the course of the cost over the iterations is illustrated in Fig. 4.11.

When applying Algorithm 4.3 (no central coordinator), an even better outcome is obtained with $J(\mathbf{x}^L) = 2.569 \cdot 10^6$ after 7 iterations within a total computation time of 0.74 *sec*.

Obviously, the computation time has been significantly reduced by using the distributed solution and the performance loss is small. Most importantly, the feasibility of the global candidate is maintained in each intermediate iteration, as well as a continuous reduction of the cost is ensured. To give an impression of the trajectories of the vehicles, their positions are shown for selected points of time in Fig. 4.15 – Fig. 4.22 as obtained by Algorithm 4.4. It can be seen from these plots that the coupling constraints are satisfied and that all vehicles have reached the target region at the end of the horizon.

In order to further evaluate the performance of the two algorithms for more general cases, they have been applied to different instances of the problem with $n_s = 2$ to $n_s = 6$ vehicles and each for 20 parameterizations (different initial position, input and state constraints, weighting matrices) of the vehicles. Fig. 4.12 illustrates the average performance loss obtained with the distributed strategy, compared to

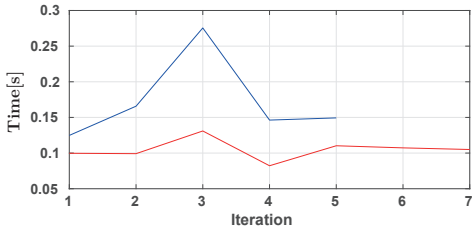

Figure 4.10.: Computation time over iterations for Algorithm 4.3 in red, and for 4.4 in blue.

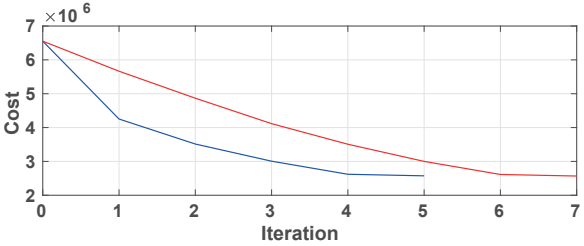

Figure 4.11.: The costs over the iterations ρ for Algorithm 4.3 in red, and for 4.4 in blue.

the globally optimal solution. Fig. 4.13 and 4.14 show the average computation time and the average number of required iterations when applying the two algorithms.

Finally, for a larger problem instance with 14 vehicles, leading to overall $14 \times 40 \times 6 = 3660$ binary variables in the centralized problem, the global optimum could not be found by centralized solution within 2 hours using CPLEX. However, it takes only 11.58 *sec* to find the first feasible candidate. Then, by applying Algorithm 4.4 once more, a cost reduction from $J(\mathbf{x}^f) = 1.03 \cdot 10^7$ to $J(\mathbf{x}^L) = 8.13 \cdot 10^6$ could be

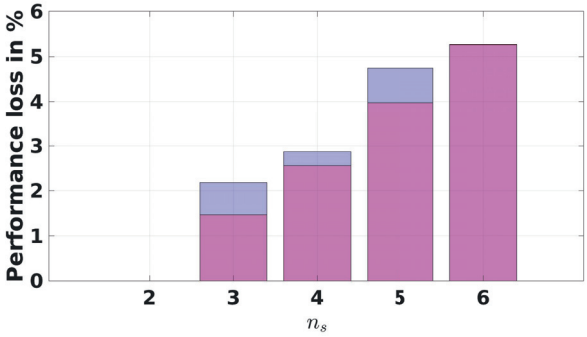

Figure 4.12.: The average relative performance loss for different numbers of vehicles; dark red for Algorithm 4.3 and light blue for 4.4.

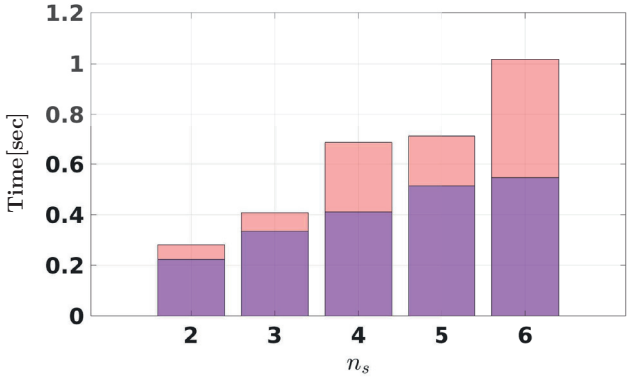

Figure 4.13.: The average computation time for different values of n_s; pink for Algorithm 4.3 and purple for 4.4.

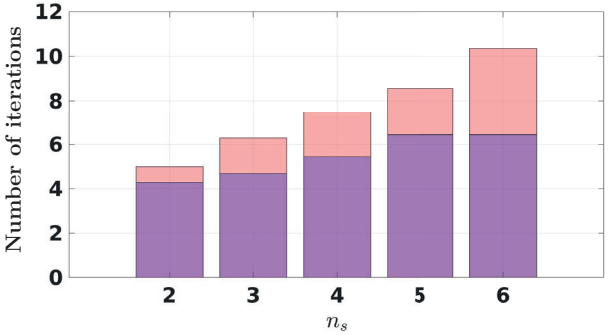

Figure 4.14.: Average number of required iterations until termination for different values of n_s; pink for Algorithm 4.3 and purple for 4.4.

achieved in only 4 iterations with a computation time of 0.40 *sec*.

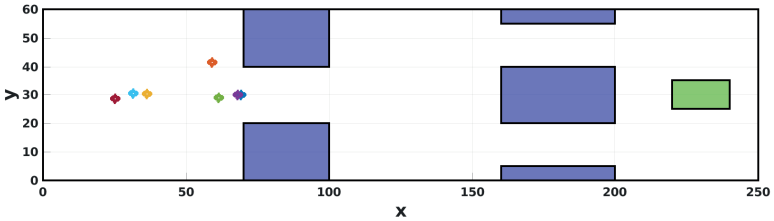

Figure 4.15.: Positions of the vehicles for $k = 6$.

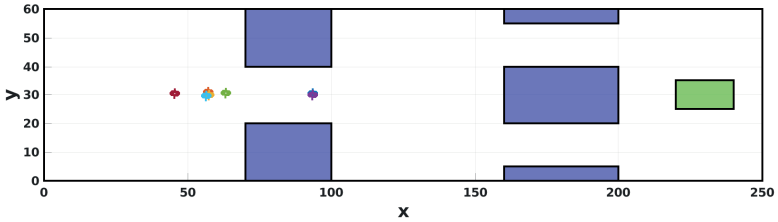

Figure 4.16.: Positions of the vehicles for $k = 9$.

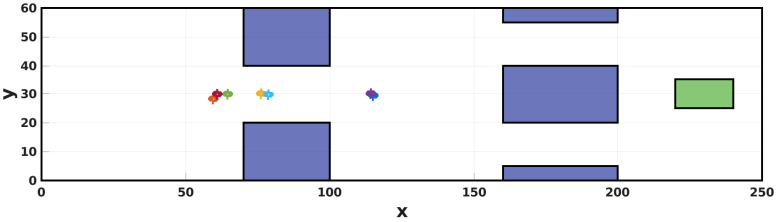

Figure 4.17.: Positions of the vehicles for $k = 11$.

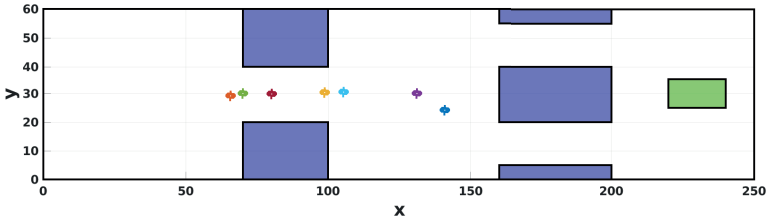

Figure 4.18.: Positions of the vehicles for $k = 13$.

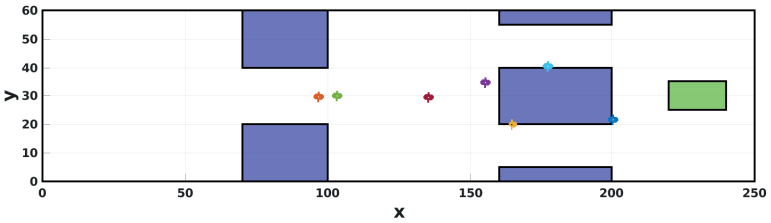

Figure 4.19.: Positions of the vehicles for $k = 19$.

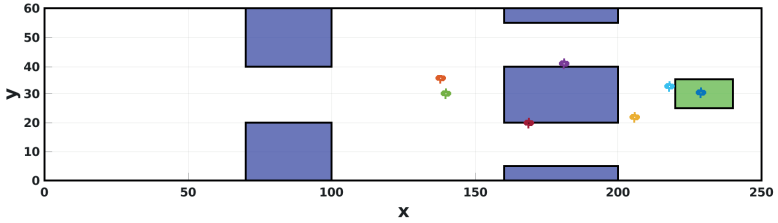

Figure 4.20.: Positions of the vehicles for $k = 22$.

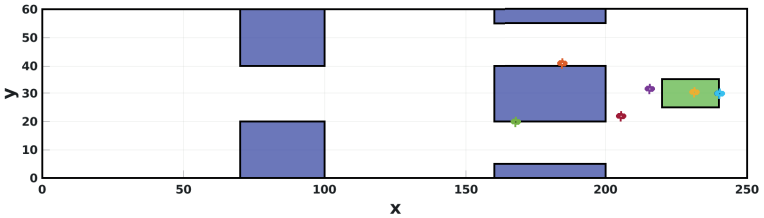

Figure 4.21.: Positions of the vehicles for $k = 26$.

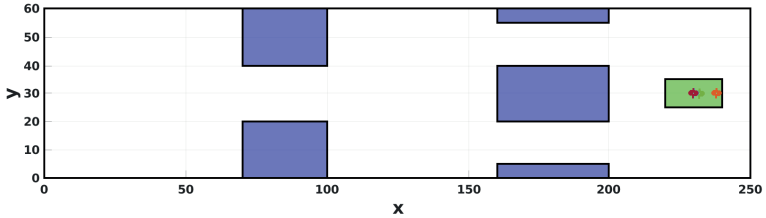

Figure 4.22.: Positions of the vehicles for $k = 40$.

4.4. Summary and Discussion

This chapter addressed the task of distributed control for interleaving hybrid systems with coupled constraints. This type of coupled constraints arises in CPS when the interaction among the subsystems cannot be cast into local constraints as in the previous chapters. The discussion started by considering an MILP problem (4.7) representing the centralized optimal control problem. A distributed solution was proposed for (4.7) based on the Shapley-Folkman-Starr theorem, so that the centralized problem can be decomposed into a set of small-scale problems to be solved in parallel. This approach reduces the overall complexity considerably and also relaxes some conservative assumptions required in state-of-the-art work. Close-to-optimal solutions could be obtained very efficiently for a large number of tests.

The distributed solution of MIQP problem (4.34) was then considered in this chapter, which represents a more typical problem in the context of optimal control than the one of MILP. The proposed distributed solution consists of totally three stages of computation, each of which stems from one of the optimality conditions of (4.34) to guide the decomposition and the solution process. Especially for the last stage, the new findings of MILP problem (4.7) are successfully applied to the distributed solution of the MIQP problem. The optimal solution (or a close-to-optimal solution) has in all cases been obtained in a series of numerical tests, which confirmed the efficiency of the method. Finally, part of the proposed methods were tested for a multi-vehicle application with passage constraints, and satisfiable results have been obtained in all the tests with respect to both the quality of the solution and the computation time.

The part that is not taken into account in this chapter, however, is the communication problem among the subsystems: the subsystems are assumed to be able to broadcast/receive information from any other subsystems, i.e., a fully connected communication graph, there is no package loss or delay, and the transmission channel is without noise. In addition, the limitation on the maximal communication frequency is also disregarded, which may indeed affect the performance of the proposed distributed solutions. Besides the communication problems, the local constraints are assumed to take the form $x_i \in X_i$, $x_i \in \mathbb{R}^{r_i} \times \mathbb{Z}^{z_i}$ in this chapter. This setting is based on the observation that all local constraints arise in the previous chapters can eventually be reformulated into this form. But by starting with this form for the proposed distributed solution, the special structure of X_i in terms of the hybrid dynamics in HA may be neglected, such as the sparsity of the matrix on the left hand side of X_i. Nevertheless, these shortages or limitations also point into meaningful research directions for the future, as summarized in the next chapter.

5. Concluding Remarks

5.1. Contributions

In this thesis, a series of modeling and control problems in CPS have been considered: for the modeling part, a general model class HA has been defined to describe the continuous and discrete dynamics in each subsystem, so that a large variety of transition mechanisms can be covered. The use of HA enhances the flexibility by modeling local dynamics in CPS compared to standard modeling techniques, such as PWA systems, switching systems or switched systems. For the control part, a set of methods has been proposed according to different interacting scheme in CPS. In general, by assuming the dynamics of each subsystem is modeled by HA, all these control methods aim at ensuring the safety of CPS and optimality of the controlled behavior, as well as a high computational performance.

In Chapter 2, details of the considered discrete-time hybrid system HA have been introduced, together with a comparison with other hybrid modeling techniques. Then, in case the influence from other subsystems in CPS can be cast into a change of the invariants and guards of local HA, two finite-horizon control strategies have been proposed for the local subsystem. This chapter has shown that both methods are able to exactly cast the hybrid dynamics and the control task into an MIP problem. To avoid the high complexity caused by the MIP problem, both methods have also constructed a set of constraints involving only the integer variables, through which the search space of underlying MIP problem can be reduced significantly. By applying the considered efficient solution of the MIP problems, each local subsystem in the CPS can determine its best control strategy in a short time, and thus be online applicable.

In Chapter 3, the influence from other subsystems in CPS has been further cast into a change of local invariants and guards, but with uncertainties varying over time. In addition, the local hybrid dynamics there has also been affected by additive disturbances and parametric uncertainties due to modeling errors. This setting, apparently, is more realistic for real-world engineering problems than in the preceding chapter. This chapter started with the case that uncertainties only arise from the local dynamics, and a robust control strategy has been proposed by constructing a robust invariant tube around a nominal trajectory. This robust control strategy enables the local subsystems in CPS to safely plan its local behavior, despite the

modeling error of the local dynamics. Then, a more complicated case has been considered, in which the environment (i.e., the invariants and guards modeling the HA) of the local subsystems is time-varying, with changes which are not precisely known. Problems of this kind have been rarely considered before, although being important for online strategies such as MPC. Hence, for two types of change mechanism of the environment, methods to guarantee the recursive feasibility and stability of MPC have been presented. In addition, a method to guarantee the same properties of MPC but without reducing the original feasible space has also been proposed, and this is realized by introducing penalty terms of the cost function. With the help of all these methods, each local subsystem in CPS can safely plan its local behavior in each online step, despite uncertainties caused by the other subsystems.

In contrast to the preceding chapters, the interaction scheme in Chapter 4 has been modeled by coupling constraints to be jointly satisfied by all subsystems. Each subsystem i there has to select its control strategy from a local input space, usually a mixed-integer space due to the hybrid dynamics. The global control goal of CPS is to minimize the global costs containing each local costs, while the coupling constraints must also be satisfied. Obviously, the centralized solution is the most straightforward way to ensure the satisfaction of all constraints and thus the safety of CPS. However, as MIP problems arise in the centralized problem, the NP-hard nature of MIP problems makes the centralized solution to become intractable if larger numbers of subsystems are contained in the CPS. To reduce the centralized complexity, a set of novel distributed solution schemes have been proposed in this chapter, which decompose the centralized problem into a set of small-scale problems solved by each subsystem in parallel. These distributed solution strategies guarantee that, for an increase of subsystems in CPS, the control complexity only grows moderately instead of exponentially. The efficiency of these strategies has also been confirmed by a series of numerical experiments made in this chapter.

Finally, besides the numerical experiments conducted in each section, a set of application-oriented (CPS relevant) studies has also been presented at the end of each chapter: In Chapter 2, an autonomous vehicle overtaking problem has been considered. It has been shown that, through the construction of a hybrid system HA, a safe overtaking path can be determined for the local vehicle. In Chapter 3, a human-robot collaboration problem has been considered, where uncertainties arise in both robot modeling and human motion prediction. By using the techniques developed in this chapter, the robot can accomplish its task while the safety of the human is also ensured. In Chapter 4, a vehicle coordinating problem has been considered, in which a set of autonomous vehicles need to travel through a narrow passage. The centralized solution of this problem is not practically feasible due to the high computation time, while the proposed distributed solution has shown the possibility to provide a sub-optimal solution in reasonable time with negligible performance loss.

5.2. Future Directions

Point-to-Set-Control of Hybrid Systems with Non-Deterministic Transitions

In addition to the disturbances and parametric uncertainties affecting the hybrid dynamics considered as in Sec. 3.1, another important class of uncertainty is when the discrete state transition is not deterministic [1, 45]. This occurs if the discrete input of HA also leads to uncertainties, e.g., the automatic gear transmission does not always transfer to the desired gear, but only has a limited success probability. The control goal in this case is to take into account all these uncertainties by planning the continuous and discrete input sequences, such that the probability of reaching a target region is maximized, while the expected running cost of reaching the target region is minimized. Thus, it is worth to investigate in how far the control strategies from Sec. 2.3 and 3.1 should be adapted, in order to handle non-deterministic probabilistic transition structures.

Numerical Program Based Falsification of Hybrid Systems

The optimal control technique developed in Chapter 2 can also be employed to the falsification of hybrid systems. Traditionally, reachability analysis of hybrid systems has been applied as an important verification tool to specify the properties of the system [8, 6]. Alternatively, a verification concept called „falsification" has been developed in the last few years [133, 123, 11], and can be regarded as the dual problem to reachability analysis. In contrast to reachability analysis, where the reachable part of the state space must be explored in order to prove the system to be safe, the goal of falsification is to find one state sequence that violates safety properties in order to show that the system is unsafe. This approach may significantly reduce the complexity of exploring the continuous state space in reachable set computation.

Current approaches to falsification mainly use heuristics, but this may lead to fail proving that the system is safe, in case of no violation of the safety properties is found while it indeed exists. Hence, as the semantics of the hybrid system HA can be exactly cast into a set of mixed-integer constraints according to Chapter 2, it is thus worth investigating whether these constraints can be embedded into the falsification procedure, so that the falsification task can be realized through a numerical program and thus provides guaranty. In addition, as the ultimate goal of either reachability analysis or falsification is to ensure the safety of the controlled systems, an efficient „safety-recovering" procedure is thus desired, if the system is proven to be unsafe. The question is, thus, in how far can one utilize the outcome of the falsification process to guide the recovery process.

MPC with Hardware Limitations and Low Frequency MPC

Most of the approach in this thesis aims at reducing the computation complexity to make the proposed control strategy online applicable. However, a concrete time

limit for the computation has not been considered in the thesis, as the computation times are usually determined by the selected hardware. According to the standard receding-horizon scheme of MPC, only the first step input signal of the optimized solution is applied, and the optimization procedure is repeated after moving to the next state. This requires that the total time to measure the environment and to solve the numerical program is smaller than one sampling time. But this requirement cannot always be satisfied due to the limitations of the hardware, e.g., the precise measurement of the environment may be time-consuming, or the computational power to solve the numerical program is timely insufficient. This may lead to the consequence that no feasible solution can be found within one sampling time, and the operation must be stopped. Thus, it is important to study how to further apply MPC and continue operation, despite those hardware limitations.

An alternative approach to enhance the real time ability of MPC is to solve the numerical program every several steps instead of in each step. This idea is similar to the one in [40, 92]. This scheme immediately makes the time interval left for the numerical program several times larger than before, and thus reduces the risk of insufficient computation time. The drawback of this strategy, however, is also obvious, since it may cause a loss of performance due to the low update rate. In addition, if the environment is varying over time (as considered in Sec. 3.2), a low optimization rate may cause a delayed response to the new environment and lead to infeasibility. Accordingly, it is worth to investigate under which circumstances lower frequencies of the optimization in MPC can be applied, while both safety and acceptable performance loss are guaranteed.

Distributed Optimization of MIP Problems with Imperfect Communication

As mentioned in the introduction, communication imperfections in CPS are not considered in this thesis. Instead, the focus is on the decomposition techniques of the centralized problem, so that the distributed solution can converge to an optimum (or sub-optimum) with reduced complexity. However, when imperfect communication is encountered in CPS, the decomposition technique should be tailored to the given communication settings, such as a time-varying communication graph, or a partially connected graph, or a directed graph. Different distributed solutions have been developed in the past to accommodate the limitations of the graph [117, 61, 119], but most of them are limited to a consensus problem. In this type of problems, the local state vector of all subsystems must converged to a common value (the global optimum in most cases), despite the limitations of the imperfect communication. The problem considered in Chapter 4, however, belongs to another type, i.e., the constraint-coupled one, in which the local optimization variable x_i of each subsystem does not necessarily converge to the same value. Nevertheless, for some steps in Algorithm 4.2 and 4.5, an agreement on the common dual variable λ is required. This implies that the subsystems may fail to agree on the same λ, if the communication is imperfect. Thus, how to preserve the desired properties of

Algorithm 4.2 and 4.5, such as feasibility and cost reduction over iterations, despite the limitations of the communication graph, turns out to be a meaningful and open direction for the future research.

Distributed Optimization of MIP Problems with Non-Cooperative Behavior

When the goals of the subsystems in the CPS are not identical, the resulting setting can be interpreted as a non-cooperative game of distributed systems. In this case, game theory is often used to model and analyze the competitive behavior of the subsystems. Note that a non-cooperative game is far more general than the cooperative one considered in Chapter 4, and reflects some real-world problems more realistically. It is because, on the one hand, the definition of cooperative behavior in the sense that all subsystems must follow the same goal is quite restricted, while non-cooperative behavior is more general with respect to the local goal (allowing also for selfish behavior). On the other hand, even if all subsystems would intend to behave in a cooperative way, the limitations of the communication graph or the errors in transmission channels may prevent them to follow this intention, leading to a competitive behavior.

In a non-cooperative game, the local costs are no more purely decided by the local variables, but also by other subsystems [138]. The corresponding optimality conditions are thus also different compared to the cooperative ones. For the optimal solution of a non-cooperative game, which is usually referred to as the „Nash equilibrium" [114], no subsystem can unilaterally optimize its local variables without changing the variables of other subsystems. For the distributed computation of the Nash equilibrium, several research efforts have been published, such as in [156, 146, 142, 169]. But they are limited to the case in which the variables are real-valued, and the distributed computation has been carried out by solving a real-valued variational inequalities problem [58]. Thus, in case the variables are of mixed-integer type instead of being real-valued only, it is worth to investigate whether the Shapley-Folkman-Starr theorem can be leveraged to solve the mixed-integer variational inequalities problems, as was done for the cooperative problems.

List of Symbols

Abbreviations

ADMM	alternating direction method of multipliers
BVP	boundary value problem
CPS	cyber-physical systems
DP	dynamic programming
EE	end-effector of a robot manipulator
HJB	Hamilton-Jacobi-Bellman
ILP	integer linear program
LP	linear program
LQR	linear quadratic regulator
MILP	mixed integer linear program
MINLP	mixed integer nonlinear program
MIP	mixed integer program
MIQP	mixed integer quadratic program
MLD	mixed logical dynamics
MPC	model predictive control
PWA	piecewise affine
QP	quadratic programming
RM	robot manipulator

Functions

$\bar{f} : X \times U \times Z \to X$ nominal discrete-time continuous dynamics in $\overline{HA}$

$\bar{r} : \mathcal{T} \times X \to X$	nominal reset function in $\overline{HA}$
$\mathcal{J} : X \times U \to \mathbb{R}$	cost function
$F : X \to \mathbb{R}$	terminal cost
$f : X \times U \times Z \to X$	discrete-time continuous dynamics in HA
$f^u : X \times U \times Z \to X$	uncertain discrete-time continuous dynamics in HA^u
$J : X \to \mathbb{R}$	global cost function of all subsystems (Chapter 4)
$J_i : X_i \to \mathbb{R}$	local cost function of the subsystem i (Chapter 4)
$L : X \times U \to \mathbb{R}$	function to assign step cost
$r : \mathcal{T} \times X \to X$	reset function in HA
$r^u : \mathcal{T} \times X \to X$	uncertain reset function in HA^u

General

$\mathcal{C}$	central coordinator
$\mathcal{M}$	model to predict the change of the state space
$\mathcal{M}_{arm}$	model to predict the motion of a human arm
$\overline{HA}$	nominal hybrid system obtained from tightening HA^u
HA	a general type of hybrid system
HA^u	hybrid system with uncertainties of the continuous dynamics and reset functions
HA_k	time-varying hybrid system measured in step k

Operators

$\mathcal{R}(\ldots)$	relaxation of the integrality constraints of a given set (Chapter 4)
$\ominus$	Pontryagin difference
$\oplus$	Minkowski addition
$Conv(\ldots)$	convex hull of a given set

Scalars and Constants

$\mathscr{L}_{(x,G_{(i,j)}(h))}$	distance between state x and the h-th facet of the guard set $G_{(i,j)}$
$\mathscr{L}_{(x,I_{(i)}(h))}$	distance between state x and the h-th facet of the invariant set $I_{(i)}$
ρ	iteration counter (Chapter 4)
$\tau_{(i,j)}$	transition from $z_{(i)}$ to $z_{(j)}$
$b_{(i)}$	binary variable assigned to the invariant set $I_{(i)}$
$b_{(i,j)}$	binary variable assigned to the guard set $G_{(i,j)}$
b_g	binary variable assigned to the terminal set X_g
k	discrete time index
L	length of given phase sequence
L_{max}	maximal length of a phase sequence
m	number of coupling constraints in the centralized problem (Chapter 4)
N_g	the first point of time in which the goal set is reached
n_s	number of subsystems
p_l	l-th phase in a discrete state sequence
q_g	weighting factor for N_g
r_i	number of real variables in x_i (Chapter 4)
$s^{[\rho]}$	step length (Chapter 4)
$v_{(i,j)}$	discrete input for the transition $\tau_{(i,j)}$
z_0	discrete initial state
z_g	discrete goal state
z_i	number of integer variables in x_i (Chapter 4)
$z_{(i)}$	discrete state i

Sets

$\bar{G}$	finite set of nominal guards in $\overline{HA}$
$\bar{G}_{(i,j)} \subseteq \mathbb{R}^{n_x}$	nominal guard set for the nominal transition $\bar{\tau}_{(i,j)}$
$\bar{I}$	finite set of nominal invariants in $\overline{HA}$
$\bar{I}_{(i)} \subseteq \mathbb{R}^{n_x}$	nominal invariant of the discrete state $z_{(i)}$
$\bar{U} \subseteq \mathbb{R}^{n_u}$	nominal continuous input space in $\overline{HA}$
$\hat{P}^{rec} \subseteq \mathbb{R}^2$	predicted pose of the human arm by using model $\mathcal{M}_{arm}$
$\mathcal{D} \subseteq \mathbb{R}^{n_x}$	disturbance invariant set
$\mathcal{T}$	finite set of transitions in HA
$\mathcal{W} \subseteq \mathbb{R}^{n_x}$	set of possible additive disturbances
$\tilde{P}^{rec} \subseteq \mathbb{R}^2$	rectangular set obtained by enlarging $\hat{P}^{rec}$ with maximal prediction error
$\varepsilon^{L,\beta^*}_{\Delta\mathbf{x}}$	the largest mixed-integer ellipsoidal set contained in $P^{L,rec}_{\Delta\mathbf{x}}$
$\varepsilon^{L,\beta}_{\Delta\mathbf{x}}$	mixed-integer ellipsoidal set of $\Delta\mathbf{x}$, selected from which relation $J(\mathbf{x}^L + \Delta\mathbf{x}) - J(\mathbf{x}^L) \leq \beta$ applies
$\varepsilon^{L}_{\Delta\mathbf{x}}$	mixed-integer ellipsoidal set of $\Delta\mathbf{x}$, selected from which relation $J(\mathbf{x}^L + \Delta\mathbf{x}) - J(\mathbf{x}^L) \leq 0$ applies
G	finite set of guards in HA
G_k	finite set of guards in HA_k
$G_{(i,j)} \subseteq \mathbb{R}^{n_x}$	guard set for the transition $\tau_{(i,j)}$
I	finite set of invariants in HA
I_k	finite set of invariants in HA_k
$I_{(i)} \subseteq \mathbb{R}^{n_x}$	invariant of the discrete state $z_{(i)}$
N	set of subsystems contained in given CPS (Chapter 4)
N_1	set of subsystems whose local solution is attained at the vertex of $Conv(X_i)$, for all $i \in N_1$ (Chapter 4)
N_2	complementary to the set N_1 of subsystems in N (Chapter 4)

$P^{L,rec}_{\Delta\mathbf{x}}$	mixed-integer hyperrectangular set inner-approximating $\varepsilon^{L}_{\Delta\mathbf{x}}$
$P^{L}_{\Delta\mathbf{x}}$	mixed-integer polyhedral set inner-approximating $\varepsilon^{L}_{\Delta\mathbf{x}}$
$P^{rec} \subseteq \mathbb{R}^2$	rectangular set used to over-approximate the space occupied by a human arm
$T_{\tilde{N}}$	extended discrete-time horizon by taking intermediate states into account
T_N	discrete-time horizon with N discrete steps
$U \subseteq \mathbb{R}^{n_u}$	continuous input space in HA
V	finite set of discrete inputs in HA
$X \subseteq \mathbb{R}^{n_x}$	continuous state space in HA
$X_g \subseteq \mathbb{R}^{n_x}$	terminal/goal set in the continuous state space
X_i	mixed-integer polyhedral set representing local constraints of subsystem i (Chapter 4)
$X_k \subseteq \mathbb{R}^{n_x}$	feasible continuous state space in step k
Z	finite set of discrete states in HA
$Z^{out}_{(i)}$	set of discrete successor states of the discrete state $z_{(i)}$

Vectors and Matrices

$\bar{u}$	nominal continuous input
$\bar{x}$	nominal continuous state
$\Delta\mathbf{x}$	increment vector (Chapter 4)
$\hat{p}^{arm}$	predicted position of the human arm
$\hat{w}_{max}$	maximal prediction error of the state space generated in a single step
κ_g	terminal controller applied in the terminal set X_g
λ	Lagrangian multiplier used to dualize the coupling constraints (Chapter 4)
$\mathbf{x}$	global optimization variable for all subsystems (Chapter 4)

$\mathbf{x}^f$	first feasible candidate for given MIP problem (Chapter 4)
$\mathbf{x}^{S1}$	global candidate obtained at the end of stage one (Chapter 4)
$\mathbf{x}^{S2}$	global candidate obtained at the end of stage two (Chapter 4)
$\mathbf{x}^{S3}$	global candidate obtained at the end of stage three (Chapter 4)
$\mathcal{B}_m$	matrix of binary variables
ϕ_u^{cd}	candidate continuous input sequence
ϕ_v^{cd}	candidate discrete input sequence
ϕ_x^{cd}	candidate continuous state sequence
ϕ_p	phase sequence
ϕ_u	continuous input sequence
ϕ_v	discrete input sequence
ϕ_x	continuous state sequence
ϕ_z	discrete state sequence
K	closed-loop controller matrix
M	a vector/factor of large constants
p_k^E	position of the end-effector measured in step k
p_k^{arm}	position of human arm measured in step k
Q	weighting matrix of the continuous state
Q_g	weighting matrix of the continuous state in the last step of the horizon
R	weighting matrix of the continuous input
u	continuous input
w	additive disturbance
w_{max}	maximal change of the state space within one step
x	continuous state

x'	intermediate continuous state triggering a discrete state transition
x_0	initial continuous state
x_g	target continuous state
x_i	local optimization variables of subsystem i (Chapter 4)
$x_{i,r}$	real variables in the local variables x_i of subsystem i (Chapter 4)
$x_{i,t}$	integer variables in the local variables x_i of subsystem i (Chapter 4)

References

[1] A. Abate, M. Prandini, J. Lygeros, and S. Sastry, "Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems," *Automatica*, vol. 44, no. 11, pp. 2724–2734, 2008.

[2] F. Adamek, M. Sobotka, and O. Stursberg, "Stochastic optimal control for hybrid systems with uncertain discrete dynamics," in *International Conference on Automation Science and Engineering*. IEEE, 2008, pp. 23–28.

[3] T. Alamo, M. Fiacchini, A. Cepeda, D. Limon, J. Bravo, and E. Camacho, "On the computation of robust control invariant sets for piecewise affine systems," in *Assessment and Future Directions of Nonlinear Model Predictive Control*. Springer, 2007, pp. 131–139.

[4] Y. Alber, A. Iusem, and M. Solodov, "On the projected subgradient method for nonsmooth convex optimization in a Hilbert space," *Mathematical Programming*, vol. 81, no. 1, pp. 23–35, 1998.

[5] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear Model Predictive Control*. Springer, 2009, pp. 345–369.

[6] M. Althoff, "Reachability analysis and its application to the safety assessment of autonomous cars," Ph.D. dissertation, Technische Universität München, 2010.

[7] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of linear systems with uncertain parameters and inputs," in *46th Conference on Decision and Control*. IEEE, 2007, pp. 726–732.

[8] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *47th Conference on Decision and Control*. IEEE, 2008, pp. 4042–4048.

[9] R. Alur, C. Courcoubetis, T. Henzinger, and P. Ho, "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems," in *Hybrid systems*. Springer, 1992, pp. 209–229.

[10] B. Anderson and J. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

[11] Y. Annpureddy, C. Liu, G. Fainekos, and S. Sankaranarayanan, "S-taliro: A tool for temporal logic falsification for hybrid systems," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2011, pp. 254–257.

[12] M. Anstreicher and A. Wolsey, "Two "well-known" properties of subgradient optimization," *Mathematical Programming*, vol. 120, no. 1, pp. 213–220, 2009.

[13] E. Asarin, T. Dang, and A. Girard, "Hybridization methods for the analysis of nonlinear systems," *Acta Informatica*, vol. 43, no. 7, pp. 451–476, 2007.

[14] L. Asselborn and O. Stursberg, "Robust control of uncertain switched linear systems based on stochastic reachability," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 310–316, 2015.

[15] L. Asselborn and O. Stursberg, "Control of discrete-time piecewise affine probabilistic systems using reachability analysis," in *Conference on Computer Aided Control System Design*. IEEE, 2016, pp. 661–666.

[16] J. Aubin and I. Ekeland, "Estimates of the duality gap in nonconvex optimization," *Mathematics of Operations Research*, vol. 1, no. 3, pp. 225–245, 1976.

[17] M. Barić, P. Grieder, M. Baotić, and M. Morari, "Optimal control of PWA systems by exploiting problem structure," *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 307–312, 2005.

[18] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations Research*, vol. 46, no. 3, pp. 316–329, 1998.

[19] A. Bemporad, "Efficient conversion of mixed logical dynamical systems into an equivalent piecewise affine form," *IEEE Trans. on Automatic Control*, vol. 49, no. 5, pp. 832–838, 2004.

[20] A. Bemporad, F. Borrelli, and M. Morari, "Piecewise linear optimal controllers for hybrid systems," in *Proceedings of the American Control Conference*, vol. 2. IEEE, 2000, pp. 1190–1194.

[21] A. Bemporad, G. Ferrari, and M. Morari, "Observability and controllability of piecewise affine and hybrid systems," *IEEE Trans. on Automatic Control*, vol. 45, no. 10, pp. 1864–1876, 2000.

[22] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[23] S. Bengea and R. DeCarlo, "Optimal control of switching systems," *Automatica*, vol. 41, no. 1, pp. 11–27, 2005.

[24] D. Bertsekas, "Nonlinear programming," *Journal of the Operational Research Society*, vol. 48, no. 3, pp. 334–334, 1997.

[25] D. Bertsekas, *Convex optimization theory*. Athena Scientific Belmont, 2009.

[26] D. Bertsekas, W. Hager, and O. Mangasarian, *Nonlinear programming*. Athena Scientific Belmont, MA, 1998.

[27] D. Bertsekas and J. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice Hall, 1989, vol. 23.

[28] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[29] A. Boccia, L. Grüne, and K. Worthmann, "Stability and feasibility of state constrained MPC without stabilizing terminal constraints," *Systems and control letters*, vol. 72, pp. 14–21, 2014.

[30] S. Bogomolov, M. Forets, G. Frehse, F. Viry, A. Podelski, and C. Schilling, "Reach set approximation through decomposition with low-dimensional sets and high-dimensional matrices," in *21st International Conference on Hybrid Systems: Computation and Control*, 2018, pp. 41–50.

[31] F. Borrelli, M. Baotić, A. Bemporad, and M. Morari, "Dynamic programming for constrained optimal control of discrete-time linear hybrid systems," *Automatica*, vol. 41, no. 10, pp. 1709–1721, 2005.

[32] F. Borrelli, A. Bemporad, and M. Morari, "Geometric algorithm for multiparametric linear programming," *Journal of optimization theory and applications*, vol. 118, no. 3, pp. 515–540, 2003.

[33] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[34] R. Bourdais, H. Guéguen, and A. Belmiloudi, "Distributed model predictive control for a class of hybrid system based on Lagrangian relaxation," *IFAC Proceedings Volumes*, vol. 45, no. 9, pp. 46–51, 2012.

[35] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[36] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[37] M. Branicky, V. Borkar, and S. Mitter, "A unified framework for hybrid control: Model and optimal control theory," *IEEE Trans. on Automatic Control*, vol. 43, no. 1, pp. 31–45, 1998.

[38] M. Branicky and S. Mitter, "Algorithms for optimal hybrid control," in *34th Conference on Decision and Control*, vol. 3. IEEE, 1995, pp. 2661–2666.

[39] F. Bünger, "A note on the boundary shape of matrix polytope products." *Reliable Computing*, vol. 20, no. 1, pp. 73–88, 2014.

[40] R. Cagienard, P. Grieder, E. Kerrigan, and M. Morari, "Move blocking strategies in receding horizon control," *Journal of Process Control*, vol. 17, no. 6, pp. 563–570, 2007.

[41] P. Caines, F. Clarke, X. Liu, and R. Vinter, "A maximum principle for hybrid optimal control problems with pathwise state constraints," in *Proceedings of the 45th Conference on Decision and Control*. IEEE, 2006, pp. 4821–4825.

[42] A. Camisa, F. Farina, I. Notarnicola, and G. Notarstefano, "Distributed constraint-coupled optimization over random time-varying graphs via primal decomposition and block subgradient approaches," in *58th Conference on Decision and Control*. IEEE, 2019, pp. 6374–6379.

[43] A. Camisa, I. Notarnicola, and G. Notarstefano, "A primal decomposition method with suboptimality bounds for distributed mixed-integer linear programming," in *Conference on Decision and Control*. IEEE, 2018, pp. 3391–3396.

[44] A. Camisa and G. Notarstefano, "Primal decomposition and constraint generation for asynchronous distributed mixed-integer linear programming," in *18th European Control Conference*. IEEE, 2019, pp. 77–82.

[45] C. Cassandras and J. Lygeros, *Stochastic hybrid systems*. CRC Press, 2018.

[46] J. Cassels, "Measures of the non-convexity of sets and the Shapley–Folkman–Starr theorem," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 78, no. 3. Cambridge University Press, 1975, pp. 433–436.

[47] T. Chang, A. Nedić, and A. Scaglione, "Distributed constrained optimization by consensus-based primal-dual perturbation method," *IEEE Trans. on Automatic Control*, vol. 59, no. 6, pp. 1524–1538, 2014.

[48] CPLEX, IBM ILOG, "User's manual for CPLEX, V12.1," *Int. Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.

[49] R. Dakin, "A tree-search algorithm for mixed integer programming problems," *The Computer Journal*, vol. 8, no. 3, pp. 250–255, 1965.

[50] T. Dang, O. Maler, and R. Testylier, "Accurate hybridization of nonlinear systems," in *Proceedings of the 13th ACM international conference on Hybrid systems: Computation and Control*, 2010, pp. 11–20.

[51] J. Desrosiers and M. Lübbecke, "A primer in column generation," in *Column generation*. Springer, 2005, pp. 1–32.

[52] M. Doan, T. Keviczky, and B. De Schutter, "A distributed optimization-based approach for hierarchical MPC of large-scale systems with coupled dynamics and constraints," in *50th Conference on Decision and Control and European Control Conference*. IEEE, 2011, pp. 5236–5241.

[53] J. Eilbrecht, M. Bieshaar, S. Zernetsch, K. Doll, B. Sick, and O. Stursberg, "Model-predictive planning for autonomous vehicles anticipating intentions of vulnerable road users by artificial neural networks," in *Symposium Series on Computational Intelligence*. IEEE, 2017, pp. 1–8.

[54] J. Eilbrecht and O. Stursberg, "Cooperative driving using a hierarchy of mixed-integer programming and tracking control," in *Intelligent Vehicles Symposium*. IEEE, 2017, pp. 673–678.

[55] J. Eilbrecht and O. Stursberg, "Optimization-based maneuver automata for cooperative trajectory planning of autonomous vehicles," in *European Control Conference*. IEEE, 2018, pp. 82–88.

[56] J. Eilbrecht and O. Stursberg, "Reducing computation times for planning of reference trajectories in cooperative autonomous driving," in *Intelligent Vehicles Symposium*. IEEE, 2019, pp. 146–152.

[57] E. Emerson, "Temporal and modal logic," in *Formal Models and Semantics*. Elsevier, 1990, pp. 995–1072.

[58] F. Facchinei and J. Pang, *Finite-dimensional variational inequalities and complementarity problems*. Springer Science & Business Media, 2007.

[59] A. Falsone, K. Margellos, S. Garatti, and M. Prandini, "Dual decomposition for multi-agent distributed optimization with coupling constraints," *Automatica*, vol. 84, pp. 149–158, 2017.

[60] A. Falsone, K. Margellos, and M. Prandini, "A decentralized approach to multi-agent MILPs: finite-time feasibility and performance guarantees," *Automatica*, vol. 103, pp. 141–150, 2019.

[61] A. Falsone, I. Notarnicola, G. Notarstefano, and M. Prandini, "Tracking-ADMM for distributed constraint-coupled optimization," *Automatica*, vol. 117, p. 108962, 2020.

[62] I. Filippidis, S. Dathathri, S. Livingston, N. Ozay, and R. Murray, "Control design for hybrid systems with TuLiP: The temporal logic planning toolbox," in *Conference on Control Applications*. IEEE, 2016, pp. 1030–1041.

[63] A. Firooznia, R. Bourdais, and B. De Schutter, "A distributed algorithm to determine lower and upper bounds in branch and bound for hybrid model predictive control," in *54th IEEE Conference on Decision and Control*. IEEE, 2015, pp. 1736–1741.

[64] E. Fogel and D. Halperin, "Exact and efficient construction of Minkowski sums of convex polyhedra with applications," *Computer-Aided Design*, vol. 39, no. 11, pp. 929–940, 2007.

[65] C. Garcia, D. Prett, and M. Morari, "Model predictive control: theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[66] M. Geoffrion, "Lagrangean relaxation for integer programming," in *Approaches to integer programming*. Springer, 1974, pp. 82–114.

[67] M. S. Ghasemi and A. Afzalian, "Robust tube-based MPC of constrained piecewise affine systems with bounded additive disturbances," *Nonlinear Analysis: Hybrid Systems*, vol. 26, pp. 86–100, 2017.

[68] P. Ghosh, "A unified computational framework for Minkowski operations," *Computers & Graphics*, vol. 17, no. 4, pp. 357–378, 1993.

[69] R. Gondhalekar, J. Imura, and K. Kashima, "Controlled invariant feasibility—a general approach to enforcing strong feasibility in MPC applied to move-blocking," *Automatica*, vol. 45, no. 12, pp. 2869–2875, 2009.

[70] D. Groß, *Distributed model predictive control with event-based communication*. Kassel University Press GmbH, 2015.

[71] D. Groß and O. Stursberg, "Distributed predictive control for a class of hybrid systems with event-based communication," *IFAC Proceedings Volumes*, vol. 46, no. 27, pp. 383–388, 2013.

[72] L. Grüne and S. Pirkelmann, "Economic model predictive control for time-varying system: Performance and stability results," *Optimal Control Applications and Methods*, vol. 41, no. 1, pp. 42–64, 2020.

[73] K. Hariprasad and S. Bhartiya, "A computationally efficient robust tube based MPC for linear switched systems," *Nonlinear Analysis: Hybrid Systems*, vol. 19, pp. 60–76, 2016.

[74] S. Hedlund and A. Rantzer, "Optimal control of hybrid systems," in *38th Conference on Decision and Control*, vol. 4. IEEE, 1999, pp. 3972–3977.

[75] W. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, no. 7, pp. 1085–1091, 2001.

[76] T. Henzinger, "The theory of hybrid automata," in *Verification of digital and hybrid systems*. Springer, 2000, pp. 265–292.

[77] S. Karaman, R. Sanfelice, and E. Frazzoli, "Optimal control of mixed logical dynamical systems with linear temporal logic specifications," in *47th Conference on Decision and Control*. IEEE, 2008, pp. 2117–2122.

[78] E. Kerrigan, "Robust constraint satisfaction: Invariant sets and predictive control," Ph.D. dissertation, University of Cambridge, 2001.

[79] E. Kerrigan and D. Mayne, "Optimal control of constrained, piecewise affine systems with bounded disturbances," in *Proceedings of the 41st Conference on Decision and Control*, vol. 2. IEEE, 2002, pp. 1552–1557.

[80] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Trans. on Automatic Control*, vol. 53, no. 1, pp. 287–297, 2008.

[81] D. Kontny and O. Stursberg, "Fast control using homotopy properties for obstacle-avoidance of systems with input constraints," in *Conference on Computer Aided Control System Design*. IEEE, 2016, pp. 654–660.

[82] D. Kontny and O. Stursberg, "Fast optimizing control for non-convex state constraints using homotopy properties," in *55th Conference on Decision and Control*. IEEE, 2016, pp. 4894–4900.

[83] D. Kontny and O. Stursberg, "Online adaption of motion paths to time-varying constraints using homotopies," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3331–3337, 2017.

[84] H. Kwakernaak and R. Sivan, *Linear optimal control systems*. Wiley-interscience New York, 1972, vol. 1.

[85] R. Langerak and J. Polderman, "Tools for stability of switching linear systems: Gain automata and delay compensation," in *IEEE 44th Conference on Decision and Control*, 2005, pp. 4867–4872.

[86] W. Langson, I. Chryssochoos, S. Raković, and D. Mayne, "Robust model predictive control using tubes," *Automatica*, vol. 40, no. 1, pp. 125–133, 2004.

[87] F. Lauer and G. Bloch, "Switched and piecewise nonlinear hybrid system identification," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2008, pp. 330–343.

[88] E. Lawler and D. Wood, "Branch-and-bound methods: A survey," *Operations Research*, vol. 14, no. 4, pp. 699–719, 1966.

[89] D. Liberzon, *Switching in systems and control*. Springer Science & Business Media, 2003.

[90] D. Limón, T. Alamo, F. Salas, and E. Camacho, "On the stability of constrained MPC without terminal constraint," *IEEE Trans. on Automatic Control*, vol. 51, no. 5, pp. 832–836, 2006.

[91] H. Lin and P. Antsaklis, "Robust regulation of polytopic uncertain linear hybrid systems with networked control system applications," in *Stability and Control of Dynamical Systems with Applications*. Springer, 2003, pp. 71–96.

[92] K. Ling, J. Maciejowski, A. Richards, and B. F. Wu, "Multiplexed model predictive control," *Automatica*, vol. 48, no. 2, pp. 396–401, 2012.

[93] Z. Liu and O. Stursberg, "Optimal trajectory planning of hybrid systems by efficient MIQP encoding," in *Conference on Decision and Control*. IEEE, 2018, pp. 1548–1553.

[94] Z. Liu and O. Stursberg, "Optimierungsbasierte Regelung und Steuerung Hybrid-Dynamischer Systeme," *at-Automatisierungstechnik*, vol. 66, no. 11, pp. 928–938, 2018.

[95] Z. Liu and O. Stursberg, "Optimizing online control of constrained systems with switched dynamics," in *European Control Conference*. IEEE, 2018, pp. 788–794.

[96] Z. Liu and O. Stursberg, "Distributed control of networked systems with coupling constraints," *at-Automatisierungstechnik*, vol. 67, no. 12, pp. 1007–1018, 2019.

[97] Z. Liu and O. Stursberg, "Efficient optimal control of hybrid systems with conditioned transitions," in *15th International Conference on Control and Automation*. IEEE, 2019, pp. 223–230.

[98] Z. Liu and O. Stursberg, "Recursive feasibility and stability of MPC with time-varying and uncertain state constraints," in *18th European Control Conference*. IEEE, 2019, pp. 1766–1771.

[99] Z. Liu and O. Stursberg, "Efficient solution of distributed MILP in control of networked systems," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6723–6729, 2020.

[100] Z. Liu and O. Stursberg, "Robust point-to-set control of hybrid systems with uncertainties using constraint tightening," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6806–6812, 2020.

[101] Z. Liu and O. Stursberg, "Distributed control of networked systems with nonlinear dynamics and coupling constraints," in *IEEE 20th European Control Conference*, 2021, p. 1945–1950.

[102] Z. Liu and O. Stursberg, "Distributed solution of MIQP problems arising for networked systems with coupling constraints," in *IEEE 20th European Control Conference*, 2021, p. 2420–2425.

[103] Z. Liu and O. Stursberg, "Efficient solution of distributed MIP in control of networked systems," *PAMM*, vol. 20, no. 1, p. e202000160, 2021.

[104] J. Löfberg, "Oops! I cannot do it again: Testing for recursive feasibility in MPC," *Automatica*, vol. 48, no. 3, pp. 550–555, 2012.

[105] M. Maiworm, T. Bäthge, and R. Findeisen, "Scenario-based model predictive control: Recursive feasibility and stability," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 50–56, 2015.

[106] T. Manrique, M. Fiacchini, T. Chambrion, and G. Millérioux, "MPC tracking under time-varying polytopic constraints for real-time applications," in *European Control Conference*, 2014, pp. 1480–1485.

[107] T. Manrique, M. Fiacchini, T. Chambrion, and G. Millérioux, "MPC-based tracking for real-time systems subject to time-varying polytopic constraints," *Optimal Control Applications and Methods*, vol. 37, no. 4, pp. 708–729, 2016.

[108] D. Mayne, "Robust and stochastic model predictive control: Are we going in the right direction?" *Annual Reviews in Control*, vol. 41, pp. 184–192, 2016.

[109] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[110] D. Mayne, M. Seron, and S. Raković, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, no. 2, pp. 219–224, 2005.

[111] J. Mitchell, "Branch-and-cut algorithms for combinatorial optimization problems," *Handbook of Applied Optimization*, vol. 1, pp. 65–77, 2002.

[112] L. Mitten, "Branch-and-bound methods: General formulation and properties," *Operations Research*, vol. 18, no. 1, pp. 24–34, 1970.

[113] T. Moor and J. Davoren, "Robust controller synthesis for hybrid systems using modal logic," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2001, pp. 433–446.

[114] J. Nash, "Non-cooperative games," *Annals of Mathematics*, pp. 286–295, 1951.

[115] I. Necoara, B. De Schutter, T. J. van den Boom, and J. Hellendoorn, "Model predictive control for perturbed continuous piecewise affine systems with bounded disturbances," in *43rd Conference on Decision and Control*, vol. 2. IEEE, 2004, pp. 1848–1853.

[116] A. Nedić and J. Liu, "Distributed optimization for control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 77–103, 2018.

[117] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Trans. on Automatic Control*, vol. 60, no. 3, pp. 601–615, 2014.

[118] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. Tsitsiklis, "Distributed subgradient methods and quantization effects," in *47th Conference on Decision and Control*. IEEE, 2008, pp. 4177–4184.

[119] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.

[120] A. Nedić, A. Olshevsky, and W. Shi, "Improved convergence rates for distributed resource allocation," in *Conference on Decision and Control*. IEEE, 2018, pp. 172–177.

[121] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[122] V. Nenchev, C. Belta, and J. Raisch, "Optimal motion planning with temporal logic and switching constraints," in *European Control Conference*. IEEE, 2015, pp. 1141–1146.

[123] T. Nghiem, S. Sankaranarayanan, G. Fainekos, F. Ivancić, A. Gupta, and G. Pappas, "Monte-carlo techniques for falsification of temporal properties of non-linear hybrid systems," in *Proceedings of the 13th ACM international conference on Hybrid systems: Computation and Control*. ACM, 2010, pp. 211–220.

[124] G. Notarstefano, I. Notarnicola, and A. Camisa, "Distributed optimization for smart cyber-physical networks," *Foundations and Trends® in Systems and Control*, vol. 7, no. 3, pp. 253–383, 2019.

[125] H. Ohtake, K. Tanaka, and H. Wang, "Piecewise nonlinear control," in *42nd Conference on Decision and Control*, vol. 5. IEEE, 2003, pp. 4735–4740.

[126] A. Pakniyat and P. Caines, "On the hybrid minimum principle: The hamiltonian and adjoint boundary conditions," *IEEE Trans. on Automatic Control*, 2020.

[127] D. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1439–1451, 2006.

[128] B. Passenberg, "Theory and algorithms for indirect methods in optimal control of hybrid systems," Ph.D. dissertation, Technische Universität München, 2012.

[129] B. Passenberg, P. Caines, M. Sobotka, O. Stursberg, and M. Buss, "The minimum principle for hybrid systems with partitioned state space and unspecified discrete state sequence," in *49th Conference on Decision and Control*. IEEE, 2010, pp. 6666–6673.

[130] B. Passenberg, M. Sobotka, O. Stursberg, M. Buss, and P. Caines, "An algorithm for discrete state sequence and trajectory optimization for hybrid systems with partitioned state space," in *49th Conference on Decision and Control*. IEEE, 2010, pp. 4223–4229.

[131] A. Pereira and M. Althoff, "Safety control of robots under computed torque control using reachable sets," in *International Conference on Robotics and Automation*. IEEE, 2015, pp. 331–338.

[132] P. Pflaum, M. Alamir, and M. Lamoudi, "Comparison of a primal and a dual decomposition for distributed MPC in smart districts," in *Conference on Smart Grid Communications*. IEEE, 2014, pp. 55–60.

[133] E. Plaku, L. Kavraki, and M. Vardi, "Hybrid systems: from verification to falsification by combining motion planning and discrete search," *Formal Methods in System Design*, vol. 34, no. 2, pp. 157–182, 2009.

[134] A. Pnueli, "The temporal logic of programs," in *18th Annual Symposium on Foundations of Computer Science*. IEEE, 1977, pp. 46–57.

[135] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[136] S. Rakovic, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen, "Parameterized tube model predictive control," *IEEE Trans. on Automatic Control*, vol. 57, no. 11, pp. 2746–2761, 2012.

[137] J. Rawlings and D. Mayne, *Model predictive control: Theory and design*. Nob Hill Pub. Madison, Wisconsin, 2009.

[138] J. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games," *Econometrica: Journal of the Econometric Society*, pp. 520–534, 1965.

[139] M. Rungger and O. Stursberg, "Optimal control for deterministic hybrid systems using dynamic programming," *IFAC Proceedings Volumes*, vol. 42, no. 17, pp. 316–321, 2009.

[140] P. Scokaert, D. Mayne, and J. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Trans. on Automatic Control*, vol. 44, no. 3, pp. 648–654, 1999.

[141] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Trans. on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[142] W. Shi and L. Pavel, "Lana: an ADMM-like nash equilibrium seeking algorithm in decentralized environment," in *American Control Conference*. IEEE, 2017, pp. 285–290.

[143] M. Silva, A. Bemporad, M. Botto, and J. da Costa, "Optimal control of uncertain piecewise affine/mixed logical dynamical systems," in *European Control Conference*. IEEE, 2003, pp. 1573–1578.

[144] A. Simonetto and H. Jamali-Rad, "Primal recovery from consensus-based dual decomposition for distributed convex optimization," *Journal of Optimization Theory and Applications*, vol. 168, no. 1, pp. 172–197, 2016.

[145] R. Soloperto, J. Köhler, F. Allgöwer, and M. Müller, "Collision avoidance for uncertain nonlinear systems with moving obstacles using robust model predictive control," in *18th European Control Conference*. IEEE, 2019, pp. 811–817.

[146] M. Stankovic, K. Johansson, and D. Stipanovic, "Distributed seeking of Nash equilibria with applications to mobile sensor networks," *IEEE Trans. on Automatic Control*, vol. 57, no. 4, pp. 904–919, 2011.

[147] R. Starr, "Quasi-equilibria in markets with non-convex preferences," *Econometrica: journal of the Econometric Society*, pp. 25–38, 1969.

[148] R. Starr, "Approximation of points of the convex hull of a sum of sets by points of the sum: an elementary approach," *Journal of Economic Theory*, vol. 25, no. 2, pp. 314–317, 1981.

[149] R. Stubbs and S. Mehrotra, "A branch-and-cut method for 0-1 mixed convex programming," *Mathematical Programming*, vol. 86, no. 3, pp. 515–532, 1999.

[150] O. Stursberg and S. Engell, "Optimal control of switched continuous systems using mixed-integer programming," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 433–438, 2002.

[151] O. Stursberg and H. Krogh, "Efficient representation and computation of reachable sets for hybrid systems," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2003, pp. 482–497.

[152] O. Stursberg and S. Panek, "Control of switched hybrid systems based on disjunctive formulations," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2002, pp. 421–435.

[153] C. Sun and R. Dai, "Distributed optimization for convex mixed-integer programs based on projected subgradient algorithm," in *Conference on Decision and Control*. IEEE, 2018, pp. 2581–2586.

[154] H. Sussmann, "A maximum principle for hybrid optimal control problems," in *Proceedings of the 38th Conference on Decision and Control*, vol. 1. IEEE, 1999, pp. 425–430.

[155] R. Takapoui, N. Moehle, S. Boyd, and A. Bemporad, "A simple effective heuristic for embedded mixed-integer quadratic programming," *International Journal of Control*, vol. 93, no. 1, pp. 2–12, 2020.

[156] T. Tatarenko, W. Shi, and A. Nedić, "Accelerated gradient play algorithm for distributed Nash equilibrium seeking," in *Conference on Decision and Control*. IEEE, 2018, pp. 3561–3566.

[157] A. Testa, A. Rucco, and G. Notarstefano, "A finite-time cutting plane algorithm for distributed mixed integer linear programming," in *56th Conference on Decision and Control*. IEEE, 2017, pp. 3847–3852.

[158] T. Tsang, D. Himmelblau, and T. Edgar, "Optimal control via collocation and non-linear programming," *International Journal of Control*, vol. 21, no. 5, pp. 763–768, 1975.

[159] C. Uribe, S. Lee, A. Gasnikov, and A. Nedić, "A dual approach for optimal algorithms in distributed optimization over networks," *Optimization Methods and Software*, pp. 1–40, 2020.

[160] O. Von Stryk, "Numerical solution of optimal control problems by direct collocation," in *Optimal Control*. Springer, 1993, pp. 129–143.

[161] R. Vujanic, M. Esfahani, J. Goulart, S. Mariéthoz, and M. Morari, "A decomposition method for large scale MILPs, with performance guarantees and a power system application," *Automatica*, vol. 67, pp. 144–156, 2016.

[162] R. Vujanic, P. Esfahani, P. Goulart, and M. Morari, "Large scale mixed-integer optimization: A solution method with supply chain applications," in *22nd Mediterranean Conference on Control and Automation*. IEEE, 2014, pp. 804–809.

[163] N. Wada, H. Tomosugi, and M. Saeki, "Model predictive tracking control for a linear system under time-varying input constraints," *Int. Journal of Robust and Nonlinear Control*, vol. 23, no. 9, pp. 945–964, 2013.

[164] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *51st Conference on Decision and Control*. IEEE, 2012, pp. 5445–5450.

[165] H. P. Williams, *Model Building in Mathematical Programming*, 1st ed. Wiley, 1978.

[166] E. Wolff, U. Topcu, and R. Murray, "Optimization-based trajectory generation with linear temporal logic specifications," in *International Conference on Robotics and Automation*. IEEE, 2014, pp. 5319–5325.

[167] X. Xu and P. Antsaklis, "Optimal control of switched systems based on parameterization of the switching instants," *IEEE Trans. on Automatic Control*, vol. 49, no. 1, pp. 2–16, 2004.

[168] B. Yang and M. Johansson, "Distributed optimization and games: A tutorial overview," in *Networked Control Systems*. Springer, 2010, pp. 109–148.

[169] M. Ye and G. Hu, "Distributed Nash equilibrium seeking by a consensus based approach," *IEEE Trans. on Automatic Control*, vol. 62, no. 9, pp. 4811–4818, 2017.

[170] B. Yordanov, J. Tumova, I. Cerna, J. Barnat, and C. Belta, "Temporal logic control of discrete-time piecewise affine systems," *IEEE Trans. on Automatic Control*, vol. 57, no. 6, pp. 1491–1504, 2011.

[171] W. Zhang, A. Abate, J. Hu, and M. Vitus, "Exponential stabilization of discrete-time switched linear systems," *Automatica*, vol. 45, no. 11, pp. 2526–2536, 2009.

[172] M. Zhu and S. Martinez, "On distributed convex optimization under inequality and equality constraints," *IEEE Trans. on Automatic Control*, vol. 57, no. 1, pp. 151–164, 2011.

ISBN 978-3-7376-0976-0