Multi-Agent Model Predictive Control

with Applications to Power Networks

R.R. Negenborn

Multi-Agent Model Predictive Control

with Applications to Power Networks

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificu prof.dr.ir. J.T. Fokkema,
voorzitter van het College van Promoties,
in het openbaar te verdedigen op dinsdag 18 december 2007 om 10:00 uur
door Rudy R. NEGENBORN,
doctorandus in de informatica,
geboren te Utrecht.

Dit proefschrift is goedgekeurd door de promotoren:
Prof.dr.ir. J. Hellendoorn
Prof.dr.ir. B. De Schutter

Samenstelling promotiecommissie:

Rector Magnificu	voorzitter
Prof.dr.ir. J. Hellendoorn	Technische Universiteit Delft, promotor
Prof.dr.ir. B. De Schutter	Technische Universiteit Delft, promotor
Prof.dr. G.J. Olsder	Technische Universiteit Delft
Prof.dr. J.-J.Ch. Meyer	Universiteit Utrecht
Prof.Dr. G. Andersson	ETH Zürich
Prof.Dr.-Ing. W. Marquardt	RWTH Aachen University
Ir. J.J.M. Langedijk	Siemens Nederland N.V.
Prof.dr. C. Witteveen	Technische Universiteit Delft (reservelid)



This thesis has been completed in partial fulfillmen of the requirements of the Dutch Institute of Systems and Control (DISC) for graduate studies. The research described in this thesis was supported by the project “Multi-agent control of large-scale hybrid systems” (DWV.6188) of the Dutch Technology Foundation STW and by an NWO Van Gogh grant (VGP79-99).

TRAIL Thesis Series T2007/14, The Netherlands TRAIL Research School

Published and distributed by: R.R. Negenborn
E-mail: http://contact.negenborn.net/
WWW: http://www.negenborn.net/mampc/

ISBN 978-90-5584-093-9

Keywords: multi-agent control, model predictive control, power networks, transportation networks.

Printed in The Netherlands

Preface

I owe a lot of thanks to the people that I have lived around and worked with over the last years. I first of all express my gratitude to my promotors Bart De Schutter and Hans Hellendoorn for supervising and promoting my research. Bart and Hans together have exactly these properties that are essential for good supervision and promotorship. Together they are not only dedicated, in time, sharp, and detailed, but also visionary, creative, social, and practical. They have always shown interest in my work, and allowed me to work in an international research environment. I thank them for enabling me to develop myself both scientifically and organizationally.

I have greatly appreciated and benefited from the cooperation with and the feedback received from partners in various projects. I thank the members of the user committee of the project "Multi-agent control of large-scale hybrid systems" for their input during the project meetings. I thank the participants of the HYCON project for introducing me to the topics in the fields of hybrid control and power systems. In particular, I thank A. Giovanni Beccuti, Gilney Damm, Gabriela Hug-Glanzmann, and Sylvain Leirens for our cooperation and inspiring discussions. Also, I thank the people involved in the project "Next Generation Infrastructures", in particular Michiel Houwing, Koen van Dam, and Zofia Lukszo, for our fruitful collaboration on infrastructure modeling and operation. Furthermore, I thank Hervé Guéguen and Jean Buisson for having me as a guest researcher at Supélec, Rennes, France, through our Van Gogh grant.

It has been a delight for me to work among my colleagues at the Delft Center for Systems and Control (DCSC), of whom I in particular thank Robert Babuška, Jelmer Braaksma, Sjoerd Dietz, Rogier Ellenbroek, Redouane Hallouzi, András Hegyi, Diederick Joosten, and Ronald van Katwijk, for sharing enjoyable times both inside and outside the office, among others while playing tennis and while juggling. Besides this, I cherish the discussions on PhD affairs and policies, and the organizing of informative events and social meetings, such as the printer's market, the PhD café, and the PhD barbecue, with my colleagues in the board of Promood. I thank in particular Loesje Bevers, Anand Dokania, Gwen van Eijk, Tom Van Helleputte, and Frederik de Wit, for their cooperation, for the good times, and for their drive to represent the PhD candidates of TU Delft.

I acknowledge the efforts of the members of my PhD committee and their constructive remarks on my research. I thank Thijs Kinkhorst and David van Prooijen for being my paranymphs, and, most of all, I am grateful to my family, in particular Kateřina Staňková, for their love and support throughout the years.

Rudy R. Negenborn,
Delft, December 2007.

Contents

Chapter 1

Introduction

In this chapter we present the background and the motivation for the research addressed in this thesis. In Section 1.1 we first introduce the type of systems that we consider: transportation networks in general, and power networks in particular. In Section 1.2 we then discuss controlling such systems and motivate the use of multi-agent control structures. In Section 1.3 the conceptual ideas of model predictive control are presented as strategy for the control agents to determine which actions to take, and various issues to be addressed in relation with model predictive control and multi-agent control structures for transportation networks are discussed. In Section 1.4 we discuss opportunities for the use of multi-agent model predictive control in the power networks of the future, and in Section 1.5 we conclude the chapter with an overview and road map of this thesis, and a list of the contributions to the state of the art.

Parts of this chapter have been published in [107].

1.1 Transportation networks

Transportation or infrastructure networks, like power distribution networks [82], traffic and transportation systems [33], water distribution networks [21], logistic operations networks [88], etc., are the corner stones of our modern society. A smooth, efficient, reliable, and safe operation of these systems is of huge importance for the economic growth, the environment, and the quality of life, not only when the systems are pressed to the limits of their performance, but also under regular operating conditions. Recent examples illustrate this. E.g., the problems in the USA and Canada [141], Italy [139], Denmark and Sweden [43], The Netherlands, Germany, Belgium, and France [140], and many other countries [114, 148] due to power outages have shown that as power network operation gets closer to its limits, small disturbances in heavily loaded lines can lead to large black-outs causing not only huge economic losses, but also mobility problems as trains and metros may not be able to operate. Also, as road traffic operation gets closer to its limits, unexpected situations in road traffic networks can lead to heavy congestion. Not only the huge traffic congestion after incidents such as bomb alerts are examples of this, also the almost daily road-traffic jams due to accidents illustrate this convincingly.

Expanding the physical infrastructure of these networks could help to relieve the issues

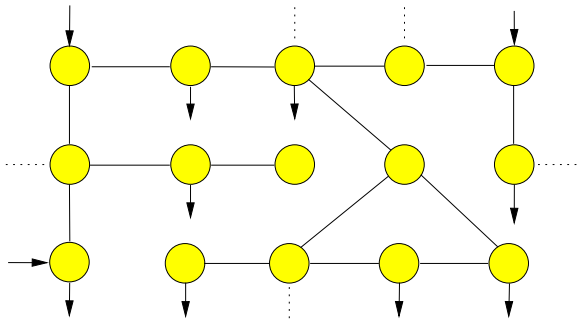


Figure 1.1: Generic transportation network. Commodity enters the network at sources (circles with an arrow pointing towards them), flows over links to other elements in the network that alter the flows (at each circle, and leaves the network at sinks (circles with an arrow pointing outward). Dotted lines represent connections with other parts of the network.

in transportation networks, although at extremely high costs. As alternative to spending this money on building new infrastructure, it is worth spending effort on investigating improved use of the existing infrastructure by employing intelligent control techniques that combine state-of-the-art techniques from f elds like systems and control engineering [6], optimization [18], and multi-agent systems [147], with domain-specif c knowledge.

The examples of networks just mentioned are only some particular types of networks within the much larger class of transportation networks. Common to transportation networks is that at a generic level they can be seen as a set of nodes, representing the components or elements of the network, and interconnections between these nodes. In addition, transportation networks have some sort of commodity, that is brought into the network at source nodes, that f ows over links to sink nodes, and that is inf uenced in its way of f owing over the network by elements inside the network, as illustrated in Figure 1.1. Other characteristics that are common to transportation networks are:

- they typically span a large geographical area;
- they have a modular structure consisting of many subsystems;
- they have many actuators and sensors;
- they have dynamics evolving over different time scales.

In addition to this, transportation networks often contain both continuous (e.g., f ow evolution) and discrete dynamics (e.g., on and off switching), and are therefore also referred to as hybrid systems [143]. This mixture of characteristics makes that transportation networks can show extremely complex dynamics.

Even though transportation networks differ in the details of commodity, sources, sinks, etc., it is worth to consider them in a generic setting. On the one hand, methods developed for generic transportation networks can be applied to a wide range of specif c domains, perhaps using additional f ne-tuning and domain-specif c enhancements to improve the per-

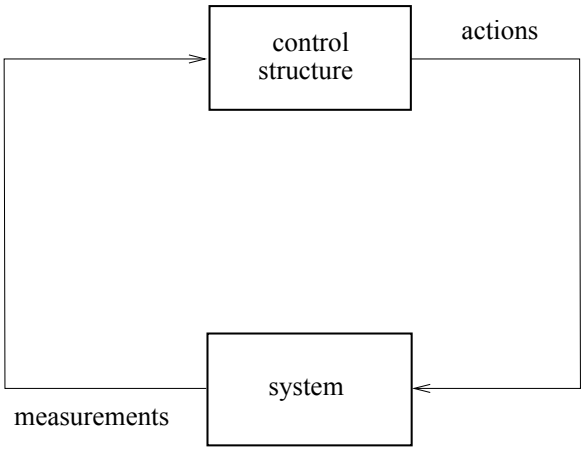


Figure 1.2: The relation between a general system and the control structure that controls the system.

formance. On the other hand, approaches specif cally developed for a particular domain can be applied to other domains after having transfered them to the generic framework.

1.2 Control structures

There are many users, controllers, players, actors, and operators involved in the evolution of transportation networks. Each of these refers to entities that directly or indirectly change the way commodity is f owing. Different users may have different objectives, and these objectives may be conf icting. Objectives that users may have involve avoiding congestion of links, maximizing throughput, minimizing costs of control actions, minimizing travel times, etc. An example of conf icting objectives in a road traff c network is given by the objectives that the individual road users have on the one hand and road authority has on the other: The individual road users want to minimize the travel time to their destination, whereas the road authority wants to minimize overall network congestion [134]. An example in the domain of power networks is given by the objectives that the individual households have on the one hand and the government has on the other: The individual households aim at minimizing the costs on energy, whereas the government aims at maximizing usage of the perhaps more expensive green energy. Also, in power networks, it may sometimes be benef cial for the overall network performance to cut off certain parts of the network from electricity consumption in a controlled way in order to prevent large black-outs [142], even though individual consumers perhaps do not want this.

In order to formalize the operation of transportation networks, consider Figure 1.2. The f gure illustrates the overall picture of a *system* on the one hand and a *control structure* on the other. The system is the entity that is under control, and the control structure is the entity that controls the system. Hence, the control structure is the concept used to indicate the structure that produces actuator settings. The control structure monitors the system by making measurements and based on these chooses control actions that are implemented on

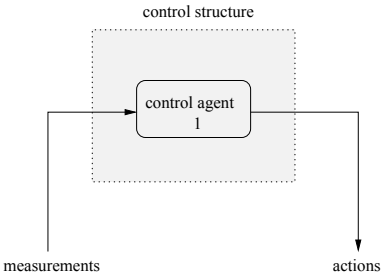
the system. The system evolves subject to these actions to a new state, which is again measured by the control structure. The control structure consists of one or more components, called *control agents*. These control agents try to determine settings for the actuators inside the system in such a way that their own objectives are met as closely as possible and any constraints are satisfied. In our case, the system consists of the transportation network, and the components of the control structure consists of all the users, controllers, operators, players, etc., from now on only referred to as the control agents.

The control structure is a very general concept and can have many different shapes. A first important distinguishing feature between control structures is the number of control agents that constitute the control structure. E.g., the control structure can consist of a single control agent or multiple control agents. Some other properties in which control structures can differ are:

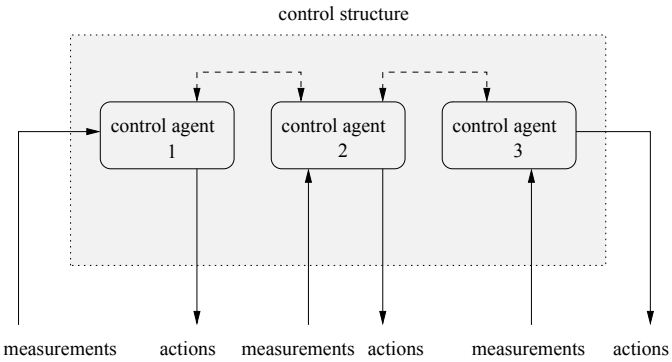
- the access that the control agents have to the sensors and actuators in the network,
- the communication that the control agents have among one another,
- the way in which the control agents process sensor data to obtain actions,
- the authority relations between the control agents,
- the beliefs, desires, and intentions of the control agents,
- etc.

Defining different types of control structures is difficult due to the large amount of properties that they can have. However, some general types of control structures can be identified, that have increasing complexity, that are commonly encountered in theory and practice, and that will also be of particular interest in the subsequent chapters:

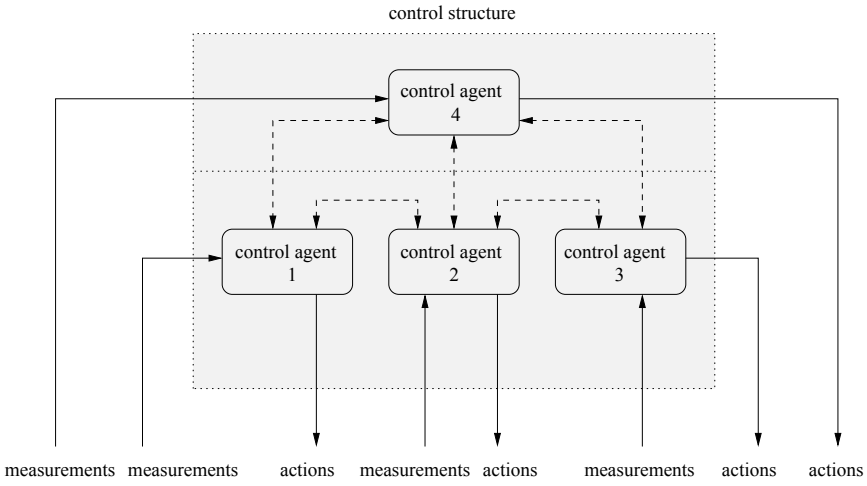
- When it is assumed that there is only one control agent, that has access to all actuators and sensors of the network and thus directly controls the physical network, then this control structure is referred to as an *ideal single-agent* control structure, as illustrated in Figure 1.3(a). The control structure is referred to as an ideal structure, since in principle such a control structure can determine actions that give optimal performance.
- When there are multiple control agents, each of them considering only its own part of the network and being able to access only sensors and actuators in that particular part of the network, then the control structure is referred to as a *multi-agent single-layer* control structure, as illustrated in Figure 1.3(b). If in addition the agents in the control structure do not communicate with each other, the control structure is *decentralized*. If the agents do communicate with each other, the control structure is *distributed*.
- When there are multiple control agents, and some of these control agents have authority over other control agents, in the sense that they can force or direct other control agents, then the control structure is a *multi-layer* control structure, as illustrated in Figure 1.3(c) A multi-layer control structure typically is present when one control agent determines set-points to a group of other control agents, that work in a decentralized or distributed way. Due to the authority relationship between agents or groups



(a) Single-agent control structure. The single control agent makes measurements of the system and provides actions to the network.



(b) Multi-agent single-layer control structure. Multiple control agents make measurements and provide actions to the network. Communication between the control agents is optionally present (dashed line).



(c) Multi-layer control structure. A higher-layer control agent can make measurements and provide actions to the network and can in addition direct or steer a lower control layer.

Figure 1.3: Some important types of control structures.

of agents, the multi-layer control structure can also be referred to as a supervisory control structure, or a hierarchical control structure.

1.2.1 Control structure design

Suppose that a particular network is given and that any control structure can be implemented on it. The question that then arises is the question of how it can be determined what the best control structure is. Unfortunately, theories for determining general control structures are lacking. However, motivations for preferring one type of control structure over another can be given.

Advantages of single-agent control structures are in general that they can deliver the best performance possible, and that they have been studied extensively in the literature, in particular for small-scale systems. However, there are several issues that complicate the use of single-agent control structures for large-scale transportation networks such as:

- undesirable properties with respect to robustness, reliability, scalability, and responsiveness;
- technical issues related to communication delays and computational requirements;
- commercial, legal, and political issues related to unavailability of information and restricted control access.

These reasons motivate the use of multi-agent control structures [135, 145, 147], which are expected to be able to deal or at least relieve these issues. Multi-agent control structures can in principle:

- improve robustness and reliability, since if one control agents fails, another can take over, and improve responsiveness, since the control agents typically use only local measurements and therefore can react quicker to changing situations;
- reduce communication delays, since the control agents operate locally and therefore solve problems that may be smaller, and since coomunication typically takes place among nearby control agents;
- deal with unavailability of information and restricted control access, since the control agents only require information of their own part of the network and since they determine actions only for their own part of the network.

However, typically multi-agent control structures have a lower performance than the performance of ideal single-agent control structures and implementing schemes that give desired performance is far from trivial.

An advantage of the decentralized over the distributed multi-agent single-layer control structures is that there is no communication between the controllers, resulting in lower computational requirements and faster control. However, this advantage will typically be at the price of decreased overall performance. The advantage of a distributed multi-agent single-layer control structure is therefore that improved performance can be obtained, although at the price of increased computation time due to cooperation, communication, and perhaps negotiation among control agents. However, even though improved performance can

be obtained, the performance will still typically be lower than the performance of an ideal single-agent control structure.

The multi-agent multi-layer control structure provides the possibility to obtain a trade-off between system performance and computational complexity. A higher layer considers a larger part of the system and can therefore direct the lower control layer to obtain coordination. Such a multi-layer control structure can thus combine the advantages of the single-agent control structure with the multi-agent single-layer control structure, i.e., overall system performance with tractability. It is noted, however, that communication in a multi-agent multi-layer control structure is typically more complex than in a single-agent control structure and a multi-agent single-layer control structure.

Note that in practice often a particular control structure is already in place, and that the control structure cannot be redesigned from scratch. The question in this case is not so much the question of what control structure is best, but of how the currently existing control structure can be changed, such that the performance is improved. Of course here it has to be def ned what the performance is, and in a control structure with control agents with conf icting objectives it may not be possible to reach consensus on this.

1.2.2 Assumptions for design and analysis

In this thesis we develop control strategies for several control structures. Due to the complexity of transportation networks, we have to narrow the scope of control problems that we will consider. Our focus will mostly be on the most fundamental of transportation network control problems: the operational control of transportation networks, in which amounts of commodity to be transported over the network are given, and controllers have to ensure that transport over the network can take place at acceptable service levels, while satisfying any constraints, both under normal and emergency operating conditions.

In order to make the analysis and the design of the control structures more tractable, assumptions have to be made, both on the network and the control structure. Assumptions relating to the network are made on the dynamics of the network, i.e., the way in which the components in the network function. E.g., the dynamics can be assumed to evolve over continuous time or in discrete-time, they can be assumed to involve only continuous dynamics, or both continuous and discrete dynamics, and they can be assumed to be instantaneous or not. In each chapter we explicitly point out which particular assumptions are made on the network.

With respect to the control structure, we assume in each of the following chapters that:

- the control agents are already present;
- the control agents control f xed parts of the network, and they can access actuators and sensors in these parts of the network;
- the control agents know what qualitative behavior is desired for the parts of the network they control;
- the control agents strive for the best possible overall performance of the network;
- the control agents can measure the state of the parts of the network that they control.

Under such assumptions it remains to be decided on how the agents in the control structure get from their measurements to actuator settings, i.e., what protocols, computations, and information exchanges take place inside the control structure. Assumptions on these are made in the subsequent chapters. In the following section we discuss the approach that we propose to be used by the control agents in a multi-agent control structure for transportation network control: model predictive control.

1.3 Model predictive control

To find the actions that meet the control objectives as well as possible, the control agents have to make a trade-off between the different available actions. In order to make the best decision and hence find the best actions, all relevant information about the consequences of choosing actions should be taken into account. For power networks, typical information that is available consists of forecasts on power consumption and exchanges [55], capacity limits on transmission lines, dynamics of components like generators, capacitor banks, transformers, and loads [82]. Furthermore, typically area-wide measurements of voltage magnitude and angles across the network can be made to provide an up-to-date status of the situation of the network. A particularly useful form of control for transportation network that in principle can use all information available is model predictive control (MPC) [27, 93].

1.3.1 Single-agent MPC

Over the last decades MPC (also knowns as receding horizon control or moving horizon control) has become an important strategy for finding control policies for complex, dynamic systems. MPC in a single-agent control structure has shown successful application in the process industry [93, 102], and is now gaining increasing attention in fields like amongst others power networks [49, 61], road traffic networks [58], railway networks [36], steam networks [94], supply chain management [146], food processing [130], mine planning [56], heat-exchanger networks [54], greenhouse control [123], and drug delivery [22].

Concept

MPC is a control strategy that is typically used in a discrete-time control context, i.e., control actions are determined in discrete control cycles of a particular duration which in itself is expressed in continuous time units[1]. From the beginning of one control cycle until the beginning of the next control cycle, the control actions stay fixed, i.e., a zero-order hold strategy is employed.

In each control cycle the MPC control agent uses the following information, as illustrated in Figure 1.4:

- an *objective function* expressing which system behavior and actions are desired;
- a *prediction model* describing the behavior of the system subject to actions;

[1] Although usually the term control sample step is used to indicate the discrete step at which a control agent determines its actions, we refer to this as control cycle, since later on we will require control step to denote certain steps inside multi-agent MPC strategies.

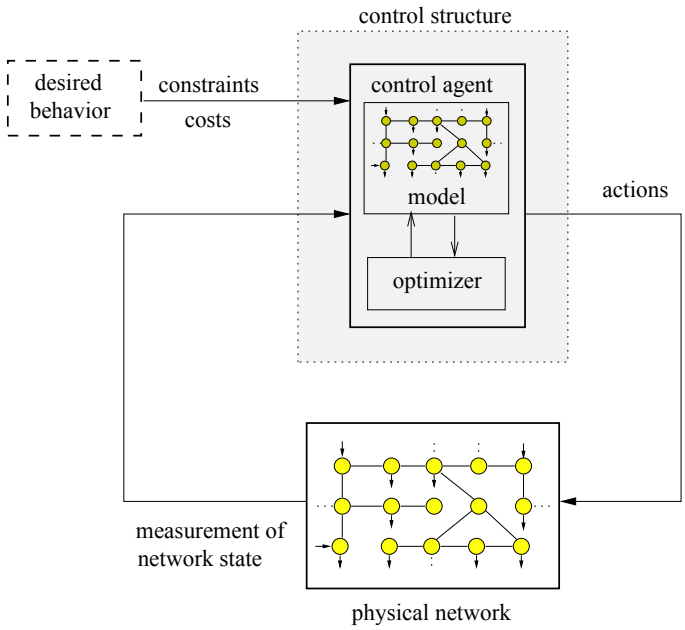


Figure 1.4: Single-agent MPC.

- possibly *constraints* on the states, the inputs, and the outputs of the system (where the inputs and the outputs of the system correspond to the actions and the measurements of the control agent, respectively);
- possibly known information about future disturbances;
- a *measurement* of the state of the system at beginning of the current control cycle.

The objective of the control agent is to determine those actions that optimize the behavior of the system and minimize costs as specif ed through the objective function. In order to f nd the actions that lead to the best performance, the control agent uses the prediction model to predict the behavior of the system under various actions over a certain prediction horizon, starting from the state at the beginning of the control cycle. Once the control agent has determined the actions that optimize the system performance over the prediction horizon, it implements the actions until the beginning of the next control cycle, at which point the control agent determines new actions over the prediction horizon starting at that point, using updated information. Hence, the control agent operates in a receding or rolling horizon fashion to determine its actions.

In general it is preferable to have a longer prediction horizon, since by considering a longer prediction horizon, the control agent can better oversee the consequences of its actions. At some length, however, increasing the length of the prediction horizon may not improve the performance, if transients in the dynamics may have become negligible. For computational reasons, determining the actions over a very long horizon typically is not tractable, and in addition due to potential uncertainty in the prediction model and in predictions of future disturbances, a smaller prediction horizon is usually considered. Hence, in

practice, the prediction horizon should be long enough to cover the most important dynamics, i.e., those dynamics dominating the performance, and short enough to give tractable computations. It should hereby also be noted that if a prediction horizon is used that is too short, the system could arrive in states from which it cannot continue due to the presence of constraints, e.g., on the actions. The prediction horizon should thus have such a length that arriving in such states can be avoided.

MPC Algorithm

Summarizing, a control agent in a single-agent control structure using MPC to determine its actions performs at each control cycle the following:

1. Measure the current state of the system.
2. Determine which actions optimize the performance over the prediction horizon by solving the following optimization problem:

minimize	the objective function in terms of actions over the prediction horizon
subject to	the dynamics of the whole network over the prediction horizon,
	the constraints on, e.g., ranges of actuator inputs and link capacities,
	the measurement of the initial state of the network at the beginning of the current control cycle.

3. Implement the actions until the next control cycle, and return to step 1.

Advantages and issues

Advantages of MPC are that in principle it can take into account all available information and that it can therefore anticipate undesirable situations in the future at an early stage. Additional advantages of MPC are [93]:

- its explicit way of handling constraints on actions, states, and outputs;
- its ability to operate without intervention for long periods;
- its ability to adapt to slow changes in the system parameters;
- its ability to control systems with multiple inputs and multiple outputs;
- its relatively easy tuning procedure;
- its built-in robustness properties.

However, there are also some issues that have to be addressed before a control agent using an MPC strategy can be implemented successfully:

- the control goals have to be specified;
- the prediction model has to be constructed;

- the measurement of the system state has to be available;
- a solution approach has to be available that can solve the MPC optimization problem;
- the solution approach has to be tractable.

Basic issues, e.g., stability and robustness, have extensively been studied for MPC in single-agent control structures [102], in particular for linear time-invariant systems. For other classes of systems there are still many open issues. E.g., tractability issues of MPC for nonlinear and discrete-event systems, and for systems in which variables take on discrete values, still deserve attention. E.g., in [106] we propose one approach to make the MPC problem for a system modeled as a Markov decision process more tractable and to deal with changing system dynamics by including experience using reinforcement learning. Another class of systems for which there are still many open questions are hybrid systems, i.e., systems including both continuous and discrete dynamics. This class of systems currently receives signif cant attention in MPC research and will be considered in more detail in Chapters 3 and 4.

1.3.2 Multi-agent MPC

As mentioned in the previous section, in a multi-agent control structure, there are multiple control agents, each of them controlling only its own subnetwork, i.e., a part of the overall network. Multi-agent MPC issues have been investigated since the 90s in [1, 2, 12, 25, 28, 38, 41, 48, 53, 72, 74, 75, 77, 117, 129, 144].

In multi-agent MPC, multiple control agents in the control structure use MPC, but now they f rst measure the subnetwork state, then they determine the best actions over the predicted subnetwork evolution, and then they implement actions. Although this may seem like a straightforward extension of single-agent MPC at f rst sight, when considering the details it is not.

The actions that an agent in a multi-agent control structure takes inf uence both the evolution of the subnetwork it controls, and the evolution of the subnetworks connected to its subnetwork. Since the agents in a multi-agent control structure usually have no global overview and can only access a relatively small number of sensors and actuators, predicting the evolution of a subnetwork over a horizon involves even more uncertainty than when a single agent is employed. In addition, when a control agent in a multi-layer control structure provides set-points to another agent, this supervisory control changes the way in which the other agent chooses its actions, and thus the higher-layer control agent changes the performance of the system. Hence, the interactions between the agents make multi-agent MPC involved.

Under the assumption that the control agents strive for an optimal overall network performance, the challenge in implementing such a multi-agent MPC strategy comes from ensuring that the actions that the individual agents choose result in a performance that is as good as when a hypothetical single-agent control structure in which all information is available would be used.

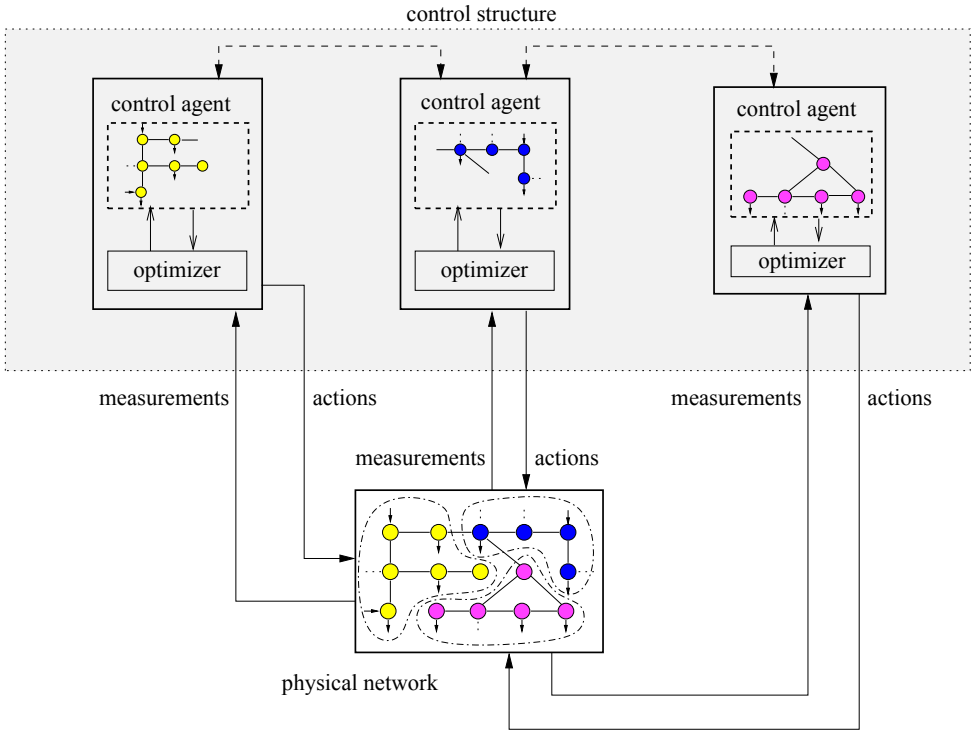


Figure 1.5: Multi-agent single-layer MPC.

Multi-agent single-layer MPC

In the multi-agent single-layer control structure each control agent only has information gathering and action capabilities that are restricted to that part of the network that a particular control agent controls, as illustrated in Figure 1.5. The challenge in implementing multi-agent single-layer MPC comes from predicting the dynamics of the subnetwork, since as mentioned, its evolution is inf uenced by the other agents. The underlying problem of MPC for multi-agent control structures can therefore be seen as optimization over a distributed simulation.

Issues To make accurate predictions of the evolution of the subnetwork, a control agent requires the current state of its subnetwork, a sequence of actions over the prediction horizon, and predictions of the evolution of the interconnections with other subnetworks. The predictions of the evolution of the interconnections with other subnetworks are based on the information communicated with the neighboring control agents. In Chapter 2 we classify how existing approaches implement this. One particular class of methods aims at achieving cooperation among control agents in an iterative way in which in each control cycle control agents perform several iterations consisting of local problem solving and communication.

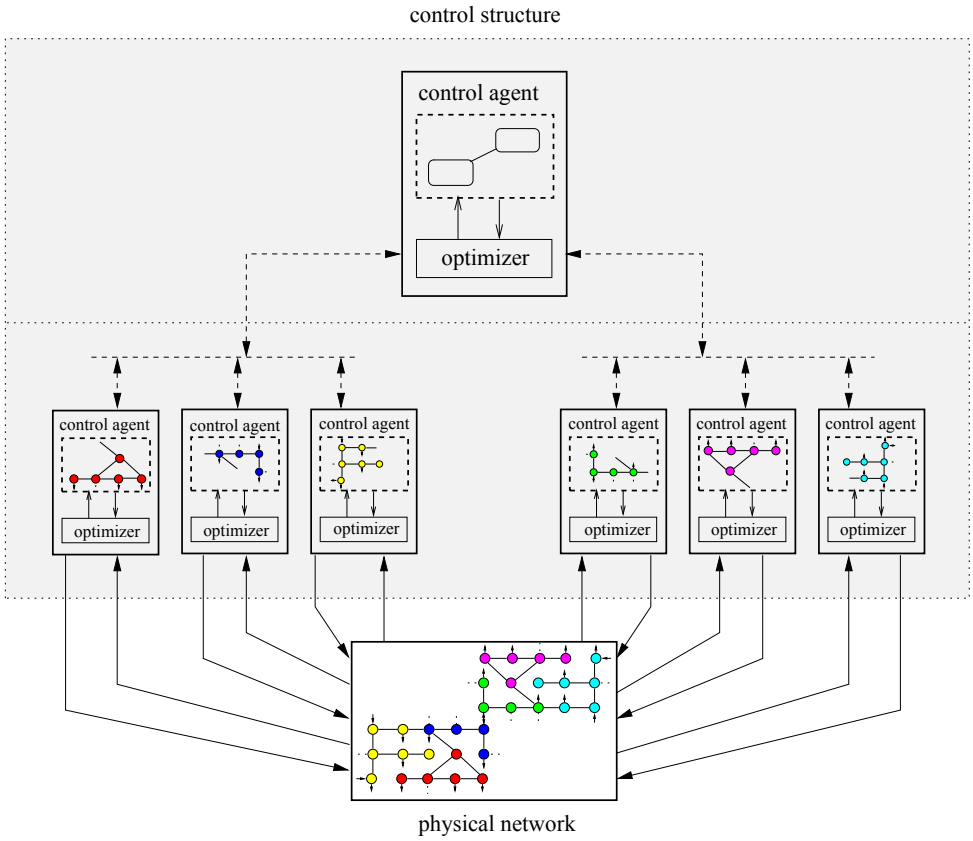


Figure 1.6: Multi-agent multi-layer MPC.

In each iteration agents obtain information about what the plans of neighboring agents are. Ideally at the end of the iterations the agents have found actions that lead to overall optimal performance. In Chapter 2 we discuss such schemes.

As is the case with MPC for single-agent control structures, having both continuous and discrete dynamics causes computational problems. In transportation networks this combination is commonly encountered, and it is therefore relevant to study models that take this into account. In Chapter 3 such models and MPC for multi-agent single-layer control structures for these models are considered.

A further complicating issue arises when the subnetworks that the agents control are overlapping. Existing strategies assume that the subnetworks that the control agents control are non-overlapping. However, in some applications the subnetworks considered by the control agents are overlapping. In Chapter 5 this issue is further addressed.

Multi-agent multi-layer MPC

In the multi-layer multi-agent MPC case there are multiple control layers in the control structure, i.e., there are authority relationships between the agents in the sense that some agents provide set-points or directions to other agents. The agents at higher layers typically consider a larger region of the network and consider slower time scales than agents in lower layers. Figure 1.6 illustrates this.

MPC can also be used by a control agent in a higher layer of the control structure. This higher-layer control agent can then coordinate the lower layer, which may consist of control agents using multi-agent single-layer MPC, or of control agents that use alternative control strategies. The higher-layer control agent then coordinates the lower control layer by enforcing penalty terms, providing additional constraints, or providing set-points. The advantage of the higher-layer control agent is in particular clear when the control agents of the lower layer are working decentralized, i.e., not communicating with one another.

Issues An important issue to be addressed when designing MPC for multi-agent multi-layer control structures is the choice of the prediction model that the higher-layer control agent uses. A higher-layer control agent has to be able to make relevant predictions of the physical system, but since the physical system is under control of the lower-control layer, the lower control layer has to be taken into account by the higher-layer control agent as well. In addition, the prediction model that the higher-layer control agent uses will typically involve both continuous and discrete elements, since it has to consider a larger part of the network than lower-layer agents. This makes the resulting MPC control problem more complex, and eff cient ways have to be found to solve it eff ciently. In Chapter 4 we address these issues.

1.4 Power networks

In this thesis we develop MPC for multi-agent control structures. In order to illustrate the performance of the developed techniques we use problems from the domain of power networks.

1.4.1 Physical power networks

Power networks [82, 92, 128] are large transportation networks consisting of a large number of components. The dynamics of the power network as a whole are the result of the interactions between the individual components. The generators produce power that is injected into the network on the one side, while the loads consume power from the network on the other. The distribution of the power in the network is dictated by Kirchhoff's laws and inf uenced by the settings of the generators, loads, transformers, and potentially also by capacitor banks and FACTS devices. This ensemble of components together produces an evolution over time of active and reactive power, and voltage magnitudes and angles. Power networks do not only exhibit continuous dynamics, but also discrete dynamics. Discrete dynamics in power networks appear due to discrete events triggered by on and off switching of generators and loads, breaking of transmission lines, discrete switching logic inside transformers, saturation effects in generators, etc. Hence, power networks are large-scale hybrid systems with complex dynamics.

1.4.2 Future power networks

Power networks are evolving towards a new structure. Conventionally, in power networks, power was generated in several large power generators. This power was then transported through the transmission and distribution network to the location where it was consumed, e.g., households and industry. Power f ows were relatively predictable, and the number of control agents was relatively low. Due to the ongoing deregulation in the power generation and distribution sector in the U.S. and Europe, the number of players involved in the generation and distribution of power has increased signif cantly. In the near future the number of source nodes of the power distribution network will even further increase as also large-scale industrial suppliers and small-scale individual households will start to feed electricity into the network [73].

As a consequence, the structure of the power distribution network is changing from a hierarchical top-down structure into a much more decentralized system with many generating sources and distributing agencies. This multi-player structure thus results in a system with many interactions and interdependencies. In addition, the following interesting developments are taking or will take place:

- At a European scale the electricity networks of the individual countries are becoming more integrated as high-capacity power lines are constructed to enhance system security [132]. The national network operators will have to cooperate and coordinate more at a European scale to operate the power network in a desirable way.
- At a national scale power does not any longer only f ow from the transmission network in the direction of the distribution network and onwards to the industrial sites and cities, but can also f ow from the industrial sites and cities to the distribution network and into the transmission network [73]. The network f ows will vary more and it will therefore be necessary to improve the coordination of decentralized local controllers, and to improve the cooperation between power regions.
- At the local scale loads at consumption nodes become controllable and it becomes possible to store energy using batteries [73]. In addition, groups of households can become independent of the large electricity suppliers by arranging energy exchanges among each other.

Hence, to still guarantee basic requirements and service levels, such as voltage levels, frequency, bounds on deviations, stability, elimination of transients, etc., and to meet the demands and requirements of the users, new control techniques have to be developed and implemented. These control techniques have to be adaptive and online as the input patterns and demands may vary over time.

1.4.3 Opportunities for multi-agent control

The developments outlined above offer many new opportunities for multi-agent control. In this thesis we deal in particular with and propose new solutions for control problems inspired by the following power domain control problems:

- distributed load-frequency control of non-overlapping power areas (Chapters 2 and 3);

- distributed FACTS devices control for security of overlapping power areas (Chapter 5);
- supervisory emergency voltage control for coordination of a layer of decentralized controllers (Chapter 4);
- decentralized control of electricity and heat usage in households (Chapter 3).

The first three problems aim at improving the operational control of power networks, ensuring adequate system performance under normal and emergency operating conditions. Here, system security is the main issue, and economical objectives are less important. The last problem aims more at economical optimization, and assumes the system operations to be reliable.

1.5 Overview of this thesis

1.5.1 Thesis outline

In this thesis current issues in model predictive control (MPC) in multi-agent control structures with applications to control problems in power networks are discussed and new solutions are proposed. This thesis is organized as follows:

- In **Chapter 2** communication and decision making schemes for multi-agent MPC are discussed, with a particular focus on serial versus parallel schemes. A novel serial scheme for multi-agent MPC is proposed and compared with an existing parallel scheme. The emphasis is on networks modeled as interconnected linear time-invariant subnetworks, a basic, yet important class of networks. The theory developed is applied to the load-frequency control problem in power networks.
- In **Chapter 3** multi-agent MPC for networked hybrid systems is studied. Translating discrete phenomena like saturation into systems of inequalities is discussed, and an extension of the schemes of Chapter 2 for dealing with interconnected linear time-invariant subnetworks with both real and integer inputs is proposed. A decentralized MPC controller for household optimization is constructed, and the load-frequency control problem of Chapter 2 is extended by including discrete switching of power generation.
- In **Chapter 4** the focus is on multi-layer multi-agent control. Creating object-oriented prediction models to construct models of complex systems is discussed, and a medium-layer MPC controller is proposed that uses such a model to determine set points for a lower decentralized control layer. The theory is applied to a voltage collapse problem in a nine-bus dynamic power network.
- In **Chapter 5** higher-layer multi-agent MPC for controlling networks in which the subnetworks are overlapping is proposed. Conventional approaches assume non-overlapping subnetworks, in which control objectives and system dynamics can be clearly assigned to individual subnetworks. An extension of a recently developed scheme for multi-agent MPC is proposed for situations in which the subnetworks are

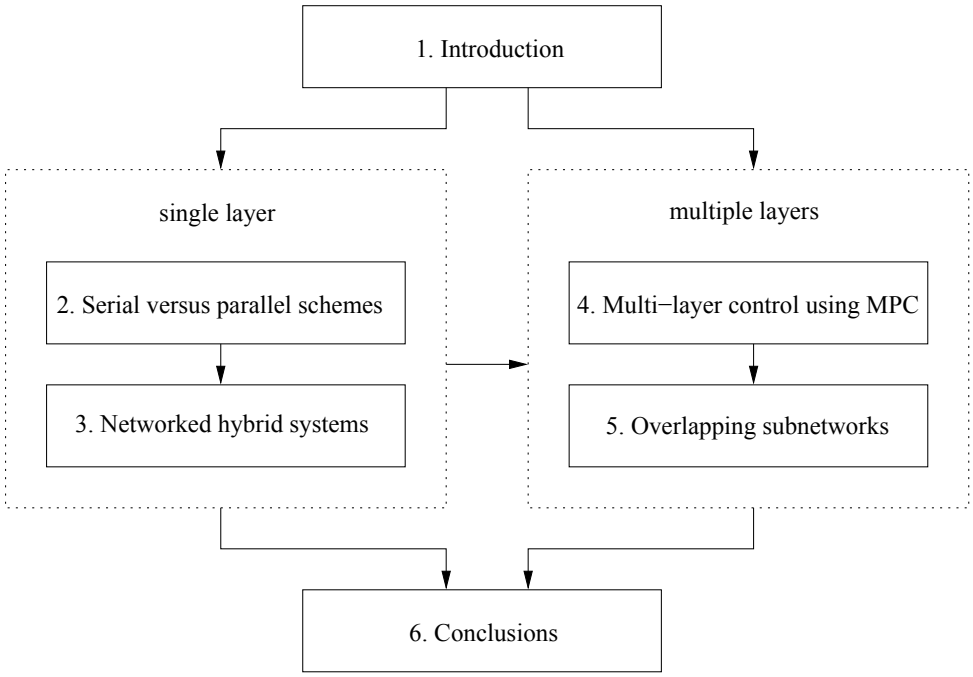


Figure 1.7: Road map. Arrows indicate read before *relations.*

overlapping. The developed scheme is used for FACTS-controlled optimal power f ow control.

- **Chapter 6** summarizes the results of this thesis and outlines directions for future research.

1.5.2 Road map

Figure 1.7 illustrates a grouping of the chapters in related subjects and an ordering in which the chapters can be read. It is suggested to read the chapters in the order as they appear in the thesis. Chapter 1 contains a general introduction to the topics in this thesis, and is therefore suggested to be read f rst. Chapters 2 and 3 focus both on issues related to control by control agents that have equal authority relationships, and therefore operate in a single layer. In addition, the schemes discussed in these chapters assume that subnetworks are non-overlapping. Chapters 4 and 5 focus on issues related to control by control agents with different authority relationships, and therefore operate in multiple layers. In addition, in Chapter 5 it is assumed that subnetworks are overlapping. It is therefore suggested to read Chapters 2 and 3 before Chapters 4 and 5. Chapter 6 summarizes the results of this thesis and gives directions for future research. This chapter can be read after the other chapters.

1.5.3 Contributions

Main contributions

The main contributions of the research described in this PhD thesis with respect to model predictive control and multi-agent systems are the following:

- A serial scheme for multi-agent single-layer MPC has been proposed for interconnected linear time-invariant systems in [109, 112], and for a class of interconnected linear hybrid systems in [108] (see also Chapters 2 and 3).
- A coordinating MPC control strategy using an object-oriented prediction model has been proposed in [113], and using a linearized object-oriented prediction model in [110] (see also Chapter 4).
- A parallel scheme for multi-agent single-layer MPC for nonlinear overlapping subnetworks has been proposed in [69] (see also Chapter 5).

With respect to power network control our main contributions are:

- A solution approach for distributed load-frequency control has been proposed for continuous problems in [109, 112], and for hybrid problems in [108] (see also Chapters 2 and 3).
- A decentralized MPC controller for optimization of energy consumption in households has been proposed in [68] (see also Chapter 3).
- Two solution approaches for coordinating decentralized controllers for emergency voltage control have been proposed in [110] and [113] (see also Chapter 4).
- A solution approach for FACTS-based security control in overlapping power areas has been proposed in [69] (see also Chapter 5).

Contributions to the state-of-the-art

Besides our main contributions, the research involved in this PhD thesis has resulted in additional contributions to the state-of-the-art in the following ways:

- A unified framework of multi-agent MPC strategies has been proposed in [107] (see also Chapter 2).
- A parallelization of the serial multi-agent MPC scheme has been proposed in [111].
- The integration of multi-level, in particular bi-level, control and multi-agent MPC has been discussed in [90].
- Challenges for process system engineering in transportation network control have been identified in [89].
- An MPC controller for Markov decision processes using experience to decrease computational requirements has been proposed in [106].

Chapter 2

Serial versus parallel schemes

In this chapter we consider multi-agent single-layer MPC, in which the network is divided into several non-overlapping subnetworks, and each subnetwork is controlled by one control agent, as shown in Figure 1.5. The agents have to locally choose those actions that give an overall optimal performance. In Section 2.1 we introduce the assumptions that we make on the network and control structure. In Section 2.2 we then formulate the MPC problem considering only one particular control agent, assuming that it knows how the surrounding network behaves. In Section 2.3 we relax this assumption and discuss how interconnections between control problems of different agents are formalized and how the multi-agent single-layer MPC approaches can differ in dealing with these interconnections. In Section 2.4 we focus on particular types of schemes, viz. *synchronous, multi-iteration, parallel*, and *serial* schemes. We propose a novel serial scheme based on Lagrange theory, and compare this scheme with a related parallel scheme. In Section 2.5 we propose the application of the approaches to the load-frequency control problem of power networks. A benchmark network is def ned and through experimental simulation studies on this network we illustrate the performance of the parallel and the serial scheme.

Parts of this chapter have been published in [89, 107, 109] and presented in [112].

2.1 Network and control setup

2.1.1 Network dynamics

As discussed in Chapter 1, transportation networks are large-scale systems with complex dynamics. In order to analyze them, assumptions have to be made on the dynamics, i.e., on the way the networks behave. Therefore, assume a network that is divided into n subnetworks, where each subnetwork consists of a set of nodes and the interconnections between these nodes. Assume furthermore that the dynamics of subnetwork $i \in \{1,\ldots,n\}$ are given by a deterministic linear discrete-time time-invariant model (possibly obtained after symbolic or numerical linearization of a nonlinear model in combination with discretization):

$$\mathbf{x}_i(k+1) = \mathbf{A}_i\mathbf{x}_i(k)+\mathbf{B}_{1,i}\mathbf{u}_i(k)+\mathbf{B}_{2,i}\mathbf{d}_i(k)+\mathbf{B}_{3,i}\mathbf{v}_i(k) \tag{2.1}$$

$$\mathbf{y}_i(k) = \mathbf{C}_i\mathbf{x}_i(k)+\mathbf{D}_{1,i}\mathbf{u}_i(k)+\mathbf{D}_{2,i}\mathbf{d}_i(k)+\mathbf{D}_{3,i}\mathbf{v}_i(k), \tag{2.2}$$

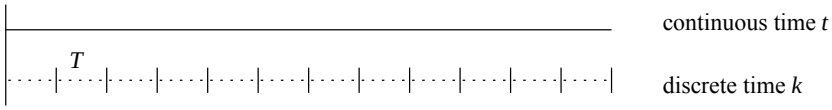


Figure 2.1: From continuous time to discrete time.

where at time step k, for subnetwork i, $\mathbf{x}_i(k) \in \mathbb{R}^{n_{\mathbf{x}_i}}$ are the local states, $\mathbf{u}_i(k) \in \mathbb{R}^{n_{\mathbf{u}_i}}$ are the local inputs, $\mathbf{d}_i(k) \in \mathbb{R}^{n_{\mathbf{d}_i}}$ are the local known exogenous inputs, $\mathbf{y}_i(k) \in \mathbb{R}^{n_{\mathbf{y}_i}}$ are the local outputs, $\mathbf{v}_i(k) \in \mathbb{R}^{n_{\mathbf{v}}}$ are the remaining variables influencing the local dynamical states and outputs, and $\mathbf{A}_i \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{x}_i}}$, $\mathbf{B}_{1,i} \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{u}_i}}$, $\mathbf{B}_{2,i} \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{d}_i}}$, $\mathbf{B}_{3,i} \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{v}_i}}$, $\mathbf{C}_i \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{x}_i}}$, $\mathbf{D}_{1,i} \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{u}_i}}$, $\mathbf{D}_{2,i} \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{d}_i}}$, $\mathbf{D}_{3,i} \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{v}_i}}$ determine how the different variables influence the local states and outputs of subnetwork i. The $\mathbf{v}_i(k)$ variables appear due to the fact that a subnetwork is connected to other subnetworks. Hence, the $\mathbf{v}_i(k)$ variables represent the influence of other subnetworks on subnetwork i. If the values of $\mathbf{v}_i(k)$ are fixed, then the dynamics of subnetwork i are decoupled from the other subnetworks.

Remark 2.1 For completeness inputs $\mathbf{u}_i(k)$ are also allowed to influence outputs $\mathbf{y}_i(k)$ at time k. A situation in which such direct feed-through terms typically appear is when algebraic relations are linearized, e.g., when linearizing equations describing instantaneous (power) flow distributions. □

Remark 2.2 In the subnetwork description that we consider here, all variables involved take on values in the real domain. This assumes that no discrete inputs, due to, e.g., switches, are present. In addition, in the subnetwork description that we consider here, the dynamics are assumed linear. Therefore, discrete behavior, e.g., due to saturation or discrete logic, cannot be included. In Chapter 3 we discuss issues related to including such discrete elements. □

Remark 2.3 In general the dynamics of the networks take place in continuous time. For computational reasons, however, it is convenient to assume that the continuous-time dynamics are adequately represented by discrete-time dynamics. Hence, instead of specifying and computing the dynamics of the network for each continuous-time instant $t \in [0, \infty)$, the dynamics are only specified and computed at discrete time or control cycle steps k, each representing T continuous-time time units, as shown in Figure 2.1. In Chapter 4 we discuss issues related to dealing with continuous-time dynamics in more detail. □

Remark 2.4 In general, the dynamics of the subnetworks are nonlinear. In Chapter 4 we discuss in more detail how to obtain linear models from nonlinear models by linearization. □

2.1.2 Control structure

We consider a multi-agent single-layer control structure as introduced in Section 1.3.2. Each of the subnetworks $i \in \{1, \ldots, n\}$ is controlled by a control agent i that:

- has a prediction model M_i of the dynamics of subnetwork i that matches the subnetwork dynamics given by (2.1)–(2.2);

- can measure the state $\mathbf{x}_i(k)$ of its subnetwork;
- can determine settings $\mathbf{u}_i(k)$ for the actuators of its subnetwork;
- can obtain exogenous inputs $\mathbf{d}_i(k+l)$ of its subnetwork over a certain horizon of length N, for $l = \{0,\ldots,N\}$;
- can communicate with neighboring agents, i.e., the agents controlling the subnetworks $j \in \mathcal{N}_i$, where $\mathcal{N}_i = \{j_{i,1},\ldots,j_{i,m_i}\}$ is the set of indexes of the m_i subnetworks connected to subnetwork i, also referred to as the *neighbors* of subnetwork or agent i.

Remark 2.5 The agents have no authority relations over one another, i.e., there is no agent that can force another agent to do something, and each agent has only information about its own subnetwork. In Chapter 4 we discuss how supervisory agents that can steer or direct other agents can be used. □

Remark 2.6 The multi-agent control structure studied here may be used not only for control of networks that span large geographical areas, but also for control of relatively small networks, when restrictions on acting and sensing make single-agent control impossible. □

2.2 MPC of a single subnetwork

Assume for now that the control agent of subnetwork i operates individually, that it therefore does not communicate with other agents, and that it knows how the surrounding network behaves. Below we will relax these assumptions.

The control agent employs MPC to determine which actions to take. In MPC, an agent determines its actions by computing optimal actions over a prediction horizon of N control cycles according to an objective function, subject to a model of the subnetwork, the behavior of the surrounding network, and additional constraints.

The MPC strategy of agent i at time k consists of measuring the initial local state[1] $\bar{\mathbf{x}}_i(k)$, determining local exogenous inputs over the horizon $\bar{\mathbf{d}}_i(k+l)$, for $l = \{0,\ldots,N-1\}$, and predicting inf uences of the rest of the network over the prediction horizon $\bar{\mathbf{v}}_i(k+l)$, for $l = \{0,\ldots,N-1\}$. Here, for notational convenience, the bar over variables indicates that the values of these variables are known. In addition, below the tilde over variables is used to denote variables over the prediction horizon, e.g., $\tilde{\mathbf{a}}_i(k) = [\ \mathbf{a}_i(k)^{\mathrm{T}},\ldots,\mathbf{a}_i(k+N-1)^{\mathrm{T}}\ ]^{\mathrm{T}}$. Control agent i then solves the following optimization problem:

$$\min_{\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k),\tilde{\mathbf{y}}_i(k)} J_{\mathrm{local},i}\left(\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k),\tilde{\mathbf{y}}_i(k)\right) = \sum_{l=0}^{N-1} J_{\mathrm{stage},i}(\mathbf{x}_i(k+1+l),\mathbf{u}_i(k+l),\mathbf{y}_i(k+l)) \tag{2.3}$$

[1] The measured initial local state is in this case the exact initial local state, since no measurement noise is considered.

subject to

$$\mathbf{x}_i(k+1+l) = \mathbf{A}_i\mathbf{x}_i(k+l)+\mathbf{B}_{1,i}\mathbf{u}_i(k+l)+\mathbf{B}_{2,i}\bar{\mathbf{d}}_i(k+l)+\mathbf{B}_{3,i}\mathbf{v}_i(k+l) \tag{2.4}$$

$$\mathbf{y}_i(k+l) = \mathbf{C}_i\mathbf{x}_i(k+l)+\mathbf{D}_{1,i}\mathbf{u}_i(k+l)+\mathbf{D}_{2,i}\bar{\mathbf{d}}_i(k+l)+\mathbf{D}_{3,i}\mathbf{v}_i(k+l) \tag{2.5}$$

$$\mathbf{v}_i(k+l) = \bar{\mathbf{v}}_i(k+l) \tag{2.6}$$

$$\text{for } l = 0,\ldots,N-1$$

$$\mathbf{x}_i(k) = \bar{\mathbf{x}}_i(k), \tag{2.7}$$

where $J_{\text{stage},i}$ is a twice differentiable function that gives the cost per prediction step given a certain local state, local input, and local output. A typical choice for the stage cost is:

$$J_{\text{stage},i}\left(\mathbf{x}_i(k+1),\mathbf{u}_i(k),\mathbf{y}_i(k)\right) = \begin{bmatrix}\mathbf{x}_i(k+1)\\ \mathbf{u}_i(k)\\ \mathbf{y}_i(k)\end{bmatrix}^{\mathrm{T}} \mathbf{Q}_i \begin{bmatrix}\mathbf{x}_i(k+1)\\ \mathbf{u}_i(k)\\ \mathbf{y}_i(k)\end{bmatrix} + \mathbf{f}_i^{\mathrm{T}} \begin{bmatrix}\mathbf{x}_i(k+1)\\ \mathbf{u}_i(k)\\ \mathbf{y}_i(k)\end{bmatrix}, \tag{2.8}$$

where $\mathbf{Q}_i$ and $\mathbf{f}_i$ are a positive def nite weighting matrix and a vector, respectively. After control agent i has solved the optimization problem and found the N actions over the horizon, it implements the actions $\mathbf{u}_i(k)$ until the next control cycle, the control cycle k moves to $k+1$, and the control agent performs the MPC strategy at that control cycle by setting up and solving the MPC optimization problem for $k+1$.

We have assumed here through (2.6) that the agent does not use communication and that it can by itself locally predict the inf uence of the surrounding network over the prediction horizon, i.e., it knows $\mathbf{v}_i(k+l)$, for $l = 0,\ldots,N-1$. However, control agent i cannot know this inf uence *a priori*, since actions taken by control agent i inf uence the dynamics of its own subnetwork and therefore also the dynamics of a neighboring subnetwork $j \in \mathcal{N}_i$, which therefore changes the decision making of neighboring agent j and, hence, changes the actions that control agent j chooses, which change the dynamics of subnetwork j, and thus changes $\mathbf{v}_i(k+l)$. Therefore, (2.6) cannot be added explicitly. To relax the assumption that this is possible, constraints between control problems and communication between control agents has to be used. Below we discuss this in more detail.

2.3 Interconnected control problems

The interconnections between control problems are modeled using so-called *interconnecting variables*. A particular variable of the control problem of agent i is an interconnecting variable with respect to the control problem of agent j if the variable of agent i corresponds to the same physical quantity as a variable in the control problem of agent j. E.g., a f ow going from subnetwork i into subnetwork j is represented with an interconnecting variable in the control problems of both agents.

Given the interconnecting variables of two agents corresponding to the same quantity, it is convenient to def ne one of these variables as an interconnecting *input* variable and the other as an interconnecting *output* variable. On the one hand, interconnecting input variables $\mathbf{w}_{\text{in},ji}(k)$ of the control problem of agent i with respect to agent j at control cycle k can be seen as inputs caused by agent j on the control problem of agent i. On the other hand, interconnecting output variables $\mathbf{w}_{\text{out},ij}(k)$ of the control problem of agent j with

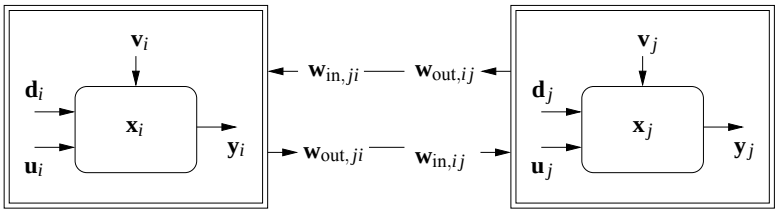


Figure 2.2: Illustration of the relation between the models and interconnecting variables of control agents i and j.

respect to the control problem of agent i can be seen as the inf uence that agent j has on the control problem of agent i. Figure 2.2 illustrates this. We consider interconnecting variables $\mathbf{w}_{\text{in},ji}(k) \in \mathbb{R}^{n_{\mathbf{w}_{\text{in},ji}}}$ and $\mathbf{w}_{\text{out},ji}(k) \in \mathbb{R}^{n_{\mathbf{w}_{\text{out},ji}}}$.

Def ne the interconnecting inputs and outputs for the control problem of agent i over a prediction horizon at control cycle k as:

$$\tilde{\mathbf{w}}_{\text{in},i}(k) = \tilde{\mathbf{v}}_i(k) \tag{2.9}$$

$$\tilde{\mathbf{w}}_{\text{out},i}(k) = \tilde{\mathbf{K}}_i \begin{bmatrix} \tilde{\mathbf{x}}_i(k+1)^{\mathrm{T}} & \tilde{\mathbf{u}}_i(k)^{\mathrm{T}} & \tilde{\mathbf{y}}_i(k)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{2.10}$$

where $\tilde{\mathbf{K}}_i$ is an interconnecting output selection matrix that contains zeros everywhere, except for a single 1 per row corresponding to a local variable that relates to an interconnecting output variable.

The variables $\tilde{\mathbf{w}}_{\text{in},i}(k)$, $\tilde{\mathbf{w}}_{\text{out},i}(k)$ are partitioned such that:

$$\tilde{\mathbf{w}}_{\text{in},i}(k) = \left[\tilde{\mathbf{w}}_{\text{in},j_{i,1}i}(k)^{\mathrm{T}}, \ldots, \tilde{\mathbf{w}}_{\text{in},j_{i,m_i}i}(k)^{\mathrm{T}} \right]^{\mathrm{T}} \tag{2.11}$$

$$\tilde{\mathbf{w}}_{\text{out},i}(k) = \left[\tilde{\mathbf{w}}_{\text{out},j_{i,1}i}(k)^{\mathrm{T}}, \ldots, \tilde{\mathbf{w}}_{\text{out},j_{i,m_i}i}(k)^{\mathrm{T}} \right]^{\mathrm{T}}. \tag{2.12}$$

The interconnecting inputs to the control problem of agent i with respect to agent j must be equal to the interconnecting outputs from the control problem of agent j with respect to agent i, since the variables of both control problems model the same quantity. For agent i this thus gives rise to the following *interconnecting constraints*:

$$\tilde{\mathbf{w}}_{\text{in},ji}(k) = \tilde{\mathbf{w}}_{\text{out},ij}(k) \tag{2.13}$$

$$\tilde{\mathbf{w}}_{\text{out},ji}(k) = \tilde{\mathbf{w}}_{\text{in},ij}(k), \tag{2.14}$$

for all $j \in \mathcal{N}_i$.

An interconnecting constraint depends on variables of two different control agents. Therefore, a particular control agent will always miss information that it requires to include the interconnecting constraint explicitly in its MPC control problem formulation. Hence, the agent has to use communication with another agent to exchange information that it uses to determine which values it should give to the interconnecting inputs and outputs. Below, we survey how schemes for multi-agent single-layer MPC differ in the type of information exchanged and the moments at which information exchange takes place.

2.3.1 Types of information exchange

The challenge is to find a suitable way for the control agents to deal with the interconnecting variables $\tilde{\mathbf{w}}_{\text{in},ji}(k)$ and $\tilde{\mathbf{w}}_{\text{out},ji}(k)$. In order to make a prediction of the evolution of the subnetwork, values of the interconnecting variables have to be known or assumed over the prediction horizon. There are several approaches to dealing with the interconnecting variables, each yielding different types of information that is exchanged:

1. Ignore the influence of the interconnecting variables. This approach is used in a completely decentralized setting. A control agent ignores the presence of other subnetworks completely. This type of control scheme can be used when interconnecting variables have negligible effect on the subnetwork dynamics. An advantage of this approach is the absence of communication overhead. However, if the influence of the interconnecting variables turns out not to be negligible, control performance will degenerate.

2. Use constant values for the values of the interconnecting variables over the full prediction horizon based on a local measurement made or obtained from a neighboring agent. This approach may be useful when the interconnecting variables change slowly. This approach may also be used to monitor the interconnecting variables online and to switch to a different way of dealing with the interconnecting variables when the variables start changing significantly. An advantage of this approach is relatively fast control, since the control agents only exchange information at the beginning of each control cycle once and after that solve their control problems decentralized. A disadvantage of this approach is that if the values of the interconnecting variables exchanged at the beginning of a control cycle are not valid over the complete prediction horizon, the performance of the control will decrease.

3. Use predictions of the values of the interconnecting variables over the full prediction horizon as obtained from a neighboring agent [28, 48, 75]. An advantage of this approach is that there is only communication at the beginning of a control cycle, after which the control agents solve their control problems decentralized. However, the neighboring agent providing the predictions has to make sure that the predictions are correct. In practice, if the subnetwork of the neighboring agent relies on other neighboring subnetworks this will be difficult to ensure. Iterations as discussed below in Section 2.3.2 are then necessary.

4. Use upper and lower bounds on the values of the interconnecting variables, as obtained from a neighboring agent. This assumes that neighboring agents do not communicate exact trajectories, but instead bounds on the values of the interconnecting variables. By enforcing these bounds, an agent can compute worst-case optimal inputs. The agent providing the bounds also has to make sure that its actual trajectory stays within the bounds it has communicated. So-called compatibility constraints can do this for certain linear-time invariant systems [37]. Hence, an advantage of this approach is that control agents do not have to make accurate predictions of the values of interconnecting variables. However, the resulting control may be conservative, since the control agents determine worst-case optimal inputs. In addition, if a control agent requires accurate values for the interconnecting variables in order to make accurate

predictions of the evolution of its subnetwork, only using upper and lower bounds may give bad predictions, and consequently, bad performance.

5. Use a model that predicts the values of the interconnecting variables based on dynamics of neighboring subsystems [37]. When this is used a control agent knows the dynamics or part of the dynamics that generate the values of the interconnecting variables [37]. This is, e.g., the case when the local agent has a copy of the subnetwork models used by its neighbors. These models will depend on variables of the neighboring subnetworks, like inputs, and perhaps interconnecting variables of neighbors of neighbors. An advantage of this approach is that more about the interconnecting variables is known. A disadvantage of this approach can be increased computational time required to determine the predictions.

6. Use a model about the evolution of the interconnecting variables that has been learned given available information from neighboring agents. This approach can be employed if the agent does not have a model of the subnetwork that generates the interconnecting variables. Instead it may employ learning techniques and build up experience to learn a model. An advantage of this approach is that the control agent may exploit the model learned from experience to improve its performance. However, learning such a model in the first place is challenging.

7. Use knowledge about the objective function of neighboring agents together with models of the dynamics of the neighboring system [79]. The control agent can use this information to compute which actions the neighbors will take [79]. It can determine the actions that will be applied to that subsystem and consequently determine the evolution of the values of the interconnecting variables. Knowledge about the objectives of neighboring subnetworks can be used to make local decisions that are not counteracting the objectives of other control agents. Hence, an advantage of this approach is that a control agent can anticipate what other control agents are going to do and therefore possibly increase the efficiency of the decision making. A disadvantage of this approach is that one controller effectively is solving the control problems of multiple subnetworks. Hence, the computational requirements will increase significantly, even more than when approach 5 is used. In an approach that somehow communicates the computed actions to the neighboring subnetworks this could become an advantage however.

2.3.2 Timing of information exchange

Schemes for multi-agent MPC do not only differ in the type of information exchanged, but also in the moment at which information exchange takes place, as shown in Figure 2.3. The schemes are distinguished by the following characteristics:

1. *Synchronous* or *asynchronous*, i.e., do agents have to wait for one another when it comes to sending and receiving information and determining which actions to take, or can they send and receive information and determine which action to take at any time.

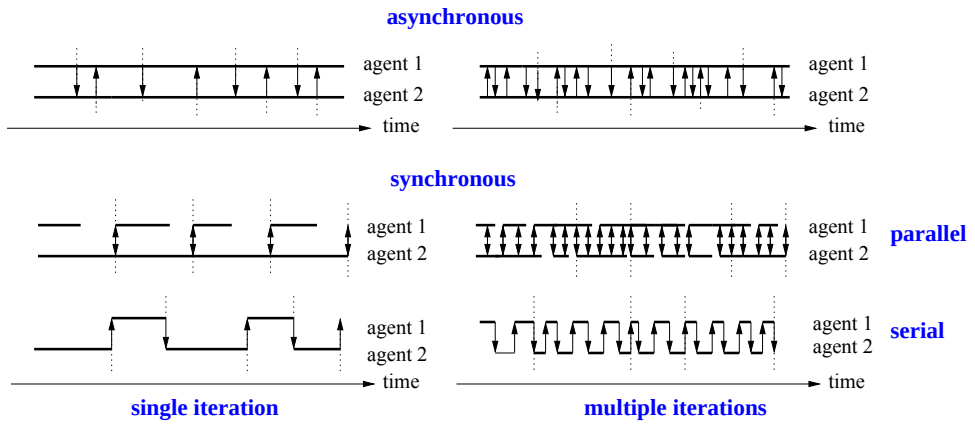


Figure 2.3: Different communication schemes between two agents. Arrows indicate information exchange. Dotted lines indicate actions being implemented. Horizontal lines indicate optimization problems being solved.

2. *Single* or *multiple* iterations, i.e., do agents decide on their actions after sending and receiving information once, or do agents decide on their actions after a number of information exchanges.

3. *Parallel* or *serial*, i.e., are multiple agents performing computations at the same time, or is there only one agent at a time performing its computations.

Asynchronous schemes have as advantage over synchronous schemes that agents do not have to wait for other agents to solve their problems and decide on which actions to take. However, agents will have to include newly received information from neighboring agents at any time while solving their own optimization problems. No multi-agent MPC methods can do this at present.

Single-iteration schemes have as advantage over multiple-iteration schemes that the amount of communication between agents is less, since information is exchanged only once after an agent has solved its problem, and that time required to make a decision is less, since only one iteration is done. Multiple-iteration schemes have as advantage over single-iteration schemes that it is more likely that interconnecting constraints are satisfied at the end of the iterations. In addition, over the iterations agents obtain implicit information about the objectives of their neighbors. Multiple-iteration schemes therefore have a larger potential to achieve overall optimal performance than single-iteration schemes.

Serial schemes have as advantage over parallel schemes that agents use the most up-to-date information from their neighbors. In parallel schemes, the information that is received is usually outdated. However, in serial schemes only one agent is performing computations at a time and therefore decision making is potentially slower than when a parallel scheme is used.

In the literature, several aspects of synchronous single-iteration parallel schemes have been considered, e.g., in [37, 75, 79]. For certain linear time-invariant systems stability can be proved when a so-called contracting stability constraint is placed on the first state of each subsystem [75]. Stability results for settings where the evolution of interconnecting

variables does not depend on neighbors of neighbors are given in [37, 79]. Synchronous multiple-iterations serial and parallel schemes have been considered in [28, 48, 74]. Conditions for convergence of iterations to local solutions and global solutions are given in [28]. A Lagrange-based scheme for the parallel case is employed in [48].

In the following we relax the assumption made in Section 2.2 that the control agent operates individually and knows what the inf uence of the neighboring agents is going to be. We extend the scheme of Section 2.2 to take into account the neighbors through an iterative procedure. The procedure uses as information predictions over the full horizon as obtained from neighboring agents, and employs multiple iterations in a synchronous fashion, aiming for satisfaction of the interconnecting constraints.

2.4 Lagrange-based multi-agent single-layer MPC

For feasible overall solutions, the interconnecting constraints as def ned in (2.13)–(2.14) have to be satisf ed at the moment that control agents decide on which action to take. As discussed above, when one agent solves its optimization problem it has to assume trajectories for the interconnecting variables of its neighboring subnetworks over the horizon. If the neighboring control agents do not respect the assumed trajectories that they communicated, it is unlikely that such a trajectory will appear in the true system evolution. The neighboring control agents will only have an incentive to respect their communicated trajectories if these trajectories yield optimal inputs for their own subsystems.

Even if the agents make an agreement in advance to respect the trajectories communicated, in practice they may not be able to implement this agreement. The reason for this is that at the time of trajectory generation the agents did not know what the values of the interconnecting variables of the other agents will be. Therefore, they may require infeasible inputs to local subsystems to respect the communicated trajectories. To deal with this, a scheme can be used in which the agents perform a number of iterations to come to an agreement on interconnecting variable trajectories that are acceptable to all agents, instead of holding on to the f rst trajectories communicated.

In each iteration each agent optimizes both over its actions and over the predictions of trajectories of neighboring subnetworks. In this way, each agent is sure that the predicted trajectories it assumes are optimal for its own subsystem. After each of the agents has in this way determined its own optimal actions and predicted interconnecting variables trajectory, it communicates the predicted interconnecting variable trajectories to the neighboring agents. This basically means that each agent tells its neighboring agents how it would like to see the interconnecting variables of those agents evolve over the horizon.

Ideally, the interconnecting variable trajectories that those neighboring agents receive will exactly correspond to their predictions of their interconnecting variable trajectories if they would implement their optimal input sequences. However, it is more likely that the received trajectories will not correspond to their predicted trajectories, as discussed before. To encourage the agents to come to an agreement on the predicted interconnecting variable trajectories a penalty term is added to the objective function of each agent. By updating the penalty terms over a series of iterations using the information received from neighboring agents, convergence may be obtained under appropriate assumptions, as we will discuss below.

To derive a scheme that implements these ideas we consider the following steps:

1. Formulate the combined overall control problem, i.e., aggregate the subproblems including the interconnecting constraints;
2. Construct an augmented Lagrange formulation by replacing each interconnecting constraint with an additional linear cost term, based on Lagrange multipliers, and a quadratic penalty term [19, 23];
3. Decompose this formulation again into subproblems for each agent.

We now focus on these steps in more detail.

2.4.1 Combined overall control problem

We def ne the combined overall control problem as the problem formed by the aggregation of the local control problems without assuming that the inf uence from the rest of the network formulated through (2.6) is known, but including the def nition of the interconnecting inputs and outputs (2.9)–(2.10) and the interconnecting constraints (2.13)–(2.14). After def ning:

$$\begin{aligned}\tilde{\mathbf{X}}(k+1) &= [\tilde{\mathbf{x}}_1(k+1)^{\mathrm{T}}, \ldots, \tilde{\mathbf{x}}_n(k+1)^{\mathrm{T}}]^{\mathrm{T}} \\ \tilde{\mathbf{U}}(k) &= [\tilde{\mathbf{u}}_1(k)^{\mathrm{T}}, \ldots, \tilde{\mathbf{u}}_n(k)^{\mathrm{T}}]^{\mathrm{T}} \\ \tilde{\mathbf{Y}}(k) &= [\tilde{\mathbf{y}}_1(k)^{\mathrm{T}}, \ldots, \tilde{\mathbf{y}}_n(k)^{\mathrm{T}}]^{\mathrm{T}},\end{aligned}$$

the control problem at control cycle k is def ned as:

$$\min_{\tilde{\mathbf{X}}(k+1), \tilde{\mathbf{U}}(k), \tilde{\mathbf{Y}}(k)} \sum_{i=1}^{n} J_{\mathrm{local},i}(\tilde{\mathbf{x}}_i(k+1), \tilde{\mathbf{u}}_i(k), \tilde{\mathbf{y}}_i(k)) \tag{2.15}$$

subject to, for $i = 1, \ldots, n$,

$$\tilde{\mathbf{w}}_{\mathrm{in},j_{i,1}i}(k) = \tilde{\mathbf{w}}_{\mathrm{out},ij_{i,1}}(k) \tag{2.16}$$

$$\vdots$$

$$\tilde{\mathbf{w}}_{\mathrm{in},j_{i,m_i}i}(k) = \tilde{\mathbf{w}}_{\mathrm{out},ij_{i,m_i}}(k) \tag{2.17}$$

and the dynamics (2.4)–(2.5) of subnetwork i over the horizon, and the initial constraint (2.7) of subnetwork i. Note that it is suff cient to include in the combined overall control problem formulation only the interconnecting input constraints (2.9) for each agent i, since the interconnecting output constraints (2.10) of agent i will also appear as interconnecting input constraints of its neighboring agents.

2.4.2 Augmented Lagrange formulation

The overall control problem (2.15) is not separable into subproblems using only local variables $\tilde{\mathbf{x}}_i(k+1)$, $\tilde{\mathbf{u}}_i(k)$, $\tilde{\mathbf{y}}_i(k)$ of one agent i alone due to the interconnecting constraints (2.16)–(2.17). In order to deal with the interconnecting constraints, an augmented Lagrange formulation of this problem can be formulated [19, 23]. An augmented Lagrange formulation

combines a penalty formulation with a Lagrange formulation and therefore can provide improved convergence [18]. Using such a formulation, the interconnecting constraints are removed from the constraint set and added to the objective function in the form of additional linear cost terms, based on Lagrange multipliers, and additional quadratic terms. The augmented Lagrange function is def ned as:

$$
\begin{aligned}
&L(\tilde{\mathbf{X}}(k+1),\tilde{\mathbf{U}}(k),\tilde{\mathbf{Y}}(k),\tilde{\mathbf{W}}_{\text{in}}(k),\tilde{\mathbf{W}}_{\text{out}}(k),\tilde{\mathbf{\Lambda}}_{\text{in}}(k))\\
&=\sum_{i=1}^{n}\Bigg(J_{\text{local},i}\left(\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k),\tilde{\mathbf{y}}_i(k),\right)\\
&\qquad+\sum_{j\in\mathcal{N}_i}\left(\tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)\left(\tilde{\mathbf{w}}_{\text{in},ji}(k)-\tilde{\mathbf{w}}_{\text{out},ij}(k)\right)+\frac{\gamma_{\text{c}}}{2}\left\|\tilde{\mathbf{w}}_{\text{in},ji}(k)-\tilde{\mathbf{w}}_{\text{out},ij}(k)\right\|_2^2\right)\Bigg),
\end{aligned}
\tag{2.18}
$$

where

$$
\begin{aligned}
\tilde{\mathbf{W}}_{\text{in}}(k)&=[\tilde{\mathbf{w}}_{\text{in},j_{1,1}1}(k)^{\text{T}},\ldots,\tilde{\mathbf{w}}_{\text{in},j_{n,m_n}n}(k)^{\text{T}}]^{\text{T}}\\
\tilde{\mathbf{W}}_{\text{out}}(k)&=[\tilde{\mathbf{w}}_{\text{out},j_{1,1}1}(k)^{\text{T}},\ldots,\tilde{\mathbf{w}}_{\text{out},j_{n,m_n}n}(k)^{\text{T}}]^{\text{T}}\\
\tilde{\mathbf{\Lambda}}_{\text{in}}(k)&=[\tilde{\boldsymbol{\lambda}}_{\text{in},j_{1,1}1}(k)^{\text{T}},\ldots,\tilde{\boldsymbol{\lambda}}_{\text{in},j_{n,m_n}n}(k)^{\text{T}}]^{\text{T}},
\end{aligned}
$$

and where γ_{c} is a positive constant, and $\tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)$ are the Lagrange multipliers associated with the interconnecting constraints $\tilde{\mathbf{w}}_{\text{in},ji}(k)=\tilde{\mathbf{w}}_{\text{out},ji}(k)$.

By duality theory [19, 23], the resulting optimization problem follows as maximization over the Lagrange multipliers while minimizing over the other variables, i.e.:

$$
\max_{\tilde{\mathbf{\Lambda}}_{\text{in}}(k)}\left\{\min_{\substack{\tilde{\mathbf{X}}(k+1),\tilde{\mathbf{U}}(k),\tilde{\mathbf{Y}}(k),\\ \tilde{\mathbf{W}}_{\text{in}}(k),\tilde{\mathbf{W}}_{\text{out}}(k)}} L\left(\tilde{\mathbf{X}}(k+1),\tilde{\mathbf{U}}(k),\tilde{\mathbf{Y}}(k),\tilde{\mathbf{W}}_{\text{in}}(k),\tilde{\mathbf{W}}_{\text{out}}(k),\tilde{\mathbf{\Lambda}}_{\text{in}}(k)\right)\right\},
\tag{2.19}
$$

subject to the dynamics (2.4)–(2.5) of subnetwork i over the horizon, and the initial constraint (2.7) of subnetwork i, for $i=1,\ldots,n$.

Under convexity assumptions on the objective functions and aff nity of the subnetwork model constraints it can be proved that a minimum of the original problem (2.15) can be found iteratively by repeatedly solving the minimization part of (2.19) for f xed Lagrange multipliers, followed by updating the Lagrange multipliers using the solution of the minimization, until the Lagrange multipliers do not change anymore from one iteration to the next [19]. These convexity assumptions are satisf ed for the linear model (2.1)–(2.2) that we assume, in combination with a linear local objective function, or in combination with a quadratic local objective function as def ned in (2.8). In Section 2.5 we show an example of such a model with a quadratic local objective function.

2.4.3 Distributing the solution approach

The iterations to compute the solution of the combined overall control problem based on the augmented Lagrange formulation (2.18) include quadratic terms and can therefore not directly be distributed over the agents. To deal with this, the non-separable problem (2.18) can be approximated by solving n separated problems, each of which is based on local

dynamics, local objectives $J_{\text{local},i}$, and an additional cost term $J_{\text{inter},i}$. The problem for the control agent controlling subnetwork i is def ned as follows:

$$\begin{aligned}\min_{\substack{\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k),\tilde{\mathbf{y}}_i(k),\\ \tilde{\mathbf{w}}_{\text{in},j_{i,1}i}(k),\ldots,\tilde{\mathbf{w}}_{\text{in},j_{i,m_i}i}(k),\\ \tilde{\mathbf{w}}_{\text{out},j_{i,1}i}(k),\ldots,\tilde{\mathbf{w}}_{\text{out},j_{i,m_i}i}(k)}} \quad & J_{\text{local},i}\left(\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k),\tilde{\mathbf{y}}_i(k)\right)\\ & + \sum_{j\in\mathcal{N}_i} J_{\text{inter},i}\left(\tilde{\mathbf{w}}_{\text{in},ji}(k),\tilde{\mathbf{w}}_{\text{out},ji}(k),\tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)},\tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)}\right),\end{aligned} \tag{2.20}$$

subject to the dynamics (2.4)–(2.5) of subnetwork i over the horizon, and the initial constraint (2.7) of subnetwork i. As we will see below, the structure of the additional cost term $J_{\text{inter},i}$ differs depending on the type of communication scheme used. At iteration s, the variables $\tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)}$ are the Lagrange multipliers computed by agent i for its interconnecting constraints $\tilde{\mathbf{w}}_{\text{in},ji}(k) = \tilde{\mathbf{w}}_{\text{out},ij}(k)$, and the variables $\tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)}$ are the Lagrange multipliers for its interconnecting constraints $\tilde{\mathbf{w}}_{\text{out},ji}(k) = \tilde{\mathbf{w}}_{\text{in},ij}(k)$. The $\tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)}$ variables are received by agent i through communication with agent j, which computed these variables for its interconnecting constraints with respect to agent i. The general multi-agent MPC scheme that results from this comprises at control cycle k the following steps:

1. For $i = 1,\ldots,n$, agent i makes a measurement of the current state of the subnetwork $\bar{\mathbf{x}}_i(k) = \mathbf{x}(k)$ and estimates the expected exogenous inputs $\bar{\mathbf{d}}_i(k+l)$, for $l = 0,\ldots,N-1$.

2. The agents cooperatively solve their control problems in the following iterative way:

   (a) Set the iteration counter s to 1 and initialize the Lagrange multipliers $\tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)}$, $\tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)}$ arbitrarily.

   (b) Either serially or in parallel, for $i = 1,\ldots,n$, agent i determines $\tilde{\mathbf{x}}_i(k+1)^{(s)}$, $\tilde{\mathbf{u}}_i(k)^{(s)}$, $\tilde{\mathbf{w}}_{\text{in},ji}(k)^{(s)}$, $\tilde{\mathbf{w}}_{\text{out},ij}(k)^{(s)}$, for $j \in \mathcal{N}_i$, by solving:

$$\begin{aligned}\min_{\substack{\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k),\tilde{\mathbf{y}}_i(k),\\ \tilde{\mathbf{w}}_{\text{in},j_{i,1}i}(k),\ldots,\tilde{\mathbf{w}}_{\text{in},j_{i,m_i}i}(k),\\ \tilde{\mathbf{w}}_{\text{out},j_{i,1}i}(k),\ldots,\tilde{\mathbf{w}}_{\text{out},j_{i,m_i}i}(k)}} \quad & J_{\text{local},i}\left(\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k),\tilde{\mathbf{y}}_i(k)\right)\\ & + \sum_{j\in\mathcal{N}_i} J_{\text{inter},i}\left(\tilde{\mathbf{w}}_{\text{in},ji}(k),\tilde{\mathbf{w}}_{\text{out},ji}(k),\tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)},\tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)}\right),\end{aligned} \tag{2.21}$$

   subject to the local dynamics (2.4)–(2.5) of subnetwork i over the horizon and the initial constraint (2.7) of subnetwork i.

   (c) Update the Lagrange multipliers,

$$\tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s+1)} = \tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)} + \gamma_{\text{c}}\left(\tilde{\mathbf{w}}_{\text{in},ji}(k)^{(s)} - \tilde{\mathbf{w}}_{\text{out},ij}(k)^{(s)}\right). \tag{2.22}$$

   (d) Move on to the next iteration $s+1$ and repeat steps 2.(b)–2.(c). The iterations stop when the following stopping condition is satisf ed:

$$\left\| \begin{bmatrix} \tilde{\boldsymbol{\lambda}}_{\text{in},j_{1,1}1}(k)^{(s+1)} - \tilde{\boldsymbol{\lambda}}_{\text{in},j_{1,1}1}(k)^{(s)} \\ \vdots \\ \tilde{\boldsymbol{\lambda}}_{\text{in},j_{n,m_n}n}(k)^{(s+1)} - \tilde{\boldsymbol{\lambda}}_{\text{in},j_{n,m_n}n}(k)^{(s)} \end{bmatrix} \right\|_{\infty} \leq \gamma_{\epsilon,\text{term}}, \tag{2.23}$$

where $\gamma_{\epsilon,\text{term}}$ is a small positive scalar and $\|\cdot\|_\infty$ denotes the inf nity norm. Note that satisfaction of this stopping condition can be determined in a distributed way, because each individual component of the inf nity norm depends only on variables of one particular agent [111].

3. The agents implement the actions until the beginning of the next control cycle.

4. The next control cycle is started.

Remark 2.7 The Lagrange multipliers can be initialized arbitrarily; however, initializing them with values close to the optimal Lagrange multipliers will increase the convergence of the decision making process. Therefore, also initializing the Lagrange multipliers with values obtained from the previous decision-making step is benef cial, since typically these Lagrange multipliers will be good initial guesses for the new solution. We refer to this as a *warm start*. □

The schemes proposed in the literature implement step 2.(b) in a parallel fashion, e.g., [28, 41, 48]. In the following we f rst discuss a scheme that implements step 2.(b) in a parallel fashion and then we propose a novel scheme that implements it in a serial fashion. We then assess the performance of both schemes experimentally.

2.4.4 Serial versus parallel schemes

Parallel implementation

The parallel implementation is the result of using the *auxiliary problem principle* [14, 81, 127] of approximating the non-separable quadratic term in the augmented Lagrange formulation of the combined overall control problem. The parallel scheme involves a number of parallel iterations in which all agents perform their local computing step at the same time.

Given for the agents $j \in \mathcal{N}_i$, the previous information $\tilde{\mathbf{w}}_{\text{in,prev},ij}(k) = \tilde{\mathbf{w}}_{\text{in},ij}(k)^{(s-1)}$ and $\tilde{\mathbf{w}}_{\text{out,prev},ji}(k) = \tilde{\mathbf{w}}_{\text{out},ji}(k)^{(s-1)}$ *of the last iteration* $s-1$, agent i solves problem (2.21) using the following additional objective function term for the interconnecting constraints:

$$
\begin{aligned}
&J_{\text{inter},i}\left(\tilde{\mathbf{w}}_{\text{in},ji}(k), \tilde{\mathbf{w}}_{\text{out},ji}(k), \tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)}, \tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)}\right) \\
&= \begin{bmatrix} \tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)} \\ -\tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \tilde{\mathbf{w}}_{\text{in},ji}(k) \\ \tilde{\mathbf{w}}_{\text{out},ji}(k) \end{bmatrix} + \frac{\gamma_{\mathrm{c}}}{2} \left\| \begin{bmatrix} \tilde{\mathbf{w}}_{\text{in,prev},ij}(k) - \tilde{\mathbf{w}}_{\text{out},ji}(k) \\ \tilde{\mathbf{w}}_{\text{out,prev},ij}(k) - \tilde{\mathbf{w}}_{\text{in},ji}(k) \end{bmatrix} \right\|_2^2 \\
&\qquad + \frac{\gamma_{\mathrm{b}} - \gamma_{\mathrm{c}}}{2} \left\| \begin{bmatrix} \tilde{\mathbf{w}}_{\text{in},ji}(k) - \tilde{\mathbf{w}}_{\text{in,prev},ji}(k) \\ \tilde{\mathbf{w}}_{\text{out},ji}(k) - \tilde{\mathbf{w}}_{\text{out,prev},ji}(k) \end{bmatrix} \right\|_2^2 .
\end{aligned}
$$

This scheme uses only information computed during the last iteration $s-1$. The parallel implementation of step 2.(b) of the general multi-agent MPC scheme therefore consists of the following steps at decision step k, iteration s:

2 (b) For all agents $i \in \{1,\ldots,n\}$, *at the same time*, agent i solves the problem (2.21) to determine $\tilde{\mathbf{x}}_i(k+1)^{(s)}, \tilde{\mathbf{u}}_i(k)^{(s)}$, $\tilde{\mathbf{w}}_{\text{in},ji}(k)^{(s)}$, $\tilde{\mathbf{w}}_{\text{out},ji}(k)^{(s)}$, and sends to agent $j \in \mathcal{N}_i$ the computed values $\tilde{\mathbf{w}}_{\text{out},ji}(k)^{(s)}$.

The positive scalar γ_c penalizes the deviation from the interconnecting variable iterates that were computed during the last iteration. This causes that when γ_c is chosen larger, the interconnecting variables that agent i computes at the current iteration will stay close to the interconnecting variables that neighboring agent $j \in \mathcal{N}_i$ computed earlier. With increasing γ_c, it becomes more expensive for an agent to deviate from the interconnecting-variable values computed by the other agents. This results in a faster convergence of the interconnecting variables to values that satisfy the interconnecting constraints. However, it may still take some iterations to obtain optimal values for the local variables. A higher γ_c results in a higher number of iterations before reaching optimality, although the interconnecting constraints will be satisf ed quickly. A lower γ_c results in a lower number of iterations before reaching optimality and interconnecting constraints that are satisf ed. However, when γ_c is chosen too small, a larger number of iterations will again be necessary, since it will take a longer time for the interconnecting constraints to be satisf ed.

As additional parameter this scheme uses a positive scalar γ_b. If $\gamma_b > \gamma_c$, then the term penalizes the deviation between the interconnecting variables of the current iteration and the interconnecting variables of the last iteration of agent i; it thus gives the agent less incentive to change its interconnecting variables from one iteration to the next. When $\gamma_b \geq 2\gamma_c$, and moreover the overall combined problem is convex, it can be proved that the iterations converge toward the overall minimum for suff ciently small $\gamma_{\epsilon,\text{term}}$ [20, 81].

Serial implementation

The novel serial implementation that we propose is the result of using *block coordinate descent* [20, 127] for dealing with the non-separable quadratic term in the augmented Lagrange formulation of the combined overall control problem (2.18). This approach minimizes the quadratic term directly, in a serial way. Contrarily to the parallel implementation, in the serial implementation one agent after another minimizes its local and interconnecting variables while the other variables stay f xed.

Given the information $\tilde{\mathbf{w}}_{\text{in,prev},ij}(k) = \tilde{\mathbf{w}}_{\text{in},ij}(k)^{(s)}, \tilde{\mathbf{w}}_{\text{out,prev},ij}(k) = \tilde{\mathbf{w}}_{\text{out},ij}(k)^{(s)}$ computed at the current iteration s for each agent $j \in \mathcal{N}_i$ that has solved its problem *before* agent i in the *current* iteration s, and given the previous information $\tilde{\mathbf{w}}_{\text{prev},ij}(k) = \tilde{\mathbf{w}}_{ij}^{(s-1)}(k)$ of the *last* iteration $s-1$ for the other agents, agent i solves problem (2.20) using the following additional objective function:

$$J_{\text{inter},i}\left(\tilde{\mathbf{w}}_{\text{in},ji}(k), \tilde{\mathbf{w}}_{\text{out},ji}(k), \tilde{\boldsymbol{\lambda}}_{\text{in},ji}(k)^{(s)}, \tilde{\boldsymbol{\lambda}}_{\text{out},ij}(k)^{(s)}\right) = \begin{bmatrix} \tilde{\boldsymbol{\lambda}}_{\text{in},ji}^{(s)}(k) \\ -\tilde{\boldsymbol{\lambda}}_{\text{out},ij}^{(s)}(k) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \tilde{\mathbf{w}}_{\text{in},ji}(k) \\ \tilde{\mathbf{w}}_{\text{out},ji}(k) \end{bmatrix} + \frac{\gamma_c}{2} \left\| \begin{bmatrix} \tilde{\mathbf{w}}_{\text{in,prev},ij}(k) - \tilde{\mathbf{w}}_{\text{out},ji}(k) \\ \tilde{\mathbf{w}}_{\text{out,prev},ij}(k) - \tilde{\mathbf{w}}_{\text{in},ji}(k) \end{bmatrix} \right\|_2^2.$$

Thus, contrarily to the parallel implementation, the serial implementation uses both information from the current iteration and from the last iteration. The serial implementation implements step 2.(b) of the general scheme as follows at decision step k, iteration s:

(ii) 2 For $i = 1, \ldots, n$, *one agent after another*, agent i determines $\tilde{\mathbf{x}}_i(k+1)^{(s)}$, $\tilde{\mathbf{u}}_i(k)^{(s)}$, $\tilde{\mathbf{w}}_{\text{in},ji}(k)^{(s)}$, $\tilde{\mathbf{w}}_{\text{out},ji}(k)^{(s)}$ by solving (2.21), and sends to each agent $j \in \mathcal{N}_i$ the computed values $\tilde{\mathbf{w}}_{\text{out},ji}(k)^{(s)}$.

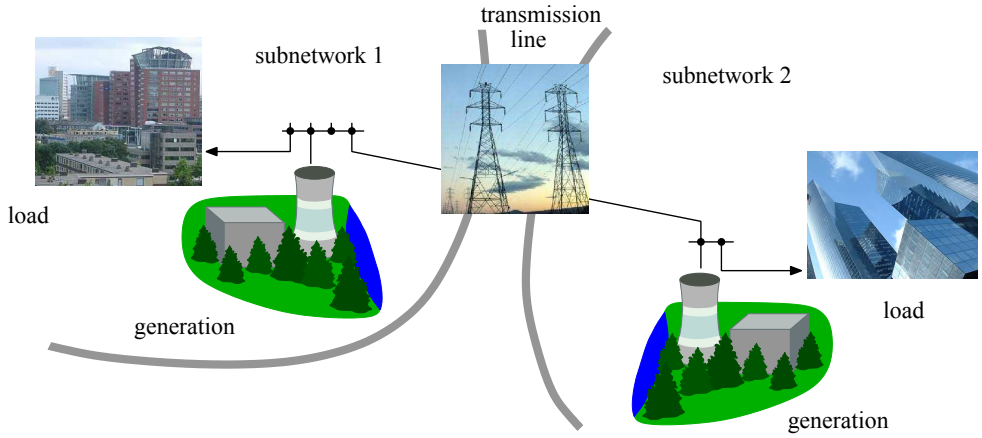


Figure 2.4: Example of two subnetworks, each with loads and power generation facilities. Power flows over the transmission line between the subnetworks. The load-frequency control problem involves adjusting the generation in each subnetwork such that the frequency deviation is maintained close to zero under load disturbances.

The role of the scalar γ_c is similar as for the parallel implementation, except that for the serial implementation γ_c penalizes the deviation from the interconnecting variable iterates that were computed by the agents before agent i in the current iteration and by the other agents during the last iteration. Note that when for the parallel scheme $\gamma_b = \gamma_c$ the additional objective functions are the same for the parallel and the serial scheme, except for the previous information used: the parallel implementation uses only information from the last iteration, the serial also from the current.

In the next section we experimentally assess the performance of the parallel and the serial scheme and discuss which of the two schemes yields a better performance.

2.5 Application: Load-frequency control

In this section we propose the use of the techniques for multi-agent single-layer MPC discussed above for a particular problem in power networks. The problem that we consider is *load-frequency control* [82]. The frequency is one of the main variables characterizing the power network. The purpose of load-frequency control is to keep power generation close to power consumption under consumption disturbances, such that the frequency is maintained close to a nominal frequency of typically 50 or 60 Hz [82]. At an international level power networks become more interconnected and in addition power flows become more unpredictable, e.g., due to large-scale unpredictable power generation using wind turbines. In order to assure correct load-frequency control in the future, current control strategies will be replaced by more advanced strategies that automatically and online determine how the actuators in the network have to be set. Since at an international level countries are not willing to give away access to actuators and sensors in their own subnetworks, they will have to

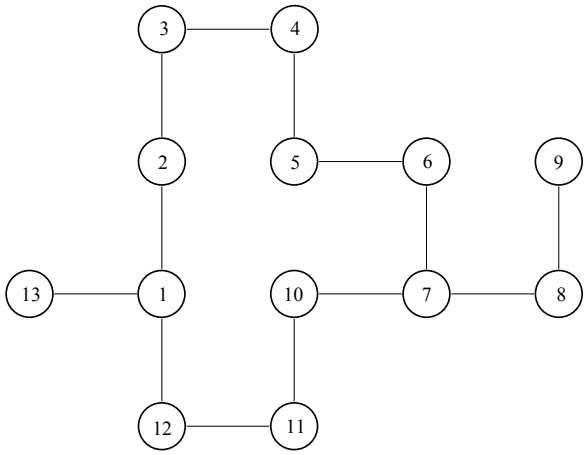


Figure 2.5: The overall network, consisting of 13 subnetworks.

install controllers that cooperatively control the overall network.

A large number of control strategies has been developed for load-frequency control [70]. In the 70s, load-frequency control started being developed with control strategies based on centralized, non-MPC control (see [42, 47, 125]). From the 80s on also, distributed, non-MPC schemes appeared [3, 78, 119, 151, 152]. Recently, also MPC-based schemes have been proposed. A centralized MPC scheme for load-frequency control was proposed in [126]. A decentralized MPC scheme for load-frequency control was proposed in [8]. The latter approach is a decentralized approach that does not take the interconnections between subnetworks explicitly into account. In [28] a distributed MPC scheme is proposed for load-frequency control assuming that only once per control step information between agents can be exchanged. Also in [144] a distributed MPC scheme is applied to a load-frequency control example. The scheme uses distributed state estimation to provide nominal stability and performance properties. We consider distributed MPC using the parallel and serial scheme of Section 2.4.4, which explicitly take into account the interconnections between subnetworks, and use multiple iterations of information exchange before deciding on which actions to take.

2.5.1 Benchmark system

Our benchmark network consists of subnetworks with consumption and generation capabilities, as illustrated in Figure 2.4 for two subnetworks. We consider a network divided into 13 subnetworks as shown in Figure 2.5. Each subnetwork is controlled by one control agent. This control agent has to keep the frequency deviation within its subnetwork close to zero under minimal generation changes. Each control agent can only make measurements and set actuators in its own subnetwork.

We consider rather simplified dynamics for the subnetwork models, that do however include the basic elements of power injection, power consumption, and power flow over

constant	value
$\eta_{\mathrm{K},i}$	120
$\eta_{\mathrm{S},ij}$	0.5
$\eta_{\mathrm{S},ji}$	0.5
$\eta_{\mathrm{T},i}$ (s)	20

Table 2.1: Values of the parameters of the subnetworks, for $i \in \{1,\ldots,n\}$ and $j \in \mathcal{N}_i$.

power lines, and that do show the basic characteristics of the load-frequency control problem. Let the continuous-time linearized dynamics of subnetwork i be described by the following second-order dynamics, as taken from [28]:

$$\begin{aligned}
\frac{\mathrm{d}x_{\Delta\delta,i}}{\mathrm{d}t}(t) &= 2\pi x_{\Delta f,i}(t)\\
\frac{\mathrm{d}x_{\Delta f,i}}{\mathrm{d}t}(t) &= -\frac{1}{\eta_{\mathrm{T},i}}x_{\Delta f,i}(t)+\frac{\eta_{\mathrm{K},i}}{\eta_{\mathrm{T},i}}u_{\Delta P_{\mathrm{gen}},i}(t)-\frac{\eta_{\mathrm{K},i}}{\eta_{\mathrm{T},i}}d_{\Delta P_{\mathrm{dist}},i}(t)\\
&\qquad +\frac{\eta_{\mathrm{K},i}}{\eta_{\mathrm{T},i}}\left(\sum_{j\in\mathcal{N}_i}\frac{\eta_{\mathrm{S},ij}}{2\pi}\left(x_{\Delta\delta,j}(t)-x_{\Delta\delta,i}(t)\right)\right)\\
\mathbf{y}_i(t) &= \begin{bmatrix} x_{\Delta\delta,i}(t)\\ x_{\Delta f,i}(t)\end{bmatrix},
\end{aligned}$$

where at time t, for subnetwork i, $x_{\Delta\delta,i}(t)$ is the incremental phase angle deviation in rad, $x_{\Delta f,i}(t)$ is the incremental frequency deviation in Hz, $u_{\Delta P_{\mathrm{gen}},i}(t)$ is the incremental change in power generation in per unit (p.u.), $d_{\Delta P_{\mathrm{dist}},i}(t)$ is a disturbance in the load in p.u., $\mathbf{y}_i(t)$ are the measurements of the states, and $\eta_{\mathrm{K},i}$ is the subnetwork gain, $\eta_{\mathrm{T},i}$ is the subnetwork time constant in s, $\eta_{\mathrm{S},ij}$ is a synchronizing coeff cient of the line between subnetwork i and j. The values for these constants are given in Table 2.1. Since we assume that the outputs $\mathbf{y}_i(t)$ measure the state variables noise-free, we will without loss of generality leave out the outputs $\mathbf{y}_i(t)$ and only focus on the states $\mathbf{x}_i(t)$ in the following.

Remark 2.8 For subnetwork i the derivative $\frac{\mathrm{d}x_{\Delta f,i}}{\mathrm{d}t}(t)$ depends on $x_{\Delta\delta,j}(t)$, for $j \in \mathcal{N}_i$, which are variables of the subnetworks $j \in \mathcal{N}_i$. The variables $x_{\Delta\delta,j}(t)$ will therefore cause an interconnecting constraint between the control problems of agents i and j. □

Def ning the local control input $u_i(k) = u_{\Delta P_{\mathrm{gen}},i}(k)$, the local exogenous input $d_i(k) = d_{\Delta P_{\mathrm{dist}},i}(k)$, the local states $\mathbf{x}_i(k) = [x_{\Delta\delta,i}(k),\ x_{\Delta f,i}(k)]^{\mathrm{T}}$, the remaining variables $\mathbf{v}_i(k) = [x_{\Delta\delta,j_{i,1}}(k),\ldots,x_{\Delta\delta,j_{i,m_i}}(k)]^{\mathrm{T}}$, and discretizing the continuous-time model using an Euler approximation (with a step size of $T_{\mathrm{p}} = 0.25$ s), the dynamics of subnetwork i can be written as:

$$\mathbf{x}_i(k+1) = \mathbf{A}_i\mathbf{x}_i(k)+\mathbf{B}_{1,i}u_i(k)+\mathbf{B}_{2,i}d_i(k)+\mathbf{B}_{3,i}\mathbf{v}_i(k), \tag{2.24}$$

where

$$\mathbf{A}_i = \begin{bmatrix} 1 & T_{\mathrm{p}}2\pi\\ \sum_{j\in\mathcal{N}_i}\left(T_{\mathrm{p}}\frac{-\eta_{\mathrm{K},i}\eta_{\mathrm{S},ij}}{2\pi\eta_{\mathrm{T},i}}\right) & 1-T_{\mathrm{p}}\frac{1}{\eta_{\mathrm{T},i}}\end{bmatrix} \qquad \mathbf{B}_{1,i} = \begin{bmatrix} 0\\ T_{\mathrm{p}}\frac{\eta_{\mathrm{K},i}}{\eta_{\mathrm{T},i}}\end{bmatrix}$$

$$\mathbf{B}_{2,i} = \begin{bmatrix} 0 \\ -T_\mathrm{p}\frac{\eta_{\mathrm{K},i}}{\eta_{\mathrm{T},i}} \end{bmatrix} \qquad \mathbf{B}_{3,i} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ T_\mathrm{p}\frac{\eta_{\mathrm{K},i}\eta_{\mathrm{S},ij_{i,1}}}{2\pi\eta_{\mathrm{T},i}} & T_\mathrm{p}\frac{\eta_{\mathrm{K},i}\eta_{\mathrm{S},ij_{i,2}}}{2\pi\eta_{\mathrm{T},i}} & \cdots & T_\mathrm{p}\frac{\eta_{\mathrm{K},i}\eta_{\mathrm{S},ij_{i,m_i}}}{2\pi\eta_{\mathrm{T},i}} \end{bmatrix}.$$

2.5.2 Control setup

The agents use the multi-agent single-layer MPC approach as discussed in Section 2.4.3. In order to implement this scheme, the prediction model, the interconnecting variables, the control objectives, and possibly additional constraints have to be specif ed:

- Prediction model. Agent i uses as prediction model M_i (2.24) over the time span from $k+1$ until $k+N$.
- Interconnecting variables. The interconnecting inputs for agent i are def ned as in (2.9), and the interconnecting outputs for agent i are def ned as in (2.10), with:

$$\tilde{\mathbf{K}}_i = \begin{bmatrix} 1 & 0 & & & & & 0 & & \\ \vdots & \vdots & & & & & \vdots & & \\ 1 & 0 & & & & & 0 & & \\ & & \ddots & & & & & \ddots & \\ & & & 1 & 0 & & & 0 & \\ & & & \vdots & \vdots & & & \vdots & \\ & & & 1 & 0 & & & 0 & \end{bmatrix},$$

  such that the interconnecting inputs are $x_{\Delta\delta,j}(k+1+l)$, and the interconnecting outputs are $x_{\Delta\delta,i}(k+1+l)$, for $j \in \mathcal{N}_i$ and $l = 0,\ldots,N-1$.
- Local control objectives. Since agent i has to minimize the frequency deviation and the control input changes in its subnetwork, it uses the following quadratic local objective function:

$$J_{\mathrm{local},i}\left(\tilde{\mathbf{x}}_i(k+1),\tilde{\mathbf{u}}_i(k)\right) = \sum_{l=0}^{N-1} \begin{bmatrix} \mathbf{x}_i(k+1+l) \\ u_i(k+l) \end{bmatrix}^\mathrm{T} \begin{bmatrix} \mathbf{Q}_{i,\mathrm{x}} & 0 \\ 0 & Q_{i,\mathrm{u}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_i(k+1+l) \\ u_i(k+l) \end{bmatrix}$$

  where

$$\mathbf{Q}_{i,\mathrm{x}} = \begin{bmatrix} 0 & 0 \\ 0 & 100 \end{bmatrix}, \quad Q_{i,\mathrm{u}} = 10.$$

  A quadratic function has the advantage that larger deviations are penalized more, and moreover that the objective function is convex.
- Additional constraints. Upper and lower bounds are imposed on the changes in power generation and on the changes in angle and frequency:

$$u_{\min,i} \leq \mathbf{u}_i(k+l) \leq u_{\max,i}$$
$$\mathbf{x}_{i,\min} \leq \mathbf{x}_i(k+1+l) \leq \mathbf{x}_{i,\max},$$

  for $l = 0,\ldots,N-1$, and $u_{\min,i} = -0.3$, $u_{\max,i} = 0.3$, $\mathbf{x}_{i,\min} = [-10,-10]^\mathrm{T}$, $\mathbf{x}_{i,\max} = [10,10]^\mathrm{T}$.

The def ned subnetwork models, interconnecting variables, local control objectives, and additional constraints lead to an overall combined control problem (2.15) that is convex.

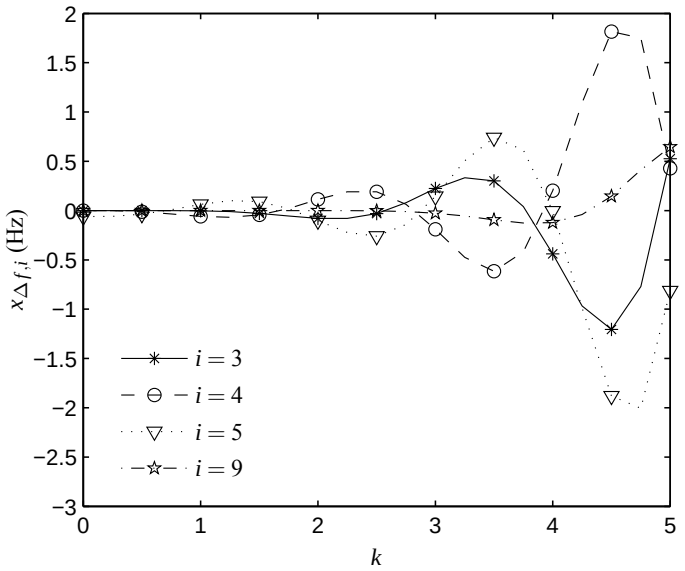


Figure 2.6: Uncontrolled simulation of frequency deviation after a small disturbance in subnetwork 5.

2.5.3 Simulations

We simulate the network in Matlab v7.3 [98]. The network is simulated in discrete time steps of 0.25 s, for $k_{\mathrm{f}} = 20$ steps. Every 0.25 s the control agents measure the state of their subnetwork after which they either employ the serial or the parallel scheme to determine which action to take next. As reference for the performance a hypothetical single agent that uses the overall combined control problem (2.15) is employed. Each of the schemes uses a warm start when possible, i.e., when the solution from a previous control cycle is available. Iterations are stopped when the stopping condition (2.23) is satisfied, or when a maximum number of 5000 iterations has been performed.

The MPC problems solved by the individual control agents at each iteration are quadratic programming problems with linear constraints. These problems are efficiently solved by the ILOG CPLEX v10 Barrier QP solver [71], which we use through the Tomlab v5.7 [66] interface in Matlab v7.3 [98].

To assess the performance of the schemes discussed above, we first illustrate the uncontrolled behavior of the network after a disturbance for a particular scenario, then we consider the performance of the schemes over the full simulation span for a particular setting of the parameters, and then we focus on how the parameters γ_{c} and $\gamma_{\epsilon,\mathrm{term}}$ influence the performance of the schemes at a single control cycle.

Scenario without control

It is easy to verify by inspection of the eigenvalues of the overall network that the network is unstable when no control is employed. To illustrate this instability, we first consider the

following uncontrolled scenario. The subnetworks start in steady state, i.e., $\mathbf{x}_i(0) = [0,0]^{\mathrm{T}}$, for $i = \{1,\ldots,n\}$). A disturbance $d_{\Delta P_{\mathrm{dist}},i}(k) = 1.10^{-2}$ is imposed at $k = -1$ in subnetwork 5.

Figure 2.6 shows the evolution of the frequency deviation right after the disturbance has appeared, i.e., starting from $k = 0$, in a number of representative subnetworks when no control actions are taken. Clearly, without control agents acting on the generation, the dynamics of the network directly after the disturbance become unstable, and the magnitudes of the oscillations of the frequency deviations increase quickly after the fault.

Performance of control over the full simulation span

We now consider the performance that the parallel and serial schemes discussed in this chapter can achieve for particular values of the control parameters. We compare the performance of the serial and parallel scheme with each other and with a hypothetical centralized control agent that solves the overall combined MPC problem.

We consider 50 scenarios in which a randomly chosen disturbance $d_{\Delta P_{\mathrm{dist}},i}$ from the domain $[-1.10^{-2}, 1.10^{-2}]$ appears in a randomly chosen subnetwork $i \in \{1,\ldots,13\}$. In each scenario, we let time step $k = 0$ correspond to the time step right after the disturbance has appeared. Hence, we consider the performance of the control agents with respect to dealing with the consequences of the disturbance.

To compare the performance of the schemes over the full simulation period, costs are computed over the full simulation as:

$$J_{\mathrm{sim}} = \sum_{i=1}^{n} \sum_{l=0}^{k_{\mathrm{f}}-1} J_{\mathrm{stage},i}\left(\bar{\mathbf{x}}_i(1+l), \bar{\mathbf{u}}_i(l), \bar{\mathbf{y}}_i(l)\right),$$

where the bar indicates that the value of the variable is the actual value as appearing in the evolution of the network, and not the predicted value as predicted by a control agent during its optimization. E.g., $\bar{\mathbf{x}}_i(k)$ refers to the actual state of subnetwork i at time k, and not to the state predicted by a control agent. No penalty term is included for violation of the upper or lower bounds on the variables.

As parameters we here consider as specif c setting for the length of the prediction horizon $N = 5$, and for the values of the parameters of the schemes $\gamma_{\mathrm{c}} = 1$, $\gamma_{\epsilon,\mathrm{term}} = 1e^{-4}$, and $\gamma_{\mathrm{b}} = 2\gamma_{\mathrm{c}}$, which for overall convex problems guarantees convergence toward an overall optimal solution. Below we will further discuss the inf uence of different values of the parameters on the performance of the control.

Table 2.2 shows over all scenarios the average results of the schemes, consisting of the average performance $J_{\mathrm{sim,avg}}$, the average number of iterations required $N_{\mathrm{iter,avg}}$, and the total computation time in seconds[2]. We observe that the average performance $J_{\mathrm{sim,avg}}$ that is obtained over a full simulation by the serial and the parallel scheme are very close to each other. In addition the performance of these multi-agent schemes is very close to

[2]For computing the total computation time required for the parallel and the serial scheme, only the time spent on solving the optimization problems is summed, since the time involved in setting up the optimization problems is negligible. The simulations are implemented in a central simulation environment. Hence, the parallel scheme is in fact executed in a serial fashion. Therefore, the computation time of a single iteration is taken as the maximum computation time required for solving either of the local optimization problems. Since the simulations are executed in a central simulation environment also no communication delays are accounted for.

scheme	$J_{\text{sim,avg}}$	$N_{\text{iter,avg}}$	$T_{\text{comp,avg}}$
centralized	0.2746	-	0.3
serial	0.2746	129	16.6
parallel	0.2746	335	5.9

Table 2.2: Results of the schemes over all experiments. The table shows over all 50 scenarios the average performance $J_{sim,avg}$, the average number of iterations $N_{iter,avg}$, and the average total computation time $T_{comp,avg}$ (s). The results have been obtained for parameter settings $N = 5$, $k_f = 20$, $\gamma_c = 1$, $\gamma_{\epsilon,term} = 1.10^{-4}$, and starting from 50 different initial states, each of which are a state appearing right after a random disturbance between -0.01 and 0.01 p.u. in one of the subnetworks has occurred.

the performance of the centralized scheme. Hence, the controls agents have obtained the performance of the centralized control agent in a distributed way.

We also observe from Table 2.2 that the serial scheme on average requires fewer iterations $N_{\text{iter,avg}}$ per simulation than the parallel scheme. This can be explained by the fact that the serial scheme uses information from both the previous and the current iteration, whereas the parallel scheme only uses information from a previous iteration.

In Table 2.2 we also observe that the total computation time in seconds per simulation on average $T_{\text{comp,avg}}$ is larger for the serial scheme than for the parallel scheme. This is explained by the fact that in the serial scheme only one agent at a time performs a computation step within an iteration, whereas in the parallel scheme multiple control agents perform computations at the same time. Compared to the centralized scheme, the parallel and serial scheme have a larger total computation time than the centralized scheme.

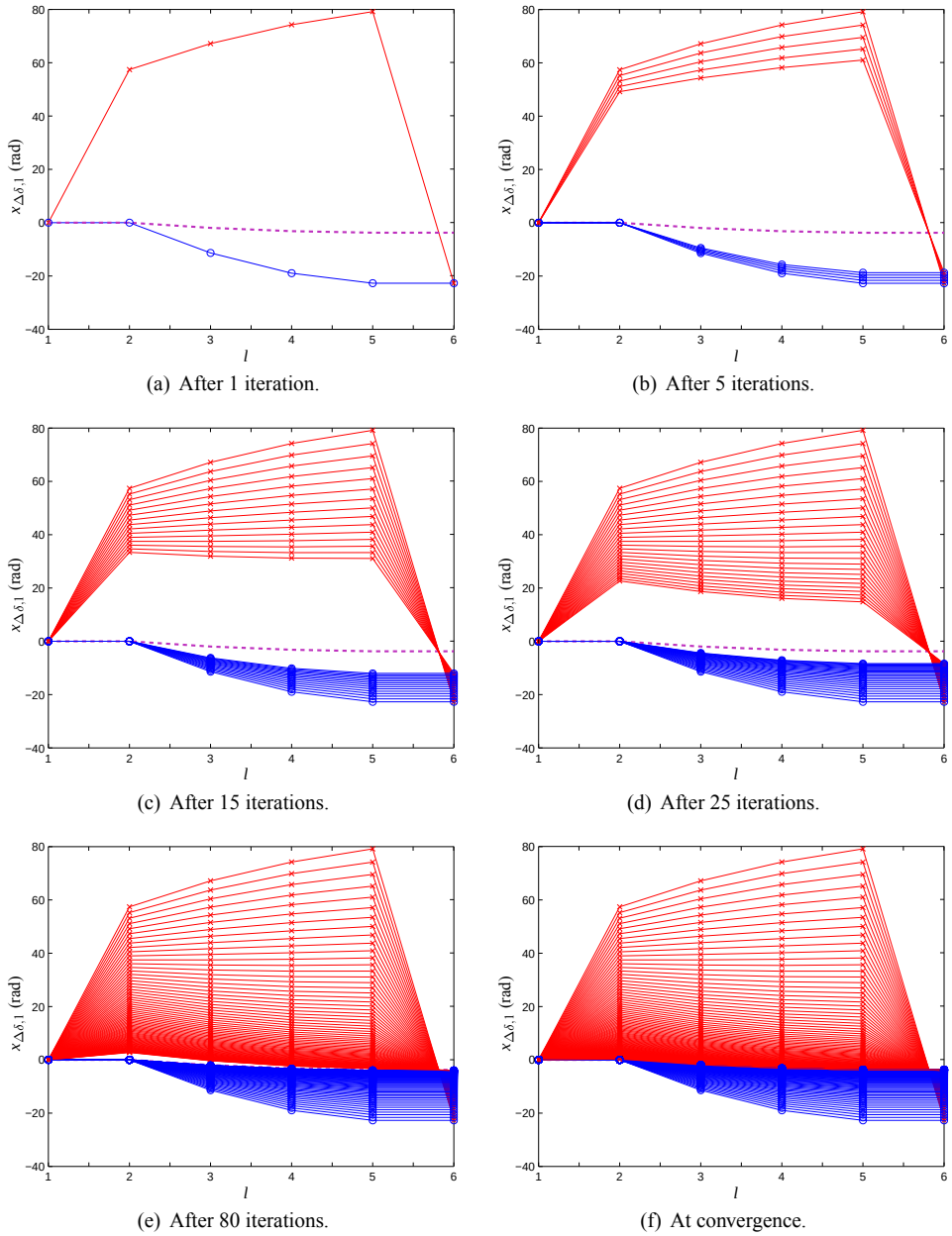
Below we will discuss these results further, after illustrating the inf uence of different parameter values on the performance of the parallel and serial scheme.

Iterations at a single control cycle

To illustrate the operation of the serial scheme at a particular control cycle, consider Figure 2.7. The f gure illustrates the typical behavior of values of interconnecting variables going toward each other over the iterations at a particular control cycle for a network consisting of two subnetworks. In this network, the values of $\mathbf{x}_i$ are unconstrained.

The f gure illustrates for a particular interconnecting input variable of agent 1 over the prediction horizon and the corresponding interconnecting output variable of agent 2 over the prediction horizon, the values that both agents would like their interconnecting variable to take on. After each local computation step, these values are communicated to the other agent, which uses these to update its interconnecting objective function. As the iterations progress the values of the interconnecting input and the corresponding interconnecting output converge to each other, indicating that the values go toward satisfying interconnecting constraints. In addition, since in our case the combined overall problem is convex, the values converge to the solution that would have been obtained with a centralized control agent that would have access to all actuators and sensors in the network.

Depending on the value of the parameter $\gamma_{\epsilon,\text{term}}$ the iterations will terminate sooner or



(a) After 1 iteration.

(b) After 5 iterations.

(c) After 15 iterations.

(d) After 25 iterations.

(e) After 80 iterations.

(f) At convergence.

Figure 2.7: Convergence of the values for interconnecting input variables of agent 1 (solid line with circle) and the corresponding interconnecting output variables of agent 2 (solid line with cross), each corresponding to the variables $x_{\Delta\delta,1}(k+l)$ over a prediction horizon of 6 steps, hence, for $l = 1,2,\ldots,6$. Over the iterations the values converge to the overall optimal solution (dashed line).

later, and depending on the value of the parameter γ_c the values of the interconnecting variables will converge sooner or later to values for which the interconnecting constraints are satisf ed. Below we go into this in more detail.

Parameter sensitivity at a single control cycle

To gain more insight into the role of the parameters γ_c and $\gamma_{\epsilon,\text{term}}$ and into the iterations that the serial and the parallel scheme perform, we illustrate the performance of the schemes for a particular representative control problem at a particular control cycle under varying parameter values. The control problem that we consider is the MPC control problem that the agents have to solve right after a disturbance $d_{\Delta P_{\text{dist}},i}$ of magnitude 1.10^{-2} has occurred in subnetwork 5.

To evaluate the solution over the prediction horizon determined by the different schemes at a single control cycle, the inputs coming from the different schemes are implemented to determine the resulting state trajectory, after which the cycle performance J_{cycle} is determined as:

$$J_{\text{cycle}} = \sum_{i=1}^{n} \sum_{l=0}^{N-1} J_{\text{stage},i}\left(\bar{\mathbf{x}}_i(1+l), \mathbf{u}_i(l), \bar{\mathbf{y}}_i(l)\right).$$

No penalty term is included for violations of the bound constraints.

Varying the penalty coefficient We f rst vary the parameter γ_c, while keeping $\gamma_{\epsilon,\text{term}}$ f xed at 1.10^{-6}. For varying values of the parameter γ_c we determine the cycle performance J_{cycle} at each intermediate iteration. Hence, after each iteration, the actions that the control agents would then choose are used to evaluate the cycle performance J_{cycle} over the prediction horizon.

Figures 2.8 and 2.9 illustrate how the cycle performance J_{cycle} of the control agents using the serial and the parallel scheme changes over the iterations, under various values for γ_c. We clearly observe that as the number of iterations increases, the performance of the solution that the control agents have determined increases as well.

We observe in Figure 2.8 that, indeed, on the one hand for very small values of the penalty term γ_c, the convergence is slow, whereas on the other hand, for larger values of the penalty term γ_c, the convergence is faster. However, we observe in Figure 2.9 that, indeed, when the penalty term γ_c is chosen too large, the convergence slows down again.

For a given value of γ_c, the serial scheme requires fewer iterations and converges faster than the parallel scheme. This behavior is best observed for larger values of γ_c in Figure 2.9. The difference in the number of iterations required is due to the fact that the serial scheme uses information earlier than the parallel scheme. For smaller values of γ_c, as those shown in Figure 2.8, the inf uence of the additional objective function $J_{\text{inter},i}$ of both the parallel and the serial scheme vanishes, making that the difference between the two schemes vanishes as well.

Varying the stopping tolerance Given a value for γ_c we determine the cycle performance J_{cycle} that the control agents obtain at termination using various values for the stopping tolerance $\gamma_{\epsilon,\text{term}}$. We vary $\gamma_{\epsilon,\text{term}}$ in the set $\{1.10^{-8}, 1.10^{-7}, \ldots, 1, 10\}$.

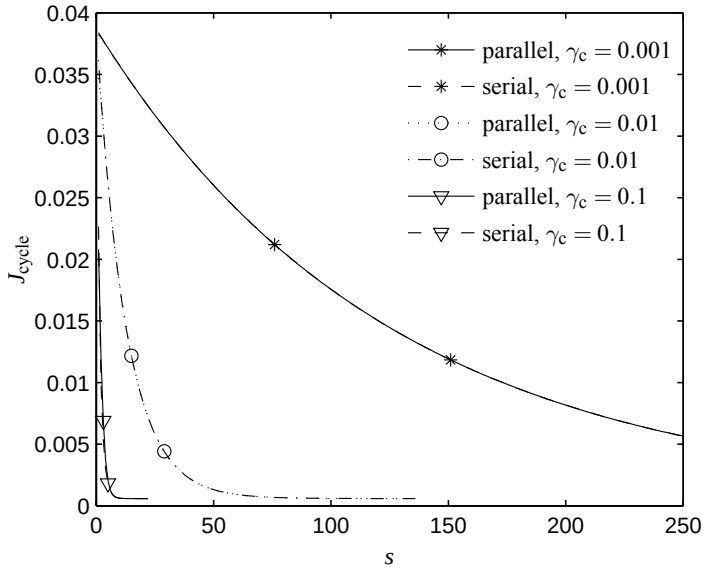


Figure 2.8: The performance of solutions after each iteration for smaller values of γ_c.

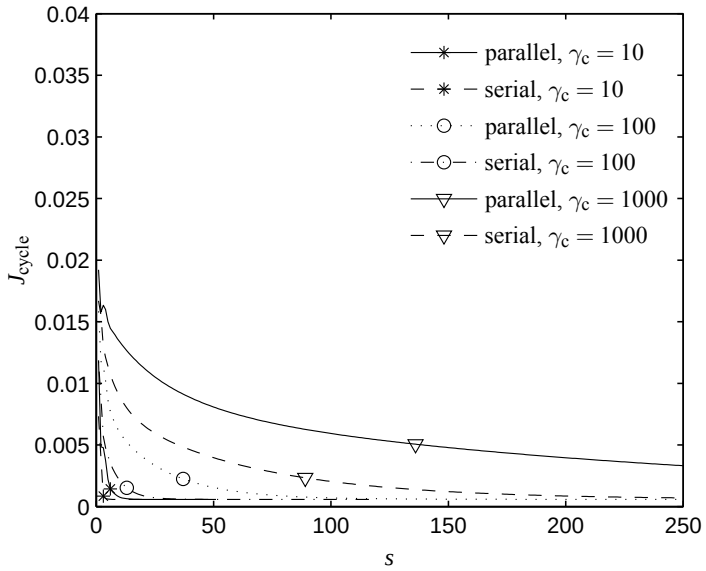


Figure 2.9: The performance of solutions after each iteration, for larger values of γ_c.

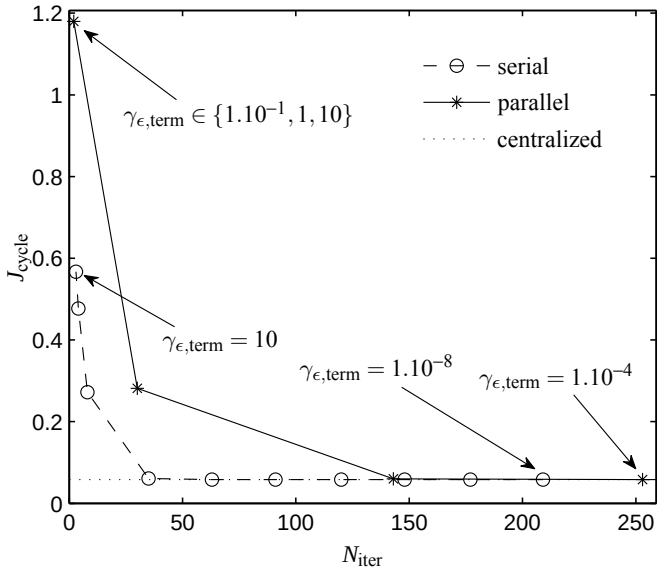


Figure 2.10: The cycle performance versus the number of iterations for $N = 5$, $\gamma_c = 100$, and varying $\gamma_{\epsilon,term}$. Each circle and star represent a scenario with a different value for $\gamma_{\epsilon,term}$. The value of $\gamma_{\epsilon,term}$ decreases when going from left to right.

Figure 2.10 shows the results for varying $\gamma_{\epsilon,\text{term}}$, while keeping γ_c fixed at 100. We observe that with a decreasing value of the stopping tolerance $\gamma_{\epsilon,\text{term}}$, more iterations are required before the stopping condition is satisfied. We also observe that if an appropriate value for $\gamma_{\epsilon,\text{term}}$ is chosen, convergence toward the centralized solution is obtained within a reasonable bound.

It is noted that there is minimal performance that is achieved when $\gamma_{\epsilon,\text{term}}$ becomes larger than a certain value. In Figure 2.10 this is observed for the parallel scheme, which for values of $\gamma_{\epsilon,\text{term}}$ larger than 0.1 achieves the same performance.

When comparing the serial scheme with the parallel scheme, we observe that the serial scheme outperforms the parallel scheme in convergence speed and performance. Furthermore, Figure 2.10 illustrates that over the iterations the performance of both schemes converges toward the performance of the centralized overall scheme.

Discussion

The experiments reported in this section represent a relatively small portion of all experiments that could have been done, involving multiple combinations of network topologies, scheme parameters, prediction horizons, etc. Nonetheless, the results obtained here give an indication of the potential of the approaches discussed in this chapter.

It is noted that both schemes discussed only communicate information common to the control problems of several agents; all other data is only used locally. Agents have only a prediction model of their own subnetwork. This gives flexibility and security, since other

agents do not have to know the exact parameters of a particular subnetwork, and in fact the subnetwork may be changed, without having to inform other agents.

The time required to complete the iterations at one control cycle in these experiments is typically larger than a real-time online implementation would allow. However, as Figures 2.8 and 2.9 illustrate, after already a few iterations a relatively good solution can have been obtained, and thus if necessary the iterations could be stopped earlier.

In our experiments we have seen that the serial scheme can outperform the parallel scheme in terms of convergence speed in terms of iterations and the performance obtained. However, we also observed that the serial scheme requires more computation time in seconds in order to perform its computations, when compared to the parallel scheme.

If the time required for one serial iteration is reduced, the serial scheme may also outperform the parallel scheme in total computation time required. Our idea to achieve this is to parallelize the serial scheme, either only within an iteration, or also over iterations. Parallelization can be done when the topology of subnetworks can be seen as a tree. This tree structure of the network makes that control problems of control agents can be solved (partially) in parallel, thus reducing total computation time. Groups of agents operating in parallel may be constructed. Within each group, the serial scheme may be employed [111].

It should be noted that the overall network that the control agents control in this section is highly unstable. As we have seen, a small disturbance in the overall network gives large oscillations if not controlled properly. For this reason, it is important for the control agents to obtain very accurate values of the interconnecting variables over the prediction horizon. For applications in which the local subnetwork dynamics and objectives do not depend as much on the values of the interconnecting variables decision making speed can be increased by lowering the value of the stopping tolerance $\gamma_{\epsilon,\text{term}}$.

The dynamics used in this section for representing the power networks dynamics are highly simplified, and the values representing the deviations therefore can also not directly be related to physical values. The linear dynamics assumed are typically valid only over small prediction horizons. However, for our purpose of showing the performance of the control schemes, this is not an issue. More advanced linear models may be used in combination with the schemes considered above to more adequately represent the actual network physics.

2.6 Summary

In this chapter we have considered multi-agent single-layer MPC for the control of transportation networks. We have started with formalizing the dynamics of the subnetworks and the control structure. Then, we have formulated the MPC problem for an individual control agent, assuming that it knows how its surrounding network behaves. Subsequently, we have relaxed this assumption and introduced interconnections between control problems. We have surveyed how these interconnections can be dealt with by discussing the various ways of information exchange and moments at which information exchange takes place. Then, we have focused on a particular type of schemes and have proposed a novel serial scheme, which we have compared with a related parallel scheme. Although under convexity assumptions on the overall combined control problem the schemes converge to overall optimal solutions, it remains to be investigated what the rate of convergence is, how the rate

of convergence can be improved, and how this scheme can be extended to other classes of models.

We have proposed the application of the schemes for a load-frequency control problem. Through experimental studies on a network consisting of 13 subnetworks, we have compared the serial scheme with the parallel and a centralized overall scheme. For the serial and the parallel schemes, the performance of the solution obtained converged toward the performance of the solution obtained by the overall control problem, provided that the overall control problem is convex.

The results of the experiments illustrate that the proposed serial scheme generally has preferable properties in terms of the solution quality and the number of iterations required. However, the parallel scheme requires less time. Through parallelization the total computation time required per iteration by the serial scheme may be made more eff cient, ultimately resulting in a scheme that requires also fewer total computation time than the parallel scheme.

In Chapter 3 we extend the serial method to situations in which the problem of controlling the transportation network cannot be formulated as a convex problem. In particular we extend the method to deal with networks modeled as hybrid systems in which both continuous and discrete dynamics appear, a situation typically appearing when, e.g., continuous f ows together with discrete actions are present.

In Chapter 4 we discuss how a supervisory control layer can control the control agents of a lower control layer, that are organized as, e.g., the structure considered in this chapter. The supervisory control layer takes into account the dynamics of both the lower control layer and the underlying physical network.

In Chapter 5 we consider how an even higher supervisory control layer can control the control agents in a lower control layer. The control agents in the higher control layer do not take into accounts the dynamics of the lower layer, but only consider steady-state characteristics. A scheme related to the schemes addressed in this chapter is used to obtain coordination among the control agents controlling subnetworks that are overlapping and may have nonlinear steady-state characteristics.

Chapter 3

Networked hybrid systems

In Chapter 2 we have considered multi-agent control of transportation networks involving only continuous variables and dynamics. In this chapter we consider multi-agent control of hybrid systems, i.e., distributed control of systems with both continuous and discrete dynamics. In Section 3.1 we introduce hybrid systems, illustrate how transportation networks can be seen as hybrid systems, and discuss which issues have to be dealt with when developing multi-agent single-layer MPC approaches for such systems. In Section 3.2 we focus on formulating prediction models of hybrid systems and discuss how transformations can be used to recast descriptions of hybrid systems into systems of linear mixed-integer constraints. In Section 3.3 we then apply these transformations to construct a model of a particular hybrid system. In Section 3.4 we focus on multi-agent control of interconnected hybrid systems and propose an extension of the serial multi-agent single-layer MPC scheme of Chapter 2.

In this chapter we apply the discussed techniques to two applications. In Section 3.3 we consider a decentralized multi-agent single-layer MPC approach for optimization of energy consumption in households. In Section 3.5 we propose an extension of the serial multi-agent approach of Chapter 2 for load-frequency control with discrete generation switching.

Parts of this chapter have been published in [68, 108].

3.1 Transportation networks as hybrid systems

Many of the transportation networks of our interest can be seen as *hybrid systems*. Hybrid systems [4, 104, 143] arise when continuous dynamics are combined with discrete dynamics. The following examples show how particular transportation networks can be seen as hybrid systems:

- In power networks, the transients and the evolution of the voltage and power levels and the demands of generators and users yield continuous dynamics, whereas the activation or deactivation of generators, lines, or users corresponds to discrete dynamics.
- In road traffic networks the flow of the cars through the network can be modeled with continuous dynamics, and elements such as ramp metering, traffic signals, lane closures, route directions, etc., yield discrete dynamics on the system.

- In water networks the evolution of the water levels can be modeled with continuous dynamics, whereas opening and closing of dams, and activating or deactivating of pumps yield discrete dynamics.

More generally speaking, hybrid dynamics are the result of the discrete dynamics caused by, e.g., saturation effects, discrete switching of actuators, discrete controller logic, priorities on control, reaching of physical bounds, etc., in combination with the continuous dynamics of, e.g., flows, pressures, speeds, levels, etc.

Conventional control approaches usually either consider only continuous or only discrete dynamics. The control approaches that do consider discrete and continuous dynamics simultaneously are mostly based on a centralized control paradigm, since multi-agent control has mostly been approached either from a computer science point of view, which focuses on discrete dynamics, or from a control engineering point of view, which focuses on continuous dynamics. Structured control design methods for large-scale hybrid systems are therefore lacking.

In a multi-agent single-layer MPC control structure the network is divided into n subnetworks, each controlled by a single control agent, cf. Section 1.3.2. Each of the control agents uses MPC to determine which actions to take. Each agent hereby uses a prediction model to predict the evolution of its subnetwork under various actuator settings over a certain prediction horizon. For transportation networks that are hybrid systems, all or some of the subnetworks will be hybrid systems. Issues that we address in the following sections are related to:

- Formalizing the hybrid behavior into suitable mathematical models. The control agents have to use prediction models that on the one hand adequately represent the hybrid dynamics, while on the other hand give MPC problems that can be solved efficiently, e.g., by making it possible to use state-of-the-art commercially available optimization problem solvers.

- Making control agents choose local actions that give performance that is as close as possible to overall optimal network performance, when the subnetworks of the control agents are hybrid systems. When the subnetwork that a control agent controls is a hybrid system, the corresponding prediction model will typically contain both continuous and discrete variables. This has as consequence that the MPC optimization problem of a particular control agent will be nonconvex, and that therefore also the overall combined control problem defined in Section 2.4 will be nonconvex. Approaches as discussed in Section 2.4 for coordinating control agents may not give satisfactory performance, and a way has to be found to improve this.

We first focus on the first issue, i.e., modeling of hybrid systems, by discussing how transformations can be used to transform discrete logic into mixed-integer equality and inequality constraints. We then employ these transformations for designing a prediction model used by a decentralized multi-agent single-layer MPC control structure to control household energy consumption. Next, we consider the second issue, i.e., multi-agent control of interconnected hybrid systems, by extending the serial approach of Section 2.4 to deal with hybrid subnetworks. The approach is experimentally assessed on a load-frequency control problem in which generation can be changed in discrete quantities.

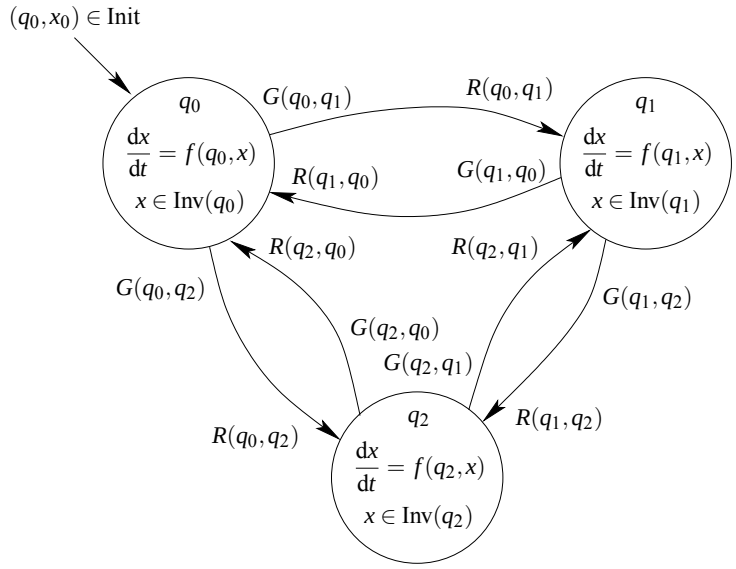


Figure 3.1: Schematic representation of a hybrid automaton.

3.2 Modeling of hybrid systems

There are many ways in which models of hybrid systems can be constructed. Typically the continuous dynamics are represented by systems of differential or difference equations and the discrete dynamics are represented by automata or finite state machines [29, 30]. Combining these two types of models results in hybrid automata [143], a type of model that can represent a large class of hybrid systems.

Figure 3.1 depicts an example of a such a hybrid automaton. Each node represents a *mode* of the system. The modes represent the discrete operating points, i.e., q_0, q_1, and q_2. In this case, each mode is governed by its own continuous dynamics, given by a system of differential equations, e.g., $\frac{dx}{dt} = f(q_1, x)$ for mode q_1. The system can stay in a particular mode as long as the continuous state stays inside the invariant set of that mode, e.g., $x \in \text{Inv}(q_1)$. The system can also transition to a different mode, and in fact has to transition to a new mode if the continuous state x is no longer inside the invariant set. The system can only transition from one mode to another, if the transition between these modes is enabled. The guard set $\mathcal{G}$ indicates for which states x the transition from one mode to another is enabled. The reset set $\mathcal{R}$ indicates which values the states can take on when a transition is made to a new mode. If each sequence of continuous state and mode transitions is uniquely determined only by the initial continuous state and mode, then the hybrid automaton is deterministic. Otherwise, the hybrid automaton is non-deterministic.

Hybrid automata have a large expressibility, in the sense that they can in principle represent the dynamics of any hybrid system. However, this expressibility comes at the price of increased difficulties for analytical studies, simulation, etc. By making assumptions on the possible mode transitions, the dynamics inside the modes, and the guard and the reset sets, different types of models can be defined. Each of these types of models will have different

characteristics when it comes to the easiness of performing time-domain simulations, the possibility for analytical analysis, and the range of hybrid systems that can be represented. Some types of models that can be considered as special cases of hybrid automata are timed Petri-nets [34], mixed-logical dynamic models [16], piecewise-affine models [133], max-min-plus scaling models [35], etc. The equivalence of some of these types of models is shown in [57].

3.2.1 Models for MPC control

In the description of dynamics of hybrid systems discrete logic statements are commonly encountered, e.g., in the form of if-then or if-then-else rules. For a deterministic hybrid automaton an example of a discrete logic statements is "if $x \notin \text{Inv}(q_1)$ and $x \in G(q_1, q_2)$, then $q = q_2$ and $x \in R(q_1, q_2)$". This statement means that if continuous state x is not in the invariant set of q_1 anymore and x is in the guard set G guarding the transition from q_1 to q_2, that then the transition to mode q_2 is made, and the continuous state obtains a value from the reset set associated with that transition. Discrete logic can be dealt with when formulating the prediction model of a control agent in the following ways:

- Software can be used that simulates the system, including the discrete logic. This software accepts a starting state and a series of inputs, and delivers an ending state and a series of outputs. Hence, the software is the prediction model of the system. The control agent can include this prediction model using nonlinear constraints in its MPC optimization problem. It can then use nonlinear optimization techniques to solve the nonlinear MPC optimization problem.

- The discrete logic can be transformed into linear equality and inequality constraints. The prediction model of the system will then consist of a system of linear equality and inequality constraints, in the case that the dynamics given fixed discrete dynamics are linear. The control agent can include this prediction model using mixed-integer linear constraints in its MPC optimization problem. It can then use mixed-integer linear or quadratic programming techniques to solve the MPC optimization problem.

In Chapter 4 we discuss the first approach. Below, we discuss the second approach, first from a more theoretical point of view in Section 3.2.2, then from a more applied point of view in Section 3.3.

3.2.2 From discrete logic to linear mixed-integer constraints

In [16, 149] it is shown how discrete logic statements can be transformed into linear mixed-integer equality and inequality constraints, i.e., constraints involving both variables that take on values from a continuous set of values, and variables that take on values from a discrete set of values. As in [16], we denote by $\mathbf{x} \in \mathbb{R}^n$ continuous variables and by $\delta \in \{0,1\}$ a binary logical variable. In addition, we denote by [exp] a logic statement, which has as value the evaluation of an expression exp to true or false. So, $[f(\mathbf{x}) \leq 0]$ evaluates to true when $f(\mathbf{x}) \leq 0$, and to false otherwise.

It would be convenient if these logic statements could be transformed into linear mixed-integer constraints, since optimization problem solvers that know how to deal with these

constraints are available. Some useful transformations from logic statements into linear mixed-integer inequality constraints are given by [16]:

$$[f(\mathbf{x}) \leq 0] \wedge [\delta = 1] \text{ is true iff } f(\mathbf{x}) - \delta \leq -1 + \gamma_{\mathrm{m}}(1-\delta) \tag{3.1}$$

$$[f(\mathbf{x}) \leq 0] \vee [\delta = 1] \text{ is true iff } f(\mathbf{x}) \leq \gamma_{\mathrm{M}}\delta \tag{3.2}$$

$$\sim [f(\mathbf{x}) \leq 0] \text{ is true iff } f(\mathbf{x}) \geq \gamma_{\epsilon,\mathrm{mach}} \tag{3.3}$$

$$[f(\mathbf{x}) \leq 0] \Rightarrow [\delta = 1] \text{ is true iff } f(\mathbf{x}) \geq \gamma_{\epsilon,\mathrm{mach}} + (\gamma_{\mathrm{m}} - \gamma_{\epsilon,\mathrm{mach}})\delta \tag{3.4}$$

$$[f(\mathbf{x}) \leq 0] \Leftrightarrow [\delta = 1] \text{ is true iff } \begin{cases} f(\mathbf{x}) \leq \gamma_{\mathrm{M}}(1-\delta) \\ f(\mathbf{x}) \geq \gamma_{\epsilon,\mathrm{mach}} + (\gamma_{\mathrm{m}} - \gamma_{\epsilon,\mathrm{mach}})\delta, \end{cases} \tag{3.5}$$

where $f : \mathbb{R}^{n_{\mathbf{x}}} \to \mathbb{R}$ is linear, $\mathbf{x} \in \mathcal{X}$, $\mathcal{X}$ is a given bounded set, $\gamma_{\epsilon,\mathrm{mach}}$ is a small positive constant, e.g., the machine precision, which indicates when a constraint is considered to be violated, and where

$$\gamma_{\mathrm{M}} = \max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \tag{3.6}$$

$$\gamma_{\mathrm{m}} = \min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}). \tag{3.7}$$

Remark 3.1 Formally $\sim [f(\mathbf{x}) \leq 0]$ is true iff $f(\mathbf{x}) > 0$. However, for numerical reasons optimization problem solvers cannot deal with such a strict inequality. Therefore in (3.3) the strict inequality $f(\mathbf{x}) > 0$ is approximated by the inequality $f(\mathbf{x}) \geq \gamma_{\epsilon,\mathrm{mach}}$. In practice, for a suff ciently small value of $\gamma_{\epsilon,\mathrm{mach}}$ this approximation is typically acceptable. □

As we will see in Section 3.3, as a byproduct of transforming logic statements into mixed-integer constraints, constraints involving products of logical variables and constraints involving products of continuous and logical variables may appear. Although these products are not linear, they can be transformed into linear inequalities. E.g., the product term $\delta_1\delta_2$ can be replaced by an auxiliary binary variable δ_3. The value of variable δ_3 should be 1, when the values of both δ_1 and δ_2 are 1, and 0 otherwise. This behavior can be expressed in a logic statement and corresponding linear inequalities as follows [16]:

$$[\delta_3 = 1] \Leftrightarrow ([\delta_1 = 1] \wedge [\delta_2 = 1]) \text{ is true iff } \begin{cases} -\delta_1 + \delta_3 \leq 0 \\ -\delta_2 + \delta_3 \leq 0 \\ \delta_1 + \delta_2 - \delta_3 \leq 1. \end{cases} \tag{3.8}$$

Also, the product term $\delta f(\mathbf{x})$, for a linear function $f : \mathbb{R}^{n_{\mathbf{x}}} \to \mathbb{R}$ and $\delta \in \{0,1\}$, can be transformed into linear inequalities. The product term $\delta f(\mathbf{x})$ is replaced by an auxiliary variable z. The value of variable z should be $f(\mathbf{x})$ when the value of δ is 1, and 0 otherwise. This behavior can be expressed and transformed into linear inequality constraints as follows [16]:

$$([\delta = 1] \Rightarrow [z = f(\mathbf{x})]) \wedge (\sim [\delta = 1] \Rightarrow [z = 0]) \text{ is true iff } \begin{cases} z \leq \gamma_{\mathrm{M}}\delta \\ z \geq \gamma_{\mathrm{m}}\delta \\ z \leq f(\mathbf{x}) - \gamma_{\mathrm{m}}(1-\delta) \\ z \geq f(\mathbf{x}) - \gamma_{\mathrm{M}}(1-\delta), \end{cases} \tag{3.9}$$

where γ_{M} and γ_{m} are as def ned in (3.6)–(3.7). Note that in fact the relations (3.8) and (3.9) transform if-then-else statements into linear inequality constraints.

3.2.3 Mixed-logical dynamic models

A prediction model M based on the transformations discussed above can be cast into mixed-logical dynamic form to obtain a compact representation of the hybrid dynamics as follows [16]:

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k)+\mathbf{B}_1\mathbf{u}(k)+\mathbf{B}_2\boldsymbol{\delta}(k)+\mathbf{B}_3\mathbf{z}(k)\\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k)+\mathbf{D}_1\mathbf{u}(k)+\mathbf{D}_2\boldsymbol{\delta}(k)+\mathbf{D}_3\mathbf{z}(k)\\ \mathbf{E}_2\boldsymbol{\delta}(k)+\mathbf{E}_3\mathbf{z}(k) &\leq \mathbf{E}_1\mathbf{u}(k)+\mathbf{E}_4\mathbf{x}(k)+\mathbf{E}_5,\end{aligned}$$

where

$$\mathbf{x}(k)=\begin{bmatrix}\mathbf{x}_\mathrm{c}(k)\\ \mathbf{x}_\mathrm{b}(k)\end{bmatrix} \qquad \mathbf{y}(k)=\begin{bmatrix}\mathbf{y}_\mathrm{c}(k)\\ \mathbf{y}_\mathrm{b}(k)\end{bmatrix} \qquad \mathbf{u}(k)=\begin{bmatrix}\mathbf{u}_\mathrm{c}(k)\\ \mathbf{u}_\mathrm{b}(k)\end{bmatrix},$$

are the state, output, and input, respectively, separated into continuous components and binary components, i.e., $\mathbf{x}_\mathrm{c}(k)\in\mathbb{R}^{n_{\mathbf{x}_\mathrm{c}}}$, $\mathbf{x}_\mathrm{b}(k)\in\mathbb{R}^{n_{\mathbf{x}_\mathrm{b}}}$, $n_\mathbf{x}=n_{\mathbf{x}_\mathrm{c}}+n_{\mathbf{x}_\mathrm{b}}$, $\mathbf{y}_\mathrm{c}(k)\in\mathbb{R}^{n_{\mathbf{y}_\mathrm{c}}}$, $\mathbf{y}_\mathrm{b}(k)\in\mathbb{R}^{n_{\mathbf{y}_\mathrm{b}}}$, $n_\mathbf{y}=n_{\mathbf{y}_\mathrm{c}}+n_{\mathbf{y}_\mathrm{b}}$, $\mathbf{u}_\mathrm{c}(k)\in\mathbb{R}^{n_{\mathbf{u}_\mathrm{c}}}$, $\mathbf{u}_\mathrm{b}(k)\in\mathbb{R}^{n_{\mathbf{u}_\mathrm{b}}}$, $n_\mathbf{u}=n_{\mathbf{u}_\mathrm{c}}+n_{\mathbf{u}_\mathrm{b}}$. In addition, $\boldsymbol{\delta}(k)$ are the binary variables and $\mathbf{z}(k)$ are the auxiliary continuous variables.

3.3 Application: Household energy optimization

In this section we consider a decentralized multi-agent single-layer MPC approach for controlling energy in households. We discuss distributed energy resources, formalize the hybrid dynamics of a household in a model, and show how this model can be used for MPC control.

3.3.1 Distributed energy resources

Distributed energy resources, comprising distributed power generators, electricity storage units, and responsive loads, can play a crucial role in supporting the European Union's key policy objectives of market liberalization, combating climate change, increasing the amount of electricity generated from renewable sources, and enhancing energy saving. Large-scale diffusion of distributed energy resources will have a profound impact on the functioning of the electricity infrastructure: It will bring radical changes to the traditional model of generation and supply as well as to the business model of the energy industry [67]. Drivers for distributed energy resources are the generation and sale of electric energy and accompanying goods, such as CO_2 emission rights, and the provision of ancillary services for network operators.

Distributed generation of electricity, e.g., via photo-voltaics, wind turbines, or combined heat and power plants, has a good chance of pervading the electricity infrastructure in the future [67, 120]. Distributed generation offers environmental benef ts (e.g., due to the use of renewable energy sources and the eff cient use of fossil fuels), reduced investment risks, fuel diversif cation and energy autonomy, and increased energy eff ciency (e.g., due to fewer line losses and co-generation options). In addition, several electricity storage technologies are under development, e.g., lithium-ion batteries and plug-in hybrid electric vehicles [91]. Furthermore, options for load response are foreseen for the future power system [24].

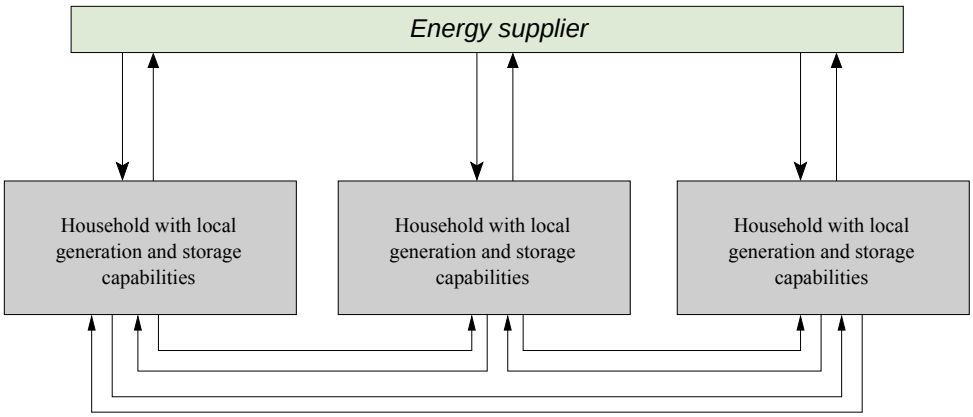


Figure 3.2: Households and an external energy supplier. Households can buy and sell energy to and from an external energy supplier or to and from neighboring households.

With an increase in distributed energy resources combined with more ICT and intelligence in the power network, the options for consumers with respect to energy demand response increase. In this section, we focus on residential distributed energy resources. Households with distributed energy resources operate more independently of energy suppliers, they can devise new contractual arrangements with suppliers and network managers, and they can buy and sell power among one another, and to and from their supplier, as shown in Figure 3.2. As a first step toward developing control structures that are installed in households for optimizing energy usage, we consider an individual household, not taking into account the possibility of energy exchange with neighboring households, i.e., we consider a decentralized multi-agent single-layer control structure[1].

3.3.2 System description

The system under study consists of a household interacting with its energy supplier, as depicted in Figure 3.3. As in conventional households, the household can buy electricity and gas from its energy supplier. In addition to this, the household can sell electricity to the energy supplier. The household can produce this electricity using a micro combined heat and power (μCHP) unit [120]. This unit can simultaneously produce heat and power for the household. It is typically located in a basement, underneath a sink, hanging from a wall, or outside. It can provide various energy needs, such as space and water heating, electricity, and, possibly, cooling.

We assume that the μCHP unit in the household is based on Stirling technology [120]. The unit provides electricity to an electricity storage unit, and heat to a heat storage unit. The μCHP unit consists of a Stirling engine prime mover, conversion unit 1, and an auxiliary burner, conversion unit 2. Conversion unit 1 converts natural gas $z_{g,1}(k)$ (in kWh) into

[1] The control agent that we will develop for control of a household could be located in a physical device such as the Qbox, which will soon become commerically available. See the website of Qurrent, the manufacturer of the Qbox, at http://www.qurrent.com/.

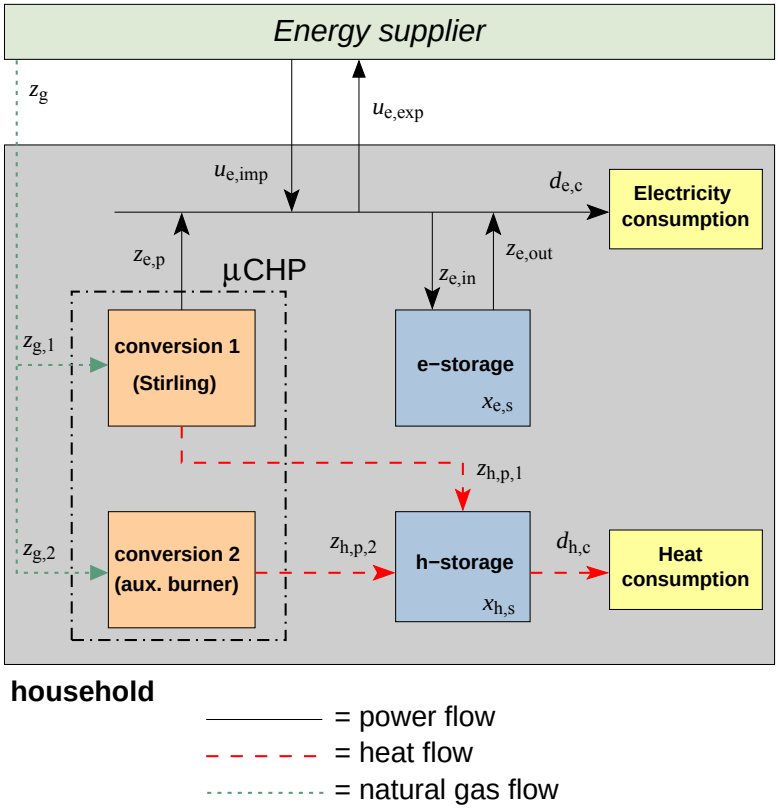


Figure 3.3: Conceptual overview of the system under study [68].

electricity produced $z_{e,p}(k)$ (in kWh) and heat produced $z_{h,p,1}(k)$ (in kWh), with a f xed ratio. The conversion unit can operate in *partial* or *full* mode and has a minimum activation constraint. Conversion unit 2 converts natural gas $z_{g,2}(k)$ (in kWh) to provide additional heat $z_{h,p,2}(k)$ (in kWh). For energy eff ciency reasons conversion unit 2 should be used as backup heat generator only. Therefore, priority has to be given to conversion unit 1. Conversion units 1 and 2 are equipped with built-in f xed controllers that are designed to keep the level of the heat storage unit $x_{h,s}(k)$ (in kWh) between certain upper and lower bounds.

The generated heat is supplied to a heat storage unit in the form of hot water. We consider an aggregated heat demand for the household, and therefore make no distinction between heat storage units for, e.g., space heating and sanitation heating. It is therefore also appropriate to assume that there is a single large heat storage unit. Such a conf guration is commercially available[2]. The level of the heat storage unit is indicated by the energy $x_{h,s}(k)$ (in kWh) in the heat storage unit. Heat consumption $d_{h,c}(k)$ (in kWh) takes heat from the storage unit, and therefore lowers the level of the heat storage unit $x_{h,s}(k)$. The level of the heat storage unit changes over time depending on the heat produced by the conversion units and the heat consumed.

[2] See, e.g., Gledhill Water Storage, http://www.gledhill.net/.

The generated electricity can be stored in a battery, e.g., a lithium-ion battery, of which the level is indicated by the energy in the battery $x_{\mathrm{e,s}}(k)$ (in kWh). Electricity can f ow to and from the battery, represented by $z_{\mathrm{e,in}}(k)$ (in kWh) and $z_{\mathrm{e,out}}(k)$ (in kWh), respectively. In addition to storing electricity, electricity can be used directly by the household for consumption, indicated by $d_{\mathrm{e,c}}(k)$ (in kWh), or it can be sold to the supplier through export, indicated by $u_{\mathrm{e,exp}}(k)$ (in kWh). Electricity can also be imported from the supplier through import, indicated by $u_{\mathrm{e,imp}}(k)$ (in kWh). The level of the electricity storage unit changes over time depending on the electricity produced by conversion unit 1, the electricity imported from or exported to the energy supplier, and the electricity consumed by the household.

System dynamics

Below we formalize the dynamics of the household. As we will see, these dynamics are hybrid, and the transformations from Section 3.2.2 can be used to obtain a prediction model consisting of only linear mixed-integer equality and inequality constraints.

Conversion unit 1 Conversion unit 1 can operate at partial generation or full generation. The control inputs are therefore $u_{1,\mathrm{part}}(k) \in \{0,1\}$ and $u_{1,\mathrm{full}}(k) \in \{0,1\}$, where input $u_{1,\mathrm{full}}(k)$ can only be used when $u_{1,\mathrm{part}}(k) = 1$. Depending on the control inputs, the conversion unit uses a different amount of gas $z_{\mathrm{g},1}(k)$. Conversion unit 1 converts this gas into electricity $z_{\mathrm{e,p}}(k)$ and heat $z_{\mathrm{h,p},1}(k)$. The gas used $z_{\mathrm{g},1}(k)$, the electricity provided to the internal network $z_{\mathrm{e,p}}(k)$, and the heat provided to the heat storage unit $z_{\mathrm{h,p},1}(k)$ are given by:

$$\begin{aligned} z_{\mathrm{g},1}(k) &= \eta_{\mathrm{g,part}} u_{1,\mathrm{part}}(k) + (\eta_{\mathrm{g,max}} - \eta_{\mathrm{g,part}}) u_{1,\mathrm{full}}(k) \\ z_{\mathrm{e,p}}(k) &= \eta_{\mathrm{e}} z_{\mathrm{g},1}(k) \\ z_{\mathrm{h,p},1}(k) &= (\eta_{\mathrm{tot}} - \eta_{\mathrm{e}}) z_{\mathrm{g},1}(k), \end{aligned}$$

where $\eta_{\mathrm{g,part}}$ (in kWh) is the gas used when the conversion unit operates partially, $\eta_{\mathrm{g,max}}$ (in kWh) is the gas used when the conversion unit operates at its maximum, η_{e} is the electric eff ciency of the unit, and η_{tot} is the total eff ciency of the unit, i.e., the electric and the heat eff ciency together.

When the conversion unit is in operation the dynamics of the household will be different from when the conversion unit is not in operation. In order to model logic rules relying on such information, a device-in-operation variable that indicates when conversion unit 1 is in operation is used. Based on the actuator setting $u_{1,\mathrm{part}}(k)$, which takes on binary values 0 and 1, the device-in-operation indicator $\delta_{\mathrm{dio},1}(k) \in \{0,1\}$ is def ned as:

$$[\delta_{\mathrm{dio},1}(k) = 1] \Leftrightarrow [u_{1,\mathrm{part}}(k) = 1],$$

which can be directly transformed into the linear equality constraint:

$$\delta_{\mathrm{dio},1}(k) = u_{1,\mathrm{part}}(k).$$

Using the device-in-operation variable $\delta_{\mathrm{dio},1}(k)$, the constraint that the full generation can only be switched on after the partial generation $u_{1,\mathrm{full}}(k)$ has been switched on is modeled with the inequality constraint:

$$u_{1,\mathrm{full}}(k) - \delta_{\mathrm{dio},1}(k) \leq 0.$$

Conversion unit 1 has a minimum activation constraint to avoid fast wear and tear of the device due to frequent on and off switching. The minimum activation constraint specifies that when the device has been switched on it has to stay in operation for at least $\eta_{\text{act,min}} \in \mathbb{N}^+$ time units, with $\mathbb{N}^+$ the positive natural numbers. In order to model the minimum activation constraint, introduce the counter $x_{\text{act}}(k) \in [0, x_{\text{act,max}}]$ (with $x_{\text{act,max}}$ a finite upper bound on the maximum time that a device can be in operation), which counts the number of time units that the device has been in operation so far. The evolution of this variable is given by the relation:

$$x_{\text{act}}(k+1) = \begin{cases} x_{\text{act}}(k)+1 & \text{if } \delta_{\text{dio},1}(k) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Using (3.9) this relation can be transformed into mixed-integer inequality constraints.

If the activation $x_{\text{act}}(k)$ is 0, then the conversion unit is allowed to stay switched off or to be switched on. However, if the activation $x_{\text{act}}(k)$ is larger than 0, then the conversion unit is not allowed to be switched off, until the activation $x_{\text{act}}(k)$ has reached the minimum activation $\eta_{\text{act,min}}$. Hence, as long as $x_{\text{act}}(k)$ is larger than 0 and smaller than $\eta_{\text{act,min}}$, the value of input $u_{1,\text{part}}(k)$ should stay at its maximum, i.e., 1. After the activation $x_{\text{act}}(k)$ has reached the minimum activation, the input $u_{1,\text{part}}(k)$ is allowed to have a different value again. To model this, introduce a constraint on the minimum value of $u_{1,\text{part}}(k)$ as follows:

$$u_{1,\text{part,min}}(k) \leq u_{1,\text{part}}(k), \tag{3.10}$$

with $u_{1,\text{part,min}}(k) \in \{0,1\}$. Using the activation variable $x_{\text{act}}(k)$ and this constraint we can enforce the minimum activation constraint by adjusting the lower limit $u_{1,\text{part,min}}(k)$ of $u_{1,\text{part}}(k)$ with the relation:

$$[1 \leq x_{\text{act}}(k) \leq \eta_{\text{act,min}} - 1] \Leftrightarrow [u_{1,\text{part,min}}(k) = 1].$$

To transform this relation we introduce auxiliary binary variables $\delta_1(k)$, $\delta_2(k)$, and $\delta_3(k)$ for which it holds that:

$$\begin{aligned} [1 \leq x_{\text{act}}(k)] &\Leftrightarrow [\delta_1(k) = 1] \\ [x_{\text{act}}(k) \leq \eta_{\text{act,min}} - 1] &\Leftrightarrow [\delta_2(k) = 1] \\ [\delta_3(k) = 1] &\Leftrightarrow [\delta_1(k) = 1] \wedge [\delta_2(k) = 1]. \end{aligned}$$

Hence, when $\delta_3(k)$ is equal to 1, then $x_{\text{act}}(k)$ is larger than 0, although it has not yet passed the minimum activation $\eta_{\text{act,min}}$, implying that the conversion unit should be kept in operation. To transform these three relations into mixed-integer inequality constraints, (3.5) and (3.8) are used.

Variable $\delta_3(k)$ is 1 if the device should be kept in operation, and 0 otherwise. This behavior is exactly the same behavior as variable $u_{1,\text{part,min}}(k)$ should have. Therefore, $u_{1,\text{part,min}}(k) = \delta_3(k)$, and the constraint that the conversion unit can only be switched off after an activation of $\eta_{\text{act,min}}$ is enforced by substituting $\delta_3(k)$ for $u_{1,\text{part,min}}(k)$ in (3.10).

A fixed controller is installed in conversion unit 1. This fixed controller is installed to guarantee a minimum level of heat in the heat storage unit. The fixed controller switches the conversion unit on when the level of the heat storage unit $x_{\text{h,s}}(k)$ is lower than a lower limit $\eta_{\text{h,s,lim,min},1}$, and switches it off when the level of the heat storage unit $x_{\text{h,s}}(k)$ is larger

than an upper limit $\eta_{\text{h,s,lim,max,1}}$. Let $u_{\text{1,part,tmp}}(k) \in \{0,1\}$ denote the actuator setting that the fixed controller would choose if the minimum activation constraint would not be present. The fixed controller determines the value for this variable as follows:

$$u_{\text{1,part,tmp}}(k) = \begin{cases} 1 & \text{for } x_{\text{h,s}}(k) \leq \eta_{\text{h,s,lim,min,1}} \\ u_{\text{1,part}}(k-1) & \text{for } \eta_{\text{h,s,lim,min,1}} < x_{\text{h,s}}(k) < \eta_{\text{h,s,lim,max,1}} \\ 0 & \text{for } x_{\text{h,s}}(k) \geq \eta_{\text{h,s,lim,max,1}}. \end{cases}$$

To transform this relation auxiliary variables $\delta_4(k)$, $\delta_5(k)$, $\delta_6(k)$, and $\delta_7(k)$ are defined such that:

$$\begin{aligned} [\delta_4(k) = 1] &\Leftrightarrow [x_{\text{h,s}}(k) \leq \eta_{\text{h,s,lim,min,1}}] \\ [\delta_5(k) = 1] &\Leftrightarrow [x_{\text{h,s}}(k) \geq \eta_{\text{h,s,lim,max,1}}] \\ [\delta_6(k) = 1] &\Leftrightarrow [\delta_4(k) = 0] \wedge [\delta_5(k) = 0] \\ \delta_7(k) &= \delta_6(k) u_{\text{1,part}}(k-1). \end{aligned}$$

Using (3.5) and (3.8) these relations are transformed into linear mixed-integer constraints. Given the values for these auxiliary variables, the fixed controller determines the value for $u_{\text{1,part,tmp}}(k)$ as:

$$u_{\text{1,part,tmp}}(k) = 1.\delta_4(k) + 0.\delta_5(k) + \delta_7(k).$$

In determining the actual setting for conversion unit 1, the fixed controller has to respect the minimum activation constraint. Therefore, the value that the fixed controller of conversion unit 1 chooses as input $u_{\text{1,part}}(k)$ to the actuator of conversion unit 1 is not the value of $u_{\text{1,part,tmp}}(k)$ directly, but the value determined as follows:

$$u_{\text{1,part}}(k) = \begin{cases} 1 & \text{if the conversion unit is not allowed to switch off} \\ u_{\text{1,part,tmp}}(k) & \text{otherwise,} \end{cases}$$

which can be written as:

$$u_{\text{1,part}}(k) = 1.\delta_3(k) + (1 - \delta_3(k)) u_{\text{1,part,tmp}}(k),$$

where $\delta_3(k)$ is defined through the minimum activation constraints. This relation can be transformed into linear mixed-integer constraints using (3.8).

Conversion unit 2 Conversion unit 2 has as control input $u_2(k) \in [0, u_{2,\max}]$. Depending on the control input, it uses a different amount of gas $z_{\text{g,2}}(k)$ and provides a different amount of heat $z_{\text{h,p,2}}(k)$ to the heat storage unit. The gas used $z_{\text{g,2}}(k)$ and the heat provided $z_{\text{h,p,2}}(k)$ are given by:

$$z_{\text{g,2}}(k) = u_2(k) \tag{3.11}$$

$$z_{\text{h,p,2}}(k) = \eta_{\text{tot}} z_{\text{g,2}}(k). \tag{3.12}$$

A device-in-operation variable $\delta_{\text{dio,2}}(k) \in \{0,1\}$ indicating when conversion unit 2 is in operation is defined as:

$$[u_2(k) \geq \gamma_{\epsilon,\text{mach}}] \Leftrightarrow [\delta_{\text{dio,2}}(k) = 1].$$

This relation can be converted into linear mixed-integer inequality constraints using (3.5). The device-in-operation variable $\delta_{\text{dio},2}(k)$ is used to enforce that conversion unit 2 is in operation only when conversion unit 1 is in operation through the following constraints:

$$\delta_{\text{dio},2}(k) - \delta_{\text{dio},1}(k) \leq 0. \tag{3.13}$$

A f xed controller is installed in conversion unit 2, similar to the f xed controller as in conversion unit 1. The f xed controller of conversion unit 2 determines an auxiliary actuator setting $u_{2,\text{tmp}}(k) \in \{0,1\}$ as follows:

$$u_{2,\text{tmp}}(k) = \begin{cases} 1 & \text{for } x_{\text{h,s}}(k) \leq \eta_{\text{h,s,lim,min,2}} \\ u_{2,\text{tmp}}(k-1) & \text{for } \eta_{\text{h,s,lim,min,2}} < x_{\text{h,s}}(k) < \eta_{\text{h,s,lim,max,2}} \\ 0 & \text{for } x_{\text{h,s}}(k) \geq \eta_{\text{h,s,lim,max,2}}. \end{cases}$$

To transform this relation, auxiliary variables $\delta_9(k)$, $\delta_{10}(k)$, $\delta_{11}(k)$, and $\delta_{12}(k)$ are def ned such that:

$$\begin{aligned} [\delta_9(k) = 1] &\Leftrightarrow [x_{\text{h,s}}(k) \leq \eta_{\text{h,s,lim,min,2}}] \\ [\delta_{10}(k) = 1] &\Leftrightarrow [x_{\text{h,s}}(k) \geq \eta_{\text{h,s,lim,max,2}}] \\ [\delta_{11}(k) = 1] &\Leftrightarrow [\delta_9(k) = 0] \wedge [\delta_{10}(k) = 0] \\ \delta_{12}(k) &= \delta_{11}(k) u_{2,\text{tmp}}(k-1). \end{aligned}$$

Using (3.5) and (3.8) these relations are transformed into linear mixed-integer constraints. The f xed controller now determines the value for the auxiliary actuator setting $u_{2,\text{tmp}}(k)$ as:

$$u_{2,\text{tmp}}(k) = 1.\delta_9(k) + 0.\delta_{10}(k) + \delta_{12}(k).$$

The auxiliary actuator setting $u_{2,\text{tmp}}(k)$ is used by the f xed controller to determine the actual input for conversion unit 2 as:

$$u_2(k) = u_{2,\text{tmp}}(k) \eta_{\text{frac}} u_{2,\text{max}},$$

where η_{frac} is the part of the maximum output $u_{2,\text{max}}$ that is activated when conversion unit 2 is switched on by the f xed controller.

Electricity and heat storage units The electricity and heat storage units are used to store energy. The storage units have a limited capacity. The level of the electricity storage unit $x_{\text{e,s}}(k)$ is determined by the amount of electricity $z_{\text{e,in}}(k)$ that goes into the storage unit, and the amount of electricity $z_{\text{e,out}}(k)$ that is taken out. It is assumed that the charging and discharging of the battery is without energy loss. The dynamics of the level of the electricity storage unit are given by:

$$x_{\text{e,s}}(k+1) = x_{\text{e,s}}(k) + z_{\text{e,in}}(k) - z_{\text{e,out}}(k).$$

The level of the heat storage unit $x_{\text{h,s}}(k)$ is inf uenced by the heat production of conversion units 1 and 2, i.e., $z_{\text{h,p,1}}(k)$ and $z_{\text{h,p,2}}(k)$, resepectively. The heat storage unit dynamics are given by:

$$x_{\text{h,s}}(k+1) = x_{\text{h,s}}(k) + z_{\text{h,p,1}}(k) + z_{\text{h,p,2}}(k) - d_{\text{h,c}}(k).$$

The levels of the electricity and heat storage units are limited by minimum and maximum values, i.e.:

$$x_{\mathrm{e,s,min}} \leq x_{\mathrm{e,s}}(k) \leq x_{\mathrm{e,s,max}}$$
$$x_{\mathrm{h,s,min}} \leq x_{\mathrm{h,s}}(k) \leq x_{\mathrm{h,s,max}}.$$

Power balance A power balance relating the power output of conversion unit 1 $z_{\mathrm{e,p}}(k)$, the input $z_{\mathrm{e,in}}(k)$ and output $z_{\mathrm{e,out}}(k)$ of the electricity storage unit, the electricity consumption $d_{\mathrm{e,c}}(k)$, and electricity bought $u_{\mathrm{e,imp}}(k)$ or sold $u_{\mathrm{e,exp}}(k)$ to the energy supplier, has to hold. This power balance is given by:

$$0 = z_{\mathrm{e,p}}(k) + u_{\mathrm{e,imp}}(k) + z_{\mathrm{e,out}}(k) - u_{\mathrm{e,exp}}(k) - z_{\mathrm{e,in}}(k) - d_{\mathrm{e,c}}(k).$$

3.3.3 MPC problem formulation

We now use the derived model as prediction model M for a control agent controlling the energy f ows of a household. The control agent has the task to automatically determine which actions should be taken in order to minimize the operational costs of fulf lling residential electricity and heat requirements, while maintaining the level of the heat storage unit between a desired upper and lower limit, and respecting the operational constraints, including a minimal activation of 2 time units. The control agent uses an MPC strategy such that the control agent can:

- optimize the usage of the heat and electricity storage units;
- take into account the decision freedom due to electricity import and export possibilities, and generation of energy by itself;
- incorporate predictions on residential electricity and heat demands;
- incorporate models of the dynamics and constraints of installed generators and storage units.

MPC scheme

At each control cycle k the control agent makes a measurement of the system state consisting of values for the level of the heat storage unit $x_{\mathrm{h,s}}(k)$, the level of the electricity storage unit $x_{\mathrm{e,s}}(k)$, and the activation counter $x_{\mathrm{act}}(k)$. Then the control agent determines values for the control inputs $u_{1,\mathrm{full}}(k)$, $u_{\mathrm{e,imp}}(k)$, and $u_{\mathrm{e,exp}}(k)$ by solving the MPC optimization problem that minimizes an objective function, subject to the prediction model M and initial constraints. Note that with respect to the conversion units, the control agent only determines $u_{1,\mathrm{full}}(k)$, since the values for $u_{1,\mathrm{part}}(k)$ and $u_2(k)$ are determined by the f xed controllers installed in the conversion units.

Objective function The main objective of the control agent is to minimize the daily operational costs of residential energy use. These costs depend on the price p_{f} (euro/kWh) for gas consumption, the price $p_{\mathrm{imp}}(k)$ (euro/kWh) at which electricity can be bought, and the price p_{exp} (euro/kWh) at which electricity can be sold. Note that in principle, the prices for

gas, electricity import and electricity output vary over the day. However, as a first step we assume that the price for gas consumption and power export are constant, whereas the price for importing electricity varies over the day.

In addition to minimizing the daily operational cost, the control agent should also maintain the level of the heat storage unit between the desired upper and lower limit. This goal is included as a soft constraint by penalizing an auxiliary variable $z_{\text{aux}}(k+l) \geq 0$, for $l = \{1, \ldots, N\}$, with a large positive cost p_{soft}. This auxiliary variable $z_{\text{aux}}(k+l)$ is defined such that:

$$z_{\text{aux}}(k+l) = \begin{cases} x_{\text{h,s}}(k+l) - \eta_{\text{h,s,lim,max}} & \text{for } x_{\text{h,s}}(k+l) \geq \eta_{\text{h,s,lim,max}} \\ 0 & \text{for } \eta_{\text{h,s,lim,min}} < x_{\text{h,s}}(k+l) < \eta_{\text{h,s,lim,max}} \\ \eta_{\text{h,s,lim,min}} - x_{\text{h,s}}(k+l) & \text{for } x_{\text{h,s}}(k+l) \leq \eta_{\text{h,s,lim,min}}, \end{cases}$$

which in combination with the minimization of the term $p_{\text{soft}} z_{\text{aux}}(k+l)$ can also be written as:

$$\eta_{\text{h,s,lim,min}} - z_{\text{aux}}(k+l) \leq x_{\text{h,s}}(k+l) \leq \eta_{\text{h,s,lim,max}} + z_{\text{aux}}(k+l).$$

The cost function at control cycle k over a prediction horizon of N control cycles, including the cost for the soft constraints, is defined as:

$$J = \sum_{l=0}^{N-1} \big(p_{\text{f}} \left(z_{\text{g,1}}(k+l) + z_{\text{g,2}}(k+l) \right) + p_{\text{imp}}(k+l) u_{\text{e,imp}}(k+l) \\ - p_{\text{exp}} u_{\text{e,exp}}(k+l) + p_{\text{soft}} z_{\text{aux}}(k+1+l) \big).$$

Note that p_{soft} should not be chosen too larger, since otherwise minimizing $z_{\text{aux}}(k+l)$ has too much weight.

Prediction model The prediction model M that the control agent uses is based on the relations that describe the system model as given in Section 3.3.2, specified over the prediction horizon. Hence, the prediction model M consists of a large system of linear mixed-integer equality and inequality constraints. The values of the parameters of the prediction model are given in Table 3.1.

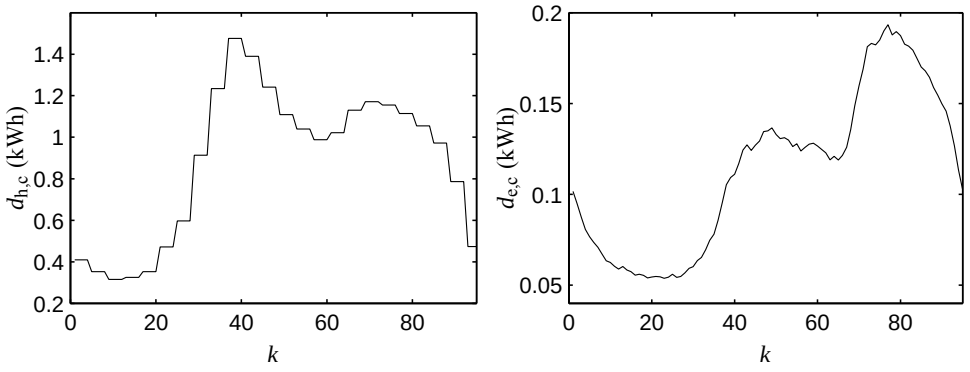
Initial constraints The initial constraints for $k = 1$ are:

$$\begin{aligned} x_{\text{e,s}}(k) &= \bar{x}_{\text{e,s}}(k) \\ x_{\text{h,s}}(k) &= \bar{x}_{\text{h,s}}(k) \\ x_{\text{act}}(k) &= \bar{x}_{\text{act}}(k) \\ u_{1,\text{part}}(k-1) &= \bar{u}_{1,\text{part}}(k-1) \\ u_{2,\text{tmp}}(k-1) &= \bar{u}_{2,\text{tmp}}(k-1), \end{aligned}$$

where the variables with a bar are known, e.g., through measurements.

parameter	value	parameter	value
$u_{2,\max}$	4.9383	$\eta_{g,\max}$	1.8333
$x_{act,\max}$	1.10^6	$\eta_{g,part}$	0.9167
$x_{e,s,\max}$	2	η_{tot}	1.0125
$x_{e,s,\min}$	0	$\eta_{h,s,lim,\max}$	8.1278
$x_{h,s,\max}$	9.1728	$\eta_{h,s,lim,max,1}$	6.9667
$x_{h,s,\min}$	0	$\eta_{h,s,lim,max,2}$	5.2250
$\gamma_{\epsilon,mach}$	1.10^{-8}	$\eta_{h,s,lim,\min}$	2.3222
$\eta_{act,\min}$	2	$\eta_{h,s,lim,min,1}$	4.0639
η_e	0.15	$\eta_{h,s,lim,min,2}$	2.9028
η_{frac}	0.6		

Table 3.1: Values of the parameters of the household system.



(a) Heat demand data.

(b) Electricity demand data.

Figure 3.4: Energy demand data for average Dutch household on January 29. One time unit corresponds to 15 minutes.

Solving the optimization problem The MPC optimization problem is a mixed-integer linear programming problem. It is linear, since the objective function and all constraints are linear and it is mixed integer, since the problem involves continuous and discrete variables. For solving the optimization problem at each control cycle we use the ILOG CPLEX v10.0 [71] linear mixed-integer programming solver through the Tomlab v5.7 interface [66] in Matlab v7.3 [98].

3.3.4 Simulations

To illustrate the operation of the proposed controller, we perform experiments for a particular winter day, January 29, 2006. For this day, average residential electricity and aggregated heat demand prof les have been created with 2006 data from 'EnergieNed', the Dutch Federation of Energy Companies. Figures 3.4(a) and 3.4(b) show the heat and electricity demand prof les of an average household on this day. Given such information, the control agent of

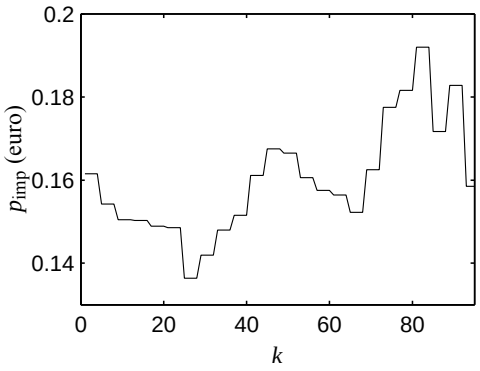


Figure 3.5: Electricity import price per kWh for January 29, 2006. One time unit corresponds to 15 minutes.

the household determines every 15 minutes new actions by solving its MPC problem at that time. To set up the control problem prices for electricity import, electricity export, and gas consumption have to be calculated first.

Price calculation

The variable electricity import price $p_{\text{imp}}(k)$ is calculated as follows. The Dutch Central Bureau of Statistics states a total electricity tariff for small consumers for 2006 of 194 euro/MWh[3] (household class: single tariff, 3000 kWh). The variable part of the total tariff (including energy and VAT taxes) is around 90 % of the total tariff[4], so this becomes 0.1746 euro/kWh. The variable supply part of the total tariff accounts for 32 % of the total tariff[3]. For this variable supply part we have substituted Dutch power exchange values taken from the Amsterdam Power Exchange data. In this way import prices as shown in Figure 3.5 were derived. For the value of the feedback tariff p_{exp} we have taken average 'EnergieNed' data for 2006, which gives 0.0601 euro/kWh.

The gas price p_{f} is determined as follows. At the website of the Dutch Central Bureau of Statistics, a total gas tariff for small consumers of 552 euro/1000 m^3 is given (for consumer class: 2000 m^3). According to the ECN website, 91 % of the gas tariff is variable (including taxes). This leads to a gas price of 0.50232 euro/m^3.

Simulations

Below we first illustrate the operation of the proposed MPC control agent for a particular setting of the prediction horizon length N. After that, we vary the length of the prediction horizon to see how this influences the performance over a day. We will then not only consider a household with fixed controllers in the conversion units installed, but also a household without these fixed controllers. This gives more freedom to the MPC control agent and is expected to improve the performance.

[3] See http://www.cbs.nl/, Dutch central bureau of statistics.
[4] See http://www.energie.nl/, Energy Research Center of The Netherlands (ECN).

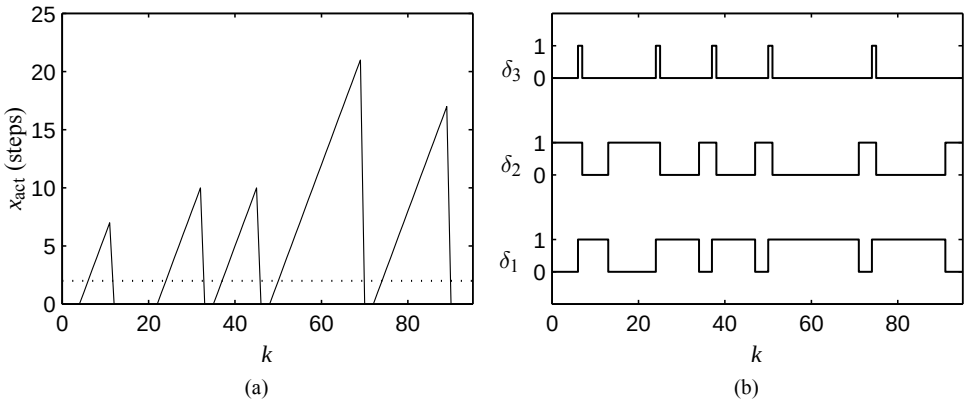


Figure 3.6: (a) Activation time $x_{act}(k)$ of conversion unit 1. The dotted horizontal line indicates the minimal activation time. (b) Evolution of δ_1, δ_2, and δ_3.

In principle the longer the prediction horizon is, the better the performance becomes. However, in practice the time required to solve the mixed-integer optimization problem restricts the length of the prediction horizon that can practically be used. To illustrate the operation of the proposed approach, we therefore below first consider a prediction horizon with length $N = 16$. The initial values for the simulation of the household are taken as:

$$\begin{aligned} \bar{x}_{\mathrm{e,s}}(k) &= 0 \\ \bar{x}_{\mathrm{h,s}}(k) &= 5.806 \\ \bar{x}_{\mathrm{act}}(k) &= 0 \\ \bar{u}_{1,\mathrm{part}}(k-1) &= 0 \\ \bar{u}_{2,\mathrm{tmp}}(k-1) &= 0. \end{aligned}$$

Results for $N = 16$

Figure 3.6(a) shows the activation time of conversion unit 1. Conversion unit 1 is switched on 5 times throughout the day, and stays in operation at least 2 time units. Hence, the constraints on the minimal activation time of 2 time units is respected. Figure 3.6(b) shows the evolution of the variables $\delta_1(k)$, $\delta_2(k)$, and $\delta_3(k)$ throughout the day. It is easy to verify that indeed, when conversion unit 1 is brought into operation, $\delta_3(k)$ becomes 1, and when conversion unit 1 has been in operation for at least 2 time units, $\delta_3(k)$ becomes 0 again.

Figure 3.7 shows the level of the heat storage unit. The fixed controllers installed in the conversion units should switch on the conversion units depending on the level of the heat storage unit. Figure 3.8(a) depicts the binary variables $\delta_4(k)$, $\delta_5(k)$, $\delta_6(k)$, and $\delta_7(k)$, which are used to indicate when conversion unit 1 should be switched on partially. In addition, Figure 3.8(b) shows the binary values used for determining the actuator values of conversion units 1 and 2. It is observed that, indeed, when the level of the heat storage unit reaches one of the lower limits, the respective conversion unit is switched on, whereas when the level reaches one of the upper limits, the respective conversion unit is switched

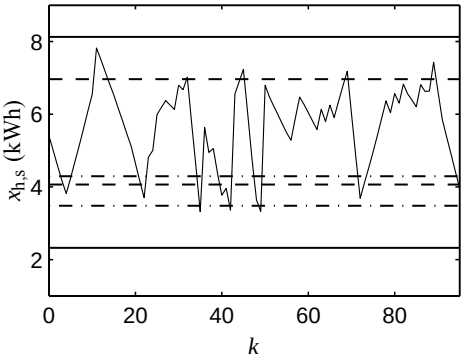


Figure 3.7: The level of the heat storage unit $x_{h,s}(k)$. The dashed and dashed-dotted lines indicate upper and lower activation bounds of the fixed controllers. The solid horizontal lines indicate physical upper and lower bounds.

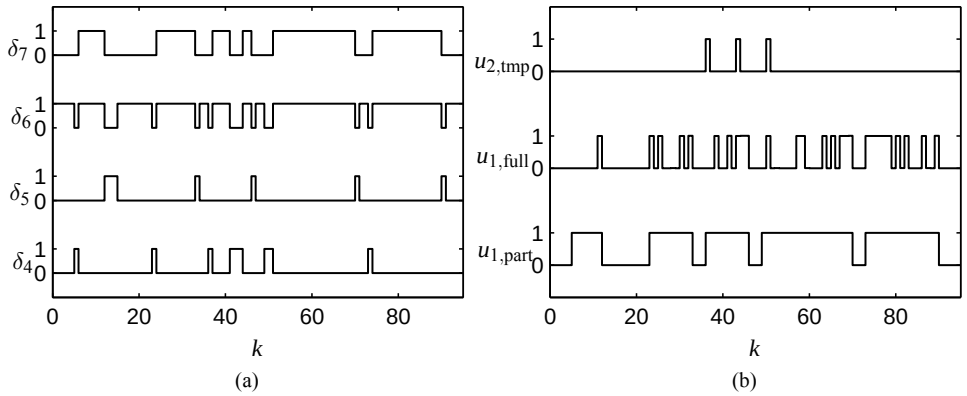


Figure 3.8: (a) Evolution of $\delta_4(k)$, $\delta_5(k)$, $\delta_6(k)$. (b) Evolution of the binary variables associated with the actuators of conversion units 1 and 2.

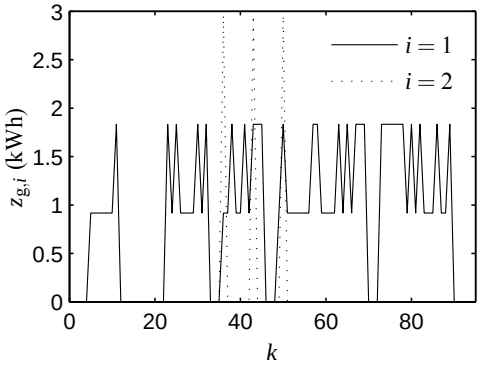


Figure 3.9: The gas consumed by the conversion units.

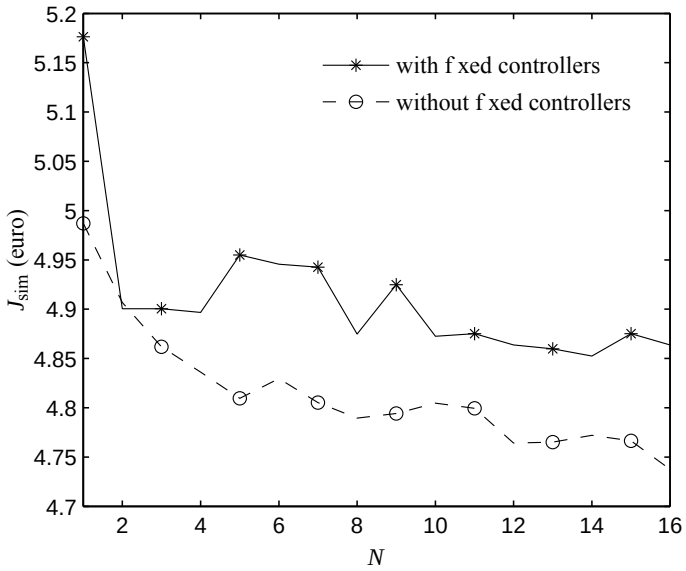


Figure 3.10: Performance J_{sim} for varying prediction horizon lengths N, both for the scenario in which the fixed controllers is installed, and for the scenario in which the fixed controllers are not installed.

off. Hence, the fixed controllers installed in the conversion units operate as they should. In addition, the MPC control agent decides to switch conversion unit 1 into full operation a number of times. When this happens, the MPC control agent has ensured that conversion unit 1 is already operating partially. Figure 3.9 shows the gas consumed by the conversion units, resulting from the actuator settings as chosen by the fixed controllers and the MPC control agent.

Results for varying prediction horizon lengths

We now consider the performance of the MPC control agent under varying lengths of the prediction horizon N. We consider two scenarios: the scenario considered so far, i.e., the scenario in which the MPC control agent controls the household including fixed controllers in the conversion units, and a scenario in which the fixed controllers in the conversion units are not present. In this second scenario, the MPC control agent has more decision freedom, since it can in the second scenario determine by itself when conversion unit 1 and conversion unit 2 should be switched on or off. Note that although the MPC control agent has this additional decision freedom, the prioritizing constraint for using conversion unit 1 before conversion unit 2, the constraint that conversion unit 1 should operate partially before switching to full operation, and the minimum activation time constraint for conversion unit 1 are still present.

Figure 3.10 shows the cost J_{sim} defined over the full simulation period for varying pre-

diction horizon lengths[5]. For both scenarios, there is a general trend that as the prediction horizon length increases, the performance increases as well. However, since the control agent does not take into account the energy consumption patterns and electricity price fluctuations after its prediction horizon, it can choose actions that are not optimal over the full simulation. Therefore, in our case it is not strictly necessary that the performance increases with a longer prediction horizon. We also observe this in Figure 3.10. From the figure we also observe that if the fixed controllers are not present, that then, indeed, the MPC control agent can expoit the the increased decision freedom. This results in a higher performance for the scenario in which the fixed controllers are not installed.

Discussion

With a longer prediction horizon, the number of binary variables increases linearly. This also implies that the computations involved in solving the corresponding MPC optimization problem increase. The household system that we consider does not go to a stable or steady state, since the electricity and heat consumption continuously keep varying. Therefore, in principle the prediction horizon should be taken over the same time span as information about consumption and prices are available. However, due to the computational requirements, this is currently not practical. In order to make computations involving prediction horizons of larger lengths approaches have to be investigated that somehow reduce the number of binary variables and possibly aggregate information regarding energy usage at prediction steps further away.

In this section we have considered energy control of an individual household, as a first step toward cooperative energy control of several interconnected households. The next step could consist of modeling interconnections between households, and developing a scheme that makes control agents of individual households obtain agreement on the values of the variables involved in modeling these interconnections. In the next section we go more into the issues involved in dealing with such interconnections.

3.4 Control of interconnected hybrid subnetworks

In the previous section we have assumed that the subnetworks, viz. the households, are independent of each other. In this section we do not make this assumption anymore, but instead allow for the subnetworks to be interconnected. Let therefore a transportation network be divided into n subnetworks. The subnetworks are interconnected as in Section 1.3.2, hence, typically the interconnections are physical links between subnetworks over which commodity flows from one subnetwork into another. Assume that each subnetwork has a control agent assigned to it. If the overall combined MPC control problem is convex, then the agents can use the multi-agent single-layer MPC approaches of Section 1.3.2.

[5] In order to compare the performance of the control for the two case studies a shrinking horizon [137] has been taken. In the shrinking horizon approach, initially the original prediction horizon N is taken, but as soon as predictions would go over the actual simulation time span, the prediction horizon will be reduced. If no shrinking horizon is taken, then comparing the performance for varying N based on the performance over 1 day is not fair, since the control agent using a larger N will at the end of the day already takes into account what will happen the next day, whereas the control agent using a smaller N will not consider this, since it optimizes over a shorter term. This will have an influence on the actions chosen at the end of the day and therefore on the performance.

In that case, the control agents locally determine in a number of iterations control actions that are overall optimal. However, when the subnetworks are hybrid systems and modeled with both continuous and discrete variables, then the overall control problem will not be convex. It is then the question what difficulties arise due to this nonconvexity, and how the approaches of Section 1.3.2 can be extended to give at best solutions that are close to or equal to overall optimal solutions, and at least solutions that are feasible solutions.

3.4.1 Hybrid subnetwork models

In Section 3.2 we have developed means to transform the dynamics of hybrid systems into linear mixed-integer equality and inequality constraints, i.e., mixed-logical dynamic models. Here, we consider a subclass of this type of models, namely those models for which the discrete dynamics are caused by inputs that can take on values from a discrete set only. In addition, we assume that all other variables, including the interconnecting variables between subnetworks, are continuous variables. Note that this type of models is an extension of the type of models considered in Chapter 2, since we now allow discrete inputs. An example of a situation in which the considered type of models appears in transportation networks is, e.g., in road traffic networks a situation in which local actions consist of discrete speed limit settings and interconnecting constraints between subnetworks are expressed in terms of continuously modeled car flows. In power networks an example of such a situation is, e.g., a situation in which local actions consist changing of power generation or consumption in discrete quantities and interconnecting constraints between subnetworks involve continuous amounts of power flowing between the subnetworks.

Remark 3.2 There are two different types of discrete inputs:

1. discrete inputs that have a direct meaning as a quantity since they are represented as numbers, typically taking on values from a set of integer or real numbers, e.g., $\{0, 0.2, \ldots, 1.0\}$;
2. discrete inputs that only have a symbolic meaning, taking on values from a set of symbolic values, e.g., $\{\text{red}, \text{yellow}, \text{green}\}$.

Although these are different types of discrete inputs, note that, however, the second class of discrete inputs can typically be transformed into the first class of inputs, and vice versa. □

Hence, assume that a network is divided into n subnetworks and that the dynamics of each subnetwork $i \in \{1, \ldots, n\}$ are given by a deterministic linear discrete-time time-invariant model, with noise-free outputs:

$$\begin{aligned} \mathbf{x}_i(k+1) &= \mathbf{A}_i\mathbf{x}_i(k) + \mathbf{B}_{1,i}\mathbf{u}_i(k) + \mathbf{B}_{2,i}\mathbf{d}_i(k) + \mathbf{B}_{3,i}\mathbf{v}_i(k) \\ \mathbf{y}_i(k) &= \mathbf{C}_i\mathbf{x}_i(k) + \mathbf{D}_{1,i}\mathbf{u}_i(k) + \mathbf{D}_{2,i}\mathbf{d}_i(k) + \mathbf{D}_{3,i}\mathbf{v}_i(k), \end{aligned} \tag{3.14}$$

where at time step k, for subnetwork i, $\mathbf{x}_i(k) \in \mathbb{R}^{n_{\mathbf{x}_i}}$ are local states, $\mathbf{u}_i(k) \in \mathcal{U}_i$ (with $\mathcal{U}_i$ a finite set of discrete values) are local inputs, $\mathbf{d}_i(k) \in \mathbb{R}^{n_{\mathbf{d}_i}}$ are known local exogenous inputs, $\mathbf{y}_i(k) \in \mathbb{R}^{n_{\mathbf{y}_i}}$ are local outputs, $\mathbf{v}_i(k) \in \mathbb{R}^{n_{\mathbf{v}_i}}$ are remaining variables influencing the local dynamical states and outputs, e.g., variables of neighboring subnetworks, and $\mathbf{A}_i \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{x}_i}}$, $\mathbf{B}_{1,i} \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{u}_i}}$, $\mathbf{B}_{2,i} \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{d}_i}}$, $\mathbf{B}_{3,i} \in \mathbb{R}^{n_{\mathbf{x}_i} \times n_{\mathbf{v}_i}}$, $\mathbf{C}_i \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{x}_i}}$, $\mathbf{D}_{1,i} \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{u}_i}}$, $\mathbf{D}_{2,i} \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{d}_i}}$, and $\mathbf{D}_{3,i} \in \mathbb{R}^{n_{\mathbf{y}_i} \times n_{\mathbf{v}_i}}$ determine how the different variables influence the local state and output of subnetwork i.

3.4.2 Non-convergence due to the discrete inputs

Suppose that we would setup the MPC control problems as in Section 1.3.2, but now based on the model including discrete inputs, i.e., (3.14). This means that the optimization problems become mixed-integer programming problems. In addition, for fixed values of the integer variables, the optimization problems are convex.

If we would use the schemes of Section 1.3.2 to solve the multi-agent control problem based on the models including the discrete inputs, it may be the case that the agents cannot come to agreement on the values of the interconnecting variables, while choosing locally optimal discrete inputs. A non-converging sequence can arise of values of the interconnecting variables on which the agents do not reach agreement.

In the original approaches, i.e., the serial and the parallel multi-agent single-layer MPC schemes with convex overall MPC problems, a control agent i receives the information from each neighboring agent $j \in \mathcal{N}_i$ regarding the values that neighboring agent j would like the interconnecting variables with respect to agent i to have. Then, control agent i processes this information by updating its interconnecting objective function $J_{\text{inter},i}$, and determines which values for the discrete inputs and interconnecting variables it prefers itself.

In the continuous case, the new values for the inputs and interconnecting variables will usually be slightly different from the values communicated in earlier iterations. However, when the inputs are discrete, the values for the *inputs* cannot slightly change, but only in discrete jumps. Hence, when a neighboring agent j suggests slightly different values for the *interconnecting* variables, control agent i will first include these values in its interconnecting objective function. After control agent i has solved its optimization problem using these new values, it will typically have obtained slightly changed values for the interconnecting variables, while having obtained values for the discrete inputs that are the same as the values at the previous iteration. So, the values of the discrete inputs will typically not change at each iteration, but only when the interconnecting objective function $J_{\text{inter},i}$ has reached such a level that switching to different discrete inputs is beneficial.

The relatively large jumps in the values of the discrete inputs have as a consequence that the values for the interconnecting variables can significantly change as well. A control agent will therefore then suggest rather different values for the interconnecting variables to its neighboring agents. This may cause that for another control agent after some more iterations a certain threshold of the interconnecting objective function has been reached, making it better for that agent to switch the values of its discrete inputs. Due to this mechanism, a series of discrete jumps in the values of discrete inputs can emerge that prevents the iterations from terminating. We will see an example of this behavior in Section 3.5.

3.4.3 Possible extensions of the original schemes

There are several ways in which the original schemes of Section 1.3.2 could be extended in order to break such a series of non-converging discrete jumps. Below we discuss some of these alternatives, based on straightforward extensions of the original schemes. We consider the following extensions:

1. Increasing the accuracy threshold The accuracy threshold $\gamma_{\epsilon,\text{term}}$ is used in the stopping condition to determine when the iterations should stop. It is linked to the maximum

allowable violation of the interconnecting constraints. Therefore, if this threshold is increased, the iterations will stop sooner since the values of interconnecting variables involved in an interconnecting constraint are allowed further apart from each other. However, this can obviously lead to predictions of the subnetwork that do not reflect the evolution of the physical subnetwork, and therefore to sub-optimally chosen inputs. In addition, it is a priori unknown to which value the accuracy threshold should be increased. If the increase is not large enough, the iterations may still continue.

2. Refining the discretization By making the discretization of the discrete inputs finer over the iterations, at some point the discretization will be fine enough to let the iterations converge to values for the interconnecting variables that make the stopping condition satisfied. By making a finer discretization for the discrete inputs, the changes in the discrete inputs from one iteration to another will be smaller, hence, approximating the case when there are only continuous inputs. In practice, however, the discretization of the discrete inputs may be given, and may not be adjustable. In that case the finer discretization can be rounded to the closest original discrete value. However, rounding of values has some consequences, as discussed in the next approach.

3. Relaxing and rounding The extreme case of refinement of the discretization appears when the discrete inputs are relaxed to continuous inputs, as is done, e.g., in [15]. In this case, the original schemes can be applied. At termination of the iterations, the resulting values for the continuous inputs can then be rounded to the closest discrete values for the discrete inputs. However, in particular when making predictions over a longer horizon this rounding can lead at least to sub-optimality and sometimes even to infeasibility. This is due to the fact that in general a rounded input has a different influence on the evolution of the subnetwork over a time step when compared to the influence that a continuous input would have. So in practice the evolution of the subnetwork will be different from the predictions made using the prediction model in the optimization.

4. Fixing the integer inputs The discrete inputs can be fixed once the non-converging series of values of the discrete inputs has been detected. The discrete inputs can be fixed to the locally most optimal values, or they can be fixed to the most frequently appearing values over a predefined number of earlier iterations. The remaining overall optimization problem will then become convex and the values of the other variables will converge to values that are optimal given the fixed discrete variables. In addition, at the end of the iterations the interconnecting constraints will be satisfied and thus the agents will have agreed on how the internetwork variables should evolve over the prediction horizon. Furthermore, the agents will have determined inputs that are feasible, and the agreements regarding the values for the interconnecting constraints will be fulfilled when these inputs are implemented. However, the fixed discrete variables may be sub-optimal from a network-wide perspective, and determining when the non-converging series of discrete values arises is a hard problem.

5. Increasing the penalty coefficient The penalty coefficient γ_c can be increased to a very high value once the non-converging series of discrete values has been detected. A large value for the penalty coefficient γ_c places all emphasis on obtaining satisfied interconnecting

constraints, and the discrete inputs that come with this can then be implemented. However, it may not be known a priori what the value of the penalty coeff cient γ_c should have in order to give convergence and in addition it is hard to determine when the non-converging series of discrete values appears. Therefore, inspired by [20], instead of increasing the penalty coeff cient γ_c abruptly when the non-converging series of discrete values has been detected, the penalty coeff cient γ_c can be increased in steps several times over the iterations. By increasing the penalty coeff cient γ_c in steps, the agents get some time to try to converge to values for the interconnecting variables that satisfy the stopping condition. If this convergence does not happen within a certain number of iterations, then the penalty coeff cient γ_c is increased again.

Discussion Comparing the alternatives, the main disadvantage of the alternatives based on increasing the accuracy threshold, and relaxing or ref ning of the discretization and then rounding, is that the values for the interconnecting variables observed in the system will be signif cantly different from those determined during the optimization. For the alternative based on increasing the accuracy threshold the reason for this disadvantage is that during the optimization the accuracy required on satisfying the interconnecting constraints is lowered, and thus the values that different control agents assign to particular interconnecting variables are allowed to be further apart. For the alternative based on relaxing or ref ning of the discretization and then rounding, the reason for this disadvantage is that the control agents have reached agreement on values for the interconnecting variables for a particular set of inputs, whereas a different set of inputs will be implemented on the system. The alternatives based on f xing the integer inputs and increasing the penalty coeff cient do not suffer from this disadvantage.

The alternative based on f xing the integer inputs requires that it can be detected when the integer inputs have to be f xed and it requires a strategy to determine to which values the inputs should be f xed. It is not straightforward to implement such strategies. The alternative based on increasing the penalty coeff cient does not have to address these issues. However, for this alternative it has to be determined at which frequency the penalty coeff cient should be increased, and with which factor. The settings that give the best performance will be problem specif c and therefore require tuning. From the alternatives discussed, this last alternative has the most natural way of dealing with the non-converging behavior, by emphasizing over the iterations more and more that a solution should be obtained with interconnecting constraints that are satisf ed. The predictions that each control agent therefore makes of its subnetwork are accurate at termination of the iterations.

Below we use the alternative based on increasing the penalty coeff cient to formulate a multi-agent single-layer MPC approach for interconnected hybrid systems.

3.4.4 Serial and parallel single-layer hybrid MPC approaches

For control of interconnected hybrid systems, in which the subnetworks are linear time-invariant systems with discrete inputs as modeled using (3.14), and the MPC overall control problem is convex for f xed values of the integer variables, we propose the serial and parallel scheme of Section 1.3.2, with the extension that the penalty coeff cient γ_c varies over the iterations, i.e., extension 5 above. Hence, the original serial and parallel scheme are followed in the sense that the agents perform local optimization steps and communication, but

the way in which the information from neighboring agents is included in updating the interconnecting objective function is different. Instead of using a fixed penalty coefficient, an iteration-varying penalty coefficient is taken. Every $N_{\Delta s}$ iterations, the penalty coefficient γ_c is multiplied by $\gamma_{\Delta c}$, with $\gamma_{\Delta c} > 1$.

Remark 3.3 The approaches proposed in this section for multi-agent MPC control of the assumed class of systems follow from rather straightforward extensions of the original approaches of Section 1.3.2. More complex extensions could be the result of combining the original approaches of Section 1.3.2 with optimization techniques for integer programming, such as distributed branch and bound or ADOPT [101]. In an iterative way of alternating between the distributed branch and bound and the approaches of Section 1.3.2, the distributed branch and bound approach could determine values for the integer variables, after which the integer values can be fixed, and the resulting convex overall problem can be solved using the approaches of Section 1.3.2. Such an approach could potentially address a larger class of systems than assumed here, although that remains to be investigated. □

In the following section we perform experiments with the proposed scheme on a load-frequency control problem with discrete power generation.

3.5 Application: Discrete-input load-frequency control

We consider the load-frequency control problem as defined in Section 2.5. In this load-frequency control problem a power network is divided into *n* subnetworks, each equipped with power generation and consumption capabilities. A control agent is assigned to each subnetwork. The objective of each control agent is to keep frequency deviations at a minimum after load disturbances. In order to achieve this objective each control agent can adjust the power generation in its subnetwork. In the original problem definition of Section 2.5, power generation was considered as a continuous input. Here, we assume that power generation can be adjusted in discrete amounts, hence, power generation is considered as a discrete input. Such discrete power generation is present, e.g., if generators can be switched on or off, or if actuators on the generator can take on values only from a discrete set of values. Furthermore, also load shedding, which can be seen in a way as negative power generation, is typically done in discrete amounts.

3.5.1 Network setup

For illustrative purposes, we consider a network consisting of 2 subnetworks, as shown in Figure 3.11. The dynamics and parameter of the subnetworks are as described in Section 2.5, with the exception that the inputs can only take on discrete values from the set $\{-1.0, -0.9, \ldots, 0.9, 1.0\}$.

3.5.2 Control setup

The control agents controlling subnetworks 1 and 2 use the objective function as defined in Section 2.5. The mixed-integer optimization problem that each control agent solves at an iteration is solved using the quadratic mixed-integer solver of ILOG CPLEX v10 [71], which we use through the Tomlab v5.7 [66] interface in Matlab v7.3 [98].

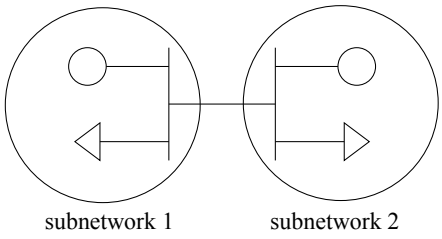


Figure 3.11: Network consisting of 2 subnetworks. Each subnetwork has generation and consumption capabilities.

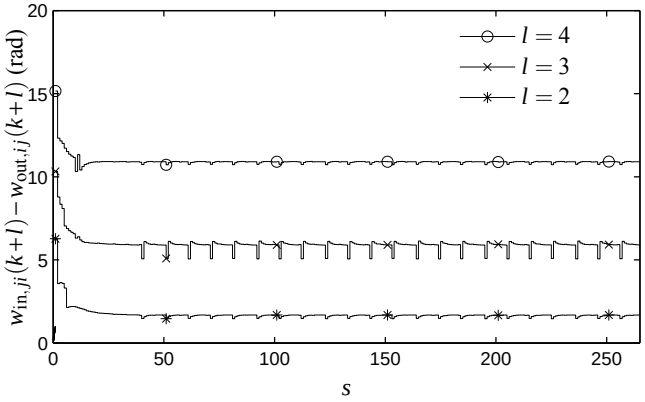


Figure 3.12: Results without *using the extended version of the serial MPC scheme. Per iteration the values of the interconnecting input variable of subnetwork 1 for prediction steps 2, 3, and 4 are shown. The values of the variable for the prediction steps 2, 3, and 4 are shifted with +5, +10, and +15, respectively.*

3.5.3 Simulations

To show the non-converging series of discrete values of the inputs, consider the experiment in which we take a prediction horizon with length $N = 5$ steps, an accuracy threshold $\gamma_{\epsilon,\text{term}} = 0.0001$, and an initial penalty coeff cient $\gamma_c(0)$ of 1. The penalty coeff cient $\gamma_c(s)$ is updated every $N_{\Delta s} = 50$ iterations, with a factor of $\gamma_{\Delta c} = 1.5$. The initial state of the network is $x_{\Delta f,1}(0) = 0$, $x_{\Delta\delta,1}(0) = 0$, $x_{\Delta f,2}(0) = 0$, and $x_{\Delta\delta,2}(0) = -1.0745$.

3.5.4 Results

When the control agents do not use the adjustment of the penalty term γ_c, then the non-converging series of discrete values appears, as illustrated in Figures 3.12 and 3.13 for the serial approach. The f gures illustrate that as the control agents exchange information, the value of the interconnecting input of control agent 1 changes, also when the values of the discrete inputs do not change. At the moments that the discrete inputs change, a clear jump is also observed in the value of the interconnecting input.

When the control agents use the penalty term increments extension, then the iterations converge, as illustrated in Figures 3.14, 3.15, and 3.16. It can be seen that in this case as the penalty coeff cient $\gamma_c(s)$ increases, the number of jumps in the discrete inputs reduces, and

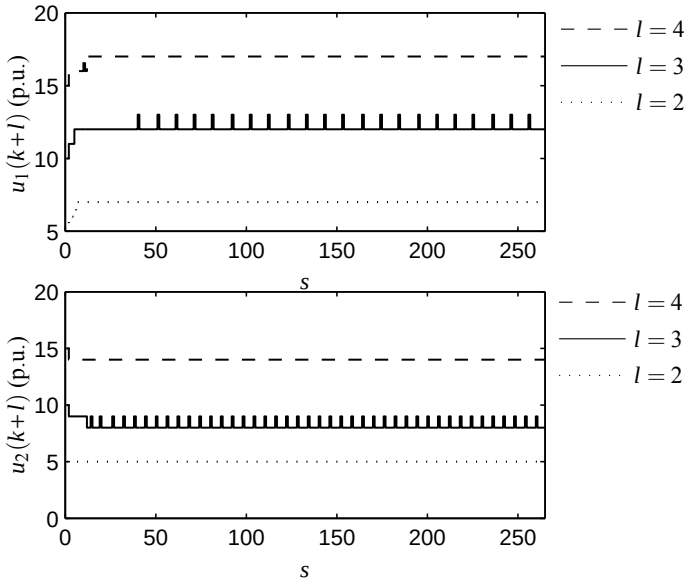


Figure 3.13: Results without *using the extended version of the serial MPC scheme. The values of the discrete inputs chosen by the agents of subnetworks 1 (top) and 2 (bottom), respectively, for prediction steps 2, 3, and 4 are shown. The values of the inputs for the prediction steps 2, 3, and 4 are shifted with +5, +10, and +15, respectively. In addition, the values of the inputs are scaled (before shifting) to take on integer values between -10 and 10.*

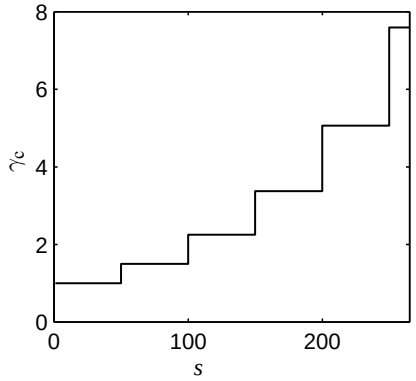


Figure 3.14: Evolution of penalty coefficient γ_c using the extended version of the serial MPC scheme.

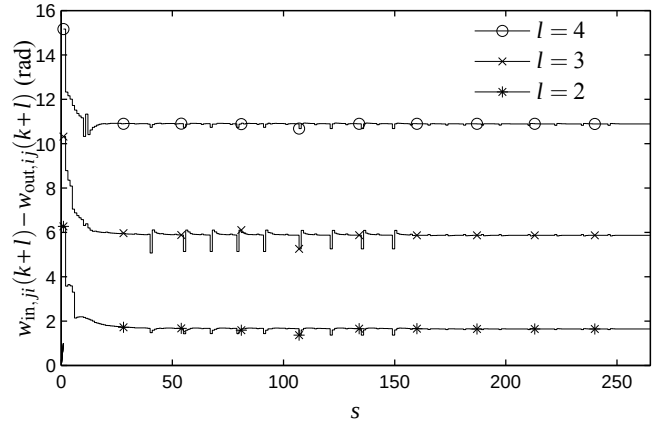


Figure 3.15: Interconnecting variable resulting from using the extended version of the serial MPC scheme. The values of the variable for the prediction steps 2, 3, and 4 are shifted with +5, +10, and +15, respectively.

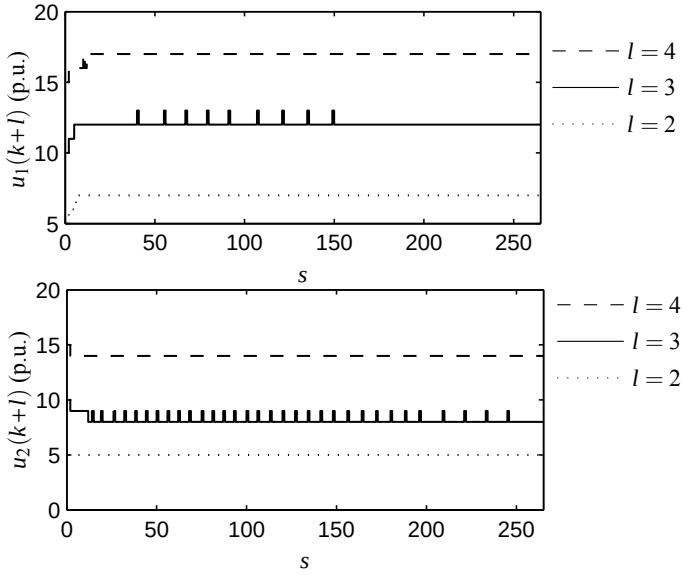


Figure 3.16: Inputs resulting from using the extended version of the serial MPC scheme. The values of the inputs for the prediction steps 2, 3, and 4 are shifted with +5, +10, and +15, respectively. In addition, the values of the inputs are scaled (before shifting) to take on integer values between -10 and 10.

that ultimately convergence is obtained. It is worth noting that the inputs that are chosen by the control agents are the same as those that would have been chosen by a centralized overall control agent.

3.6 Summary

In this chapter we have discussed multi-agent MPC control of transportation networks modeled as interconnected hybrid systems. In this setting, the network is divided into a number of subnetworks, each being controlled by a control agent that uses a model of its subnetwork and MPC to determine its actions.

We have first focused on modeling of hybrid systems and discussed how logic statements, which commonly appear in the description of hybrid systems, can be transformed into linear mixed-integer equality and inequality constraints. Then, we have illustrated the use of the transformations to construct a prediction model for an a single MPC control agent. Subsequently, we have focused on multi-agent control of networks consisting of subnetworks that are modeled as hybrid systems. We have focused on a particular type of hybrid subnetworks, viz. subnetworks with linear time-invariant dynamics that accept inputs that take on values from a discrete set of values only. Furthermore, we have discussed the problems that arise when the serial and parallel scheme of Chapter 2 would be applied to this type of system without modification. Moreover, we have discussed several alternative extensions of the original schemes to deal with these problems, and we have chosen one extension that results in control agents choosing feasible integer inputs, based on accurate subnetwork predictions. Several issues still have to be addressed in future research, including among others investigating formally whether the proposed scheme converges, determining formally what the quality of the solutions is, and determining when the penalty coefficient should be increased and with what value it should be increased. In addition, how to combine distributed optimization problem solvers for continuous and integer variables should be investigated.

In this chapter we have applied the topics discussed on two applications: energy control in households, and load-frequency control with discrete generation switching. For the energy control in households application we have used the transformations to derive a model for a household equipped with its own power generation (via a micro combined heat and power unit) and storage capabilities (via a water tank and a battery). As a first step toward a control structure in which multiple control agents, each representing a single household, jointly control the energy usage in a district, we have proposed a decentralized multi-agent single-layer MPC approach in which the control agents only consider their own household and no communication with other control agents takes place. In the application of the load-frequency control with discrete generation switching we have considered how the proposed extension of the serial scheme of Chapter 2 performs when the subnetworks do have interconnections, and the respective control agents do communicate with one another. We have illustrated that the extension proposed for dealing with non-convergence of the iterations of the MPC scheme can make the iterations converge.

In this chapter, as well as in Chapter 2, we have focused on issues particular to single-layer control, i.e., control in which the control agents have equal authority relationships with respect to one another. In Chapters 4 and 5 we focus more on how to take into account also control agents with different authority relationships.

Chapter 4

Multi-layer control using MPC

In the previous chapters we have discussed common issues arising due to the nature of large-scale transportation networks. In those chapters we have focused on particular issues in single-layer control, i.e., control in which control agents have equal authority relationships with respect to one another and control dynamics that take place at similar time scales. In this chapter we consider particular issues involved in multi-layer control, i.e., control in which control agents of higher control layers have authority over control agents in lower control layers, and control agents in higher control layers typically control dynamics at lower time scales. In Section 4.1 we introduce multi-layer MPC control for transportation networks, and in particular discuss how the prediction models that the control agents use can to be constructed. In Section 4.2 we discuss prediction models constructed for a highea-layer control agent using object-oriented modeling, which is suited for making prediction models of large-scale systems, and prediction models derived from such object-oriented models by linearization. We formulate an MPC problem based on such an object-oriented prediction model in Section 4.3. As we will see, the MPC problem based on the object-oriented prediction model leads to a nonconvex MPC problem, with an objective function that is expensive to evaluate. We consider two approaches for addressing this issue: i) the nonlinear MPC optimization problem is solved directly, using pattern search as solver; ii) a linearized approximation of the nonlinear optimization problem is solved, using an eff cient linear programming solver.

In this chapter we consider as application emergency voltage control. In Section 4.4 we develop an object-oriented model of a 9-bus dynamic power network and experimentally assess the performance of the proposed approaches on an emergency voltage control problem.

Parts of this chapter have been published in [110] and presented in [113].

4.1 Multi-layer control of transportation networks

As we have discussed in Chapter 1, there are several characteristics of transportation networks that make their control challenging. In Chapters 2 and 3, we have discussed how to deal with the large geographical region and the hybrid dynamics that transportation networks typically have. In this chapter, we discuss how to deal with the wide range of time

scales over which the dynamics of transportation networks typically evolve. Multi-layer control can be used for this.

4.1.1 Multi-layer control

If dynamics evolve over a wide range of time scales, then control of such dynamics can be done using multiple control agents that each consider a particular range of time scales. The control agents can be grouped into layers depending on the time scales they control.

Figure 4.1 illustrates multi-layer control of a network, in which the control structure consists of a higher, medium, and lower control layer. At lower layers, control agents that control faster dynamics are located. The faster dynamics will typically require faster control, hence at the lower control layers the time available to determine control actions is relatively small. However, to adequately describe the fast dynamics, more detailed dynamics have to be considered. Therefore at lower control layers, typically, more localized models of the dynamics will be used. Control agents that control slower dynamics are located at higher control layers. There more time is available to determine actions. However, the slower dynamics considered at the higher layers will typically involve larger regions of the network. Therefore, at higher control layers less detailed models are used. The result is a multi-layer or hierarchical control structure in which control takes place at different control layers based on space and/or time division [17]. The higher-layer control agents determine both actions to be implemented directly in the physical network, and set points to be provided to the control agents in a lower control layer. Hence, control agents in higher control layers can be seen as supervisory control agents.

In principle each control layer can consist of multiple control agents, each controlling their own group of control agents in a lower control layer. Communication among the control agents in each layer may or may not be present.

4.1.2 Multi-layer control in power networks

As an example of the multi-layer control of transportation networks, we consider power networks. Power networks in general are controlled using multi-layer control in which control of the physical network is the result of the joint effort of several control layers at local, regional, national, and sometimes international level [45, 60]. The physical power network consists of multiple interconnected subsystems, like generators, loads, transmission lines, etc. This physical network is controlled by several control layers in order to control the network in a desired way. The lowest control layer consists of control agents that locally control the actuators in the physical network. The higher control layers consists of control agents that determine actions and set-points for lower control layers. The set-points can be used to obtain coordination between the control agents of the lower control layers. The higher control layers typically consist of, e.g., regional or national human network operators. These human operators decide on the actions to take based on off ine studies, experience, heuristics, knowledge bases, and actual system conditions obtained via telemetry or obtained from state estimators and soft sensors. The set-points should be determined in such a way that objectives def ned for the higher control layer are achieved [100, 131]. The higher control layer hereby typically takes into account nonlinear behavior of the system, behavior that may be neglected by lower control layers.

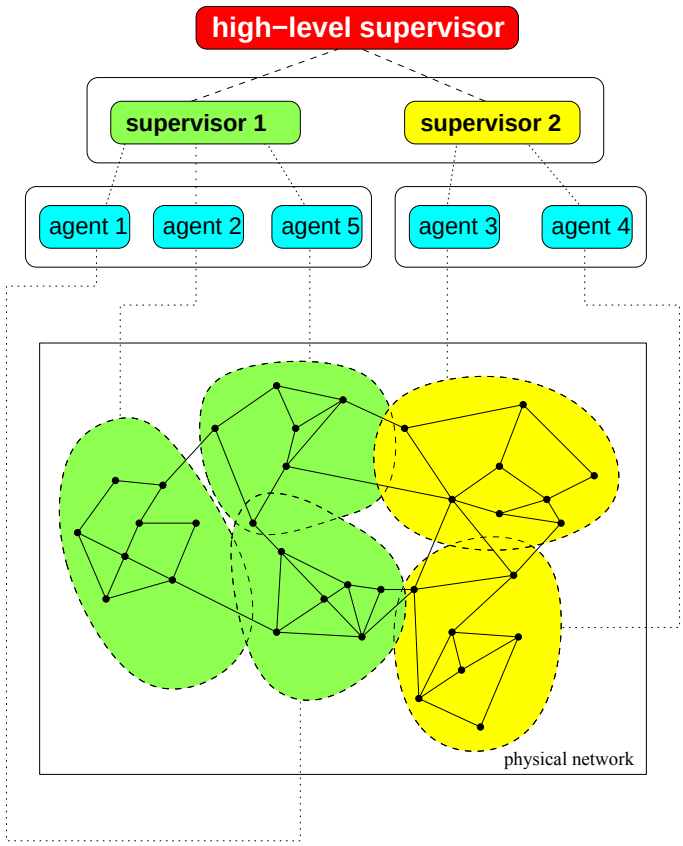


Figure 4.1: Illustration of multi-layer control. A higher layer provides set-points to a lower layer (dashed lines). The lower layer controls the actuators in the physical system (dotted lines).

It is in general not possible to rapidly change the set-points used by a lower control layer in an online and coordinated manner to achieve improved performance [45]. As it becomes more complex for human operators to adequately predict the consequences of faults and disturbances in the network (e.g., for power networks, due to deregulation of the energy market, the increase in power demands, and the emergence of embedded generation [73]), the need for intelligent automatic online control systems increases. These automatic control systems can be used to determine which set-points should be provided, at a first stage outside the control loop in the form of a decision support system, and at a later stage inside the control loop in the form of closed-loop control.

4.1.3 MPC in multi-layer control

Although in general there can be many control layers, and each control layer can consist of multiple control agents, in this chapter we restrict our focus to two control layers, a medium and a lower control layer. The medium control layer consists of a single control agent, and

the lower control layer consists of multiple decentralized control agents. In Chapter 5 we consider the control by a higher control layer that consists of multiple control agents that can communicate with one another.

To be able to obtain its control objectives, a control agent in a particular layer has to monitor the current state of the part of the lower control layer of its interest and the underlying physical network. Based on this, the control agent has to foresee when the behavior of the system is going into an undesirable direction such that it can provide adequate set-points to that part of the lower control layer that it considers. We propose a medium-layer control agent that at each control cycle uses MPC to determine which set-points to provide to the lower control layer.

In order for the medium-layer MPC control agent to meet its control objectives, it has to be able to predict how set-point changes influence the dynamics of the network. The performance of the control agent relies for a large part on the accuracy of the prediction model that it uses. The prediction model has to describe well how the actions of the control agent affect the behavior of the network and the lower-layer control agents. Ideally, the control agent should have a model of the complete dynamics of the network, including the behavior of the other control agents. However, such an ideal model can be very complex or impossible to construct, thus making the optimization procedure in the control agent slow or impossible. Instead, the control agent has to use an approximation of the model. If this approximation fits in a suitable form, relatively efficient optimization techniques can be used to determine the actions to take (e.g., linear or mixed-integer linear programming).

Suppose that the dynamics of the transportation network can be represented by a system of ordinary differential equations (ODEs) as:

$$\begin{bmatrix} \frac{\mathrm{d}\mathbf{x}_{\text{very slow}}}{\mathrm{d}t}(t) \\ \frac{\mathrm{d}\mathbf{x}_{\text{slow}}}{\mathrm{d}t}(t) \\ \frac{\mathrm{d}\mathbf{x}_{\text{fast}}}{\mathrm{d}t}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{\text{very slow}}(\mathbf{x}_{\text{very slow}}(t), \mathbf{x}_{\text{slow}}(t), \mathbf{x}_{\text{fast}}(t)) \\ \mathbf{f}_{\text{slow}}(\mathbf{x}_{\text{very slow}}(t), \mathbf{x}_{\text{slow}}(t), \mathbf{x}_{\text{fast}}(t)) \\ \mathbf{f}_{\text{fast}}(\mathbf{x}_{\text{very slow}}(t), \mathbf{x}_{\text{slow}}(t), \mathbf{x}_{\text{fast}}(t)) \end{bmatrix},$$

where the dynamics have been grouped into "very slow", "slow", and "fast dynamics". Suppose that the medium-layer control agent has as objective to control the slow dynamics only. The question is whether and how this control agent has to take into account the very slow and the fast dynamics. Although the control agent is not directly interested in the very slow and fast dynamics, these dynamics can influence the slow dynamics in which the control agent is interested. Simply ignoring the very slow and the fast dynamics may lead to unacceptable loss of model accuracy. Instead of ignoring the very slow and the fast dynamics completely, the control agent can approximate the very slow dynamics with constants, and the fast dynamics with instantaneous dynamics. The model that the control agent then considers can be described as:

$$\begin{bmatrix} \frac{\mathrm{d}\mathbf{x}_{\text{very slow}}}{\mathrm{d}t}(t) \\ \frac{\mathrm{d}\mathbf{x}_{\text{slow}}}{\mathrm{d}t}(t) \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{f}_{\text{slow}}(\mathbf{x}_{\text{very slow}}(t), \mathbf{x}_{\text{slow}}(t), \mathbf{x}_{\text{fast}}(t)) \\ \mathbf{f}_{\text{fast}}(\mathbf{x}_{\text{very slow}}(t), \mathbf{x}_{\text{slow}}(t), \mathbf{x}_{\text{fast}}(t)) \end{bmatrix}, \tag{4.1}$$

which constitutes a system of differential-algebraic equations (DAEs). Note that with respect to the fast dynamics a model such as discussed in Chapter 5 emerges. Note also that the very slow dynamics can include changes in set-points used by the medium-layer control agent. The medium-layer control agent can receive updates from higher-layer control agents

with respect to the very slow dynamics that it assumes constant, including the set-points, and it can in addition determine the set points for the lower-layer control agents, e.g., in such a way that the objectives related to its time scale dynamics are achieved.

Constructing the prediction model

Due to the complexity of transportation networks, constructing appropriate prediction models that can be used in control of these networks is a diff cult task. Constructing a model implementing (4.1) involves formalizing many components, differential equations, algebraic equations, mixed continuous and discrete elements, and dynamics at different time scales. Over the last decade modeling languages and simulation environments have been introduced that allow general-purpose physical modeling based on acausal modeling, mixing physical modeling using equations with the use of *object-oriented* constructs, and therewith significantly easing the development of such complex prediction models [13, 39, 99, 122]. In the next section we discuss object-oriented modeling and its use for constructing object-oriented prediction models implementing models such as (4.1). In addition, we discuss how prediction models approximating these object-oriented prediction models can be derived using linearization. These models will then be used in Section 4.3 for setting up MPC control problems.

4.2 Constructing prediction models with object-oriented modeling

4.2.1 Object-oriented modeling

To face the diff culty of constructing models of complex systems, object-oriented approaches for analysis and simulation of such networks have received increasing attention [95]. In object-oriented modeling, the structure of models of complex systems are determined by def ning objects for subsystems in these complex systems. The objects are used to map the structure of the model as closely as possible to the structure of the system. The objects are described in a declarative way by def ning only local equations of objects and the connections between the objects. To facilitate modeling, an object-oriented approach for modeling offers inheritance and composition concepts. Inheritance offers the possibility to form new classes of objects using classes that have already been def ned. The new classes take over or inherit attributes and behavior, e.g., dynamics, of the already existing classes. Extended models can then be constructed by inheriting dynamics and properties of more basic or more general models. E.g., for power networks, advanced generator objects are designed in this way by extending a basic generator objects that only contains the basic dynamics of a synchronous machine. Composition offers the possibility to combine simple objects into more complex ones. E.g., for power networks, when composing an object of a voltage regulator and an object of a turbine governor with an object of a basic generator, an object for a regulated generator with complex dynamics is obtained. Objected-oriented concepts enable proper structuring of models and generally lead to more f exible, modular, and reusable models.

4.2.2 Modeling tools

Several object-oriented approaches have been developed over the years, e.g., [13, 39, 99, 122, 136]. The approaches typically support both high-level modeling by composition and detailed component modeling using equations. Models of system components are typically organized in model libraries. A component model may be a composite model to support hierarchical modeling and may specify the system topology in terms of components and connections between the components. Using a graphical model editor, e.g., Dymola [39], a model can be def ned by drawing a composition diagram, by simply positioning icons that represent the models of the components and drawing connections between the icons. Parameter values of the underlying models are then conveniently specif ed in dialog boxes.

Most of the object-oriented simulation software packages assume that a system can be decomposed into objects with f xed causal relations [7]. Causal relations are relations between causes and effects. E.g., if there is a causal relationship between two objects A and B, then this means that if the variables of object A change, that then the variables of object B change as a consequence of the change of the variables of object A. In a f xed causal relations this behavior is def ned in one direction only. Hence, for objects A and B, the variables of object A do not change as a consequence of changes in variables of object B. In general, causality implies that the model of the system can be expressed as the interconnection of objects with an explicit state-space representation, in which algebraic relations as in (4.1) cannot be present. Often a signif cant effort in terms of analysis and analytical transformations is required to obtain a model in this form [39], in particular for systems in which causality is not naturally present, as is the case, e.g., in power networks. Setting the causality in a voltage-current formulation would mean that currents are expressed as function of voltages, or vice versa. Acausal modeling permits to relax the causality constraint and allows to focus on the elements and the way these elements are connected to each other, i.e., the system's topology. An environment that allows acausal modeling, is Dymola [39], which implements the object-oriented modeling language Modelica [136]. In Section 4.4 we will develop an object-oriented Modelica model for power networks using Dymola.

4.2.3 Object-oriented prediction models

Using an object-oriented modeling approach, each of the objects of a transportation network can be modeled with a mixture of differential equations, algebraic equations, and discrete logic. The model of the overall system then consists of the models for the objects and in addition algebraic equations interconnecting the individual objects.

For the object-oriented model to be useful as a prediction model that can be used by an MPC control agent, a method has to be available that can evaluate the model over a time horizon from time t_0 until t_f given the initial state of the system at time t_0. So, it should be possible to solve a so-called initial value problem that given the initial states $\mathbf{x}(t_0) \in \mathbb{R}^{n_\mathbf{x}}$, the initial inputs $\mathbf{u}(t_0) \in \mathbb{R}^{n_\mathbf{z}}$, and inputs $\mathbf{u}(t)$ specif ed over the full time interval, computes the outputs $\mathbf{y}(t) \in \mathbb{R}^{n_\mathbf{y}}$, for $t \in [t_0, t_f]$.

Note that a medium-layer control agent in fact does not provide set-points to a lower control layer continuously, but only at discrete control cycles k_c, for $k_c = \{0, 1, \ldots\}$, where control cycle k_c corresponds to continuous time $k_c T_c$, with T_c the control cycle time in continuous time units, as shown in Figure 4.2. A zero-order hold is used to make the trans-

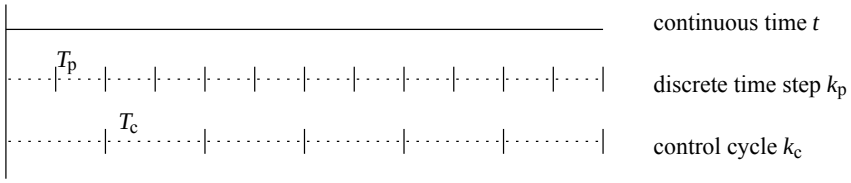


Figure 4.2: Overview of different time scales.

formation between the continuous-time input signal $\mathbf{u}(t)$ and the discrete-time input signal $\mathbf{u}(k_c)$. So, $\mathbf{u}(k_c) = \mathbf{u}_{k_c}$ becomes in continuous time:

$$\mathbf{u}(t) = \mathbf{u}_{k_c}, \quad \text{for } t \in [k_c T_c, (k_c+1)T_c).$$

Therefore, instead of specifying the continuous-time input signal $\mathbf{u}(t)$ to the prediction model M, a sequence of N_c inputs is specif ed to the prediction model M. The N_c inputs are collected in $\tilde{\mathbf{u}}(k_c)$ as $[(\mathbf{u}(k_c))^T, \ldots, (\mathbf{u}(k_c+N_c-1))^T]^T$, where $N_c = \frac{t_f - t_0}{T_c} + 1$ is the length of the prediction horizon in control cycles, and where for the sake of simplicity it is assumed that $t_f - t_0$ is an integer multiplier of T_c.

In general there is no analytic expression for the solution of the initial value problem. Instead, the trajectories of the variables of interest have to be approximated by numerical means to obtain values for these variables at discrete points in time. For control purposes we are typically interested in the outputs $\mathbf{y}(t)$. Assume that computing a sample of the continuous-time output $\mathbf{y}(t)$ for every T_p time units is suff cient to adequately represent the underlying continuous signals, where T_p is the length of one discrete time step, as illustrated in Figure 4.2. We then def ne the prediction horizon with a length $N_p = \frac{t_f - t_0}{T_p} + 1$ in discrete time steps, where for the sake of simplicity it is assumed that $t_f - t_0$ is an integer multiplier of T_p. We denote the outputs over the prediction horizon with length N_p by $\tilde{\mathbf{y}}(k_p) = [\mathbf{y}(k_p)^T, \ldots, \mathbf{y}(k_p+N_p-1)^T]^T$, where discrete time step $k_p = 0$ corresponds to continuous time $t = 0$ and discrete time step $k_p + l$ corresponds to continuous time $(k_p+l)T_p$.

Transition between t, T_p, and T_c

Below the notations $\mathbf{v}(t)$, $\mathbf{v}(k_p)$, and $\mathbf{v}(k_c)$, for some variables $\mathbf{v}$ each have to be interpreted in their own way. The notation $\mathbf{v}(t)$ refers to the variables $\mathbf{v}$ def ned at continuous time t, the notation $\mathbf{v}(k_p)$ refers to the variables $\mathbf{v}$ def ned at discrete time steps k_p, and the notation $\mathbf{v}(k_c)$ refers to the variables $\mathbf{v}$ def ned at control cycle k_c. In particular, if the continuous-time signal $\mathbf{y}(t)$ is sampled with a sample size T_p, the signal $\mathbf{y}(T_p)$ is obtained. If the continuous-time signal $\mathbf{y}(t)$ is sampled with a sample size T_c, the signal $\mathbf{y}(T_c)$ is obtained. The variables $\tilde{\mathbf{x}}(t)$, and $\tilde{\mathbf{y}}(t)$ can be transitioned in a similar way. Furthermore, if the control inputs at control cycle $\mathbf{u}(k_c)$ are subjected to a zero-order hold, the signals $\mathbf{u}(k_p)$ and $\mathbf{u}(t)$ can be obtained. The variables $\tilde{\mathbf{u}}(k_c)$ can be transitioned in a similar way. The zero-order hold for the control inputs to make the transition between $\mathbf{u}(k_c)$ and $\mathbf{u}(k_p)$ can be implemented as:

$$\mathbf{u}(k_p+l+l_2) = \mathbf{u}(k_p+l), \text{for } l = \{0, L, 2L, \ldots, N_c-1\}, \text{and } l_2 = \{1, 2, \ldots, L-1\}, \qquad (4.2)$$

where $L = \frac{N_p}{N_c}$, and where $\mathbf{u}(k_p+l)$ at discrete time k_p+l corresponds to $\mathbf{u}(k_c + \frac{l}{L})$ at control cycle $k_c + \frac{l}{L}$.

General object-oriented prediction model

Given the previous considerations, in the following we assume without loss of generality that the object-oriented prediction model of the transportation network is given by the mapping:

$$\tilde{\mathbf{y}}(k_\mathrm{p}) = M\left(\bar{\mathbf{x}}, \bar{\mathbf{u}}, \tilde{\mathbf{u}}(k_\mathrm{c})\right), \tag{4.3}$$

where the prediction model M maps the initial states $\bar{\mathbf{x}} = \mathbf{x}(k_\mathrm{c})$, the previous inputs $\bar{\mathbf{u}} = \mathbf{u}(k_\mathrm{c}-1)$, and the N_c inputs collected in $\tilde{\mathbf{u}}(k_\mathrm{c})$ to the N_p outputs collected in $\tilde{\mathbf{y}}(k_\mathrm{p})$. The prediction model thus includes the procedure to perform the time-domain simulation of the object-oriented model.

Remark 4.1 Here we have assumed that the initial derivatives $\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}(k_\mathrm{c})$ and initial algebraic variables $\mathbf{z}(k_\mathrm{c})$ can be uniquely determined when $\bar{\mathbf{x}}$ and $\bar{\mathbf{u}}$ are given. If this is not the case, then the initial derivatives and algebraic variables have to be provided to the prediction model M as well. □

A transformed prediction model

For the interconnected individual objects modeled with differential equations, algebraic equations, and discrete-event logic, there is no direct initial value problem solver. However, the object-oriented model can be transformed into a system of synchronous differential, algebraic, and discrete equations [39], leading to deterministic behaviour and automatic synchronization of the continuous and discrete parts of the model. The continuous dynamics are modeled using a system of DAEs. For handling discrete event dynamics, the synchronous data flow principle is employed [44]. The idea of this principle is that at each time instant all active equations have to be fulfilled concurrently. The active equations at a particular time instant consist of those equations representing the continuous dynamics at that time and possibly the equations related to the discrete events at that time [118].

If no discrete events would be present, and thus only a purely continuous system of DAEs is considered, a time domain simulation can be performed using the DAE solver DASSL [26, 121]. DASSL implements a variable integration step and variable order version of the backward differentiation formula [121]. Due to the variable integration step size, DASSL is in particular suited for performing simulations of dynamics involving fast and slow dynamics. Variable step size methods are well-suited for such dynamics, since these methods automatically choose a larger step size when no fast dynamics are present, and a smaller step size when they are [26]. The solver uses a predictor-corrector scheme. First, the predictor makes a guess of the solution at a new integration point. Then, the corrector determines the final solution by solving a system of algebraic equation, which is obtained after substituting the derivative with the backward differentiation formula. To use DASSL, the functions of the system of DAEs have to be specified. The Jacobian of this system of DAEs, which is used in the solution of the system of DAEs, can be supplied as a function, or it can be approximated numerically by DASSL.

To be able to adequately handle the discrete events present in the systems of our model, the solver DASSL-RT can be used. DASSL-RT is an extended version of the DASSL solver, including a root finder [121, 124]. The root finder is necessary to allow efficient simulation

of the discrete events. The root f nder checks mathematical indicator expressions that indicate when discrete events should be simulated. These indicator expressions are given in the same variables as the dynamics, and will therefore change values when the dynamics are simulated. If one of the indicator expressions changes sign during the simulation, the root f nder will back track the solution until the time instance when the indicator expression is equal to zero. The values of the simulation at that time will be returned. At event instants mixed continuous and discrete systems of equations are then solved to determine new values for the discrete variables and possibly the continuous variables.

4.2.4 Linearized object-oriented prediction models

The prediction model M in (4.3) typically is nonlinear and non-smooth, involving the numerical solution of systems of DAEs in combination with discrete logic. Therefore, computing the predictions is a costly process. This will have its effect on the time required to compute control actions. Instead of using the object-oriented prediction model directly, we can also try to derive an approximate prediction model from the object-oriented prediction model. This will result in optimization problems that are more eff cient to solve.

One way to approximate the object-oriented prediction model is by deriving a discrete-time linearized prediction model from the continuous-time dynamics represented in the system of DAEs, assuming small variations of the variables around the operation point for which the model is linearized. At each control cycle k_c, corresponding to continuous time $k_c T_c$ the continuous-time linearization for the system of DAEs:

$$\begin{aligned}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}(t) &= \mathbf{f}(\mathbf{x}(t),\mathbf{z}(t),\mathbf{u}(t))\\ 0 &= \mathbf{g}(\mathbf{x}(t),\mathbf{z}(t),\mathbf{u}(t))\\ \mathbf{y}(t) &= \mathbf{h}(\mathbf{x}(t),\mathbf{z}(t),\mathbf{u}(t)),\end{aligned}$$

around $\bar{\mathbf{x}} = \mathbf{x}(k_c)$, $\bar{\mathbf{u}} = \mathbf{u}(k_c - 1)$, $\bar{\mathbf{z}} = \mathbf{z}(k_c)$, and $\bar{\mathbf{y}} = \mathbf{y}(k_c)$ is given by the system:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}(t) = \mathbf{A}_c\mathbf{x}(t) + \mathbf{B}_c\mathbf{u}(t) + \mathbf{F}_c \tag{4.4}$$

$$\mathbf{z}(t) = \mathbf{C}_{c,z}\mathbf{x}(t) + \mathbf{D}_{c,z}\mathbf{u}(t) + \mathbf{G}_{c,z} \tag{4.5}$$

$$\mathbf{y}(t) = \mathbf{C}_{c,y}\mathbf{x}(t) + \mathbf{D}_{c,y}\mathbf{u}(t) + \mathbf{G}_{c,y}, \tag{4.6}$$

where

$$\mathbf{A}_c = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) + \frac{\partial \mathbf{f}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \left(\frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)$$

$$\mathbf{B}_c = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) + \frac{\partial \mathbf{f}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})$$

$$\mathbf{C}_{c,z} = \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})$$

$$\mathbf{D}_{c,z} = \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})$$

$$\mathbf{C}_{c,y} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) + \frac{\partial \mathbf{h}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})$$

$$\mathbf{D}_{c,y} = \frac{\partial \mathbf{h}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) + \frac{\partial \mathbf{h}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})$$

$$\begin{aligned}\mathbf{F}_c = &-\frac{\partial \mathbf{f}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \left(-\mathbf{g}(\bar{\mathbf{x}},\bar{\mathbf{u}},\bar{\mathbf{z}}) + \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{x}} + \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{u}}\right. \\ &\left. + \frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{z}}\right) - \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{x}} + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{u}} + \frac{\partial \mathbf{f}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{z}} - \mathbf{f}(\bar{\mathbf{x}},\bar{\mathbf{u}},\bar{\mathbf{z}})\right)\end{aligned}$$

$$\mathbf{G}_{c,y} = -\left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \left(-\mathbf{g}(\bar{\mathbf{x}},\bar{\mathbf{u}},\bar{\mathbf{z}}) + \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{x}} + \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{u}} + \frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{z}}\right)$$

$$\begin{aligned}\mathbf{G}_{c,y} = &-\left(\frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{x}} + \frac{\partial \mathbf{h}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{z}} + \frac{\partial \mathbf{h}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{u}} - \mathbf{h}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right) \\ &- \frac{\partial \mathbf{h}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}}) \left(-\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\right)^{-1} \left(-\mathbf{g}(\bar{\mathbf{x}},\bar{\mathbf{u}},\bar{\mathbf{z}}) + \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{x}} + \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{u}}\right. \\ &\left. + \frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})\bar{\mathbf{z}}\right),\end{aligned}$$

when $\frac{\partial \mathbf{g}}{\partial \mathbf{z}}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{u}})$ is invertible. The required Jacobians can either be derived analytically [83] or computed numerically. Using the modeling tool Dymola, the linearized model of the object-oriented model is conveniently obtained using symbolic differentiation.

Remark 4.2 It is assumed that initial algebraic variables $\mathbf{z}(k_c)$ can be uniquely determined given $\bar{\mathbf{x}}$ and $\bar{\mathbf{u}}$. If this is not the case, then $\mathbf{z}(k_c)$ should be specif ed to the prediction model M. □

Remark 4.3 The linearized prediction model can give adequate approximations when the discrete dynamics do not have a too large impact on the dynamics, and the changes in the continuous values are not too large. If the variations are not small, mode changes have to be considered in the model, e.g., by using piecewise aff ne or similar models [83]. □

The continuous-time linearization can be discretized with the sampling interval T_p, to obtain the following discrete-time linearized model in the aff ne expressions of $\mathbf{x}(k_p)$, $\mathbf{u}(k_p)$, and $\mathbf{y}(k_p)$:

$$\mathbf{x}(k_p+1) = \mathbf{A}\mathbf{x}(k_p) + \mathbf{B}\mathbf{u}(k_p) + \mathbf{F} \tag{4.7}$$

$$\mathbf{y}(k_p) = \mathbf{C}\mathbf{x}(k_p) + \mathbf{D}\mathbf{u}(k_p) + \mathbf{G}, \tag{4.8}$$

where k_p denotes the discrete time step, and where

$$\mathbf{A} = e^{\mathbf{A}_c T_p}$$

$$\mathbf{B} = \int_0^{T_p} e^{\mathbf{A}_c \tau} \mathrm{d}\tau \mathbf{B}_c$$

$$\mathbf{F} = \int_0^{T_p} e^{\mathbf{A}_c \tau} \mathrm{d}\tau \mathbf{F}_c$$

$$\mathbf{C} = \mathbf{C}_{\mathrm{c},y}$$
$$\mathbf{D} = \mathbf{D}_{\mathrm{c},y}$$
$$\mathbf{G} = \mathbf{G}_{\mathrm{c},y}.$$

The value of T_p determines how well the dynamics of the discrete-time model approximate the dynamics of the continuous-time linearized model (4.4)–(4.6). With a smaller value for T_p the approximation will be more accurate than with a larger value for T_p. However, with a smaller value for T_p the number of variables over a prediction horizon will become larger, which yields increased computational requirements for performing a simulation over a prediction horizon.

The discrete-time prediction model for $\tilde{\mathbf{x}}(k_\mathrm{p}+1)$ over a prediction horizon with length N_p discrete time steps is given by:

$$\tilde{\mathbf{x}}(k_\mathrm{p}+1) = \begin{bmatrix} \mathbf{A} & & & \\ & \mathbf{A} & & \\ & & \ddots & \\ & & & \mathbf{A} \end{bmatrix} \tilde{\mathbf{x}}(k_\mathrm{p}) + \begin{bmatrix} \mathbf{B} & & & \\ & \mathbf{B} & & \\ & & \ddots & \\ & & & \mathbf{B} \end{bmatrix} \tilde{\mathbf{u}}(k_\mathrm{p}) + \begin{bmatrix} \mathbf{F} \\ \mathbf{F} \\ \vdots \\ \mathbf{F} \end{bmatrix},$$

where the empty entries represent blocks of zeros. Substituting the expression for $\mathbf{x}(k_\mathrm{p}+l-1)$ in the expression for $\mathbf{x}(k_\mathrm{p}+l)$, for $l = \{1,\ldots,N_\mathrm{p}-1\}$, we can rewrite these equations as:

$$\tilde{\mathbf{x}}(k_\mathrm{p}+1) = \tilde{\mathbf{B}}\tilde{\mathbf{u}}(k_\mathrm{p}) + \tilde{\mathbf{F}}(\bar{\mathbf{x}}),$$

where

$$\tilde{\mathbf{B}} = \begin{bmatrix} \mathbf{B} & & & & \\ \mathbf{AB} & \mathbf{B} & & & \\ \mathbf{A}^2\mathbf{B} & \mathbf{AB} & \mathbf{B} & & \\ \vdots & \vdots & \vdots & \ddots & \\ \mathbf{A}^{N_\mathrm{p}-1}\mathbf{B} & \mathbf{A}^{N_\mathrm{p}-2}\mathbf{B} & \mathbf{A}^{N_\mathrm{p}-3}\mathbf{B} & \ldots & \mathbf{B} \end{bmatrix}$$

and

$$\tilde{\mathbf{F}}(\bar{\mathbf{x}}) = \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \mathbf{A}^3 \\ \vdots \\ \mathbf{A}^{N_\mathrm{p}} \end{bmatrix} \bar{\mathbf{x}} + \begin{bmatrix} \mathbf{F} \\ (\mathbf{A}+\mathbf{I})\,\mathbf{F} \\ \left(\mathbf{A}^2+\mathbf{A}+\mathbf{I}\right)\mathbf{F} \\ \vdots \\ \left(\mathbf{A}^{N_\mathrm{p}-1}+\ldots+\mathbf{A}+\mathbf{I}\right)\mathbf{F} \end{bmatrix}.$$

The discrete-time prediction model for $\tilde{\mathbf{y}}(k_\mathrm{p})$ over the prediction horizon of length N_p in discrete time steps is given by:

$$\tilde{\mathbf{y}}(k_\mathrm{p}) = \begin{bmatrix} \mathbf{C} & & & \\ & \mathbf{C} & & \\ & & \ddots & \\ & & & \mathbf{C} \end{bmatrix} \tilde{\mathbf{x}}(k_\mathrm{p}) + \begin{bmatrix} \mathbf{D} & & & \\ & \mathbf{D} & & \\ & & \ddots & \\ & & & \mathbf{D} \end{bmatrix} \tilde{\mathbf{u}}(k_\mathrm{p}) + \begin{bmatrix} \mathbf{G} \\ \mathbf{G} \\ \vdots \\ \mathbf{G} \end{bmatrix},$$

which after substitution of the prediction model for $\tilde{\mathbf{x}}(k_\mathrm{p})$ yields:

$$\tilde{\mathbf{y}}(k_\mathrm{p}) = \tilde{\mathbf{D}}\tilde{\mathbf{u}}(k_\mathrm{p}) + \tilde{\mathbf{G}}(\bar{\mathbf{x}}),$$

where

$$\tilde{\mathbf{D}} = \begin{bmatrix} \mathbf{D} & & & & \\ \mathbf{CB} & \mathbf{D} & & & \\ \mathbf{CAB} & \mathbf{CB} & \mathbf{D} & & \\ \vdots & \ddots & \ddots & \ddots & \\ \mathbf{CA}^{N_\mathrm{p}-2}\mathbf{B} & \mathbf{CA}^{N_\mathrm{p}-3}\mathbf{B} & \dots & \mathbf{CB} & \mathbf{D} \end{bmatrix}$$

and

$$\tilde{\mathbf{G}}(\bar{\mathbf{x}}) = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \mathbf{CA}^2 \\ \vdots \\ \mathbf{CA}^{N_\mathrm{p}-1} \end{bmatrix} \bar{\mathbf{x}} + \begin{bmatrix} \mathbf{CIF}+\mathbf{G} \\ (\mathbf{CA}+\mathbf{CI})\mathbf{F}+\mathbf{G} \\ \vdots \\ \left(\mathbf{CA}^{N_\mathrm{p}-2}+\ldots+\mathbf{CA}+\mathbf{CI}\right)\mathbf{F}+\mathbf{G} \end{bmatrix}.$$

To take into account that the control inputs can not be adjusted at each discrete time step k_p, but only at each control cycle k_c, the equalities def ning the zero-order hold on the (4.2) are added to the model. We can then denote the prediction model for $\tilde{\mathbf{y}}(k_\mathrm{p})$ by:

$$\tilde{\mathbf{y}}(k_\mathrm{p}) = M_\mathrm{lin}\left(\bar{\mathbf{x}}, \bar{\mathbf{u}}, \tilde{\mathbf{u}}(k_\mathrm{c})\right), \tag{4.9}$$

where $M_\mathrm{lin} = \left[\tilde{\mathbf{D}}\tilde{\mathbf{u}}\left(k_\mathrm{p}\right) + \tilde{\mathbf{G}}\left(\bar{\mathbf{x}}\right)\right]$. The obtained discrete-time approximation can be employed as a prediction model in the MPC problem formulation of the medium-layer control agent. It approximates the object-oriented prediction model (4.3).

4.3 Supervisory MPC control problem formulation

We now use the prediction models as discussed in the previous section to formulate the MPC problems that a medium-layer control agent can use. Every T_c time units the control agent has to determine inputs and set-points for the coming T_c time units. These variables have to be chosen in such a way that costs over a prediction horizon of N_c control cycles, i.e., over a time span of $N_\mathrm{c}T_\mathrm{c}$ time units, are minimized. Let the control objectives of the control agent consist of determining inputs and set-points such that over the entire prediction horizon:

- the values of the output variables $\tilde{\mathbf{y}}(k_\mathrm{p})$ are maintained between given upper and lower bounds;
- the changes in the values of the set-points $\tilde{\mathbf{u}}(k_\mathrm{c})$ are minimized.

We formulate the MPC problems as a nonlinear optimization problem and a linear optimization problem.

4.3.1 Nonlinear MPC formulation

To formulate the MPC problem as a nonlinear optimization problem, we first transform the control objectives in a straightforward way into a nonlinear objective function as follows:

$$J(\tilde{\mathbf{y}}(k_\mathrm{p}),\tilde{\mathbf{u}}(k_\mathrm{c})) = \sum_{l=0}^{N_\mathrm{p}-1} \|\mathbf{Q}_\mathbf{y}\mathbf{y}_\mathrm{err}(\mathbf{y}(k_\mathrm{p}+l))\|_\infty + \|\mathbf{Q}_\mathbf{u}(\mathbf{u}(k_\mathrm{c})-\bar{\mathbf{u}})\|_1 + \sum_{l=1}^{N_\mathrm{c}-1} \|\mathbf{Q}_\mathbf{u}(\mathbf{u}(k_\mathrm{c}+l)-\mathbf{u}(k_\mathrm{c}+l-1))\|_1, \quad (4.10)$$

where $\bar{\mathbf{u}}$ are the set-points provided at the last control cycle, i.e., $\bar{\mathbf{u}} = \mathbf{u}(k_\mathrm{c}-1)$, $\mathbf{Q}_\mathbf{y}$ and $\mathbf{Q}_\mathbf{u}$ are penalty matrices, $\|\mathbf{v}\|_\infty$ and $\|\mathbf{v}\|_1$ denote the infinity and one norm of vector $\mathbf{v}$, respectively, and where $\mathbf{y}_\mathrm{err}(\mathbf{y}(k_\mathrm{p}))$ are the violations of the desired output bounds, the entries of which are computed as:

$$\mathbf{y}_{q,\mathrm{err}}(y_q(k_\mathrm{p})) = \begin{cases} y_{q,\mathrm{desired,min}} - y_q(k_\mathrm{p}) & \text{for } y_q(k_\mathrm{p}) \le y_{q,\mathrm{desired,min}} \\ 0 & \text{for } y_{q,\mathrm{desired,min}} < y_q(k_\mathrm{p}) < y_{q,\mathrm{desired,max}} \\ y_q(k_\mathrm{p}) - y_{q,\mathrm{desired,max}} & \text{for } y_q(k_\mathrm{p}) \ge y_{q,\mathrm{desired,max}}, \end{cases} \quad (4.11)$$

where v_q indicates entry q of vector $\mathbf{v}$, and $y_{q,\mathrm{desired,min}}$ and $y_{q,\mathrm{desired,max}}$ are the desired upper and lower bounds of y_q. The infinity norm is taken for minimization of the variables $\mathbf{y}_\mathrm{err}(\mathbf{y}(k_\mathrm{p}))$, such that the worst error is minimized. The one norm is used for the changes in the inputs $\mathbf{u}(k_\mathrm{c}+l)-\mathbf{u}(k_\mathrm{c}+l-1)$, such that the changes in each of the inputs are minimized.

The values of the output variables $\tilde{\mathbf{y}}(k_\mathrm{p})$ are related to the inputs $\tilde{\mathbf{u}}(k_\mathrm{c})$ through the prediction model, as specified in (4.3). Hence, the supervisory MPC control problem can be formulated as:

$$\min_{\tilde{\mathbf{y}}(k_\mathrm{p}),\tilde{\mathbf{u}}(k_\mathrm{c})} J(\tilde{\mathbf{y}}(k_\mathrm{p}),\tilde{\mathbf{u}}(k_\mathrm{c})) \quad (4.12)$$

subject to

$$\tilde{\mathbf{y}}(k_\mathrm{p}) = M(\bar{\mathbf{x}},\bar{\mathbf{u}},\tilde{\mathbf{u}}(k_\mathrm{c})) \quad (4.13)$$

$$\tilde{\mathbf{u}}_\mathrm{min} \le \tilde{\mathbf{u}}(k_\mathrm{c}) \le \tilde{\mathbf{u}}_\mathrm{max}, \quad (4.14)$$

where $\tilde{\mathbf{u}}_\mathrm{min}$ and $\tilde{\mathbf{u}}_\mathrm{max}$ are vectors with bounds on the elements of $\tilde{\mathbf{u}}(k_\mathrm{c})$, and the variables with a bar are given. Instead of keeping the relation (4.13) for the prediction model as an explicit equality relation, this relation can be eliminated by substituting it into the objective function, since only the objective function depends on $\tilde{\mathbf{y}}(k_\mathrm{p})$. This substitution has computational advantages, since after the substitution the optimization problem has fewer variables and no nonlinear equality constraints. Hence, the MPC problem reduces to:

$$\min_{\tilde{\mathbf{u}}(k_\mathrm{c})} J(M(\bar{\mathbf{x}},\bar{\mathbf{u}},\tilde{\mathbf{u}}(k_\mathrm{c})),\tilde{\mathbf{u}}(k_\mathrm{c})) \quad (4.15)$$

subject to

$$\tilde{\mathbf{u}}_\mathrm{min} \le \tilde{\mathbf{u}}(k_\mathrm{c}) \le \tilde{\mathbf{u}}_\mathrm{max}. \quad (4.16)$$

Since the objective function of this problem includes the prediction model M and due to the definition of $\mathbf{y}_\mathrm{err}(\mathbf{y}(k_\mathrm{p}))$ the optimization problem is in general a nonconvex optimization problem subject to simple bound constraints.

Below we consider two approaches to solve the problem at hand. First, we propose to use the direct-search method pattern search as an appropriate solver for directly solving the nonlinear MPC problem. Pattern search has as advantage its more effective way of dealing with the problem at hand when compared to solvers for nonlinear optimization that require gradient or Hessian information. However, computational time requirements may be large. As an alternative, we consider solving the nonlinear MPC problem by using a linearized approximation of the problem. The advantage of this approach is that it may be more efficient in terms of computational requirements. However, the restricted validity of linearized models may jeopardize the quality of the resulting control actions. In Section 4.4 we experimentally compare the two approaches.

4.3.2 Direct-search methods for nonlinear optimization

In the MPC problem (4.15)–(4.16), evaluating the objective function is expensive due to the evaluation of the prediction model. In practice, computation time is limited and within the available computation time a solution that is as good as possible has to be determined. Many nonlinear optimization methods rely on gradient and Hessian information [18, 115]. However, the saturation and the use of the infinity norm in the objective function make that the objective function has many flat areas in which the gradient and Hessian are both equal to zero and thus not informative. Solvers that use this first-order or second-order information will therefore perform unnecessary numerical approximation of the gradient and the Hessian, involving numerous objective function evaluations. In addition, the MPC problem (4.15)–(4.16) typically has many local minima in which gradient-based solvers typically quickly can get stuck.

Instead of using gradient or Hessian-based solvers, we propose to use so-called direct-search optimization methods, which do not explicitly require gradient and Hessian information [32, 150]. The only property that these methods require is that the values of the objective function can be ranked [87]. This feature together with the feature that direct-search methods are suitable for non-smooth problems [32], make that these methods are suitable for solving the nonlinear problem (4.15).

Pattern search

For solving the nonlinear MPC problem (4.15)–(4.16), which is based on the object-oriented prediction model, we propose to use the direct-search method pattern search [87], for its straightforward implementation and its ability to yield good solutions, even for objective functions with many local minima, in combination with a multi-start method [96], to improve the probability of obtaining a solution close to a globally optimal solution. Several theoretical issues of pattern search have been discussed in [9, 10, 84, 138].

Pattern search works in an iterative way. Given the solution $\mathbf{s}^{(s-1)}$ at iteration $s-1$, if a new solution $\mathbf{s}^{+}$ is found for which it holds that $J(\mathbf{s}^{+}) < J(\mathbf{s}^{(s-1)})$, then the solution at iteration s becomes $\mathbf{s}^{+}$. If such a new solution is not found, then the solution at iteration s equals the solution at the previous iteration. The new solution $\mathbf{s}^{+}$ has to be selected from a finite set of candidate solutions in a mesh $\mathcal{M}^{(s)}$ that is updated at each iteration. An iteration of pattern search for an unconstrained problem is summarized in the following steps [87]:

- A mesh $\mathcal{M}^{(s)}$ around the last solution $\mathbf{s}^{(s-1)}$ is constructed, consisting of a discrete set of candidate solutions in $\mathbb{R}^{n_\mathbf{s}}$ in which the algorithm searches for a new solution. The coarseness of the mesh is determined by the mesh size $\gamma_{\text{mesh}}^{(s-1)} \in \mathbb{R}^+$.
- The mesh $\mathcal{M}^{(s)}$ is explored in one or two phases:
  - In the *search phase* any strategy can be used to find a solution $\mathbf{s}^+ \in \mathcal{M}^{(s)}$ for which $J(\mathbf{s}^+) < J(\mathbf{s}^{(s-1)})$, as long as a finite number of points is considered. If a solution $\mathbf{s}^+$ is found, the search was successful and the next phase is not invoked.
  - In the *polling phase* a new solution $\mathbf{s}^+$ for which $J(\mathbf{s}^+) < J(\mathbf{s}^{(s-1)})$ is searched for in a subset of solutions in $\mathcal{M}^{(s)}$, consisting of those solutions that are in the direct neighborhood of the last solution $\mathbf{s}^{(s-1)}$. This neighborhood is defined through a set of vectors called a pattern and the current solution. If a solution $\mathbf{s}^+$ is found in this neighborhood then the polling phase was successful.
- If either of the phases was successful, then $\mathbf{s}^{(s)} = \mathbf{s}^+$, the coarseness of the mesh is set to $\gamma_{\text{mesh}}^{(s)} = \gamma_{\text{exp}}\gamma_{\text{mesh}}^{(s-1)}$, with expansion factor $\gamma_{\text{exp}} > 1$, and the next iteration starts. If $\mathbf{s}^+$ was not found, then $\mathbf{s}^{(s)} = \mathbf{s}^{(s-1)}$, the coarseness of the mesh is set to $\gamma_{\text{mesh}}^{(s)} = \gamma_{\text{contr}}\gamma_{\text{mesh}}^{(s-1)}$, with contraction factor $\gamma_{\text{contr}} \in (0,1)$, and the next iteration starts.

The iterations continue until a stopping condition is satisfied, e.g., the mesh size is less than a given tolerance, the total number of objective function evaluations reaches a given maximum, or the distance between the point found at one successful poll and the point at the next successful poll is less than a given tolerance.

Approaches of pattern search for solving constraint optimization problems have been addressed in the literature, e.g., for optimization problems with bound constraints [86], linear constraints [84], and nonlinear constraints [85].

Multi-start pattern search

The combination of pattern search with multi-start for solving the control problem at control cycle k_c consists of solving the control problem from N_{init} different initial solutions, with N_{init} a positive integer. In general, the larger N_{init}, the larger the chance of obtaining a solution close to a globally optimal solution. However, in practice computation time is limited, since control set-points have to be provided at each control cycle. Therefore, our multi-start implementation involves starting from different initial solutions as long as time is available. The first initial solution is based on the (perhaps shifted) solution of control cycle $k_\text{c}-1$, since the solution of control cycle $k_\text{c}-1$ typically gives a good guess of the solution at control cycle k_c. The solution with the minimal objective function value after optimization with pattern search when the maximum computation time has elapsed is used as the final solution at control cycle k_c. See [96] for an overview of further characteristics of multi-start methods.

Although multi-start methods generally increase the time required to solve an optimization problem significantly, multi-start methods can typically be executed in a highly parallel fashion. In particular when a straightforward multi-start method is chosen that relies on randomly generated initial solutions, then each optimization problem involved in the multi-start method can be solved on an independent processor. For N_{init} initial solutions executed

on p processors the overall execution time is then expected to improve with approximately a factor p compared to when a single processor is used.

In Section 4.4 we experimentally compare the performance of multi-start pattern search with a multi-start gradient-based optimization.

4.3.3 Linear MPC formulation

The approach proposed above for solving the nonlinear optimization problem based on multi-start pattern search may still require a signif cant amount of computation time. Instead of solving the nonlinear optimization problem directly, we here discuss solving an approximation of the nonlinear optimization problem by linearization. This approach has the potential to require a signif cantly smaller amount of computation time, although possibly at the price of reduced performance.

To obtain a linear approximation of the MPC formulation of (4.12)–(4.14), the linearized prediction model (4.9) can be used instead of the object-oriented prediction model (4.3), and a transformation of the nonlinear objective function (4.10) and the expression for $\mathbf{y}_{\mathrm{err}}(\mathbf{y}(k_{\mathrm{p}}))$ in (4.11) can be made into linear objective terms and inequality constraints. First, note that the following optimization problem:

$$\min_{\mathbf{y}_{\mathrm{err}}} \|\mathbf{y}_{\mathrm{err}}(\bar{\mathbf{y}}(k_{\mathrm{p}}))\|_{\infty}$$

where $\mathbf{y}_{\mathrm{err}}$ as def ned in (4.11), for any f xed $\bar{\mathbf{y}}(k_{\mathrm{p}})$, is equivalent to the optimization problem:

$$\begin{aligned} &\min_{\mathbf{y}_{\mathrm{err}}} \|\mathbf{y}_{\mathrm{err}}\|_{\infty} \\ &\text{subject to} \\ &\qquad \begin{aligned} \bar{\mathbf{y}}(k_{\mathrm{p}}) &\geq \mathbf{y}_{\mathrm{desired,min}} - \mathbf{y}_{\mathrm{err}} \\ \bar{\mathbf{y}}(k_{\mathrm{p}}) &\leq \mathbf{y}_{\mathrm{desired,max}} + \mathbf{y}_{\mathrm{err}} \\ \mathbf{y}_{\mathrm{err}} &\geq \mathbf{0}, \end{aligned} \end{aligned}$$

where $\mathbf{0}$ is a zero vector of length $n_{\mathbf{y}_{\mathrm{err}}}$. Note also that the inf nity-norm based optimization problem:

$$\min_{\mathbf{v}} \|\mathbf{Q}\mathbf{v}\|_{\infty},$$

where $\mathbf{v} \in \mathbb{R}^{n_{\mathbf{v}}}$, and $\mathbf{Q} \in \mathbb{R}^{n_{\mathbf{v}} \times n_{\mathbf{v}}}$, is equivalent to the linear programming problem:

$$\begin{aligned} &\min_{\mathbf{v}, z_{\infty}} z_{\infty} \\ &\text{subject to } -\mathbf{1} z_{\infty} \leq \mathbf{Q}\mathbf{v} \\ &\qquad\qquad \mathbf{Q}\mathbf{v} \leq \mathbf{1} z_{\infty}, \end{aligned}$$

where $z_{\infty} \in \mathbb{R}$, and $\mathbf{1}$ is a one vector of length $n_{\mathbf{v}}$. In addition, note that the one-norm based optimization problem:

$$\min_{\mathbf{v}} \|\mathbf{Q}\mathbf{v}\|_{1},$$

where $\mathbf{v} \in \mathbb{R}^{n_\mathbf{v}}$, $\mathbf{Q} \in \mathbb{R}^{n_\mathbf{v} \times n_\mathbf{v}}$, is equivalent to the linear programming problem:

$$\begin{aligned} &\min_{\mathbf{v},\mathbf{z}_1} \mathbf{1}^\mathrm{T}\mathbf{z}_1 \\ &\text{subject to } -\mathbf{z}_1 \leq \mathbf{Q}\mathbf{v} \\ &\qquad\qquad \mathbf{Q}\mathbf{v} \leq \mathbf{z}_1, \end{aligned}$$

where $\mathbf{z}_1 \in \mathbb{R}^{n_{\mathbf{z}_1}}$ and $\mathbf{1}$ is a one vector of length $n_{\mathbf{z}_1}$. Using these equivalences the nonlinear optimization problem as def ned in (4.12)–(4.14) is transformed into:

$$\min_{\tilde{\mathbf{y}}(k_\mathrm{p}),\tilde{\mathbf{u}}(k_\mathrm{c}),\tilde{\mathbf{y}}_\mathrm{err}(k_\mathrm{p}),\tilde{\mathbf{z}}_\infty(k_\mathrm{p}),\tilde{\mathbf{z}}_1(k_\mathrm{c})} \sum_{l=0}^{N_\mathrm{p}-1} z_\infty(k_\mathrm{p}+l) + \sum_{l=0}^{N_\mathrm{p}-1} \mathbf{1}^\mathrm{T}\mathbf{z}_1(k_\mathrm{p}+l) \tag{4.17}$$

subject to

$$\begin{aligned} &\mathbf{y}(k_\mathrm{p}+l) \geq \mathbf{y}_{\text{desired,min}} - \mathbf{y}_\mathrm{err}(k_\mathrm{p}+l) \\ &\mathbf{y}(k_\mathrm{p}+l) \leq \mathbf{y}_{\text{desired,max}} + \mathbf{y}_\mathrm{err}(k_\mathrm{p}+l) \\ &\mathbf{y}_\mathrm{err}(k_\mathrm{p}+l) \geq \mathbf{0} \\ &-z_\infty(k_\mathrm{p}+l) \leq \mathbf{Q}_\mathbf{y}\mathbf{y}_\mathrm{err}(k_\mathrm{p}+l) \\ &\mathbf{Q}_\mathbf{y}\mathbf{y}_\mathrm{err}(k_\mathrm{p}+1) \leq z_\infty(k_\mathrm{p}+l) \\ &\qquad\qquad \text{for } l = 0,\ldots,N_\mathrm{p}-1 \\ &-\mathbf{z}_1(k_\mathrm{c}) \leq \mathbf{Q}_\mathbf{u}(\mathbf{u}(k_\mathrm{c}) - \bar{\mathbf{u}}) \\ &\mathbf{Q}_\mathbf{u}(\mathbf{u}(k_\mathrm{c}) - \bar{\mathbf{u}}) \leq \mathbf{z}_1(k_\mathrm{c}) \\ &-\mathbf{z}_1(k_\mathrm{c}+l) \leq \mathbf{Q}_\mathbf{u}(\mathbf{u}(k_\mathrm{c}+l) - \mathbf{u}(k_\mathrm{c}+l-1)) \\ &\mathbf{Q}_\mathbf{u}(\mathbf{u}(k_\mathrm{c}+l) - \mathbf{u}(k_\mathrm{c}+l-1)) \leq \mathbf{z}_1(k_\mathrm{c}+l) \\ &\qquad\qquad \text{for } l = 1,\ldots,N_\mathrm{c}-1 \\ &\tilde{\mathbf{y}}(k_\mathrm{p}) = M_\mathrm{lin}(\bar{\mathbf{x}},\bar{\mathbf{u}},\tilde{\mathbf{u}}(k_\mathrm{c})) \\ &\tilde{\mathbf{u}}_\mathrm{min} \leq \tilde{\mathbf{u}}(k_\mathrm{c}) \leq \tilde{\mathbf{u}}_\mathrm{max}. \end{aligned} \tag{4.18}$$

Since we have a linear objective function with linear equality and inequality constraints, and since all variables are continuous, this MPC optimization problem is a linear programming problem, for which there exist good commercial and free solvers [103].

4.4 Application: Voltage control in a 9-bus power network

A major source of power outages is voltage instability [142]. Voltage instability in general stems from the attempt of load dynamics to restore power consumption beyond the capability of the combined transmission and generation system after a fault. The control problem we are dealing with in this section is emergency voltage control, i.e., control to prevent a particular type of voltage instability. After a fault, e.g., a partial or total outage of a line, the generation and transmission network may not have suff cient capacity to provide the loads with power. A lower layer of decentralized control agents will try to restore the behavior of the system to an acceptable level. However, due to the reduced transmission capacity of the network the requested load demand together with the given system conf guration place the network under an excessive amount of strain and voltages may start to drop. Corrective

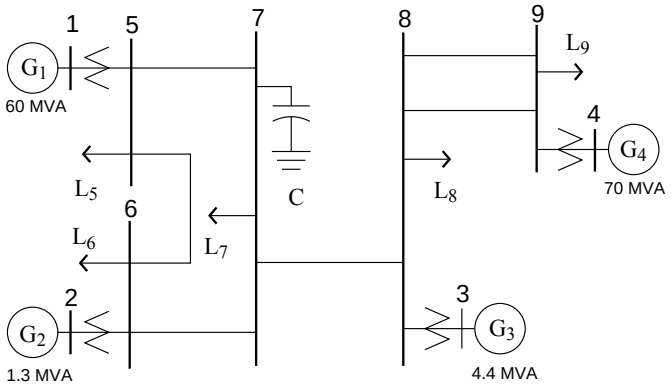


Figure 4.3: Network topology of the 9-bus dynamic network. The generators are shown with their nominal apparent power ratings.

actions have to be taken to coordinate the decentralized control agents in the lower control layer such that negative effects of this voltage instability are minimized and such that the induced transients that drive the system to collapse or cause unwanted and hazardous sustained oscillations are avoided.

Traditionally, off ine static stability studies are carried out in order to avert the occurrence of voltage instability. The approach we propose in this section is an application of *online* control that takes into account both the inherent temporal dynamics and that determines the most appropriate control sequence required to reach an acceptable and secure operating point. We therefore propose the use of the MPC schemes discussed in Section 4.3 by a medium-layer control agent to determine the set-points for lower-layer control agents in such a way that negative effects due to voltage instability after faults are minimized.

In the following we describe the power network and control setup, formulate MPC problems based on an object-oriented and a linearized prediction model, and experimentally assess the performance of the medium-layer control agent using these MPC formulations.

4.4.1 The 9-bus dynamic benchmark network

We perform simulation studies on a 9-bus power network. Figure 4.3 shows the topology of the physical network. This system is an adjusted version of the 9-bus Anderson-Farmer network [46], taken from the Dynamical Systems Benchmark Library [63]. The following list contains more details on the dynamics of the network:

- Synchronous machines: The network consists of 4 synchronous machines G_1, G_2, G_3 and G_4. The synchronous machines are connected to the network via lossless step-up transformers featuring a f xed turns ratio. Synchronous machines G_2 and G_3 represent single physical unit, whereas synchronous machines G_1 and G_4 denote aggregate machines comprising several physical units. The mechanical power and the level of the excitation f eld can be adjusted for each machine.

- Loads: There are 5 loads, L_5, L_6, L_7, L_8, L_9. Part of the loads can be disabled by using load shedding.

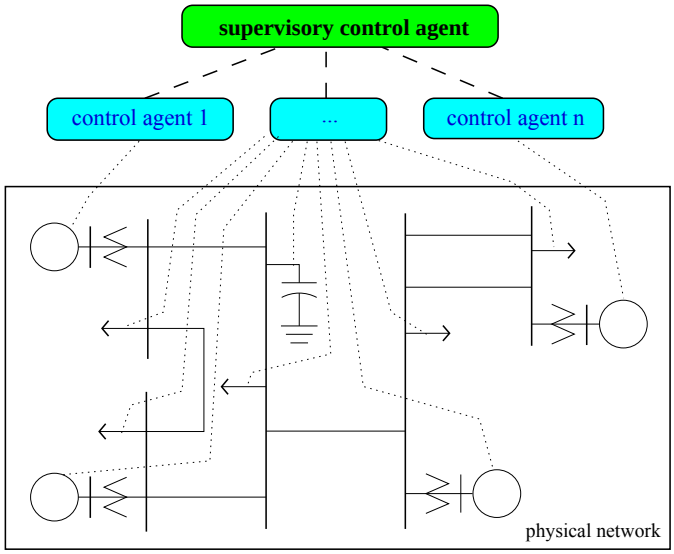


Figure 4.4: Illustration of the control of the power network.

- Capacitor bank: A capacitor bank C at bus 7 provides additional reactive power to the network to locally stabilize bus voltage magnitudes. Capacitors can be connected or disconnected from the network in discrete quantities.

- Transmission lines: The transmission lines between the buses and components transfer the power from one location to another.

Note that this power network contains very fast dynamics, due to the transmission lines, fast dynamics, due to the generators, and slow dynamics, due to the loads. Control of the physical network is done through two-layered control, consisting of a lower, primary, control layer, and a medium, secondary, control layer. Figure 4.4 illustrates this control structure.

Lower control layer

The lower control layer in the network regulates power f ows and voltage levels at the bus terminals of generators. The lower control layer consists of the following elements:

- Turbine governors: All generators feature a turbine governor controlling the mechanical power acting on the shaft of the machine in order to satisfy the active power demand of the network and maintain a desired frequency. The turbine governors act on a time scale of tens of seconds. The turbine governors accept set-points for the mechanical power and frequency.

- Automatic voltage regulators: All generators feature an automatic voltage regulator (AVR) maintaining the level of the excitation f eld in the rotor windings at the value

required to keep the bus voltage magnitude close to the desired set-point. The maximum current in the excitation system is limited. Once a machine has reached one of its limits it cannot produce additional reactive power and can therefore no longer participate in sustaining the voltage magnitudes in the network [82]. The AVR voltage references of the generators can be set in the range 0.9–1.1 p.u. The AVRs act on a time scale of seconds. The AVRs accept set-points for the voltage magnitudes of the generators' terminal buses.

- Power system stabilizers: Generators G_2 and G_3 feature a power system stabilizer (PSS) eliminating the presence of unwanted rotor oscillations by measuring the rotational speed and adding a corrective factor to the voltage magnitude reference for the AVRs. The corrective factor saturates at a lower and upper bound. Generators G_1 and G_4 feature no power system stabilizer since these generators represent multiple physical generators. The PSSs act on a time scale of tenths of seconds. The PSSs accept set-points for the frequency.

Control handles available to a medium control layer

Given the description of the network and the lower control layer, the control handles available to a higher control layer in the form of set-point and reference settings are summarized as follows:

- the voltage references for the AVRs;
- the mechanical power set-points for the turbine governors;
- the reference frequency for the turbine governors and the PSSs;
- the amount of load to shed;
- the amount of capacitor banks to connect to the grid.

Depending on the particular control problem a higher-layer control agent will adjust the values of these control handles. In particular for the voltage control problem at hand the amount of load shed and the set points of the AVRs will be taken as the available control handles.

4.4.2 Object-oriented model of the network

To construct an object-oriented model of the network, we first define several classes to describe the components in the power network. Using the definition of the classes we formalize the structure of the network. To each class we assign a set of variables and a set of equations, typically consisting of a system of DAEs. The equations of a class constrain the values of variables over time, and therefore add to the behavior of the object-oriented model. The equations of a particular class first of all typically constrain variables of that particular class. In addition, the equations of a particular class can also constrain the variables in classes from which that particular class is a subclass. After having defined the classes and the associated constraints, the classes can be instantiated into objects to form the object-oriented representation of the 9-bus network.

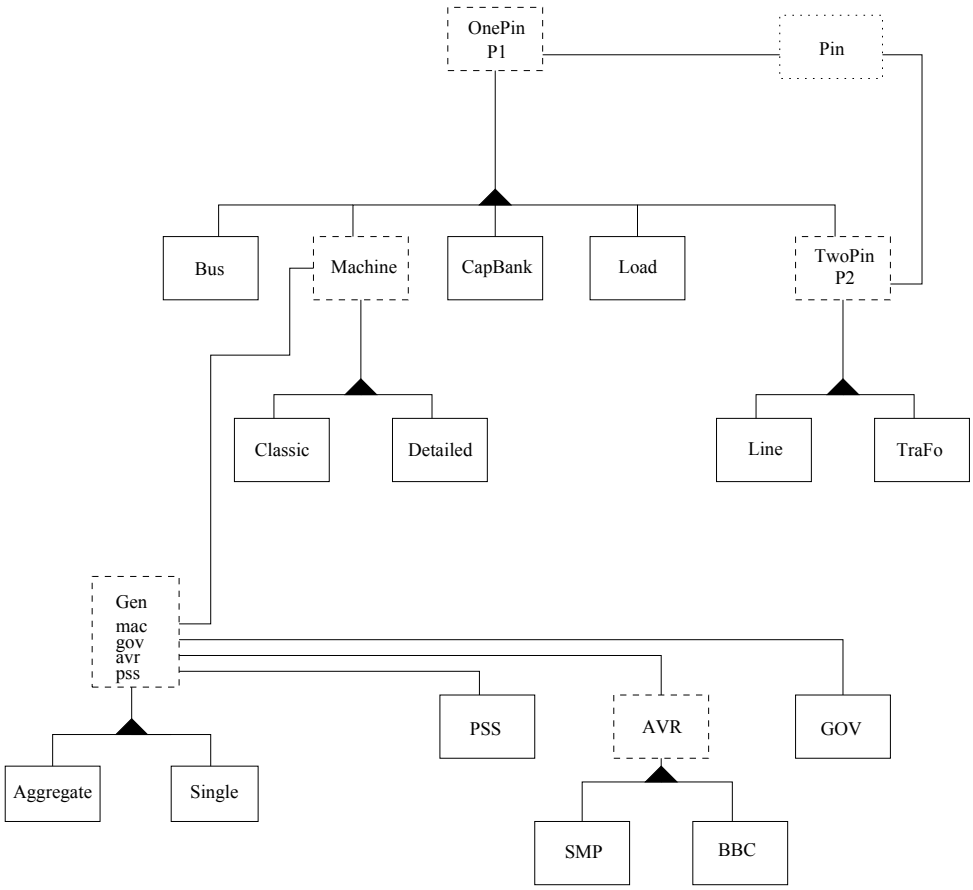


Figure 4.5: Class diagram for a power network.

Class definitions

Figure 4.5 shows the class diagram of the components that we consider. Below we motivate the def nition of the classes shown in the f gure.

The components in the physical network all have in common that they are connected to other components. To model this, the connector class [39] *Pin* is def ned. The *Pin* connector class def nes variables for the voltage magnitude and angle, and the current magnitude and angle. No additional constraints on the values of these variables are def ned, however when two components are connected to each other through the *Pin* connector class, four constraints are def ned that force the voltage magnitudes and angles of both components to be equal, and that force the sum of the current magnitudes and angles of both components to be zero.

Components like buses, machines, loads, and capacitor banks are connected to the network at one point. We therefore def ne the basic class *OnePin*. This class has a single variable $P1$, which refers to an object of class *Pin*, and has no further additional constraints. Components like transmission lines and transformers are connected to the network at two

points. Therefore, we def ne the class *TwoPin* as an extension of the *OnePin* class. This class has a single variable $P2$, which refers to an object of class *Pin*. Note that by extending the *OnePin* class, the *TwoPin* class inherits access to the variables of the *OnePin* class, hence it inherits access to variable $P1$ and the *Pin* class variables associated with this variable.

Since no additional constraints are def ned in the *OnePin* and *TwoPin* classes, the values for the variables involved in either of the two classes, consisting of the four variables of the $P1$ variable, cannot be determined. These classes are therefore *partial* classes [39]. Subclasses of partial classes have to be def ned containing constraints to give these variables values.

Subclasses of the class *OnePin* represent those components in the network that are connected at a single point, i.e., buses, machines, loads, and capacitor banks. Therefore, classes *Bus*, *Machine*, *Load*, and *CapBank* are def ned as subclasses of class *OnePin*. There are different types of machines and we therefore def ne as subclasses of the class *Machine* the classes *Classic* and *Detailed*. The following lists the most important characteristics of the dynamics associated with these classes, and the variables that the classes expect as control inputs or provide as outputs:

- The *Bus* class involves two constraints that force the current magnitude and angle of the pin of the bus to be zero.
- The *Classic* class is equipped with classical 2nd-order mechanical dynamics [63, 82]. The dynamics of this machine depend on the level of f eld voltage $u_E(t)$ and mechanical power $u_{\mathrm{Pm}}(t)$. The value of the voltage magnitude $y_V(t)$ of the bus of the machine and the frequency deviation $y_{\Delta\omega}(t)$ are made available to other classes.
- The *Detailed* class is equipped with a detailed 6th-order model [63, 82] including the mechanical equations and the electrical transient and sub-transient dynamics of the machine, since it represents a single physical unit. The variables that the dynamics depend on are the same as for the *Classic* class. Also the values that are available to other classes are the same.
- If the original benchmark def nition would be used, the *Load* class would be equipped with a static voltage dependent and constant impedance load model [76]. However, to model the loads in more detail and to obtain slow load dynamics, a 2nd-order ZIP model [59] is assigned to the *Load* class. Among others, two constraints are included describing the relation between the current angle and magnitude and the voltage angle and magnitude under different amounts of active and reactive power consumption. The class accepts as input the amount of load to shed $u_{\mathrm{shed}}(t)$.
- The *CapBank* class is equipped with two static constraints relating the number of capacitors $u_{\mathrm{cap}}(t)$ connected to the power network to the current magnitude and angle and the voltage magnitude and angle of its pin [63]. The class accepts as input the number of capacitors $u_{\mathrm{cap}}(t)$ to connect to the network. This input is a variable that can take on only discrete values.

Subclasses of the class *TwoBus* represent those components in the network that are connected to two buses, i.e., transmission lines and transformers. Therefore, classes *Line* and *TraFo* are def ned as subclasses of class *TwoPin*. The most important characteristics of the

dynamics associated with these classes and the variables that the classes expect as control inputs or provide as outputs are:

- The *Line* class is equipped with the static equations of the π model for transmission lines [63, 82]. Four constraints relate the eight variables of the two pins.
- The *TraFo* class is equipped with the static equations of the π model for transmission lines [63, 82], but with the resistance and susceptance set to zero. Four constraints relate the eight variables of the two pins.

Each of the classes def ned so far contains variables related to the particular component being modeled, i.e., input, state, algebraic, and output variables, and equations describing the behavior of the components. Moreover, each of these classes def nes constraints involving the variables of the *OnePin* or *TwoPin* class.

There are also several components that do not directly connect to the power network, and that therefore are not def ned as a subclass of the *OnePin* or *TwoPin* class. These components consist, e.g., of the components in the lower control layer, which determine the inputs to components directly connected to the power network. Examples of these are the AVRs, turbine governors, and possibly PSSs. Corresponding classes *AVR*, *GOV*, and *PSS* are therefore def ned. For the class *AVR* subclasses *SMP* and *BBC* are def ned, depicting two different types of AVRs. The most important characteristics of the dynamics of these classes, and the variables that the classes expect as control inputs or provide as outputs, are the following:

- The *SMP* class is equipped with the equations of a 3rd-order AVR [63, 82]. The *SMP* class accepts as inputs a bus voltage magnitude $u_{V,\mathrm{mac}}(t)$ of the bus of which the AVR should regulate the voltage magnitude, and a voltage magnitude set-point $u_{V,\mathrm{PSS}}(t)$. In addition, the *SMP* class accepts as input voltage magnitude set-point $u_{V,\mathrm{ref}}(t)$. The *SMP* class provides the excitation f eld voltage $y_E(t)$ as output. The excitation f eld voltage $y_E(t)$ saturates at a lower limit $y_{E,\min}$ and an upper limit $y_{E,\max}$.
- The *BBC* class is equipped with 2nd-order dynamics [63, 82]. This class has the same inputs and outputs as the *SMP* class. Also this AVR class considers saturation of the excitation f eld voltage $y_E(t)$.
- The *GOV* class is equipped with 3rd-order dynamics [63, 82]. The dynamics have as input a frequency deviation $u_\omega(t)$ of a machine. The *GOV* class accepts $u_{\mathrm{Torder}}(t)$ as set-point for the mechanical power. The class provides mechanical power $y_{\mathrm{Pm}}(t)$ as output.
- The *PSS* class is equipped with 3rd-order dynamics [63, 82]. It uses as input a frequency deviation $u_\omega(t)$ to determine a voltage magnitude $y_{V,\mathrm{PSS}}(t)$. The voltage magnitude $y_{V,\mathrm{PSS}}(t)$ saturates at upper bound $y_{V,\mathrm{PSS},max}$ and lower bound $\eta_{V,\mathrm{PSS},\min}$.

Having def ned the classes for these individual components, it is convenient to def ne some classes by composition. E.g., the class *Gen* is def ned as the composition of a machine with a specif c lower control conf guration. As subclasses we def ne the classes *Aggregate* and *Single*. The classes *Aggregate* and *Single* include references to specif c *AVR*, *GOV*, and *PSS* classes.

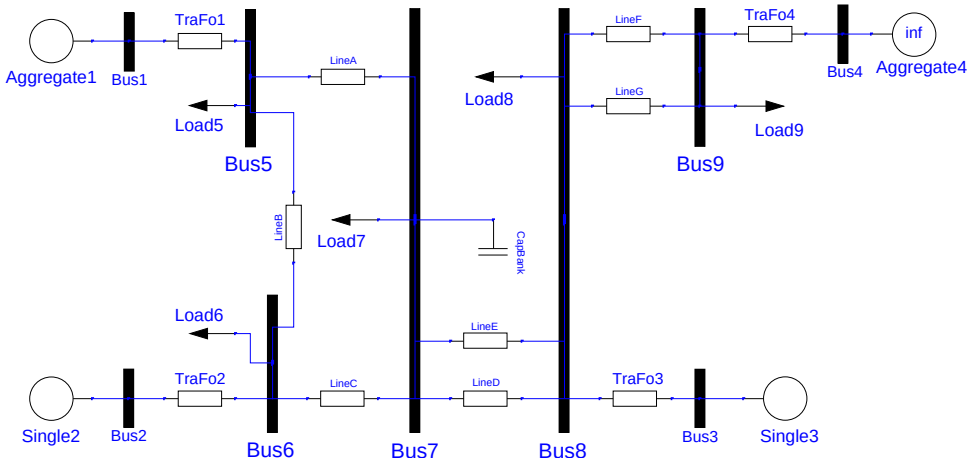


Figure 4.6: The 9-bus power network as object diagram.

Remark 4.4 In the example of the division of the power network into classes and subclasses given here only the components that will be used later have been def ned. It is straightforward, however, to include many more subclasses, e.g., for describing different loads, transformers, and additional generators. In [105] several examples of additional components that can be added can be found. The classif cation of components into classes facilitates easy experimenting with models with different levels of detail. □

Object diagram

Given the classes and the associated dynamics, we can now instantiate the classes into objects to form an object diagram for the power network under study. Generators G_1 and G_4 are of class *Aggregate*. Generators G_2 and G_3 are of class *Single*. The loads are of class *Load*. The capacitor bank is of class *CapBank*. The buses are of class *Bus*. The transmission lines are of class *Line*, and the transformers are of class *TraFo*. Figure 4.6 shows the layout of the resulting object diagram as created in Dymola. The Dymola model can be obtained from the author on request.

4.4.3 Control problem formulation for the higher control layer

To illustrate the control problem, we consider a typical scenario with no medium-layer MPC control agent installed, in which we use the model constructed in the previous section as model of the physical network. In the scenario that we consider, the network is initially in steady state. Then, at $t_{\text{fault}} = 26.5$ s a fault of 600% impedance increase in the transformer between bus 1 and 5 occurs. Figure 4.7 illustrates the evolution of the voltage magnitudes of three representative buses. The fault occurring in the transformer between bus 1 and 5 changes the transmission capacity of the network. Due to the changed transmission capacity of the network and due to the dynamics of the loads, the voltage magnitudes start to start oscillating, despite the actions of the lower control layer. If the set-points to the control agents

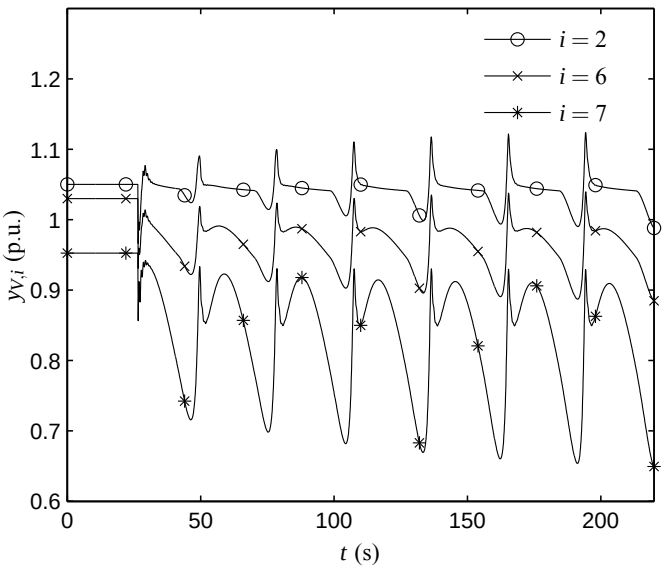


Figure 4.7: Voltage magnitude profiles $y_{V,i}$ of three representative buses, ($i = 2$, $i = 6$, and $i = 7$), for a typical scenario in which no medium control layer is present. After a fault of 600% impedance increase occurs in the transformer between buses 1 and 5 at $t_{fault} = 26.5$ s, the voltage magnitudes $y_{V,i}$ start oscillating, ultimately resulting in a network collapse.

of the lower control layer are not changed, perhaps in combination with other measures, the network ultimately collapses.

To prevent such a collapse from occurring, a higher-layer control agent should be installed with the task to [49]:

1. Maintain the voltage magnitudes between 0.9 and 1.1 p.u., i.e., sufficiently close to nominal values to ensure a safe operation of the system by keeping the voltage magnitudes sufficiently distant from low voltages.

2. Effectively achieve a steady-state point of operation, while minimizing changing of the control inputs so that a constant and appropriate set of input values is ultimately applied to the power network and the lower control layer.

For the second objective, in particular the option of shedding load is to be avoided unless absolutely necessary in order to fulfill the primary objective, as load shedding is the most disruptive countermeasure available. Since typical slow voltage collapses without a medium-layer control agent installed emerge over time spans of several tens of seconds up to several minutes [142], a control cycle time of 20 s is acceptable. It should be noted that the speed at which a voltage collapse unfolds depends on the magnitude of the fault occurring. A collapse will take place sooner with a larger fault than with a smaller fault. So, depending on the range of faults that should be adequately dealt with, the control cycle time will have to be decreased or increased.

Below we formulate the nonlinear and linear higher-layer MPC problem of the 9-bus power network, and we assess the performance of the resulting closed-loop control structure in experiments.

Nonlinear MPC problem formulation

The control problem of the supervisory control agent using the object-oriented prediction model is based on the formulation specif ed in Section 4.3.1. For $l = 0,\ldots,N_c-1$, the inputs $\tilde{\mathbf{u}}(k_c+l)$ correspond to the AVR set-points $u_{\mathrm{AVR},i}(k_c+l)$, for $i = \{1,\ldots,4\}$, and the amounts of load to shed $u_{\mathrm{shed},i}(k_c+l)$, for $i = \{5,\ldots,9\}$. For $l = 0,\ldots,N_p-1$, the outputs $\tilde{\mathbf{y}}(k_p+l)$ correspond to the voltage magnitudes $y_{V,i}(k_p)$, for $i = \{1,\ldots,9\}$, at the 9 buses.

One control cycle takes 20 s, hence T_c is 20 s. Although in principle a the prediction horizon should include all important dynamics, for computational reasons a prediction horizon with a length of only 2 control cycles is taken. The continuous voltage signal is sampled every 0.5 s, hence T_p is 0.5 s, and the length of the prediction horizon N_p is therefore 40 prediction steps.

The MPC control problem is formulated as in (4.15)–(4.16), where $y_{q,\mathrm{desired,min}}$ is 0.9 p.u. and $y_{q,\mathrm{desired,max}}$ is 1.1 p.u. for each element of $\tilde{\mathbf{y}}(k_p+l)$. The elements of $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$ corresponding to AVR settings $u_{\mathrm{AVR},i}(k_c+l)$ are set to 0.9 and 1.1 p.u., respectively. The elements of $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$ corresponding to load settings $u_{\mathrm{shed},i}(k_c+l)$ are set to 0 and 1, respectively, corresponding to full load shedding or no load shedding, respectively.

The cost matrix $\mathbf{Q_y}$ contains on its diagonal elements $\frac{1}{N_p/N_c}200$ and $\mathbf{Q_u}$ contains the value 1 on the diagonal elements corresponding to AVR settings $u_{\mathrm{AVR},i}(k_c+l)$, and the value 20 on the diagonal elements corresponding to load shedding settings $u_{\mathrm{shed},i}(k_c+l)$. This way of penalizing the voltage bound violations, the AVR settings, and the load shedding settings ensures that the main objective of the control agent is to satisfy the voltage objectives, and that load shedding should only be chosen as a last resort.

Linear MPC problem formulation

The control problem of the supervisory control agent using the linearized prediction model is based on the formulation given in Section 4.3.3. The MPC control is formulated using (4.17)–(4.18). The length of the prediction horizon in prediction steps N_p is 40, and the length of the prediction horizon in control cycles N_c is 2. The inputs $\tilde{\mathbf{u}}(k_c)$ correspond to the set-points for the AVRs $u_{\mathrm{AVR},i}(k_c+l)$ and the amounts of load to shed $u_{\mathrm{shed},i}(k_c+l)$ over the prediction horizon. The outputs $\tilde{\mathbf{y}}(k_p)$ correspond to the voltage magnitudes $y_{V,i}(k_p+l)$ at the 9 buses.

Similar as for the nonlinear MPC formulation, the cost matrices $\mathbf{Q_y}$ and $\mathbf{Q_u}$ are def ned such that a weight of $\frac{1}{N_p/N_c}200$ is placed on the violation of each soft constraint. The inputs are weighted with the penalty coeff cients 1 and 20 for the AVR settings $u_{\mathrm{AVR},i}(k_c)$ and the load shedding settings $u_{\mathrm{shed},i}(k_c)$, respectively.

The linearized prediction model is obtained at each control cycle k_c by linearizing the object-oriented prediction model around the current state $\mathbf{x}(k_c)$ and the inputs applied at the preceding time instant $\mathbf{u}(k_c-1)$. The sampling interval $T_p = 0.5$ s.

In the following we f rst focus on the performance of the control agent when it uses

the nonlinear MPC formulation. We illustrate the difference between pattern search and gradient-based optimization methods, and illustrate how the proposed approach chooses adequate set points that prevent the network from collapsing. Then, we also consider the performance of the control agent, when it uses the linear MPC formulation. We illustrate how the two strategies compare.

4.4.4 Control using the nonlinear MPC formulation

Direct search versus gradient-based optimization

We compare pattern search as part of Matlab's Direct Search and Genetic Algorithms toolbox in Matlab v7.3 [97] with the derivative-based solver SNOPT v5.8 [50], as implemented in Tomlab v5.7 [65], and accessed from Matlab. SNOPT uses a sparse sequential quadratic programming method, using limited-memory quasi-Newton approximations to the Hessian of the Lagrange. In principle it requires gradient information, but this information can be approximated numerically if it is not available.

To compare the performance of the solvers, we perform 50 experiments in which a single fault occurs at varying locations in the power network (i.e., at the 4 transformers and the lines), with varying magnitudes (i.e., an impedance increase of 100% up to 800%), and at varying times (i.e., the fault time varies between second 20 and 28). The control problems of the first control cycle after a fault has been applied are solved by both pattern search and SNOPT, allowing a decision making time of 300 s[1].

In Figure 4.8 we see that SNOPT considers far more initial solutions within the given time span. The time that SNOPT requires to obtain a locally optimal solution is much lower than the time required by pattern search. This is explained by the fact that SNOPT uses much fewer prediction model evaluations per optimization, since it does not explore the search space as much as pattern search does.

Figure 4.9 shows, as decision time progresses, the average over all experiments of the best objective value of pattern search so far divided by the best objective value of SNOPT so far. This fraction is 1, if the best objective values of pattern search and SNOPT are on average the same. It is larger than 1, if SNOPT on average has a better solution, and smaller than 1 if pattern search has a better solution on average. The figure considers only points for which the fraction can be computed, i.e., both pattern search and SNOPT have finished at least one optimization problem. We observe that pattern search on average has a best objective value so far that is about a factor 5 smaller than the best objective value so far of SNOPT.

The comparison shows that pattern search, although it does not require gradient or Hessian information and is straightforward to implement, generally provides solutions that outperform the solutions provided by SNOPT.

[1] This relatively long decision making time is taken to illustrate how the performance of both solvers varies over time. In practice, multiple processors can be employed to parallelize the multi-start approach and to obtain acceptable solutions in a more realistic time frame. In addition all code can be optimized for speed and implemented in object code (currently only the SNOPT code is in object code). This is in particular important for the objective function evaluations, since these consume the most significant part of the computation time.

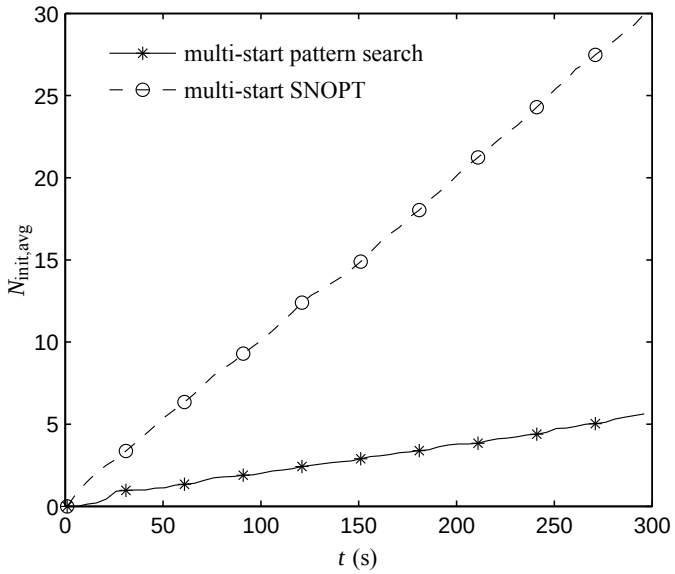


Figure 4.8: Average $N_{init,avg}$ of accumulated number of initial solutions considered by the solvers as decision time t progresses.

Control for a single scenario

To illustrate the performance of the medium-layer control agent using the nonlinear MPC formulation, we now discuss a single scenario. We reconsider the fault of 600% impedance increase at $t_{\text{fault}} = 26.5$ s in the transformer in the line from bus 1 to 5. Figure 4.7 shows the evolution of three representative buses when no medium-layer control agent is installed. We now consider using a supevisory control agent that uses the nonlinear MPC formulation. The supervisory control agent operates at $T_{\text{c}} = 20$ s using multi-start pattern search as discussed before to solve the nonlinear MPC problem. The supervisory control agent uses a prediction horizon with a length of of 40 s, and samples the voltage magnitudes from its prediction model every 0.5 s.

Figure 4.10 shows the resulting voltage magnitude prof les and Figure 4.11 shows the chosen set-points. It should be noted that the load shedding set-point is scaled to take on values between 0 and 50, corresponding to 100% load shedding and no load shedding, respectively, and that the AVR set-points for the automatic voltage regulators are scaled to take on values between 0 and 20, corresponding to 0.9 p.u. and 1.1 p.u., respectively.

After the fault has appeared, the control agent is able to stabilize the voltage magnitude between 0.9 and 1.1 p.u. with a low number of set-point changes and thus achieves its objectives. The control agent obtains a total performance[2] of 98.7, and it takes the control agent in total 157.4 s to determine its control actions.

[2]The total performance is obtained by evaluating the nonlinear objective function over the full day with $T_{\text{p}} = 0.1$ s.

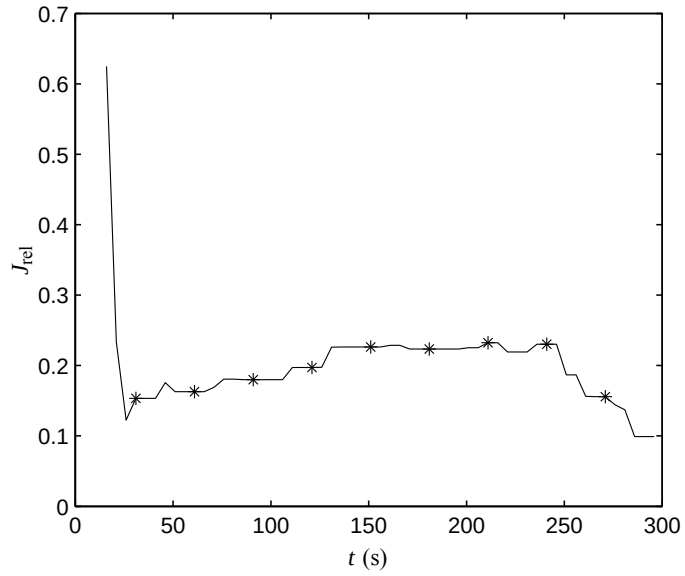


Figure 4.9: Average relative performance J_{rel} of pattern search compared to SNOPT over all experiments. The average relative performance J_{rel} at a particular time t is computated as follows: best objective value of pattern search so far divided by best objective value of SNOPT so far, averaged over all experiments.

4.4.5 Control using the linear MPC formulation

As alternative to solving the optimization problem using pattern search, we now use the linear MPC problem formulation. To solve the linear programming problems at each control cycle, we use the ILOG CPLEX v10 linear programming problem solver [71], which we access through the Tomlab v5.7 [66] interface in Matlab v7.3 [98].

We consider the following scenario. The network is in steady state, when at $t_{\text{fault}} = 26.5$ a fault appears, which increases the impedance in the transformer between buses 1 and 5 with 600%. The medium-layer control agent again operates at $T_{\text{c}} = 20$ s, and uses the linear MPC formulation with a prediction horizon with a length of 40 s, while sampling the voltage magnitudes every 0.5 s.

Figures 4.12 and 4.12 show the evolution of the voltages over the simulation and the set-points chosen by the control agent, respectively. We observe that the control agent can determine actions that stabilize the voltages at acceptable levels, despite the linearized approximation that the control agents uses for the prediction model. The control agent using the linear MPC formulation obtains a total performance of 142.4, and it takes the control agent in total 26.3 s to determine its control actions. Hence, although the control agent does not obtain an improved performance when compared to the control agent using the nonlinear MPC formulation, it does achieve stabilizing the voltage magnitudes using signif cantly fewer computation time.

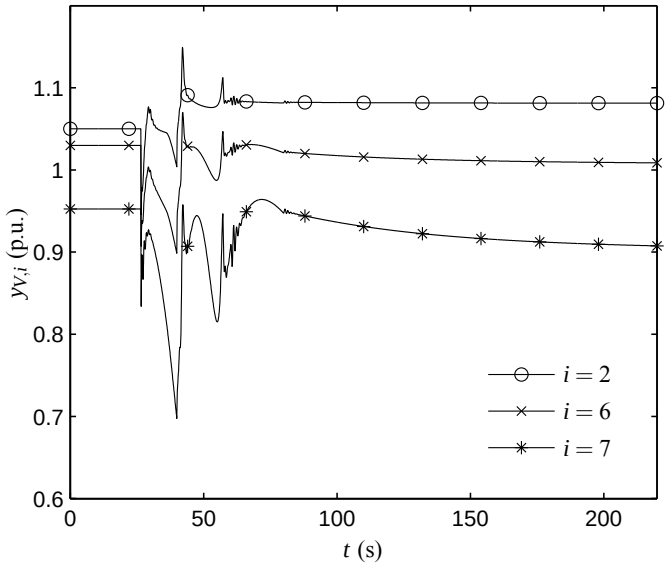


Figure 4.10: Voltage magnitude profiles for simulation including a medium-layer nonlinear MPC control agent.

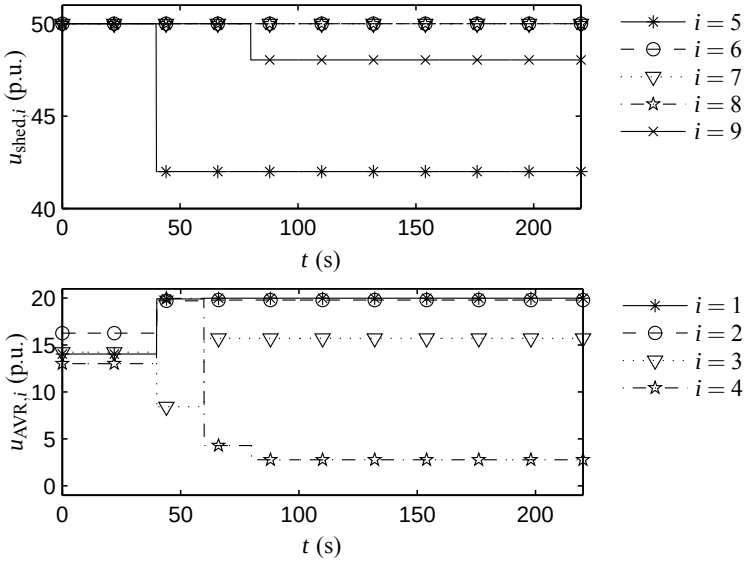


Figure 4.11: Set-points provided by the supervisory control agent for simulation including the nonlinear medium-layer MPC control agent. Load shedding values are scaled to lie within 0 and 50. AVR set-points are scaled to lie within 0 and 20.

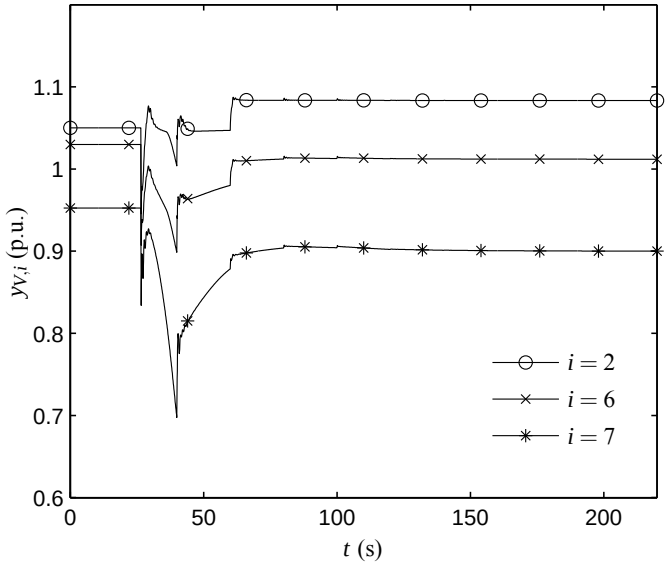


Figure 4.12: Voltage magnitude profiles for controlled simulation using the linear MPC formulation.

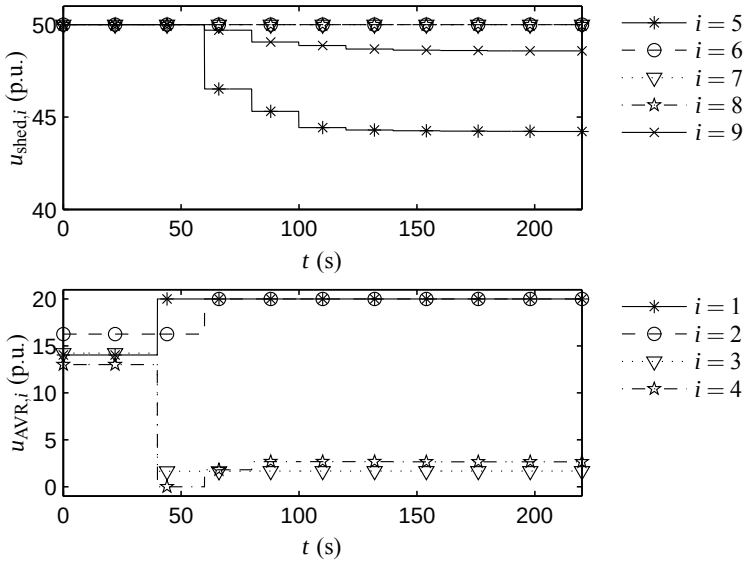


Figure 4.13: Set-points provided by the control agent using the linear MPC formulation for controlled simulation. Load shedding values are scaled to lie within 0 and 50. AVR set-points are scaled to lie within 0 and 20.

4.5 Summary

In this chapter we have discussed MPC in multi-layer control. In particular we have focused on issues related to the model that a medium-layer MPC control agent uses and discussed why object-oriented modeling is suitable for this. We have proposed an MPC strategy in which the prediction model is formulated either as an object-oriented model, allowing relatively easy construction of models of complex systems, or as a linearized approximation of such a model, allowing the use of eff cient optimization problem solvers. Due to the nature of power networks, the object-oriented prediction model involves differential, algebraic, and logic relations and is nonlinear, non-smooth, and costly to evaluate.

To solve the nonlinear MPC problem of the medium-layer control agent using the constructed prediction model, we have proposed to use pattern search as optimization method. Pattern search is a direct-optimization method that does not compute or approximate gradients and/or Hessians, which are not available in analytical form in the situation considered. Moreover, due to the discrete elements, e.g., saturation, the MPC optimization problem is non-smooth, making approaches using gradient or Hessian information less suitable.

We have applied the proposed control strategy for the control agent in a medium control layer of a power network. The medium-layer control agent provides set-points to a lower control layer with the aim of preventing voltage collapses from occurring. Simulation studies on a 9-bus dynamic power network have shown the potential of the proposed approaches. For the MPC formulation based on the object-oriented model, we have illustrated the difference in performance between a gradient-based and the pattern search method and we have shown that the voltage collapses can be prevented from occurring. For the MPC problem based on the linearized model, we have compared the performance of the control for a specif c example with the performance obtained by the MPC control agent using the original model. We have observed that the MPC control agent using the linearized prediction model can determine set-points that stabilize the voltage magnitudes, despite the linearized model used. Although the control actions that the MPC control agent using the linearized model chooses result in higher costs than the actions that the original MPC control agent would choose, the total computation time is signif cantly lower for the MPC control agent using the linearized model. It is therefore interesting to investigate further what the performance loss is due to the linearization, and for which type of disturbances the control agent using the linearized model can yield good performance.

Chapter 5

Overlapping subnetworks

In Chapter 4 we have considered the control of several control agents in a lower control layer by a single control agent in a medium control layer. The control agent in the medium control layer has used a prediction model including both the behavior of the lower control layer and the physical network. In this chapter we consider control by multiple control agents in a higher control layer. The control agents assume that the dynamics of the lower control layers and the physical network are instantaneous. We focus on the question of how nodes of a network should be assigned to subnetworks. In Chapters 2 and 3 the subnetworks into which the transportation networks were divided were not overlapping. In this chapter we will discuss how subnetworks can be defined that are overlapping.

We first formalize the way in which we model general transportation networks in this chapter in Section 5.1. We then discuss several approaches for defining subnetworks and the properties of the resulting subnetworks in Section 5.2. In Section 5.3 we focus on a particular approach for defining subnetworks based on the influence that actuators in these subnetworks have. Currently existing approaches for multi-agent control assume that the subnetworks that control agents control are not overlapping. However, as we will see, the influence-based approach might result in subnetworks that are overlapping. To deal with this, in Section 5.4 we first discuss a recently proposed approach that can be used for the higher-layer multi-agent control of subnetworks that are not overlapping, but that do have links among them. We then propose an extension of this approach for application to higher-layer multi-agent control of subnetworks that are overlapping in Section 5.5.

In this chapter we consider as application optimal power flow control of large power networks. In particular, in Section 5.6 we apply the approach for overlapping subnetworks to an optimal power flow control problem using FACTS devices, in which each FACTS device is controlled by a different control agent. Experiments are carried out on an adjusted IEEE 57-bus power network.

Parts of this chapter have been published in [69].

5.1 Steady-state models of transportation networks

As explained in Chapter 1, in a transportation network there is some commodity flowing through the network over links between nodes inside the network. The nodes can be ar-

ranged in a topology to ref ect how the elements inside the network are connected to each other. Depending on the f ows of the commodity in the network, the values of the variables associated with the nodes, e.g., pressures, speeds, etc., take on different values. By changing the values of actuators that are located in the network, the f ows and, hence, the values of the variables can be changed. Control agents are used to determine how the values of the actuators should be changed in order to achieve desired behavior, which is directly related to desired values for the variables associated with the nodes inside the network.

In Chapter 4, we have discussed multi-layer control, and made a distinction between lower, medium, and higher control layers, as depicted in Figure 4.1. In that chapter, we have in particular considered control of an individual medium-layer control agent, that uses a model of the dynamics of the lower control layer and physical network. Here we consider the control of multiple control agents in a higher control layer. The control agents in this higher control layer are interested in controlling the very slow dynamics or the long term behavior, and therefore assume that dynamics of the lower control layers and physical network can be represented by instantaneous dynamics. Therefore, the control agents in the higher control layer consider only steady-state characteristics, i.e., the characteristics of the lower control layers and the network when transients have faded out and the network has settled in a steady state, e.g., after a change in the settings of an actuator.

To model the steady-state characteristics, each of the nodes in the network has associated with it variables and constraints used to compute the steady-state values for these variables, given values for actuator settings and exogenous inputs. Let the network consist of ν nodes, and let ι, for $\iota \in \{1,\ldots,\nu\}$ denote a particular node. The constraints of a particular node ι involve variables of that particular node ι and possibly variables of the nodes of neighboring nodes $\omega \in \mathcal{N}^\iota$, where $\mathcal{N}^\iota = \{\omega_{\iota,1},\ldots,\omega_{\iota,n_{\mathcal{N}^\iota}}\}$ is the set of neighboring nodes of node ι. The set of neighboring nodes $\mathcal{N}^\iota$ of node ι contains those nodes that can be reached from node ι by going over one link in the topology.

Let for node $\iota \in \{1,\ldots,\nu\}$, the variables $\mathbf{z}^\iota \in \mathbb{R}^{n_{\mathbf{z}^\iota}}$, $\mathbf{u}^\iota \in \mathbb{R}^{n_{\mathbf{u}^\iota}}$, and $\mathbf{d}^\iota \in \mathbb{R}^{n_{\mathbf{d}^\iota}}$, denote the algebraic[1], the input, and the exogenous input variables associated with node ι, respectively, and let the constraints of node ι be given by:

$$0 = \mathbf{g}^\iota(\mathbf{z}^\iota, \mathbf{u}^\iota, \mathbf{d}^\iota, \mathbf{z}^{\omega_{\iota,1}}, \ldots, \mathbf{z}^{\omega_{\iota,n_{\mathcal{N}^\iota}}}) \tag{5.1}$$

where $\mathbf{z}^\omega$ are the variables of neighboring node $\omega \in \mathcal{N}^\iota$, and $\mathbf{g}^\iota$ are smooth constraint functions of node ι. A steady-state model for the overall network is obtained by aggregating the constraints (5.1) for all nodes $\iota \in \{1,\ldots,\nu\}$, and is compactly represented as:

$$0 = \mathbf{g}(\mathbf{z}, \mathbf{u}, \mathbf{d}), \tag{5.2}$$

where $\mathbf{z}$, $\mathbf{u}$, and $\mathbf{d}$ are the algebraic, input, and exogenous input variables of the overall network, and $\mathbf{g}$ def nes the steady-state characteristics of the network. Given the inputs $\mathbf{u}$ and the exogenous inputs $\mathbf{d}$, the steady state in which the network settles is determined by solving the system of equations (5.2).

Assume that there are multiple control agents, with the objective to reach overall network objectives, like safety and security. With each node a number of objective terms can be associated. These objective terms are used to indicate which behavior is desired for the

[1] Sometimes the algebraic variables are also referred to as static states.

variables $\mathbf{z}^\iota$ and $\mathbf{u}^\iota$ of that node. The terms involve the variables of node ι and possibly the variables of the neighboring nodes $\omega \in \mathcal{N}^\iota$. The aggregation of the objectives terms of each node gives the objective for the control of the overall network.

The nodes that a control agent considers in its decision making form the subnetwork of that control agent. Given that each control agent has access to a particular actuator, the issue that we address below is how to determine which nodes of the overall network a control agent should consider, i.e., how should the subnetwork of a control agent be determined.

5.2 Subnetworks and their properties

We first introduce some properties of subnetworks, and then we discuss different approaches for defining subnetworks and the properties of the resulting subnetworks.

5.2.1 Properties of subnetworks

We make distinctions among *non-overlapping*, *touching*, and *overlapping* subnetworks. If for each subnetwork, the nodes belonging to that subnetwork do not coincide with the nodes belonging to any other subnetwork, and if there are no links going from nodes in one subnetwork into nodes of another subnetwork, then the subnetworks are non-overlapping. If for each subnetwork, the nodes belonging to that subnetwork do not coincide with the nodes of any other subnetwork, but if there are links between nodes of one subnetwork and nodes of another subnetwork, then the subnetworks are touching. If the nodes belonging to some subnetworks partially coincide with the nodes belonging to other subnetworks, then the subnetworks are overlapping. In that case, *common* subnetworks of particular subnetworks are defined as the subnetworks consisting of those nodes that belong to each of these particular subnetworks. Figure 5.1 illustrates the different types of subnetwork divisions. Note that it is not strictly necessary that each node is part of a subnetwork.

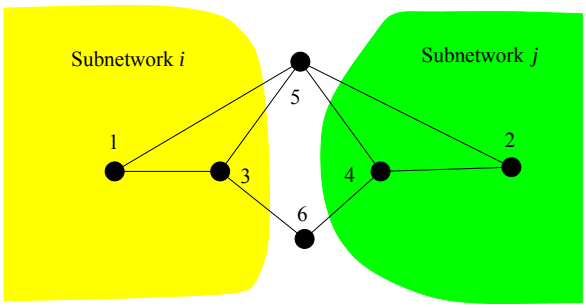
In addition to non-overlapping, touching, and overlapping subnetworks, we make a distinction between *time-invariant* and *time-varying* subnetworks. With a time-invariant subnetwork we refer to a subnetwork of which the assignment of nodes does not change over time. With a time-varying subnetwork we refer to a subnetwork of which the assignment of nodes does change over time.

5.2.2 Defining subnetworks

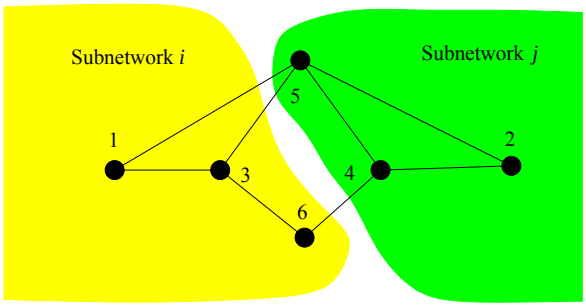
Given an overall transportation network, there are several approaches that can be taken to define subnetworks inside that transportation network, i.e., how to determine which nodes belong to which subnetwork. Some examples of approaches to define subnetworks are the following:

1. Subnetworks can be defined through geographical borders, e.g., of cities, provinces, countries, etc., i.e., based on an existing grouping of nodes.

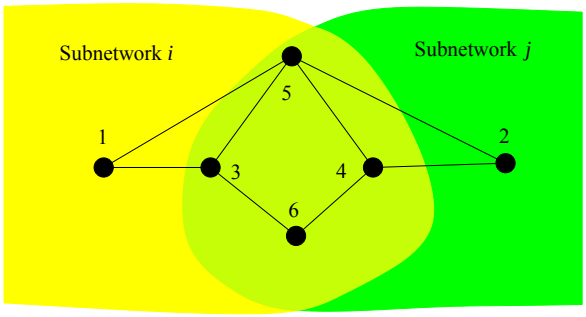
2. Subnetworks can be defined through clustering of nodes into a predefined number of groups, in such a way that the number of interconnections among the resulting subnetworks is minimized.



(a) Non-overlapping subnetworks.



(b) Touching subnetworks.



(c) Overlapping subnetworks.

Figure 5.1: Illustration of different types of subnetworks.

3. Subnetworks can be def ned based on a radius around nodes, i.e., nodes reachable within a certain number of links from a particular main node (e.g., the node with an actuator) are included in a particular subnetwork.

4. Subnetworks can be def ned by including in the subnetwork of a control agent only nodes that can be inf uenced by the actuators of that control agent.

The f rst approach can lead to subnetworks that are non-overlapping, touching, or overlapping. E.g., if the subnetworks are def ned based on city borders, then the subnetworks can be non-overlapping; if the subnetworks are def ned based on country borders, then the subnetworks can be touching; and, if the subnetworks are def ned based on country and

city borders, the subnetworks can be overlapping. In the case that subnetworks are def ned in this way, each subnetwork is typically already controlled by a control authority. The subnetworks resulting from this approach are typically time-invariant, unless wars, city restructuring, breakdowns, etc., are taken into account.

The second approach can lead to non-overlapping or touching subnetworks. If not all nodes of the network are assigned to a subnetwork, then the subnetworks can be non-overlapping. However, if all nodes of the network are assigned to a subnetwork, the subnetworks are touching. Note that using this approach, it may be the case that actuators owned by different control authorities are placed in one subnetwork. The subnetworks resulting from this approach are typically time invariant.

The third approach can lead to non-overlapping, touching, and overlapping subnetworks, depending on the number of nodes that is taken to belong to a particular subnetwork. The underlying idea of considering a radius is that the dynamics topologically far from an actuator are not relevant, since these far away dynamics do not have a signif cant inf uence on the dynamics around the actuator. The resulting subnetwork is typically time invariant.

The fourth approach can also lead to non-overlapping, touching, and overlapping subnetworks. In this approach, f rst it is determined how much the variables of each node can be inf uenced by actuators, and then depending on the inf uence on the nodes it is determined which nodes should be included in a subnetwork. If the inf uence varies over time, then the resulting subnetwork is time-varying. Otherwise it is not.

In the following sections we consider the fourth approach for def ning subnetworks, and discuss how coordination among control agents that control subnetworks def ned in that way can be achieved, in particular when the resulting subnetworks are overlapping.

5.3 Influence-based subnetworks

The idea of inf uence-based subnetworks is that the subnetworks are def ned based on the nodes that a certain actuator and, hence, a control agent controlling that actuator, can inf uence. When the nodes that can be inf uenced have been computed for each actuator, the inf uence-based subnetwork is def ned as the union of these nodes over all actuators of a control agent.

5.3.1 Using sensitivities to determine subnetworks

To determine which dynamics an actuator can inf uence, sensitivities can be used [51]. The sensitivity of a variable z^ω associated with a node $\omega \in \{1,\dots,\nu\}$ in the network with respect to an input u^ι indicates how much the value of variable z^ω changes when the input u^ι changes. Therefore, an input u^ι with respect to which variable z^ω has a high sensitivity, i.e., a sensitivity with an absolute value relatively far from zero, has a large inf uence on the value of variable z^ω, whereas an input u^ι with respect to which the variable z^ω has a low sensitivity, i.e., a sensitivity with an absolute value close to zero, has a low inf uence on the value of the variable z^ω. Knowledge of those variables that have a relatively high sensitivity to the inputs is more important than accurate knowledge of variables that have a relatively low sensitivity. Given the sensitivities, sensitivity thresholding can be used to determine which variables have to be known and which may be neglected. In general, it is to

be expected that variables representing dynamics appearing geographically far from a particular input, will have relatively low sensitivity with respect to that input, when compared to variables representing dynamics in the geographical vicinity of that input.

5.3.2 Computing the sensitivities

To determine the sensitivity of the steady-state characteristics of the network, i.e., the sensitivity of the algebraic variables $\mathbf{z}$ with respect to a particular input u^ι at node ι, consider the constraint functions $\mathbf{g}_{u^\iota}(\mathbf{z}, u^\iota)$, where $\mathbf{g}_{u^\iota}$ are the constraint functions in $\mathbf{g}$ in which all elements of $\mathbf{u}$, except for the element corresponding to u^ι, and all elements of $\mathbf{d}$ have been set to f xed values. Since $0 = \mathbf{g}(\mathbf{z}, \mathbf{u}, \mathbf{d})$, also $0 = \mathbf{g}_{u^\iota}(\mathbf{z}, u^\iota)$. In addition, since $\mathbf{z}$ depends on u^ι, it follows by the chain rule that:

$$0 = \frac{\partial \mathbf{g}_{u^\iota}}{\partial \mathbf{z}}(\mathbf{z}, u^\iota)\frac{\partial \mathbf{z}}{\partial u^\iota}(\mathbf{z}, u^\iota) + \frac{\partial \mathbf{g}_{u^\iota}}{\partial u^\iota}(\mathbf{z}, u^\iota),$$

and therefore:

$$\frac{\partial \mathbf{z}}{\partial u^\iota}(\mathbf{z}, u^\iota) = \left(-\frac{\partial \mathbf{g}_{u^\iota}}{\partial \mathbf{z}}(\mathbf{z}, u^\iota)\right)^{-1}\frac{\partial \mathbf{g}_{u^\iota}}{\partial u^\iota}(\mathbf{z}, u^\iota), \tag{5.3}$$

under the assumption that the inverse term exists. The term $\frac{\partial \mathbf{z}}{\partial u^\iota}(\mathbf{z}, u^\iota)$ is the sensitivity of $\mathbf{z}$ with respect to u^ι. From this sensitivity we can determine which terms of the algebraic variables $\mathbf{z}$ are signif cantly inf uenced by input u^ι. If the absolute value of the sensitivity of a particular element of $\mathbf{z}$ with respect to input u^ι is larger than a sensitivity threshold γ_s, then that element of $\mathbf{z}$ cannot be neglected. The elements of $\mathbf{z}$ that cannot be neglected can be linked to their corresponding nodes, giving a set of nodes that can be signif cantly inf uenced by input u^ι.

The set of nodes that can be inf uenced by an actuator depends on the sensitivity threshold γ_s used. On the one hand, if a sensitivity threshold γ_s of 0 is used, all nodes will be selected. Hence, the subnetwork resulting from this approach will correspond to the full network. On the other hand, if a very large sensitivity threshold γ_s is used, no nodes will be selected, and the subnetwork resulting from this approach will be empty. In Section 5.6.2 we give an illustration of this.

5.3.3 Control of influence-based subnetworks

The settings of the actuators in the network should be adjusted in such a way that the objectives associated with the nodes are achieved as well as possible. Let each actuator be controlled by a control agent, and let the task of each control agent be to determine new set points for its actuators. Control agent i considers as its subnetwork the union of the nodes that can be inf uenced by the actuators that control agent i can control. The prediction model M_i that control agent i considers therefore also consists of the union of the constraints in the inf uence-based models for each actuator that it controls.

Remark 5.1 Since in this chapter we consider only steady-state characteristics, it is not benef cial to formulate the control problem of each control agent in an MPC setting. If we would formulate the control problem as an MPC problem, then the MPC problem would

consist of the combination of static problems for each prediction step, without having coupling between the static problems. Hence, this would effectively mean solving N independent static optimization problems without any coupling between them each time a control agent has to determine actions. However, note that if dynamics depending on time, or if objective terms depending on inputs implemented earlier are included, that then it does make sense to formulate an MPC control problem. □

When the influence-based approach is used to determine for each control agent which subnetwork it should consider, the resulting subnetworks can be non-overlapping, touching, or overlapping. In addition, the influence-based approach uses the sensitivity (5.3), which is a function of the operating point, to select which nodes should belong to a subnetwork. Since the operating point can change over time, the nodes that would be assigned to a subnetwork can differ as well. Hence, the subnetworks can be time-varying.

If the subnetworks are non-overlapping, then the values of the variables of the nodes that control agents can influence significantly do not overlap, so no coordination among control agents is necessary. Adequate control performance can then be obtained, as illustrated in [51]. If the subnetworks are touching, then techniques based, e.g., on the ideas of Chapter 2 can be used to obtain coordination. For subnetworks that are overlapping, no techniques have been proposed so far for obtaining coordination. For overlapping subnetworks, the control agents will have to find agreement on how the variables involved in the dynamics of the common subnetworks will evolve over time. In the following we first discuss an approach that can be used for controlling time-invariant touching subnetworks. Then we extend this approach to be able to deal with time-invariant overlapping subnetworks. For sake of simplicity we assume below that all nodes in the network are assigned to a subnetwork.

5.4 Multi-agent control of touching subnetworks

In Chapters 2 and 3 we have discussed two approaches for coordinating multiple control agents when subnetworks are touching, based on a decomposition of an augmented Lagrange function. Below we discuss a technique for coordinating such control agents based on the ideas of the modified Lagrange technique proposed in [31]. The underlying idea is to determine subproblems in such a way that the first-order optimality conditions for the subproblems of all control agents together are equivalent to the first-order optimality conditions of a hypothetical overall control problem [31].

5.4.1 Internal and external nodes

Before explaining how the approach for multi-agent control of touching subnetworks works, we first define some concepts that will be frequently used in the following:

- We categorize the nodes that control agent i considers based on their location. For touching subnetworks, the nodes that control agent i considers can be *internal* nodes or *external* nodes. The internal nodes of control agent i are those nodes that belong exclusively to its subnetwork. The external nodes of control agent i are those nodes that do not belong to its subnetwork.

type	location	variables involved in constraint
$\mathcal{C}_{i,\text{int}}^{\text{int}}$	internal	internal
$\mathcal{C}_{i,\text{int}}^{\text{int+ext}}$	internal	internal+external
$\mathcal{C}_{i,\text{ext}}^{\text{ext}}$	external	external
$\mathcal{C}_{i,\text{ext}}^{\text{int+ext}}$	external	internal+external

Table 5.1: Localized constraint types of constraints associated with nodes in a subnetwork that touches other subnetworks. The location indicates the location of the node from the point of view of control agent i. The variables involved in the constraint indicate which variables are involved in the constraint, from the point of view of control agent i.

- Based on the distinction between internal and external nodes of control agent i, we make a distinction between internal and external variables of control agent i. The internal variables are those variables associated with the internal nodes of control agent i. The external variables are those variables associated with the external nodes of control agent i.

- For control agent i, the *localized constraint type* of a particular constraint associated with a node ι that control agent i considers is formed by the combination of the location and the types of variables involved in that constraint. The localized constraint type of a constraint associated with a node ι considered by control agent i is denoted by $\mathcal{C}_{i,\text{Loc}}^{\text{Vars}}$, where $Loc \in \{\text{int}, \text{ext}\}$ indicates the location of the node to which the constraint is associated, and $Vars \in \{\text{int}, \text{int+ext}\}$ indicates the variables involved in the constraint. Recall that a constraint associated with a particular node ι involves variables of that particular node and possibly variables of neighboring nodes. The constraints associated with the nodes considered by control agent i can therefore have the localized constraint types as depicted in Table 5.1. Figure 5.2 illustrates for some nodes the localized constraint types that can be found at these nodes.

- In a similar way as we def ned localized constraint types $\mathcal{C}_{i,\text{Loc}}^{\text{Vars}}$, we also def ne localized objective term types $\mathcal{J}_{i,\text{Loc}}^{\text{Vars}}$, referring to the location of the node to which an objective term is associated and the variables that are involved in the objective function term.

5.4.2 Control problem formulation for one agent

The optimization problem of control agent i at time step k consists of minimizing the objective function J_i, subject to the steady-state characteristics of subnetwork i and additional constraints on inputs and outputs. Below we focus on the diff culties that arise with respect to the prediction model and the objective function due to the existence of other control agents that control subnetworks that are touching the subnetwork of control agent i.

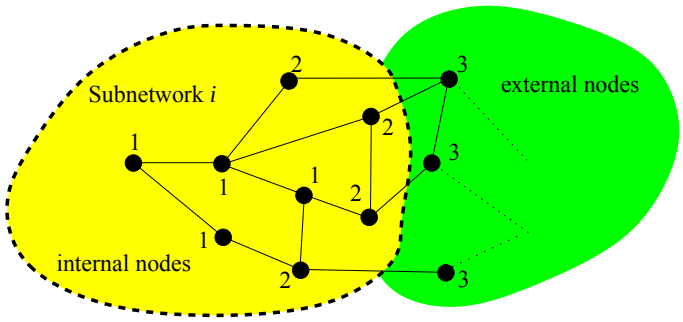


Figure 5.2: Illustration of different localized constraint types that can be found at nodes considered by control agent i. The number next to a node in the figure corresponds as follows to the localized constraint types of the constraints that can be associated to that node: (1) $C_{i,int}^{int}$; (2) $C_{i,int}^{int}$, $C_{i,int}^{int+ext}$; (3) $C_{i,ext}^{int+ext}$, $C_{i,ext}^{ext}$.

Prediction model

The prediction model of control agent i consists of the constraints associated with all its internal nodes. The internal nodes that do not have external neighboring nodes do not require special attention, since the variables involved in the constraints of these internal nodes are of localized constraint type $C_{i,\text{int}}^{\text{int}}$ and therefore only involve variables that are inf uenced by control agent i. However, the internal nodes that are connected to external nodes do require special attention, since the constraints associated with these internal nodes can be of localized constraint type $C_{i,\text{int}}^{\text{int+ext}}$, and therefore involve not only variables of the subnetwork of control agent i, but also variables of the subnetwork of a neighboring agent $j \in \mathcal{N}_i$. In order to make predictions over its prediction horizon, control agent i has to know accurate values for the external variables involved in the constraints of these nodes. Therefore, control agent i has to coordinate with the neighboring agents which values external variables should have. To obtain coordination on the values of the external variables, we apply an idea that was f rst proposed in [31] as follows.

Control agent i considers the constraints that are associated with its internal nodes and that are of localized constraint type $C_{i,\text{int}}^{\text{int+ext}}$, using f xed values for the external variables. The values for these external variables have been obtained from the neighboring agent j that has the node of these external variables as an internal node. Control agent i solves its local optimization problem using these values for the external variables. The optimization yields values for the internal variables of control agent i, and for the Lagrange multipliers that are associated with the constraints of localized constraint type $C_{i,\text{int}}^{\text{int+ext}}$. The Lagrange multipliers of these constraints and the values of the internal variables involved in these constraints are sent to each neighboring agent j that has a node to which the external variables of these constraints correspond as an internal node.

Each neighboring agent j considers the constraints of the internal nodes of control agent i that involve external variables of control agent i in its decision making by including these associated constraints as soft constraints in its objective function. Note that internal and external nodes of control agent i correspond to external and internal nodes, respectively, of a control agent j. In the soft constraints of such a control agent j, the external variables, which

localized constraint type	constraint
$\mathcal{C}_{i,\text{int}}^{\text{int}}$	hard
$\mathcal{C}_{i,\text{int}}^{\text{int+ext}}$	hard
$\mathcal{C}_{i,\text{ext}}^{\text{int+ext}}$	soft

Table 5.2: The constraints that control agent i can have and how it deals with these constraints. For the hard and soft constraints, the external variables are fixed to values obtained from neighboring agents. For the hard constraints with external variables Lagrange multipliers are determined. The soft constraints are weighted using the Lagrange multipliers received from neighboring agents.

localized objective term type	how deal with the objective term
$\mathcal{J}_{i,\text{int}}^{\text{int}}$	include as is
$\mathcal{J}_{i,\text{int}}^{\text{int+ext}}$	include as is

Table 5.3: The localized objective term types that control agent i considers and how it deals with these terms. External variables are fixed to values obtained from neighboring agents.

correspond to internal variables of control agent *i*, are f xed to the values that control agent *i* has sent to control agent *j*. In addition, the soft constraints are weighted by the Lagrange multipliers as given by control agent *i*. Neighboring agent *j* solves its optimization problem, yielding values for its internal variables. It sends the values of the internal variables that appear in the soft constraints to control agent *i*, such that control agent *i* can update its information about the corresponding external variables.

Based on this idea, Table 5.2 shows how control agent *i* deals with the different constraints, when formulating its optimization problem.

Objectives

The objective function for control agent *i* consists of objective function terms that are associated with the nodes in its subnetwork. Objective terms associated with internal nodes that are only connected to internal nodes do not give rise to issues, since no other control agents consider these objective terms. However, objective terms associated with internal nodes that are also connected to external nodes cause problems for the same reason as with the constraints associated with such nodes. Coordination on the values of these variables is obtained by obtaining the desired values for the external variables from neighboring agents.

Table 5.3 shows the different localized objective term types that control agent *i* considers, and how it deals with these, when formulating its optimization problem.

5.4.3 Control scheme for multiple agents

The multi-agent control scheme taking into account the prediction model and objective function discussed above operates in an iterative way. When the control agents have to determine actions, they perform a series of iterations, in each of which the control agents perform a

local optimization step and communicate information. The outline of the scheme is as follows:

1. Each control agent i measures the current values for the algebraic variables $\mathbf{z}_i$ and the input variables $\mathbf{u}_i$ that are associated with the nodes in its subnetwork. In addition, it obtains predictions of known exogenous inputs $\mathbf{d}_i$. Furthermore, it obtains through communication from its neighbors values for the external variables and Lagrange multipliers associated with the external nodes that control agent i considers.

2. The iteration counter s is set to 1.

3. Let $\mathbf{w}_{\text{in},i}^{(s-1)}$ and $\tilde{\boldsymbol{\lambda}}_{\text{soft},i}^{(s-1)}$ denote the external variables and Lagrange multipliers, respectively, of which control agent i has received the values from neighboring agents. Given $\mathbf{w}_{\text{in},i}^{(s-1)}$ and $\tilde{\boldsymbol{\lambda}}_{\text{soft},i}^{(s-1)}$, each control agent $i \in \{1,\ldots,n\}$ performs the following steps in parallel:

   (a) Control agent i solves the optimization problem:

$$\min_{\mathbf{z}_i,\mathbf{u}_i,\mathbf{w}_{\text{out},i}} J_i\left(\mathbf{z}_i,\mathbf{u}_i,\mathbf{w}_{\text{in},i}^{(s-1)}\right) + \left(\tilde{\boldsymbol{\lambda}}_{\text{soft},i}^{(s-1)}\right)^{\mathrm{T}} \tilde{\mathbf{g}}_{\text{soft},i}\left(\mathbf{z}_i,\mathbf{u}_i,\mathbf{w}_{\text{in},i}^{(s-1)}\right)$$

   subject to

$$\tilde{\mathbf{g}}_{\text{hard},i}\left(\mathbf{z}_i,\mathbf{u}_i,\mathbf{d}_i\right) = 0$$

$$\tilde{\mathbf{g}}_{\text{hard,ext},i}\left(\mathbf{z}_i,\mathbf{u}_i,\mathbf{d}_i,\mathbf{w}_{\text{in},i}^{(s-1)}\right) = 0 \tag{5.4}$$

$$\mathbf{w}_{\text{out},i} = \tilde{\mathbf{K}}_i\left[\mathbf{z}_i^{\mathrm{T}} \quad \mathbf{u}_i^{\mathrm{T}} \quad \mathbf{d}_i^{\mathrm{T}}\right]^{\mathrm{T}} \tag{5.5}$$

$$\mathbf{z}_{i,\min} \leq \mathbf{z}_i \leq \mathbf{z}_{i,\max}$$

$$\mathbf{u}_{i,\min} \leq \mathbf{u}_i \leq \mathbf{u}_{i,\max},$$

   where $\mathbf{z}_{i,\min}$ and $\mathbf{z}_{i,\max}$ are upper and lower bounds on $\mathbf{z}_i$, $\mathbf{u}_{i,\min}$ and $\mathbf{u}_{i,\max}$ are upper and lower bounds on $\mathbf{u}_i$, $\tilde{\mathbf{g}}_{\text{soft},i}$ are the constraints of localized constraint type $\mathcal{C}_{i,\text{ext}}^{\text{int+ext}}$, $\tilde{\mathbf{g}}_{\text{hard},i}$ are the constraints of localized constraint type $\mathcal{C}_{i,\text{int}}^{\text{int}}$, $\tilde{\mathbf{g}}_{\text{hard,ext},i}$ are the constraints of localized constraint type $\mathcal{C}_{i,\text{int}}^{\text{int+ext}}$, and $\mathbf{w}_{\text{out},i}$ are the variables that control agent i uses in communication to neighboring agents, selected using selection matrix $\tilde{\mathbf{K}}_i$. The optimization results in values for the variables $\mathbf{z}_i^{(s)}$ and $\mathbf{u}_i^{(s)}$, Lagrange multipliers $\tilde{\boldsymbol{\lambda}}_{\text{hard,ext},i}^{(s)}$ associated with the constraints (5.4) for current iteration s, and values for $\mathbf{w}_{\text{out},i}^{(s)}$.

   (b) Control agent i sends the values of the Lagrange multipliers $\tilde{\boldsymbol{\lambda}}_{\text{hard,ext},i}^{(s)}$ of the hard constraints of localized constraint type $\mathcal{C}_{i,\text{int}}^{\text{int+ext}}$ and the values of $\mathbf{w}_{\text{out},i}$ corresponding to internal variables of these nodes to the neighboring agents that consider the involved external variables.

   (c) Control agent i receives from the neighboring agents $j \in \mathcal{N}_i$ those Lagrange multipliers related to the localized constraint type $\mathcal{C}_{i,\text{ext}}^{\text{int+ext}}$ and those values of the internal variables of the neighboring agents that control agent i requires to

f x its external variables. Control agent i uses this received information at the next iteration as $\tilde{\boldsymbol{\lambda}}^{(s)}_{\text{soft},i}$ and $\mathbf{w}^{(s)}_{\text{in},i}$.

4. The next iteration is started by increasing s and going back to step 3, unless a stopping condition is satisf ed. The stopping condition is def ned as the condition that the absolute changes in the Lagrange multipliers from iteration $s-1$ to s are smaller than a pre-def ned small positive constant $\gamma_{\epsilon,\text{term}}$.

Although the approach discussed above can coordinate control agents that control touching subnetworks, a shortcoming of this method is that it requires that the subnetworks are touching, since it assumes that each node in the network is assigned to only one of the subnetworks. However, in the case of control of overlapping subnetworks, some of the nodes are included in more than one subnetwork and the identif cation of internal and external nodes of a control agent is not straightforward any more. Therefore, the method is not directly applicable to overlapping subnetworks. In the following, we consider an extension of the method discussed above to control of overlapping subnetworks.

5.5 Multi-agent control for overlapping subnetworks

Now, we extend the approach for control of touching subnetworks to control of overlapping subnetworks. We f rst propose some new def nitions, then consider the issues appearing due to the overlap, and then propose a way to deal with these issues.

5.5.1 Common nodes

In addition to internal and external nodes as def ned before, for control of overlapping subnetworks we make the following def nitions:

- *Common* nodes are nodes that belong to the subnetwork of control agent i and that also belong to the subnetwork of another control agent j. A subnetwork def ned by the nodes common to several subnetworks is referred to as a common subnetwork.

- The variables associated with the common nodes are referred to as the common variables.

- Given the def nition of a common node, the number of possibilities for localized constraint types increases. Table 5.4 lists the localized constraint types that can be considered by a control agent when subnetworks can be overlapping. In total there are 12 different localized constraint types. Figure 5.3 illustrates some of the possible localized constraint types.

- In addition to the extension of the localized constraint types, the localized objective term types are extended also accordingly.

type	location	variables involved in constraint
$\mathcal{C}^{\text{int}}_{i,\text{int}}$	internal	internal
$\mathcal{C}^{\text{int+com}}_{i,\text{int}}$	internal	internal+common
$\mathcal{C}^{\text{int+ext}}_{i,\text{int}}$	internal	internal+external
$\mathcal{C}^{\text{int+com+ext}}_{i,\text{int}}$	internal	internal+common+external
$\mathcal{C}^{\text{int+com}}_{i,\text{com}}$	common	internal+common
$\mathcal{C}^{\text{int+com+ext}}_{i,\text{com}}$	common	internal+common+external
$\mathcal{C}^{\text{com}}_{i,\text{com}}$	common	common
$\mathcal{C}^{\text{com+ext}}_{i,\text{com}}$	common	common+external
$\mathcal{C}^{\text{ext}}_{i,\text{ext}}$	external	external
$\mathcal{C}^{\text{int+ext}}_{i,\text{ext}}$	external	internal+external
$\mathcal{C}^{\text{com+ext}}_{i,\text{ext}}$	external	common+external
$\mathcal{C}^{\text{int+com+ext}}_{i,\text{ext}}$	external	internal+common+external

Table 5.4: Localized constraint types for overlapping subnetworks.

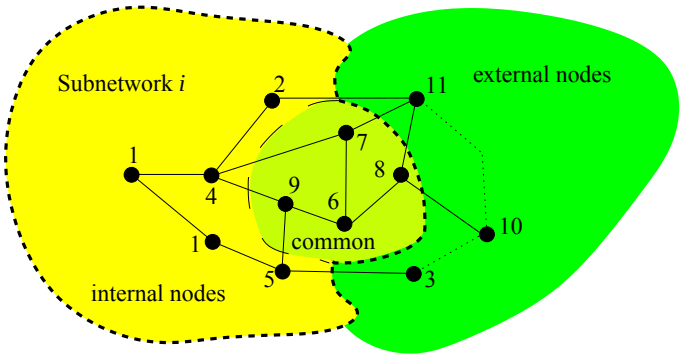


Figure 5.3: Illustration of different localized constraint types that can be found at particular nodes. The number next to a node in the figure corresponds as follows to the localized constraint types of the constraints that can be associated to that node: (1) $\mathcal{C}^{\text{int}}_{i,\text{int}}$; (2) $\mathcal{C}^{\text{int}}_{i,\text{int}}$, $\mathcal{C}^{\text{int+ext}}_{i,\text{int}}$; (3) $\mathcal{C}^{\text{int+ext}}_{i,\text{ext}}$, $\mathcal{C}^{\text{ext}}_{i,\text{ext}}$; (4) $\mathcal{C}^{\text{int}}_{i,\text{int}}$, $\mathcal{C}^{\text{int+com}}_{i,\text{int}}$; (5) $\mathcal{C}^{\text{int}}_{i,\text{int}}$, $\mathcal{C}^{\text{int+com}}_{i,\text{int}}$, $\mathcal{C}^{\text{int+ext}}_{i,\text{int}}$, $\mathcal{C}^{\text{int+com+ext}}_{i,\text{int}}$; (6) $\mathcal{C}^{\text{com}}_{i,\text{com}}$; (7) $\mathcal{C}^{\text{int+com}}_{i,\text{com}}$, $\mathcal{C}^{\text{com+ext}}_{i,\text{com}}$, $\mathcal{C}^{\text{com}}_{i,\text{com}}$, $\mathcal{C}^{\text{int+com+ext}}_{i,\text{com}}$; (8) $\mathcal{C}^{\text{com}}_{i,\text{com}}$, $\mathcal{C}^{\text{com+ext}}_{i,\text{com}}$; (9) $\mathcal{C}^{\text{com}}_{i,\text{com}}$, $\mathcal{C}^{\text{int+com}}_{i,\text{com}}$; (10) $\mathcal{C}^{\text{ext}}_{i,\text{ext}}$, $\mathcal{C}^{\text{ext+com}}_{i,\text{ext}}$; (11) $\mathcal{C}^{\text{int+ext}}_{i,\text{ext}}$, $\mathcal{C}^{\text{com+ext}}_{i,\text{ext}}$, $\mathcal{C}^{\text{ext}}_{i,\text{ext}}$, $\mathcal{C}^{\text{int+com+ext}}_{i,\text{ext}}$.

5.5.2 Control problem formulation for one agent

For multi-agent control of overlapping subnetworks an approach has to be found to deal with the common nodes. Since the common nodes are considered by several control agents, also the constraints associated with these common nodes appear in the subnetwork models of multiple control agents. Even though we assume that the control agents have the same objective with respect to these nodes, combined with the objective for their internal nodes, conf icting intentions for the common nodes can be the result. Below we discuss how to extend the scheme of the previous section for control of overlapping subnetworks.

Prediction model

Similar as for control of touching subnetworks, for control of overlapping subnetworks, internal nodes of control agent i that are connected to external nodes require special attention, since the constraints associated to these nodes may involve external variables. In addition to this, also common nodes of control agent i that are connected to external nodes require special attention. The extension of the approach for control of touching subnetworks to the control of overlapping subnetworks consists of the following with respect to the prediction model.

Control agent i considers as prediction model the constraints of all internal *and* common nodes. For the constraints of localized constraint types $\mathcal{C}_{i,\text{int}}^{\text{int+ext}}$, $\mathcal{C}_{i,\text{int}}^{\text{int+ext+com}}$, $\mathcal{C}_{i,\text{com}}^{\text{com+ext}}$, and $\mathcal{C}_{i,\text{com}}^{\text{int+com+ext}}$ the control agent takes for the external variables values that it has received from neighboring agents. When control agent i has solved its optimization problem, it sends the values of the internal *and* the common variables of the constraints of these specialized constraint types to neighboring agents.

Each neighboring agent j considers the constraints of the internal and common nodes of control agent i that involve external variables of control agent i in its optimization problem as soft constraints by including them in the objective function, weighted by the Lagrange multipliers provided by control agent i, and with f xed values for the external *and* common values in the soft constraints as received from control agent i. The result of solving the optimization problem of neighboring agent j yields values for the internal, common, and external variables of control agent j. The internal variables of control agent j related to the soft constraints are sent to control agent i.

Table 5.5 summarizes how control agent i deals with the different localized constraint types.

Objectives

With the nodes that control agent i has in its subnetwork objective terms are associated. The objective function terms associated with each node can depend on the variables associated with that node and its neighboring nodes. As before, the objective terms involving only internal variables require no special attention. The objective terms involving both internal and external variables can be dealt with by f xing the external variables, as is also done for control of touching subnetworks. However, the common variables appearing in control of overlapping subnetworks do require special attention.

For control of overlapping subnetworks, multiple control agents will try to control the values of the common variables. To allow control agents to jointly achieve performance

localized constraint type	constraint
$\mathcal{C}_{i,\text{int}}^{\text{int}}$	hard
$\mathcal{C}_{i,\text{int}}^{\text{int+ext}}$, $\mathcal{C}_{i,\text{int}}^{\text{int+com}}$	hard
$\mathcal{C}_{i,\text{int}}^{\text{int+com+ext}}$	hard
$\mathcal{C}_{i,\text{com}}^{\text{int+com}}$	hard and soft
$\mathcal{C}_{i,\text{com}}^{\text{int+com+ext}}$	hard and soft
$\mathcal{C}_{i,\text{com}}^{\text{com}}$	hard
$\mathcal{C}_{i,\text{com}}^{\text{com+ext}}$	hard
$\mathcal{C}_{i,\text{ext}}^{\text{int+ext}}$	soft
$\mathcal{C}_{i,\text{ext}}^{\text{int+ext+com}}$	soft

Table 5.5: The way in which control agent i considers the constraints of particular localized constraint types in its optimization problem. For the hard constraints all common variables are fixed to values obtained from neighboring agents. For the soft constraints all external and common variables are fixed. For the hard constraints with external variables Lagange multipliers are determined. The soft constraints are weighted with Lagrange multipliers obtained from neighboring agents.

localized objective term type	how deal with the objective term
$\mathcal{J}_{i,\text{int}}^{\text{int}}$	include as is
$\mathcal{J}_{i,\text{int}}^{\text{int+ext}}$	include as is
$\mathcal{J}_{i,\text{int}}^{\text{int+com}}$	include as is
$\mathcal{J}_{i,\text{com}}^{\text{com}}$	include partially: $1/N_\iota$
$\mathcal{J}_{i,\text{com}}^{\text{int+com}}$	include partially: $1/N_\iota$

Table 5.6: The localized objective term types that control agent i considers and how it deals with the associated objective terms. External variables are fixed. Variable N_ι is the number of control agents considering node N_ι as common node.

comparable to the performance that an overall centralized control agent can achieve, the responsibility for the objective terms involving only common variables, i.e., of localized objective term type $\mathcal{C}_{i,\text{com}}^{\text{com}}$, is shared equally by the control agents. Hence, each control agent i that considers a particular common node ι, takes in its objective function $1/N_\iota$ times the objective function terms of that common node that involve only common variables, where N_ι is the number of control agents considering node N_ι as common node. Control agent i in addition includes the objective terms of internal and common nodes that involve only internal and common variables, i.e., of localized objective term types $\mathcal{C}_{i,\text{int}}^{\text{int}}$, $\mathcal{C}_{i,\text{int}}^{\text{int+com}}$, $\mathcal{C}_{i,\text{com}}^{\text{com}}$, and $\mathcal{C}_{i,\text{com}}^{\text{int+com}}$.

Table 5.6 summarizes how control agent i deals with the different localized objective term types.

5.5.3 Control scheme for multiple agents

We have discussed how each control agent formulates its prediction model and objective function. The scheme that we propose for multi-agent control for overlapping subnetworks consists of the scheme proposed in Section 5.4 for touching subnetworks, with the following changes:

- Control agent i receives from the neighboring agents the following information at initialization and after each iteration:
  - Lagrange multipliers with respect to the constraints of localized constraint type $\mathcal{C}_{i,\text{ext}}^{\text{ext+int}}$, $\mathcal{C}_{i,\text{ext}}^{\text{ext+com}}$, $\mathcal{C}_{i,\text{ext}}^{\text{ext+com+int}}$.
  - Values for the external variables *and* the common variables involved in these constraints.
- The optimization problem that each agent solves is changed accordingly to reflect the extensions discussed in this section, i.e., to take into account the constraints as given in Table 5.5 and the objective terms as given in Table 5.6.

The result is a control scheme that can be used by higher-layer control agents that control subnetworks that are overlapping. In the next section we apply this scheme on an optimal flow control problem in power networks.

5.6 Application: Optimal flow control in power networks

In this section we propose to use the scheme discussed in Section 5.5 for multi-agent control of overlapping subnetworks to the problem of optimal power flow control in power networks. Optimal power flow control is a well known-method to control and optimize the operation of a power network [82]. Optimal power flow control is typically used to improve steady-state network security by improving the voltage profile, preventing lines from overloading, and minimizing active power losses. Usually settings for generators are determined by solving an optimization problem that minimizes an objective function encoding the system security objectives, subject to the steady-state characteristics of the network.

Typically only steady-state characteristics at on time step are considered, not taking into account future known exogenous inputs. The conventional optimal power flow control can be easily generalized to an optimal power flow control taking into account future known exogenous inputs. In this way, indeed, the optimal power flow control can be seen as an application of model predictive control, in which the prediction model consists of the steady-state characteristics defined over a particular prediction horizon.

Flexible alternating current transmission systems (FACTS) are devices that can improve power network operation. They can be used for dynamic control of voltage, impedance, and phase angle. The usage of FACTS devices has the potential to improve the security of the network, to increase the dynamic and transient stability, to increase the quality of supply for sensitive industries, and to enable environmental benefits, all without changing the topology of the existing network [62]. Some frequently used types of FACTS devices, and the types of FACTS devices that we consider below, are Static Var Compensators (SVCs) and Thyristor Controlled Series Compensators (TCSCs) [40].

Traditional approaches for multi-agent optimal power f ow control assume that a decomposition of the overall network and control objectives into touching subnetworks is possible [80, 116], as shown in Figure 5.1(b). When the optimal power f ow control problem involves multiple subnetworks and each bus in these subnetworks is assigned to exactly one subnetwork, then the assumption of touching subnetworks is appropriate to make. However, when a bus in a subnetwork is assigned to multiple subnetworks, then this assumption no longer holds. In our case, we are interested in control using FACTS devices of subnetworks that have been determined by sensitivity analysis, as discussed in Section 5.3. As we discussed in that section, the resulting subnetworks can be non-overlapping, touching, or overlapping. Indeed, if FACTS devices are positioned topologically far from each other, their inf uence-based subnetworks will typically not overlap, whereas if they are positioned topologically close to each other, their inf uence-based subnetworks will typically overlap. Hence, an approach that can be used by the control agents controlling the FACTS devices in such overlapping subnetworks is required. The approach proposed in Section 5.5 is suitable for this.

Simulations are carried out on the IEEE 57-bus power network with additional FACTS devices installed at various locations [5]. The base parameters of the IEEE 57-bus network can be obtained from the Power Systems Test Case Archive[2]. Line limits have been assigned to the lines in such a way that no lines are overloaded. In order to f nd an interesting and meaningful situation for FACTS control, the grid was adapted by placing an additional generator at bus 30 leading to increased power f ows in the center of the grid. The values of all parameters of the used power network are available from the author on request.

Below we formulate the steady-state models used to describe the network behavior, we assign the constraints to buses, we set up the objective terms associated with the buses, we discuss the way in which the subnetworks can be determined using the inf uence-based approach, and we show the workings of the proposed approach.

5.6.1 Steady-state characteristics of power networks

As the focus lies on improving the steady-state network security, the power network is modeled using equations describing the steady-state characteristics of the power network. As we will see, the aspects of the steady-state security that we are interested in can be determined from the voltage magnitude $z_{V,\iota}$ per unit (p.u.) and voltage angle $z_{\theta,\iota}$ (degrees) associated with each bus ι in the network. In order to determine the values for these variables under different exogenous inputs and actuator values, models for the components and their inf uence on the voltage magnitude and angle are def ned. We model the transmission lines, the generators, the loads, and the FACTS devices.

Transmission lines

For the transmission lines the well known π-model is used [82]. The active power $z_{P,\iota\omega}$ (p.u.) and the reactive power $z_{Q,\iota\omega}$ (p.u.) f owing from bus ι over the transmission line to

[2] http://www.ee.washington.edu/research/pstca/pf57/pg_tca57bus.htm

bus ω are then given by, respectively:

$$z_{P,\iota\omega} = (z_{V,\iota})^2 \left(\frac{\eta_{\mathrm{R},\iota\omega}}{(\eta_{\mathrm{R},\iota\omega})^2 + (\eta_{\mathrm{X},\iota\omega})^2} \right) - z_{V,\iota} z_{V,\omega} \left(\frac{\eta_{\mathrm{R},\iota\omega}}{(\eta_{\mathrm{R},\iota\omega})^2 + (\eta_{\mathrm{X},\iota\omega})^2} \cos(z_{\theta,\iota} - z_{\theta,\omega}) \right) + z_{V,\iota} z_{V,\omega} \left(\frac{\eta_{\mathrm{X},\iota\omega}}{(\eta_{\mathrm{R},\iota\omega})^2 + (\eta_{\mathrm{X},\iota\omega})^2} \sin(z_{\theta,\iota} - z_{\theta,\omega}) \right) \tag{5.6}$$

$$z_{Q,\iota\omega} = (z_{V,\iota})^2 \left(\frac{\eta_{\mathrm{X},\iota\omega}}{(\eta_{\mathrm{R},\iota\omega})^2 + (\eta_{\mathrm{X},\iota\omega})^2} - \frac{\eta_{\mathrm{B},\iota\omega}}{2} \right) + z_{V,\iota} z_{V,\omega} \left(\frac{\eta_{\mathrm{R},\iota\omega}}{(\eta_{\mathrm{R},\iota\omega})^2 + (\eta_{\mathrm{X},\iota\omega})^2} \sin(z_{\theta,\iota} - z_{\theta,\omega}) \right) - z_{V,\iota} z_{V,\omega} \left(\frac{\eta_{\mathrm{X},\iota\omega}}{(\eta_{\mathrm{R},\iota\omega})^2 + (\eta_{\mathrm{X},\iota\omega})^2} \cos(z_{\theta,\iota} - z_{\theta,\omega}) \right), \tag{5.7}$$

where $\eta_{\mathrm{B},\iota\omega}$ (p.u.) is the shunt susceptance, $\eta_{\mathrm{R},\iota\omega}$ (p.u.) is the resistance, and $\eta_{\mathrm{X},\iota\omega}$ (p.u.) is the reactance of the line between buses ι and ω.

The constraints for each transmission line going from bus ι to bus ω, for $\omega \in \mathcal{N}^{\iota}$, are assigned to bus ι.

Generators

Generators are modeled with constant active power injection and constant voltage magnitude. Hence, if a generator is connected to bus ι, then the following constraints are assigned to that bus:

$$z_{P,\mathrm{gen},\iota} = d_{P,\mathrm{gen},\iota}$$
$$z_{V,\iota} = d_{V,\mathrm{gen},\iota},$$

where $d_{P,\mathrm{gen},\iota}$ is the given active power that the generator produces, and $d_{V,\mathrm{gen},\iota}$ is the given voltage magnitude that the generator maintains. At most one generator can be connected to a bus, since a generator directly controls the voltage magnitude of that bus.

A single generator is used as slack generator, i.e., a generator with inf nite active and reactive power capacity, with f xed voltage magnitude and angle [82]. Hence, if the slack generator is connected to bus ι, the following constraints are assigned to that bus:

$$z_{V,\iota} = d_{V,\mathrm{gen},\iota}$$
$$z_{\theta,\iota} = d_{\theta,\mathrm{gen},\iota},$$

where $d_{\theta,\mathrm{gen},\iota}$ is the given voltage angle ensured by the generator.

Loads

The loads are modeled with constant active and constant reactive power injections. Hence, if a load is connected to bus ι, then the following constraints are associated to that bus:

$$z_{P,\mathrm{load},\iota} = d_{P,\mathrm{load},\iota}$$
$$z_{Q,\mathrm{load},\iota} = d_{Q,\mathrm{load},\iota},$$

where $d_{P,\text{load},\iota}$ and $d_{Q,\text{load},\iota}$ are the given active and reactive power consumption, respectively, of the load connected to bus ι. For simplicity, one load can be connected to a node. Multiple loads can easily be aggregated to obtain a single load.

FACTS devices

SVC An SVC is a FACTS device that is shunt-connected to a bus ι and that injects or absorbs reactive power $z_{Q,\text{SVC},\iota}$ to control the voltage $z_{V,\iota}$ at that bus [62]. The SVC connected to bus ι is modeled as a shunt-connected variable susceptance, which accepts as control input the effective susceptance $u_{B,\text{SVC},\iota}$, as shown in Figure 5.4(a). The injected reactive power $z_{Q,\text{SVC},\iota}$ of the SVC is:

$$z_{Q,\text{SVC},\iota} = -(z_{V,\iota})^2 u_{B,\text{SVC},\iota}.$$

The control input $u_{B,\text{SVC},\iota}$ is limited to the domain:

$$u_{B,\text{SVC,min},\iota} \leq u_{B,\text{SVC},\iota} \leq u_{B,\text{SVC,max},\iota},$$

where the values of $u_{B,\text{SVC,min},\iota}$ and $u_{B,\text{SVC,max},\iota}$ are determined by the size of the device [52].

The constraints of an SVC are assigned to the bus to which the SVC is connected.

TCSC A TCSC is a FACTS device that can control the active power f owing over a line [62]. It can change the line reactance $z_{X,\text{line},\iota\omega}$, and hence the conductance $\eta_{\text{G},\iota\omega}$ and susceptance $\eta_{\text{B},\iota\omega}$ involved in (5.6)–(5.7). The TCSC is therefore modeled as a variable reactance $u_{X,\text{TCSC},\iota\omega}$ connected in series with the line, as shown in Figure 5.4(b). If a TCSC is connected in series with a transmission line between buses ι and ω, the total reactance $z_{X,\text{line},\iota\omega}$ of the line including the TCSC is given by:

$$z_{X,\text{line},\iota\omega} = \eta_{\text{X},\iota\omega} + u_{X,\text{TCSC},\iota\omega},$$

where $\eta_{\text{X},\iota\omega}$ is the reactance of the line without the TCSC installed. The reactance $u_{X,\text{TCSC},\iota\omega}$ is limited to the domain:

$$u_{X,\text{TCSC,min},\iota\omega} \leq u_{X,\text{TCSC},\iota\omega} \leq u_{X,\text{TCSC,max},\iota\omega},$$

where the values of $u_{X,\text{TCSC,min},\iota\omega}$ and $u_{X,\text{TCSC,max},\iota\omega}$ are determined by the size of the TCSC device and the characteristics of the line in which it is placed, since due to the physics the allowed compensation rate of the line $u_{X,\text{TCSC},\iota\omega}/\eta_{\text{X},\iota\omega}$ is limited [52].

The constraints of the TCSC at the line between bus ι and ω are assigned to bus ι.

Power balance

By Kirchhoff's laws, at each bus the total incoming power and the total outgoing power has to be equal. This yields for bus ι the following additional constraints:

$$0 = \sum_{\omega \in \mathcal{N}^\iota} (z_{P,\iota\omega}) + z_{P,\text{load},\iota} - z_{P,\text{gen},\iota}$$
$$0 = \sum_{\omega \in \mathcal{N}^\iota} (z_{Q,\iota\omega}) + z_{Q,\text{load},\iota} + z_{Q,\text{SVC},\iota}.$$

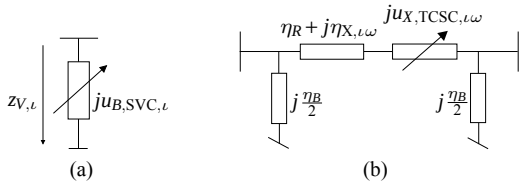


Figure 5.4: (a) Model of an SVC and (b) of a TCSC.

If no generator is connected to bus ι, then $d_{P,\mathrm{gen},\iota}$ and $z_{Q,\mathrm{gen},\iota}$ are zero. If no load is connected to bus ι, then $z_{P,\mathrm{load},\iota}$ and $z_{Q,\mathrm{load},\iota}$ are zero. If no SVC is connected to bus ι, then $z_{Q,\mathrm{SVC},\iota}$ is zero.

5.6.2 Control objectives

The objectives of the control are to improve the system security through minimization of deviations of bus voltages from given references to improve the voltage profile, minimization of active power losses, and preventing lines from overloading, by choosing appropriate settings for the FACTS devices. These objectives are translated into objective terms associated with the buses as follows:

- To minimize the deviations of the bus voltage magnitude $z_{V,\iota}$ of bus ι from a given reference $d_{V,\mathrm{ref},\iota}$, an objective term $p_V\,(z_{V,\iota} - d_{V,\mathrm{ref},\iota})^2$ is associated with bus ι, where p_V is a weighting coefficient.
- To minimize the active power losses over a line between bus ι and bus ω, an objective term $p_{\mathrm{loss}} z_{P,\mathrm{loss},\iota\omega}$ is associated to bus ι, where p_{loss} is a weighting coefficient, and where $z_{P,\mathrm{loss},\iota\omega} = z_{P,\iota\omega} + z_{P,\omega\iota}$.
- To minimize the loading of the line between buses ι and ω, an objective term is associated to bus ι as $p_{\mathrm{load}} \left(\frac{z_{S,\iota\omega}}{z_{S,\max,\iota\omega}} \right)^2$, where p_{load} is a weighting coefficient, and where $z_{S,\iota\omega} = \sqrt{(z_{P,\iota\omega})^2 + (z_{Q,\iota\omega})^2}$ is the apparent power flowing over the line from bus ι to bus ω. The relative line loading is penalized in a quadratic way such that an overloaded line is penalized more severely than a line that is not overloaded.

The weighting coefficients p_V, p_{loss}, and p_{load} allow to put change the weight given to each objective. In the following we take $p_V = 1000$, $p_{\mathrm{loss}} = 100$, and $p_{\mathrm{load}} = 1$.

5.6.3 Setting up the control problems

Each FACTS device is controlled by a different control agent. The influence-based subnetworks of the control agents controlling the FACTS devices can be overlapping, and therefore the control problems of the control agents are set up using the approach discussed in Section 5.5. To solve their subproblems at each iteration the control agents use the nonlinear problem solver SNOPT v5.8 [50], as implemented in Tomlab v5.7 [65], and accessed from Matlab v7.3 [98].

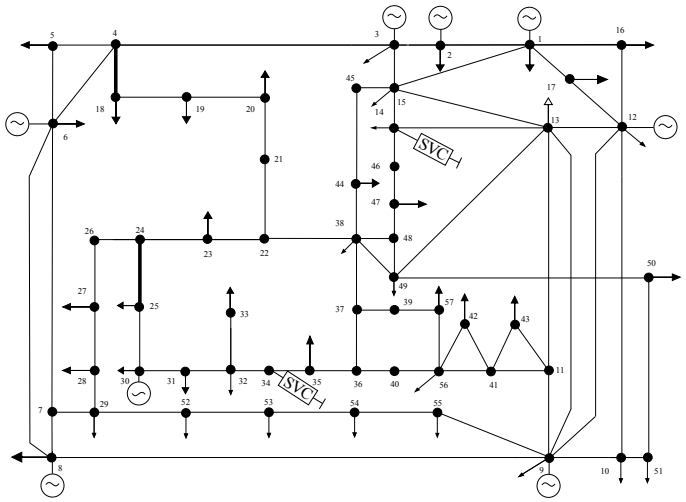


Figure 5.5: IEEE 57-bus network with SVCs installed at buses 14 and 34.

In the following we illustrate how the subnetwork of a control agent changes depending on the sensitivity threshold γ_s, and how the approach works for a particular assignment of buses to subnetworks in two representative scenarios.

5.6.4 Illustration of determination of subnetworks

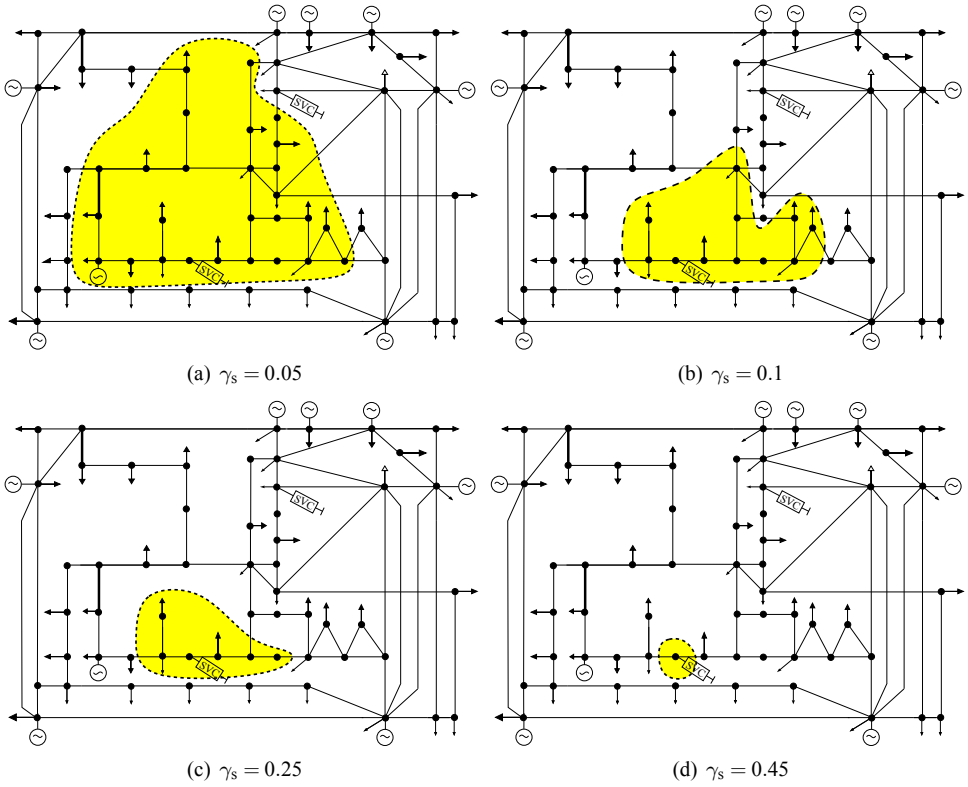
To illustrate the way in which inf uence-based subnetworks can be def ned for a power network, consider the adjusted IEEE 57-bus power network depicted in Figure 5.5 with SVCs installed at buses 14 and 34. We illustrate how the inf uence of the SVC at bus 34 on the buses in the network changes depending on the sensitivity threshold γ_s.

Remark 5.2 Instead of computing the gradients of the constraint functions of the network with respect to the SVC input analytically, we have numerically approximated them. The approximation is made by initializing the network in a particular operating point $\bar{\mathbf{z}}, \bar{\mathbf{u}}$, increasing the value of the SVC input by $\gamma_{\Delta u_{B,\mathrm{SVC}}}$, determining the values of $\mathbf{z}$, and computing the sensitivity of $\mathbf{z}$ with respect to the SVC input as: $\frac{1}{\gamma_{\Delta u_{B,\mathrm{SVC}}}}(\mathbf{z}-\bar{\mathbf{z}})$, where $\gamma_{\Delta u_{B,\mathrm{SVC}}} = 10^{-6}$. Since we are interested in the sensitivity of the SVC input with respect to the voltage magnitudes, the sensitivity criterion is checked only for the elements of $\mathbf{z}$ corresponding to the voltage magnitudes. □

Figure 5.6 shows the subnetworks and Figure 5.7 shows the number of nodes in the subnetworks, as the sensitivity threshold γ_s is increased. We observe that, indeed, with a lower threshold, more buses are included in the subnetwork, and with a higher threshold, fewer buses are included.

5.6.5 Simulations

Various test scenarios with different FACTS devices and subnetworks have been examined. Here we present two representative scenarios. The subnetworks used in these scenarios are



(a) $\gamma_s = 0.05$ (b) $\gamma_s = 0.1$

(c) $\gamma_s = 0.25$ (d) $\gamma_s = 0.45$

Figure 5.6: Subnetworks constructed with different sensitivity threshold settings.

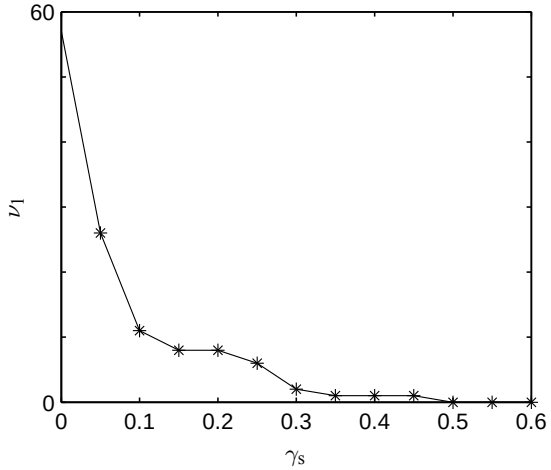


Figure 5.7: Number of nodes ν_1 selected for subnetwork 1 for different values of the ensitivity threshold γ_s.

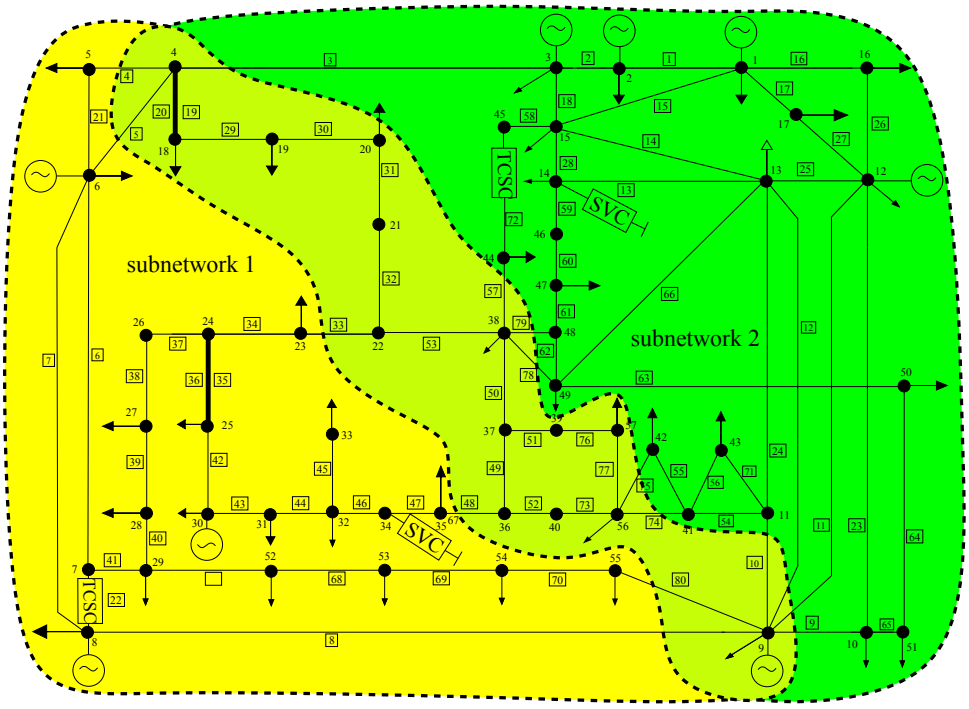


Figure 5.8: IEEE 57-bus system with decomposition into 2 subnetworks. Scenario 1: SVCs at buses 14 and 34, scenario 2: TCSCs in lines 22 and 72.

shown in Figure 5.8. It can be seen that these subnetworks are overlapping, since there are several buses that are included in both subnetworks.

Scenario 1: Control of SVCs

In the f rst scenario, SVCs are placed at buses 14 and 34. As the SVCs are mainly used to inf uence the voltage prof le, the line limits are chosen such that no line is at the risk of being overloaded.

Figure 5.9 shows the convergence of the SVC device settings over the iterations. As can be seen, the settings of the SVC devices converge within only a few iterations to the f nal values, which in this case are equal to the values obtained from an overall optimization. Figure 5.10 shows the evolution of the deviations between the values determined by both subnetworks for the voltage magnitudes and angles at some common buses. In the f gure the error $z_{V,\mathrm{err},\iota}$ is def ned as the absolute difference between the values that control agents 1 and 2 want to give to the voltage magnitude $z_{V,\iota}$. Similarly, the error $z_{\theta,\mathrm{err},\iota}$ is def ned as the absolute difference between the values that control agents 1 and 2 want to give to the voltage angles. As can be seen fast convergence is observed.

Scenario 2: Control of TCSCs

In the second scenario, TCSCs are installed in lines 72 and 22. Since TCSCs are mainly used to inf uence active power f ows and to resolve congestion, the line limits are chosen

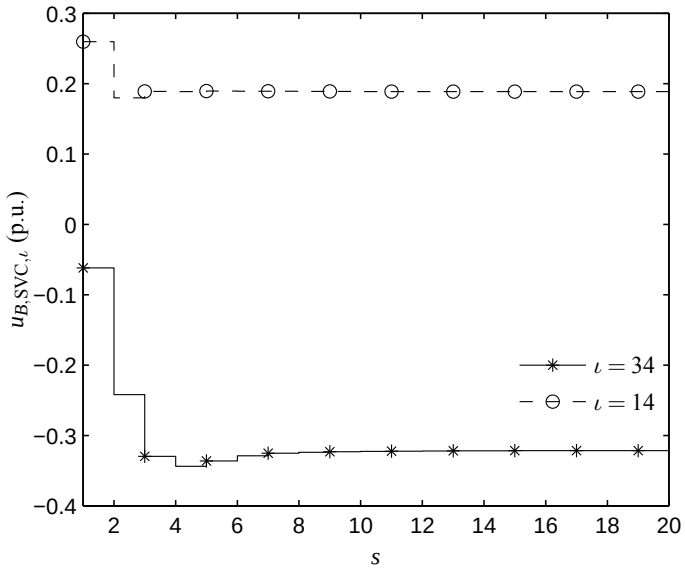


Figure 5.9: Convergence of the FACTS device settings over the iterations for the SVCs at buses 14 and 34 for scenario 1.

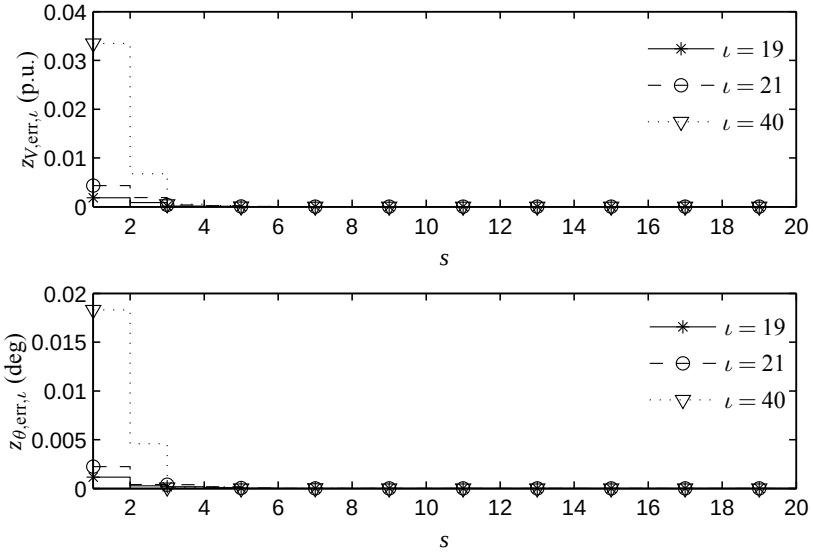


Figure 5.10: Convergence of the difference between the values of the voltage magnitudes (top) and the voltage angles (bottom) as considered by both control agents over the iterations for buses 19, 21, 40 for scenario 1.

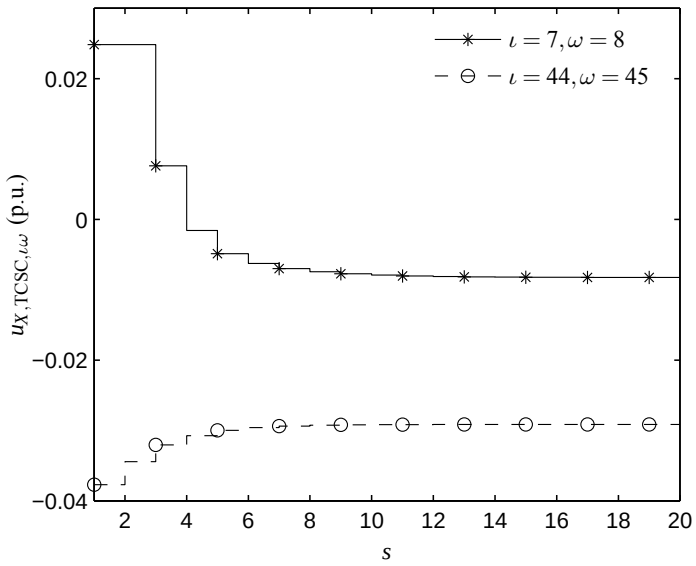


Figure 5.11: FACTS device settings for the TCSCs in lines 22 and 72, i.e., the lines between buses 7 and 8, and buses 44 and 45, respectively.

such that lines 7 and 60 are overloaded in the base case when the FACTS devices are set out of operation.

The results for the TCSC settings and the difference between the voltage magnitudes and angles for some common buses over the iterations are given in Figures 5.11 and 5.12, respectively. The control agent of subnetwork 1 sets the TCSC to its upper limit at the first few iterations. But after some additional iterations, the values that the control agents choose converge to their final values, which are again equal to the values obtained from an overall control agent.

In Figure 5.13 the line loadings of lines 7 and 60, i.e., the lines which are overloaded without FACTS devices in operation, are shown. Line 7 is immediately brought below its limit whereas for line 60, the loading approaches 100% in the course of the optimization process.

5.7 Summary

In this chapter we have focused on higher-layer multi-agent control using alternative ways to define subnetworks. While in Chapter 4 the medium control layer has used a model of the dynamics of the lower control layer and physical network, here the higher control layer uses steady-state characteristics only. While in the previous chapters we have defined subnetworks based on already existing control regions, in this chapter we have discussed how subnetworks can be defined based on the influence of actuators on the variables of nodes. When such an approach is used to define subnetworks, some subnetworks could be

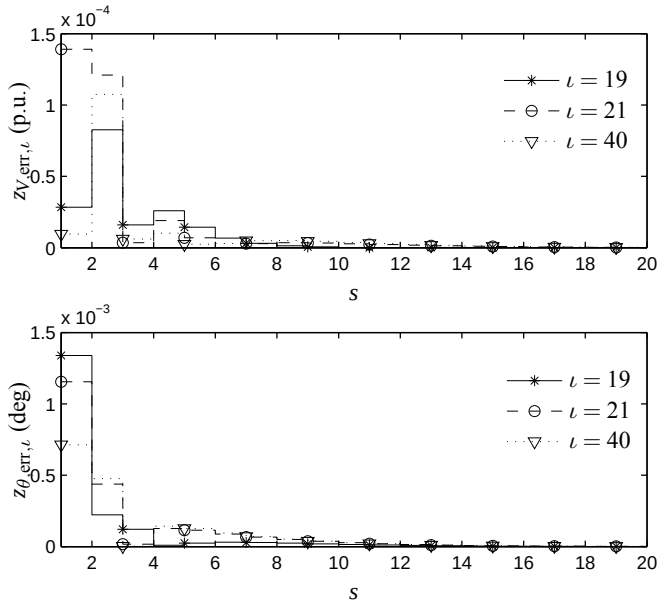


Figure 5.12: Convergence of the difference between the values of the voltage magnitudes (top) and the voltage angles (bottom) as considered by both control agents over the iterations for buses 19, 21, 40 for scenario 2.

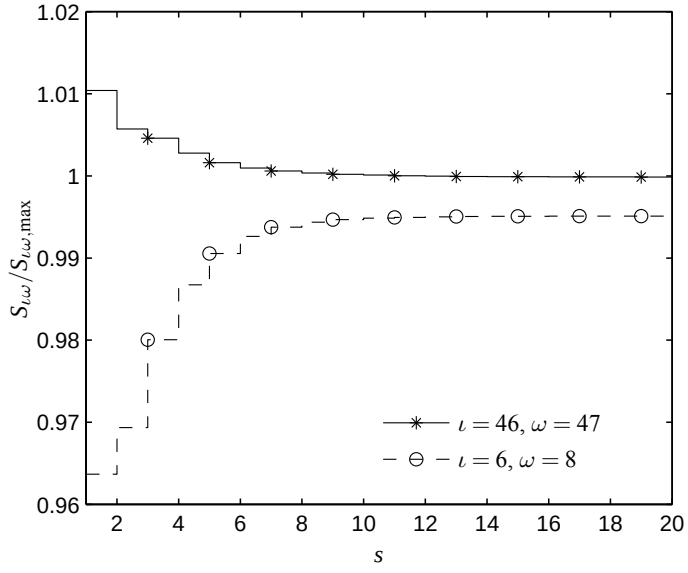


Figure 5.13: Relative line loadings of lines 7 and 60, i.e., the lines between buses 6 and 8, and 46 and 47, respectively, for scenario 2.

overlapping. Issues involving how to deal with the emerging common subnetwork then have to be dealt with. We have discussed these issues and proposed a method for higher-layer multi-agent control that can be used by control agents that control overlapping subnetworks. With simulation studies we have illustrated the potential of the approach. However, further research is still required, e.g., to determine formally when the approach converges and what the quality of the obtained solutions is, in particular when compared to an overall combined approach.

As application we have considered FACTS control in an adjusted version of the IEEE 57-bus power network. We have illustrated how the subnetwork of an actuator varies depending on the sensitivity threshold used, and we have applied the control approach that we proposed in this chapter for overlapping subnetworks to an optimal f ow control problem using FACTS devices. The simulations illustrate that the proposed approach can in the considered cases achieve fast convergence to actuator values that are overall optimal. Future research should address further comparison with an overall single-agent control scheme, to gain more insight in the quality of the solutions and the time required to obtain these solutions.

Chapter 6

Conclusions and future research

In this thesis we have discussed multi-agent model predictive control of transportation networks in general, and power networks in particular. We have discussed how control agents have to make decisions given different constraints on the type of systems they control, the actuators they can access, the information they can sense, and the communication and cooperation they can perform. In this chapter we summarize our main contributions and formulate future research directions.

6.1 Conclusions

Our main contributions with respect to the control approaches discussed are:

- **Serial versus parallel schemes.** In Chapter 2 we have formalized the dynamics of subnetworks as interconnected linear time-invariant systems, and def ned their control using an MPC control agent for each subnetwork. We have discussed why the control agent has to communicate with neighboring agents about how the variables involved in interconnecting constraints evolve. Furthermore, we have surveyed several ways of how to perform such communication, and have proposed a novel serial scheme, which converges toward an overall optimal solution under convexity of the overall MPC control problem. It has hereby been assumed that the subnetworks have discrete-time linear time-invariant dynamics, involving only variables taking on continuous values, and that the control objectives can be formulated as aff ne or convex functions. We have contrasted the scheme with a related parallel scheme. Experiments have conf rmed that the proposed approach can achieve performance close to overall performance.

- **Networked hybrid systems.** In Chapter 3 we have discussed issues related to modeling and control of hybrid systems, i.e., systems including both discrete and continuous elements. With respect to modeling of hybrid systems we have illustrated how discrete logic statements can be transformed into linear mixed-integer equality and inequality constraints. We have discussed issues arising in multi-agent control of interconnected hybrid systems and we have proposed an extension of the serial approach of Chapter 3 for control of such systems. This extension relaxes the assumptions

made on the original approach (i.e., discrete-time linear time-invariant dynamics for subnetworks with variables taking on continuous values, in combination with aff ne or convex objective functions) by allowing input variables to take on integer values instead of continuous values. Experiments using the proposed approach have given an indication that the proposed extension can resolve the discussed issues and can result in actions that give adequate performance.

- **Multi-layer control using MPC.** In Chapter 4 we have discussed MPC in multi-layer control. We have discussed the layered control of transportation networks using higher, medium, and lower-layer control, based on a time-scale decomposition of the dynamics. Then, we have focused in particular on issues related to the prediction model that a medium-layer MPC control agent uses and discussed why object-oriented modeling is suitable for constructing such a prediction model. Subsequently, we have proposed an MPC approach using such an object-oriented prediction model, or using a linearized approximation of such a model. To cope with the nonlinear, non-smooth, and costly-to-evaluate objective function of the MPC problem based on the object-oriented model, we have proposed the use of multi-start pattern search as optimization method. In experiments we have illustrated that the multi-start pattern search can outperform a state-of-the-art multi-start gradient-based approach. In addition, we have illustrated that using the MPC problem based on the linearized approximation of an object-oriented prediction model can result in signif cantly faster control, although at the price of reduced performance.

- **Overlapping subnetworks.** In Chapter 5 we have focused on the control by a higher-layer control agent. It is hereby assumed that at this higher layer the dynamics of the underlying control layers and physical network can be assumed instantaneous. We have discussed various ways of def ning subnetworks, and have in particular focused on how subnetworks can be def ned based on the inf uence that actuators of a control agent have. Using such inf uence-based subnetworks, it could happen that several subnetworks are overlapping. We have discussed issues that arise due to this overlap, and have proposed an approach for multi-agent control of overlapping subnetworks, using the nonlinear steady-state characteristics of the subnetworks as prediction models. Experiments have illustrated for a given example that the proposed approach can choose actions close to overall optimal actions.

We have considered several applications to which the proposed control approaches can be applied. Our main contributions with respect to these applications are:

- **Load-frequency control.** In Chapter 2 we have proposed the application of the serial MPC control scheme for a load-frequency control problem. Through experimental studies on a network consisting of 13 subnetworks, we have compared the serial scheme with the related parallel scheme and an overall scheme. The serial scheme showed to have preferable properties in terms of speed of convergence and quality of solutions. However, the parallel scheme outperformed the serial scheme in terms of total computation time. For the serial and the parallel schemes, the performance of the solutions obtained converged toward the performance of the solutions obtained by the overall scheme.

Furthermore, in Chapter 3 we have considered how the proposed extension of the serial scheme for interconnected hybrid systems performs when applied to the load-frequency control problem of Chapter 2, extended with discrete generation switching. We have illustrated that the approach has the potential to yield control actions that are overall optimal.

- **Household energy control.** In Chapter 3 we have used the transformations for discrete dynamics to derive a model for a household equipped with its own power generation and storage capabilities. As a first step toward a control structure in which multiple control agents, each representing a single household, jointly control the energy usage in a district, we have then proposed MPC for control of a single household using this model. In its decision making, the control agent uses expected energy consumption profiles and electricity export prices. We have illustrated that the MPC control agent can adequately take into account the discrete dynamics and yield a reduction in operational costs.

- **Emergency voltage control.** In Chapter 4 we have considered a control agent in a medium control layer of a power network that provides set-points to a lower control layer with the aim of preventing voltage collapses. For the MPC formulation of the higher-layer control agent based on the object-oriented model, using experiments we have illustrated the that multi-start pattern can outperform a multi-start state-of-the-art gradient-based method and we have illustrated that the voltage collapses can be prevented from occurring. For the MPC problem based on the linearized model, we have illustrated the performance of the control and related this performance to the quality of the predictions of the linearized model under faults of varying magnitude.

- **FACTS-based optimal flow control.** In Chapter 5, we have considered the problem of control of overlapping subnetworks using FACTS devices on an adjusted version of the IEEE 57-bus power network. We have illustrated how the region of influence of an actuator varies depending on the sensitivity threshold used, and we have applied the control approach proposed for control of overlapping subnetworks. Simulations have illustrated that the proposed approach has the potential to achieve fast convergence to actuator values that are overall optimal.

6.2 Future research

In principle, a multi-agent control approach for a transportation network will have to integrate solutions to each of the issues discussed in this thesis. However, even then several issues remain unsolved or can be investigated further. With respect to the control approaches addressed in this thesis, some challenging issues that require future research are:

- **Serial versus parallel schemes.** With respect to the serial multi-agent MPC scheme as discussed in Chapter 2, analytical bounds on the rate of convergence should be derived to give guarantees on the speed at which decisions are made. In addition, ways to speed up the decision making should be investigated, e.g., by forming groups and control agents that cooperate in coalitions.

- **Networked hybrid systems.** In Chapter 3, the transformations that have been used to transform discrete logic into linear constraints yielded a large number of binary variables. Research should address how the number of binary variables can be reduced. This may be done, e.g., by reformulating the underlying discrete logic, or by examining for which discrete dynamics it is strictly necessary to explicitly include these discrete dynamics; it may be the case that some discrete dynamics have a negligible effect on the continuous dynamics (e.g., dynamics appearing further away on a prediction horizon) and that these discrete dynamics therefore can be neglected or approximated with continuous dynamics. For the proposed extension of the serial MPC scheme for hybrid systems, it should be investigated formally whether and under which assumptions the scheme is guaranteed to converge to an overall optimal solution. In addition, it should be investigated how exactly the penalty coeff cient should be increased, and with what value this should be done. Furthermore, it should be investigated what the range of systems is for which the proposed approach could work, and if for a larger range of systems combinations between techniques from distributed integer and distributed real optimization could be useful.

- **Multi-layer control using MPC.** With respect to Chapter 4, the performance loss when using the MPC control problem based on the linearized prediction model due to the approximation of the linearization should be further investigated. In addition, the solution techniques should be extended to deal with both continuous and discrete variables, such that hybrid systems can be controlled. Furthermore, analysis has to be done regarding the performance of the proposed approach for a medium-layer control agent when the model of the medium-layer control agent is an abstraction of the dynamics of the physical network and lower control layer. Model order reduction techniques may be used to determine which dynamics have to be taken into account by a medium-layer control agent, and which may be removed. In addition, topological reduction techniques may be used to determine which dynamics a medium-layer control agent can aggregate in order to obtain a simplif ed model. Furthermore, it has to be determined how control agents using MPC in a lower control layer should be taken into account by a higher-layer control agent, and how ultimately multiple MPC control agents in a higher layer should control multiple MPC control agents in a lower layer. Techniques such as those discussed in Chapters 2 and 5 may be extended to obtain agreement between control agents at different layers about certain variables. The techniques of Chapters 2 and 5 for the control agents in the lower control layer should be mixed with similar techniques to obtain coordination between lower and medium control layers.

- **Overlapping subnetworks.** For Chapter 5, the quality of the predictions made with subnetworks based on the inf uence of actuators under different sensitivity parameter values should be analyzed formally. In addition, investigation has to be performed on the assumptions under which the scheme proposed for control of overlapping subnetworks converges, and what the quality of the solutions obtained is. The scheme should be extended to include dynamic models, instead of only steady-state models, and to be able to deal with time-varying subnetworks, instead of f xed subnetworks.

In addition to these topics, more general fundamental further future research directions consist of:

- **Scalability.** It remains to be addressed how the convergence speed of the approaches discussed changes when applied to control structures with large numbers of control agents. If the approaches do not scale well, then ways to make them scalable should be investigated, e.g., by clustering control agents in groups in combination with coordination between the groups.

- **Robustness.** It should be investigated how robust the discussed approaches are against modeling errors and noise. In addition the control schemes that we have discussed silently assume that the decision making is done instantaneously or that at least the information used to initiate the decision making at a particular control cycle is valid also at moment at which actions are actually implemented. Future research should address how the schemes could be made robust to delays. In addition, fault-tolerance against failing control agents is still an unsolved issue.

- **Non-cooperative agents.** When some of the control agents are not cooperative the control agents may not be able to reach agreement on which actions to take. It should be investigated how the cooperative agents could deal with this and if they could, e.g., manipulate the non-cooperative control agents in order to reach the cooperative objectives. It would hereby be interesting to related concepts from multi-agent MPC to concepts from the f eld of non-cooperative game theory [11], such as Stackelberg games [11] and inverse Stackelberg games [64, 134].

- **Alternative control methodologies.** The techniques that have been discussed in this thesis should be compared with alternative, non-MPC-based techniques (both from the f eld of control engineering and from the f eld of computer science), to determine the advantages and disadvantages of each. In this way possibly new techniques can be proposed, combining the best of several techniques.

With respect to the applications discussed in this thesis, future research directly related to these applications consists of:

- **Load-frequency control.** The models that are used for the load-frequency control problem in Chapters 2 and 3 could be extended to more adequately represent the dynamics of the power networks, e.g., by including more detailed models of generators and loads, and by modeling explicitly the presence of tie-line power control devices. In addition, the model representing the physical network could be replaced by a continuous-time nonlinear model, instead of the currently used discrete-time linearized model. Furthermore, forecasts about expected power f ows between subnetworks could be included, such the MPC strategy can be exploit at an early time expected changes in power f ows.

- **Household energy control.** The model of a single household in Chapter 3 could be extended by including, e.g., disposable loads, i.e., load shedding within a household. In addition, the control problem could be reformulated to include variable gas and electricity export prices, and to schedule when consumption and generation should

take place. Furthermore, several households could be connected to one another, allowing energy to be exchanged between neighboring households. The control problem could then be extended such that the control agents of the households cooperatively optimize their energy usage. The models of the individual households will then depend on variables of other households. The control agents will have to reach agreement on the values of these variables in order to successfully implement MPC. It should be investigated how the proposed scheme in Section 3.4 for control of interconnected hybrid subnetworks performs on such a system of interconnected households. Moreover, from the practical point of view, steps toward implementation in practice can be made by first implementing the household control agents on a laboratory setup and then implementing the household control agents in physical households.

- **Emergency voltage control.** With respect to the emergency voltage control scenario in Chapter 4, it would be interesting to further investigate the range of situations in which the MPC control agent using the linearized prediction model performs adequately and would be a good replacement for the control agent using the nonlinear prediction model. In addition, a larger benchmark network could be constructed after which the extended control approaches proposed for future research could be applied. In this larger benchmark network, control agents are used to control parts of the network using MPC, and a higher-layer control agent coordinates these MPC control agents. Furthermore, it would be interesting to investigate the potential of the proposed approach for emergency voltage control within large industrial sites.
- **FACTS-based optimal flow control.** The way in which the subnetworks based on the influence of the FACTS devices change under varying network conditions should be investigated. In addition, instead of considering steady-state characteristics of the power network under consideration in Chapter 5, dynamics could be included, e.g., in the generators and loads, to more adequately model the dynamics of the network.

In addition, future general application-oriented research should investigate the use of the discussed approaches in other fields besides power networks. In this respect, the following future research directions should be considered:

- **Model development and validation.** The control schemes that we have discussed all require a model that adequately represents the dynamics of the system. For application of the approaches discussed on practical examples, models will have to be constructed and validated. It will then also have to be investigated which quantities can be measured in practice, and which quantities will have to be estimated.
- **Alternative application domains.** The application of the control approaches presented in this thesis is not restricted to the applications from the domain of power networks only. Domains in which the control approaches presented could be applied include not only transportation networks, such as water distribution networks, road traffic networks, railway networks, gas distribution networks, etc., but also the process industry (e.g., for multi-agent control of production lines), supply chains (e.g., for multi-agent control of stocks), and autonomous guided or flying vehicles. Investigation of the application of the approaches discussed in this thesis to these domains will give interesting insights into the similarities and dissimilarities between the operation of transportation networks.

Bibliography

[1] L. Acar. Some examples for the decentralized receding horizon control. In *Proceedings of the 31st IEEE Conference on Decision and Control*, pages 1356–1359, Tucson, Arizona, 1992.

[2] M. Aicardi, G. Casalino, R. Minciardi, and R. Zoppoli. On the existence of stationary optimal receding-horizon strategies for dynamic teams with common past information structures. *IEEE Transactions on Automatic Control*, 37:1767–1771, November 1992.

[3] M. Aldeen and J. F. Marsh. Observability, controllability and decentralized control of interconnected power systems. *International Journal on Computers and Electrical Engineering*, 16(4):207–220, 1990.

[4] P.J. Antsaklis and A. Nerode, editors. Special issue on hybrid systems. *IEEE Transactions on Automatic Control*, 43(4), April 1998.

[5] Power Systems Test Case Archive. Parameters of the IEEE 57-bus grid. http://www.ee.washington.edu/research/pstca/.

[6] K. J. Åström and B. Wittenmark. *Computer-Controlled Systems*. Prentice-Hall, Upper Saddle River, New Jersey, 1997.

[7] K. J. Åström, H. Elmqvist, and S. E. Mattsson. Evolution of continuous-time modeling and simulation. In *Proceedings of the 12th European Simulation Multiconference*, pages 9–18, Manchester, UK, June 1998.

[8] N. Atic, D. Rerkpreedapong, A. Hasanovic, and A. Feliachi. NERC compliant decentralized load frequency control design using model predictive control. In *Proceedings on the IEEE Power Engineering Society General Meeting*, Toronto, Canada, July 2003.

[9] C. Audet and J. E. Dennis Jr. Pattern search algorithms for mixed variable programming. *SIAM Journal on Optimization*, 11(3):573–594, 2000.

[10] C. Audet and J. E. Dennis Jr. Analysis of generalized pattern searches. *SIAM Journal on Optimization*, 13(3):889–903, 2007.

[11] T. Başar and G. J. Olsder. *Dynamic Non-Cooperative Game Theory*. Academic Press, London, UK, 1998.

[12] M. Baglietto, T. Parisini, and R. Zoppoli. Neural approximators and team theory for dynamic routing: A receding-horizon approach. In *Proceedings of the 38th IEEE Conference on Decision and Control*, pages 3283–3288, Phoenix, Arizona, 1999.

[13] P. Barton and C. Pantelides. Modeling of combined discrete/continuous processes. *AIChE Journal*, 40(6):966–979, 1994.

[14] J. Batut and A. Renaud. Daily generation scheduling optimization with transmission constraints: a new class of algorithms. *IEEE Transactions on Power Systems*, 7(3): 982–989, August 1992.

[15] A. G. Beccuti and M. Morari. A distributed solution approach to centralized emergency voltage control. In *Proceedings of the 2006 IEEE American Control Conference*, pages 3445–3450, Minneapolis, Minnesota, June 2006.

[16] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, March 1999.

[17] J. Bernussou and A. Titli. *Interconnected Dynamical Systems: Stability, Decomposition and Decentralisation*. North-Holland Publishing Company, Amsterdam, The Netherlands, 1982.

[18] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientif c, Beltmore, Massachusetts, 2003.

[19] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, London, UK, 1982.

[20] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientif c, New Hampshire, 1997.

[21] P. R. Bhave and R. Gupta. *Analysis of Water Distribution Networks*. Alpha Science International, Oxford, UK, 2006.

[22] L. G. Bleris, P. D. Vouzis, J. G. Garcia, M. G. Arnold, and M. V. Kothare. Pathways for optimization-based drug delivery. *Control Engineering Practice*, 15(10):1280–1291, October 2007.

[23] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

[24] S. D. Braithwait. Real-time pricing and demand response can work within limits. *Natural Gas and Electricity*, 21(11):1–9, 2005.

[25] M. W. Braun, D. E. Rivera, M. E. Flores, W. M. Carlyle, and K. G. Kempf. A model predictive control framework for robust management of multi-product, multi-echelon demand networks. *Annual Reviews in Control*, 27:229–245, 2003.

[26] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. SIAM, Philadelphia, Pennsylvania, 1996.

[27] E. F. Camacho and C. Bordons. *Model Predictive Control in the Process Industry*. Springer-Verlag, Berlin, Germany, 1995.

[28] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 1:44–52, February 2002.

[29] C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, Boston, Massachusetts, 1999.

[30] C. G. Cassandras, S. Lafortune, and G. J. Olsder. Introduction to the modelling, control and optimization of discrete event systems. In A. Isidori, editor, *Trends in Control: A European Perspective*, pages 217–291. Springer-Verlag, Berlin, Germany, 1995.

[31] A. J. Conejo, F. J. Nogales, and F. J. Prieto. A decomposition procedure based on approximate newton directions. *Mathematical Programming, Series A*, 93(3):495–515, December 2002.

[32] A. R. Conn, K. Scheinberg, and P. L. Toint. Recent progress in unconstrained nonlinear optimization without derivatives. *Mathematical Programming*, 79(1–3):397–414, 1997.

[33] C. F. Daganzo. *Fundamentals of Transportation and Traffic Operations*. Pergamon Press, New York, New York, 1997.

[34] R. David. Modeling of dynamic systems by Petri nets. In *Proceedings of the 1st European Control Conference*, pages 136–147, Grenoble, France, July 1991.

[35] B. De Schutter and T. J. J. van den Boom. Model predictive control for max-min-plus-scaling systems. In *Proceedings of the 2001 American Control Conference*, pages 319–324, Arlington, Virginia, June 2001.

[36] B. De Schutter, T. van den Boom, and A. Hegyi. A model predictive control approach for recovery from delays in railway systems. *Transportation Research Record*, (1793):15–20, 2002.

[37] W. B. Dunbar and R. M. Murray. Model predictive control of coordinated multi-vehicle formations. In *Proceedings of the 41st IEEE Conference on Decision and Control*, pages 4631–4636, Las Vegas, Nevada, December 2002.

[38] W. B. Dunbar and R. M. Murray. Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42(4):549–558, April 2006.

[39] Dynasim. Dymola – User's Manual. Technical report, Dynasim AB, Lund, Sweden, 2004.

[40] A. Edris, R. Adapa, M. H. Baker, L. Bohmann, K. Clark, K. Habashi, L. Gyugyi, J. Lemay, A. S. Mehraban, A. K. Meyers, J. Reeve, F. Sener, D. R. Torgerson, and R. R. Wood. Proposed terms and def nitions for f exible AC transmission system (FACTS). *IEEE Transactions on Power Delivery*, 12(4):1848–1853, October 1997.

[41] H. El Fawal, D. Georges, and G. Bornard. Optimal control of complex irrigation systems via decomposition-coordination and the use of augmented Lagrangian. In *Proceedings of the 1998 International Conference on Systems, Man, and Cybernetics*, pages 3874–3879, San Diego, California, 1998.

[42] O. I. Elgerd and C. Fosha. Optimum megawatt frequency control of multi-area electric energy systems. *IEEE Transactions on Power Apparatus and Systems*, PAS-89 (4):556–563, February 1970.

[43] Elkraft Systems. Power failure in Eastern Denmark and Southern Sweden on 23 September 2003 – preliminary report on the course of events. Technical report, Elkraft Systems, Holte, Denmark, 2003.

[44] H. Elmqvist, F. E. Cellier, and M. Otter. Object-oriented modeling of hybrid systems. In *Proceedings of the European Simulation Symposium*, pages xxxi–xli, Delft, The Netherlands, October 1998.

[45] B. Fardanesh. Future trends in power system control. *IEEE Computer Applications in Power*, 15(3):24–31, July 2002.

[46] R. G. Farmer and P. M. Anderson. *Series Compensation of Power Systems*. PBLSH, Encinitas, California, 1996.

[47] C. E. Fosha and O. I. Elgerd. The megawatt frequency control problem: A new approach via optimal control theory. *IEEE Transactions on Power Apparatus and Systems*, PAS-89(4):563–577, April 1970.

[48] G. Georges. Decentralized adaptive control for a water distribution system. In *Proceedings of the 3rd IEEE Conference on Control Applications*, pages 1411–1416, Glasgow, UK, 1999.

[49] T. Geyer, M. Larsson, and M. Morari. Hybrid emergency voltage control in power systems. In *Proceedings of the European Control Conference 2003*, Cambridge, UK, September 2003.

[50] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimisation*, 12(4):979–1006, 2002.

[51] G. Glanzmann and G. Andersson. FACTS control for large power systems incorporating security aspects. In *Proceedings of X SEPOPE*, Florianopolis, Brazil, May 2006.

[52] G. Glanzmann and G. Andersson. Using FACTS devices to resolve congestions in transmission grids. In *Proceedings of the CIGRE/IEEE PES International Symposium*, San Antonio, Texas, October 2005.

[53] M. Gomez, J. Rodellar, F. Vea, J. Mantecon, and J. Cardona. Decentralized predictive control of multireach canals. In *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*, pages 3885–3890, San Diego, California, 1998.

[54] A. H. González, D. Odloak, and J. L. Marchetti. Predictive control applied to heat-exchanger networks. *Chemical Engineering and Processing*, 45(8):661–671, August 2006.

[55] E. González-Romera, M. Á. Jaramillo-Morán, and D. Carmona-Fernández. Forecasting of the electric energy demand trend and monthly f uctuation with neural networks. *Computers and Industrial Engineering*, 52:336–343, April 2007.

[56] G. C. Goodwin, M. M. Seron, R. H. Middleton, M. Zhang, B. F. Hennessy, P. M. Stone, and M. Menabde. Receding horizon control applied to optimal mine planning. *Automatica*, 42(8):1337–1342, August 2006.

[57] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, July 2001.

[58] A. Hegyi, B. De Schutter, and J. Hellendoorn. Optimal coordination of variable speed limits to suppress shock waves. *IEEE Transactions on Intelligent Transportation Systems*, 6(1):102–112, March 2005.

[59] D. J. Hill. Nonlinear dynamic load models with recovery for voltage stability studies. *IEEE Transactions on Power Systems*, 8(1):166–176, February 1993.

[60] D. J. Hill, Y. Guo, M. Larsson, and Y. Wang. Global control of complex power systems. In G. Chen, D. J. Hill, and X. Yu, editors, *Bifurcation Control: Theory and Applications*, Lecture Notes in Control and Information Sciences, pages 155–187. Springer, Berlin, Germany, 2003.

[61] P. Hines, L. Huaiwei, D. Jia, and S. Talukdar. Autonomous agents and cooperation for the control of cascading failures in electric grids. In *Proceedings of the 2005 IEEE International Conference on Networking, Sensing and Control*, pages 273–278, Tucson, Arizona, March 2005.

[62] N. G. Hingorani and L. Gyugyi. *Understanding FACTS concepts and technology of flexible AC transmission systems*. IEEE Press, New York, New York, 2000.

[63] I. A. Hiskens and K. Mitsumoto. Dynamical systems benchmark library. URL: http://psdyn.ece.wisc.edu/IEEE_benchmarks/, 2005.

[64] Y. C. Ho, P. B. Luh, and G. J. Olsder. A control-theoretic view on incentives. In *Proceedings of the 19th IEEE Conference on Decision and Control*, pages 1160–1170, Albuquergue, New Mexico, December 1980.

[65] K. Holmström, A. O. Göran, and M. M. Edvall. User's guide for Tomlab /SNOPT, December 2006.

[66] K. Holmström, A. O. Göran, and M. M. Edvall. User's guide for Tomlab /CPLEX, June 2007.

[67] M. Houwing, A. N. Ajah, P. M. Herder, and I. Bouwmans. Addressing uncertainties in the design and operation of residential distributed energy resources: Case study

of a micro-CHP system. In *Proceedings of the 10th Conference on Process Integration, Modelling and Optimisation for Energy Saving and Pollution Reduction*, Ischia Island, Italy, June 2007.

[68] M. Houwing, R. R. Negenborn, P. Heijnen, B. De Schutter, and J. Hellendoorn. Least-cost model predictive control of residential energy resources when applying μCHP. In *Proceedings of Power Tech 2007*, Lausanne, Switzerland, July 2007. Paper 291.

[69] G. Hug-Glanzmann, R. R. Negenborn, G. Andersson, B. De Schutter, and J. Hellendoorn. Multi-area control of overlapping areas in power systems for FACTS control. In *Proceedings of Power Tech 2007*, Lausanne, Switzerland, July 2007. Paper 277.

[70] Ibraheem, P. Kumar, and D. P. Kothari. Recent philosophies of automatic generation control strategies in power systems. *IEEE Transactions on Power Systems*, 20(1): 346–357, February 2005.

[71] ILOG. CPLEX. URL: http://www.ilog.com/products/cplex/, 2007.

[72] R. Irizarry-Rivera and W. D. Seider. Model-predictive control of the Czochralski crystallization process. Part I. Conduction-dominated melt. *Journal of Crystal Growth*, 178(4):593–611, 1997.

[73] N. Jenkins, R. Allan, P. Crossley, D. Kirschen, and G. Strbac. *Embedded Generation*. TJ International, Padstow, UK, 2000.

[74] D. Jia and B. Krogh. Min-max feedback model predictive control for distributed control with communication. In *Proceedings of the 2002 American Control Conference*, pages 4507–4512, Anchorage, Alaska, May 2002.

[75] D. Jia and B. H. Krogh. Distributed model predictive control. In *Proceedings of the 2001 American Control Conference*, pages 2767–2772, Arlington, Virginia, June 2001.

[76] D. Karlsson and D. J. Hill. Modelling and identif cation of nonlinear dynamic loads in power systems. *IEEE Transactions on Power Systems*, 9(1):157–163, February 1994.

[77] M. R. Katebi and M. A. Johnson. Predictive control design for large-scale systems. *Automatica*, 33(3):421–425, 1997.

[78] H. Kawabata and M. Kido. A decentralized scheme of load frequency control power system. *Electrical Engineering Japan*, 102(4):100–106, July 1982.

[79] T. Keviczky, F. Borrelli, and G. J. Balas. A study on decentralized receding horizon control for decoupled systems. In *Proceedings of the 2004 American Control Conference*, volume 6, pages 4921–4926, Boston, Massachusetts, June 2004.

[80] B. H. Kim and R. Baldick. A comparison of distributed optimal power f ow algorithms. *IEEE Transactions on Power Systems*, 15(2):599–604, May 2000.

[81] B. H. Kim and R. Baldick. Coarse-grained distributed optimal power f ow. *IEEE Transactions on Power Systems*, 12(2):932–939, May 1997.

[82] P. Kundur. *Power System Stability and Control*. McGraw-Hill, New York, New York, 1994.

[83] S. Leirens, J. Buisson, P. Bastard, and J.-L. Coullon. A hybrid approach for voltage stability of power systems. In *Proceedings of the 15th Power Systems Computation Conference*, Liège, Belgium, August 2005. Paper 291.

[84] R. M. Lewis and V. Torczon. Pattern search methods for linearly constrained minimization. *SIAM Journal on Optimization*, 10(3):917–941, 2000.

[85] R. M. Lewis and V. Torczon. A globally convergent augmented Lagrangian pattern search algorithm for optimization with constraints and simple bounds. *SIAM Journal on Optimization*, 12(4):1075–1089, 2002.

[86] R. M. Lewis and V. Torczon. Pattern search algorithms for bound constrained minimization. *SIAM Journal on Optimization*, 9(4):1082–1099, 1999.

[87] R. M. Lewis, V. Torczon, and M. W. Trosset. Direct search methods: then and now. *Journal of Computational and Applied Mathematics*, 124(1–2):191–207, December 2000.

[88] G. Lodewijks. *Dynamics of Belt Systems*. PhD thesis, Delft University of Technology, The Netherlands, 1996.

[89] Z. Lukszo, M. P. C. Weijnen, R. R. Negenborn, B. De Schutter, and M. Ilić. Challenges for process system engineering in infrastructure operation and control. In W. Marquardt and C. Pantelides, editors, *16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering* (Garmisch-Partenkirchen, Germany, July 2006), volume 21 of *Computer-Aided Chemical Engineering*, pages 95–100. Elsevier, Amsterdam, The Netherlands, 2006.

[90] Z. Lukszo, M. P. C. Weijnen, R. R. Negenborn, and B. De Schutter. Tackling challenges in infrastructure operation and control using multi-level and multi-agent control. Technical Report 07-027, Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands, 2007. Submitted to a journal.

[91] E. Lysen, S. Van Egmond, and S. Hagedoorn. Opslag van elektriciteit: Status en toekomstperspectief voor Nederland. Technical Report NEO 0268-05-05-01-002, Utrecht Centrum voor Energieonderzoek – SenterNovem, Utrecht, The Netherlands, 2006. In Dutch.

[92] J. Machowski, J. Bialek, and J. R. Bumby. *Power System Dynamics and Stability*. John Wiley & Sons, New York, New York, 1997.

[93] J. M. Maciejowski. *Predictive Control with Constraints*. Prentice-Hall, Harlow, UK, 2002.

[94] Y. Majanne. Model predictive pressure control of steam networks. *Control Engineering Practice*, 13(12):1499–1505, December 2005.

[95] A. Manzoni, A. S. de Silva, and I. C. Decker. Power systems, dynamics simulation using object-oriented programming. *IEEE Transactions on Power Systems*, 14(1): 249–255, 1999.

[96] R. Martí. Multi-Start Methods. In F. Glover and G. A. Kochenberger, editors, *Handbook of Metaheuristics*, chapter 12, pages 355–368. Springer, New York, New York, 2006.

[97] MathWorks. Genetic Algorithm and Direct Search Toolbox 2 – User's Guide, 2007.

[98] Mathworks. Matlab. URL: http://www.mathworks.com/, 2007.

[99] S. E. Mattsson, H. Elmqvist, and M. Otter. Physical system modeling with Modelica. *Control Engineering Practice*, 6(4):501–510, April 1998.

[100] M. D. Mesarovic, D. Macko, and Y. Takahara. *Theory of Hierarchical Multilevel Systems*. Academic Press, New York, New York, 1970.

[101] P. J. Modi, W. M. Shen, M. Tambe, and M. Yokoo. ADOPT: Asynchronous Distributed Constraint Optimization with quality guarantees. *Artificial Intelligence*, 161 (1-2):149–180, January 2005.

[102] M. Morari and J. H. Lee. Model predictive control: past, present and future. *Computers and Chemical Engineering*, 23(4):667–682, 1999.

[103] J. J. Moré and S. J. Wright. *Optimization Software Guide*. SIAM, Philadelphia, Pennsylvania, 1993.

[104] A. S. Morse, C. C. Pantelides, S. Sastry, and J. M. Schumacher, editors. Special issue on hybrid systems. *Automatica*, 35(3), March 1999.

[105] I. R. Navarro, M. Larsson, and G. Olsson. Object-oriented modeling and simulation of power systems using modelica. In *Proceedings of the IEEE Power Engineering Society Winter Meeting*, pages 790–795, Singapore, January 2000.

[106] R. R. Negenborn, B. De Schutter, M.A. Wiering, and H. Hellendoorn. Learning-based model predictive control for Markov decision processes. In *Proceedings of the 16th IFAC World Congress*, Prague, Czech Republic, July 2005. Paper 2106 / We-M16-TO/2.

[107] R. R. Negenborn, B. De Schutter, and H. Hellendoorn. Multi-agent model predictive control of transportation networks. In *Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC 2006)*, pages 296–301, Fort Lauderdale, Florida, April 2006.

[108] R. R. Negenborn, B. De Schutter, and H. Hellendoorn. Multi-agent model predictive control for transportation networks with continuous and discrete elements. In *Proceedings of the 11th IFAC Symposium on Control in Transportation Systems*, pages 609–614, Delft, The Netherlands, August 2006.

[109] R. R. Negenborn, B. De Schutter, and J. Hellendoorn. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. In *Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 2006)*, pages 339–344, Saint-Etienne, France, May 2006.

[110] R. R. Negenborn, A. G. Beccuti, T. Demiray, S. Leirens, G. Damm, B. De Schutter, and M. Morari. Supervisory hybrid model predictive control for voltage stability of power networks. In *Proceedings of the American Control Conference 2007*, pages 5444–5449, New York, New York, July 2007.

[111] R. R. Negenborn, B. De Schutter, and J. Hellendoorn. Eff cient implementation of serial multi-agent model predictive control by parallelization. In *Proceedings of the 2007 IEEE International Conference on Networking, Sensing, and Control*, pages 175–180, London, UK, July 2007.

[112] R. R. Negenborn, B. De Schutter, and J. Hellendoorn. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. Technical Report 07-024, Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands, 2007. To appear in Engineering Applications of Artif cial Intelligence.

[113] R. R. Negenborn, S. Leirens, B. De Schutter, and J. Hellendoorn. Supervisory nonlinear MPC for emergency voltage control using pattern search. Technical Report 07-025, Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands, 2007. Submitted to a journal.

[114] P. G. Neumann. Widespread network failures. *Communications of the ACM*, 50(2): 112, 2007.

[115] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, New York, 1999. ISBN 0-387-98793-2.

[116] F. J. Nogales, F. J. Prieto, and A. J. Conejo. Multi-area AC optimal power f ow: A new decomposition approach. In *Proceedings of the 13th Power Systems Control Conference (PSCC)*, pages 1201–1206, Trondheim, Germany, 1999.

[117] S. Ochs, S. Engell, and A. Draeger. Decentralized vs. model predictive control of an industrial glass tube manufacturing process. In *Proceedings of the 1998 IEEE Conference on Control Applications*, pages 16–20, Trieste, Italy, 1998.

[118] M. Otter, H. Elmqvist, and S. E. Mattson. Hybrid modeling in Modelica based on the synchronous data f ow principle. In *Proceedings of the International Symposium on Computer Aided Control System Design*, pages 151–157, Kohala Coast-Island of Hawai'i, Hawai'i, August 1999.

[119] Y. M. Park and K. Y. Lee. Optimal decentralized load frequency control. *Electrical Power Systems Research*, 7(4):279–288, September 1984.

[120] M. Pehnt, M. Cames, C. Fischer, B. Praetorius, L. Schneider, K. Schumacher, and J. Vob. *Micro Cogeneration: Towards Decentralized Energy Systems*. Springer, Berlin, Germany, 2006.

[121] L. R. Petzold. A description of DASSL - A differential/algebraic system solver. In *Proceedings of the 10th World Congress on System Simulation and Scientific Computation*, pages 430–432, Montreal, Canada, August 1983.

[122] P. C. Piela, T. G. Epperly, K. M. Westerberg, and A. W. Westerberg. ASCEND: an object-oriented computer environment for modeling and analysis: The modeling language. *Computers and Chemical Engineering*, 15(1):53–72, January 1991.

[123] S. Piñón, E. F. Camacho, B. Kuchen, and M. Peña. Constrained predictive control of a greenhouse. *Computers and Electronics in Agriculture*, 49(3):317–329, December 2005.

[124] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes*. Cambridge University Press, Cambridge, UK, 2007.

[125] G. Quazza. Noninteracting controls of interconnected electric power systems. *IEEE Transactions on Power Apparatus and Systems*, PAS-85(7):727–741, July 1966.

[126] D. Rerkpreedapong, N. Atic, and A. Feliachi. Economy oriented model predictive load frequency control. In *Proceedings of the 2003 Large Engineering Systems Conference on Power Engineering*, pages 12–16, Montreal, Canada, May 2003.

[127] C. B. Royo. *Generalized Unit Commitment by the Radar Multiplier Method*. PhD thesis, Technical University of Catalonia, Barcelona, Spain, May 2001.

[128] P. W. Sauer and M. A. Pai. *Power System Dynamics and Stability*. Prentice-Hall, London, UK, 1998.

[129] S. Sawadogo, R. M. Faye, P. O. Malaterre, and F. Mora-Camino. Decentralized predictive controller for delivery canals. In *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*, pages 3380–3884, San Diego, California, 1998.

[130] N. I. Shaikh and V. Prabhu. Model predictive controller for cryogenic tunnel freezers. *Journal of Food Engineering*, 80(2):711–718, May 2007.

[131] M. G. Singh and A. Titli. *Systems Decomposition, Optimisation and Control*. Pergamon Press, Oxford, UK, 1978.

[132] J.-E. Skog, K. Koreman, and B. Pääjärvi. The Norned HVDC cable link – A power transmission highway between Norway and The Netherlands. In *Proceedings of Energex 2006*, Stavanger, Norway, June 2006.

[133] E. D. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control*, 26(2):346–358, April 1981.

[134] K. Staňková, M. C. J. Bliemer, and G. J. Olsder. Inverse Stackelberg games and their application to dynamic bilevel optimal toll design problem. In *Proceedings of the 12th International symposium on dynamic games and applications*, Sophia Antipolis, France, July 2006.

[135] K. P. Sycara. Multiagent systems. *AI Magazine*, 2(19):79–92, 1998.

[136] The Modelica Association. Modelica - A Unif ed Object-Oriented Language for Physical Systems Modeling – Language specif cation. URL: http://www.modelica.org/documents/ModelicaSpec22.pdf, 2005.

[137] M. M. Thomas, J. L. Kardos, and B. Joseph. Shrinking horizon model predictive control applied to autoclave curing of composite laminate materials. In *Proceedings of the American Control Conference*, pages 505–509, Baltimore, Maryland, June 1994.

[138] V. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7(1):1–25, 1997.

[139] UCTE. Final report of the investigation committee on the 28 September 2003 blackout in Italy. Technical report, Union for the Coordination of Transmission of Electricity (UCTE), Brussels, Belgium, 2003.

[140] UCTE. Final report system disturbance on 4 November 2006. Technical report, Union for the Coordination of Transmission of Electricity (UCTE), Brussels, Belgium, 2006.

[141] U.S.-Canada Power System Outage Task Force. Final report on the August 14, 2003 blackout in the United States and Canada: causes and recommendations. Technical report, April 2004.

[142] T. Van Cutsem and C. Vournas. *Voltage Stability of Electric Power Systems*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[143] A. J. van der Schaft and J. M. Schumacher. *An Introduction to Hybrid Dynamical Systems*, volume 251 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, London, 2000.

[144] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, and S. J. Wright. Distributed output feedback MPC for power system control. In *Proceedings of the 45th IEEE Conference on Decision and Control*, San Diego, California, December 2006.

[145] D. D. Šiljak. *Decentralized Control of Complex Systems*. Academic Press, Boston, Massachusetts, 1991.

[146] W. Wang, D. E. Rivera, and K. G. Kempf. Model predictive control strategies for supply chain management in semiconductor manufacturing. *International Journal of Production Economics*, 107(1):56–77, May 2007.

[147] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, Massachusetts, 2000.

[148] Wikipedia. List of famous wide-scale power outages. URL: http://en.wikipedia.org/wiki/List_of_power_outages, 2007.

[149] H. P. Williams. *Model Building in Mathematical Programming*. Wiley, New York, New York, 1993.

[150] M. H. Wright. Direct search methods: Once scorned, now respectable. In D. F. Griffths and G. A. Watson, editors, *Numerical Analysis 1995*, pages 191–208. Addison Wesley, Harlow, UK, 1996.

[151] T. C. Yang, H. Cimen, and Q. M. Zhu. Decentralised load-frequency controller design based on structured singular values. *IEE Proceedings Generation, Transmission and Distribution*, 145(1):7–14, January 1998.

[152] T. C. Yang, Z. T. Ding, and H. Yu. Decentralised power system load frequency control beyond the limit of diagonal dominance. *International Journal on Electrical Power Energy Systems*, 24(3):173–184, March 2002.

Glossary

Conventions

The following conventions are used in this thesis for notation and symbols:

- A lower case character typeset in boldface, e.g., $\mathbf{x}$, represents a column vector.
- The number of elements in a vector $\mathbf{x}$ is indicated by $n_{\mathbf{x}}$.
- An upper case character typeset in boldface, e.g., $\mathbf{A}$, represents a matrix.
- A character typeset in calligraphics, e.g., $\mathcal{N}$, represents a set.
- A tilde over a variable, e.g., $\tilde{x}$, indicates a variable specif ed over a prediction horizon.
- A bar over a variable, e.g., $\bar{x}$, indicates that the value of the variable is known.
- A subscript i or j of a variable, e.g., x_i or x_j, refers to a variable of a control agent or subnetwork i or j, respectively.
- Subscripts max and min of a variable, e.g., $x_{\max}$ and $x_{\min}$, represent the maximum and minimum value of that variable, respectively.
- A subscript avg, e.g., x_{avg}, indicates that an average is considered.
- A superscript ι or ω of a variable, e.g., x^{ι} or x^{ω}, refers to a variable beloning to node ι or ω, respectively.
- A superscript T, e.g., $\mathbf{x}^{\mathrm{T}}$, indicates that a transpose is taken.

List of symbols and notations

Below follows a list of the most frequently used symbols and notations in this thesis. Symbols particular to power network applications are explained only in the relevant chapters.

$\mathbf{A}, \mathbf{A}_{\mathrm{c}}$	system matrices of linear time-invariant models
$\mathbf{B}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3$	input matrices of linear time-invariant models
$\mathbf{C}, \mathbf{C}_{\mathrm{c},y}, \mathbf{C}_{\mathrm{c},z}$	output matrices of linear time-invariant models

$\mathcal{C}_{i,\mathrm{Loc}}^{\mathrm{Vars}}$	localized constraint type
d	exogeneous input
$\mathbf{D}, \mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_{\mathrm{c},y}, \mathbf{D}_{\mathrm{c},z}$	direct-feedthrough matrices of linear time-invariant models
$\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3, \mathbf{E}_4, \mathbf{E}_5$	matrices of mixed-logical dynamic models
f	function
$\mathbf{f}$	vector with linear cost coeff cients
$\mathbf{F}, \mathbf{F}_{\mathrm{c}}$	state-offset vectors of linear time-invariant models
g	equality constraint function
g_u	equality constraint function with all variables except u f xed
$g_{\mathrm{hard},i}$	equality constraint function of subnetwork i for an internal node
$g_{\mathrm{hard,ext},i}$	equality constraint function of subnetworks i for an internal node that is connected to an external node
$g_{\mathrm{soft},i}$	equality constraint function of subnetwork i for an external node
$\mathbf{G}, \mathbf{G}_{\mathrm{c},y}, \mathbf{G}_{\mathrm{c},z}$	output-offset vectors of linear time-invariant models
h	inequality constraint function
i	index of a control agent or a subnetwork
$\mathbf{I}$	identity matrix
j	index of a neighboring agent
J	objective function
$J_{\mathrm{add}}, J_{\mathrm{cycle}}, J_{\mathrm{rel}}, J_{\mathrm{sim}}$	additional, cycle, relative, and full simulation cost
$\mathcal{J}_{i,\mathrm{Loc}}^{\mathrm{Vars}}$	localized objective function term type
k	discrete time step or control cycle counter
k_{c}	control cycle counter
k_{f}	control cycle f nishing step
k_{p}	prediction step counter
$\tilde{\mathbf{K}}_i$	interconnecting-output selection matrix of agent i
l	a control cycle counter within predictions
L_{aug}	an augmented Lagrange function
m_i	number of neighbors of control agent i
M, M_{lin}	prediction model and linearized prediction model
$\mathcal{M}$	mesh with candidate solutions
n	number of subnetworks
$n_{\mathbf{a}}$	number of elements in vector $\mathbf{a}$
N	length of a prediction horizon
N_{c}	length of a prediction horizon in control cycles
N_{init}	number of initial solutions
N_{iter}	number of iterations

N_p	length of a prediction horizon in discrete time steps
$N_{\Delta s}$	number of iterations between parameter updates
N_ι	number of control agents that have node ι in their subnetwork
$\mathbb{N}$	set of natural numbers
$\mathbb{N}^+$	set of positive natural numbers
$\mathcal{N}_i$	set of indexes of neighboring agents of agent i
$\mathcal{N}^\iota$	set of indexes of neighboring nodes of node ι
p	parameter
$\mathbf{Q}, \mathbf{Q_a}$	weight matrices for quadratic costs
$\mathbb{R}$	set of real numbers
s	iteration number
$\mathbf{s}, \mathbf{s}^+$	solution vector and a new solution vector
t	continuous time instant
$t_0, t_\mathrm{f}, t_\mathrm{fault}$	initial, f nishing, and fault continuous time instant
T_c	length of a discrete control cycle in seconds
T_comp	computation time in seconds
T_opt	f nishing time of an optimization in seconds
T_p	length of a discrete time step in seconds
u	input variable
u_b	binary input variable
u_c	continuous input variable
$\mathbf{U}$	vector with input vectors of all agents
$\mathcal{U}$	domain with integer values
v_i	local remaining variable of subnetwork i
$w_{\mathrm{in},ji}$	interconnecting input of subnetwork i
$w_{\mathrm{out},ij}$	interconnecting output of subnetwork j
$\mathbf{w}_{\mathrm{in},i}$	vector with all interconneting inputs of agent i
$\mathbf{w}_{\mathrm{out},i}$	vector with all interconnecting outputs of agent i
$\mathbf{W}_\mathrm{in}$	vector with the interconnecting inputs of all agents
$\mathbf{W}_\mathrm{out}$	vector with the interconnecting outputs of all agents
x	state variable
x_b	binary state variable
x_c	continuous state variable
$\mathbf{X}$	vector with states of all agents
y	output variable
y_b	binary output
y_c	continuous output
$y_\mathrm{desired,max}, y_\mathrm{desired,min}$	desired upper and lower bound

y_{err}	maximum of the violation of an upper and lower bound
$\mathbf{Y}$	vector with output variables of all agents
z	auxiliary continuous variable
z_∞	auxiliary variable for computing an ∞-norm
$\mathbf{z}_1$	auxiliary variables for computing a 1-norm
$\gamma_{\text{b}}, \gamma_{\text{c}}$	positive penalty coeff cients
γ_{contr}	contraction factor
γ_{exp}	expansion factor
$\gamma_{\text{m}}, \gamma_{\text{M}}$	minimum and maximum of a function
γ_{mesh}	mesh size change
γ_{s}	sensitivity threshold
$\gamma_{\Delta\text{c}}$	multiplication factor for γ_{c}
$\gamma_{\epsilon,\text{mach}}$	small postive constant close to machine precision
$\gamma_{\epsilon,\text{term}}$	small postive constant used for determining termination
δ	binary variable
ι	index of a node
$\lambda_{\text{in},ji}$	Lagrange multiplier of an interconnecting input constraint
$\lambda_{\text{out},ij}$	Lagrange multiplier of an interconnecting output constraint
$\lambda_{\text{hard,ext},i}$	Lagrange multiplier of a constraint of subnetwork i for an internal node that is connected to an external node
$\lambda_{\text{soft},i}$	Lagrange multiplier of a constraint of subnetwork i for an external node
$\mathbf{\Lambda}_{\text{in}}$	vector with Lagrange multipliers of all agents
ν	number of nodes in a network
ω	index of a neighboring node

List of abbreviations

The following abbreviations are used in this thesis:

AVR	Automatic Voltage Regulator
DAE	Differential-Algebraic Equations
FACTS	Flexible Alternating-Current Transmission System
MPC	Model Predictive Control
PSS	Power System Stabilizer
SVC	Static Var Compensator
TCSC	Thyristor Controlled Series Compensators
μCHP	micro Combined Heat and Power

TRAIL Thesis Series

A series of The Netherlands TRAIL Research School for theses on transport, infrastructure and logistics.

Nat, C.G.J.M., van der, *A Knowledge-based Concept Exploration Model for Submarine Design*, T99/1, March 1999, TRAIL Thesis Series, Delft University Press, The Netherlands

Westrenen, F.C., van, *The Maritime Pilot at Work*: E*valuation and Use of a Time-to-boundary Model of Mental Workload in Human-machine Systems*, T99/2, May 1999, TRAIL Thesis Series, Eburon, The Netherlands

Veenstra, A.W., *Quantitative Analysis of Shipping Markets,* T99/3, April 1999, TRAIL Thesis Series, Delft University Press, The Netherlands

Minderhoud, M.M., *Supported Driving: Impacts on Motorway Traffic Flow*, T99/4, July 1999, TRAIL Thesis Series, Delft University Press, The Netherlands

Hoogendoorn, S.P., *Multiclass Continuum Modelling of Multilane Traffic Flow*, T99/5, September 1999, TRAIL Thesis Series, Delft University Press, The Netherlands

Hoedemaeker, M., *Driving with Intelligent Vehicles: Driving Behaviour with Adaptive Cruise Control and the Acceptance by Individual Drivers*, T99/6, November 1999, TRAIL Thesis Series, Delft University Press, The Netherlands

Marchau, V.A.W.J., *Technology Assessment of Automated Vehicle Guidance - Prospects for Automated Driving Implementation*, T2000/1, January 2000, TRAIL Thesis Series, Delft University Press, The Netherlands

Subiono, *On Classes of Min-max-plus Systems and their Applications*, T2000/2, June 2000, TRAIL Thesis Series, Delft University Press, The Netherlands

Meer, J.R., van, *Operational Control of Internal Transport*, T2000/5, September 2000, TRAIL Thesis Series, Delft University Press, The Netherlands

Bliemer, M.C.J., *Analytical Dynamic Traffic Assignment with Interacting User-Classes: Theoretical Advances and Applications using a Variational Inequality Approach,* T2001/1, January 2001, TRAIL Thesis Series, Delft University Press, The Netherlands

Muilerman, G.J., *Time-based logistics: An analysis of the relevance, causes and impacts,* T2001/2, April 2001, TRAIL Thesis Series, Delft University Press, The Netherlands

Roodbergen, K.J., *Layout and Routing Methods for Warehouses*, T2001/3, May 2001, TRAIL Thesis Series, The Netherlands

Willems, J.K.C.A.S., *Bundeling van infrastructuur, theoretische en praktische waarde van*

een ruimtelijk inrichtingsconcept, T2001/4, June 2001, TRAIL Thesis Series, Delft University Press, The Netherlands

Binsbergen, A.J., van, J.G.S.N. Visser, *Innovation Steps towards Efficient Goods Distribution Systems for Urban Areas,* T2001/5, May 2001, TRAIL Thesis Series, Delft University Press, The Netherlands

Rosmuller, N., *Safety analysis of Transport Corridors*, T2001/6, June 2001, TRAIL Thesis Series, Delft University Press, The Netherlands

Schaafsma, A., *Dynamisch Railverkeersmanagement, besturingsconcept voor railverkeer op basis van het Lagenmodel Verkeer en Vervoer*, T2001/7, October 2001, TRAIL Thesis Series, Delft University Press, The Netherlands

Bockstael-Blok, W., *Chains and Networks in Multimodal Passenger Transport. Exploring a design approach*, T2001/8, December 2001, TRAIL Thesis Series, Delft University Press, The Netherlands

Wolters, M.J.J., *The Business of Modularity and the Modularity of Business*, T2002/1, February 2002, TRAIL Thesis Series, The Netherlands

Vis, F.A., *Planning and Control Concepts for Material Handling Systems*, T2002/2, May 2002, TRAIL Thesis Series, The Netherlands

Koppius, O.R., *Information Architecture and Electronic Market Performance,* T2002/3, May 2002, TRAIL Thesis Series, The Netherlands

Veeneman, W.W., *Mind the Gap; Bridging Theories and Practice for the Organisation of Metropolitan Public Transport*, T2002/4, June 2002, TRAIL Thesis Series, Delft University Press, The Netherlands

Van Nes, R., *Design of multimodal transport networks, a hierarchical approach,* T2002/5, September 2002, TRAIL Thesis Series, Delft University Press, The Netherlands

Pol, P.M.J., *A Renaissance of Stations, Railways and Cities, Economic Effects, Development Strategies and Organisational Issues of European High-Speed-Train Stations*, T2002/6, October 2002, TRAIL Thesis Series, Delft University Press, The Netherlands

Runhaar, H., *Freight transport: at any price? Effects of transport costs on book and newspaper supply chains in the Netherlands*, T2002/7, December 2002, TRAIL Thesis Series, Delft University Press, The Netherlands

Spek, S.C., van der, *Connectors. The Way beyond Transferring*, T2003/1, February 2003, TRAIL Thesis Series, Delft University Press, The Netherlands

Lindeijer, D.G., *Controlling Automated Traffic Agents*, T2003/2, February 2003, TRAIL Thesis Series, Eburon, The Netherlands

Riet, O.A.W.T., van de, *Policy Analysis in Multi-Actor Policy Settings. Navigating Between Negotiated Nonsense and Useless Knowledge*, T2003/3, March 2003, TRAIL Thesis Series, Eburon, The Netherlands

Reeven, P.A., van, *Competition in Scheduled Transport*, T2003/4, April 2003, TRAIL Thesis Series, Eburon, The Netherlands

Peeters, L.W.P., *Cyclic Railway Timetable Optimization*, T2003/5, June 2003, TRAIL The-

sis Series, The Netherlands

Soto Y Koelemeijer, G., *On the behaviour of classes of min-max-plus systems*, T2003/6, September 2003, TRAIL Thesis Series, The Netherlands

Lindveld, Ch..D.R., *Dynamic O-D matrix estimation: a behavioural approach*, T2003/7, September 2003, TRAIL Thesis Series, Eburon, The Netherlands

Weerdt, M.M., de, *Plan Merging in Multi-Agent Systems,* T2003/8, December 2003, TRAIL Thesis Series, The Netherlands

Langen, P.W, de, *The Performance of Seaport Clusters*, T2004/1, January 2004, TRAIL Thesis Series, The Netherlands

Hegyi, A., *Model Predictive Control for Integrating Traffic Control Measures*, T2004/2, February 2004, TRAIL Thesis Series, The Netherlands

Lint, van, J.W.C., *Reliable Travel Time Prediction for Freeways*, T2004/3, June 2004, TRAIL Thesis Series, The Netherlands

Tabibi, M., *Design and Control of Automated Truck Traffic at Motorway Ramps*, T2004/4, July 2004, TRAIL Thesis Series, The Netherlands

Verduijn, T. M., *Dynamism in Supply Networks: Actor switching in a turbulent business environment*, T2004/5, September 2004, TRAIL Thesis Series, The Netherlands

Daamen, W., *Modelling Passenger Flows in Public Transport Facilities*, T2004/6, September 2004, TRAIL Thesis Series, The Netherlands

Zoeteman, A., *Railway Design and Maintenance from a Life-Cycle Cost Perspective: A Decision-Support Approach*, T2004/7, November 2004, TRAIL Thesis Series, The Netherlands

Bos, D.M., *Changing Seats: A Behavioural Analysis of P&R Use*, T2004/8, November 2004, TRAIL Thesis Series, The Netherlands

Versteegt, C., *Holonic Control For Large Scale Automated Logistic Systems*, T2004/9, December 2004, TRAIL Thesis Series, The Netherlands

Wees, K.A.P.C. van, *Intelligente voertuigen, veiligheidsregulering en aansprakelijkheid. Een onderzoek naar juridische aspecten van Advanced Driver Assistance Systems in het wegverkeer*, T2004/10, December 2004, TRAIL Thesis Series, The Netherlands

Tampère, C.M.J., *Human-Kinetic Multiclass Traffic Flow Theory and Modelling: With Application to Advanced Driver Assistance Systems in Congestion*, T2004/11, December 2004, TRAIL Thesis Series, The Netherlands

Rooij, R.M., *The Mobile City. The planning and design of the Network City from a mobility point of view*, T2005/1, February 2005, TRAIL Thesis Series, The Netherlands

Le-Anh, T., *Intelligent Control of Vehicle-Based Internal Transport Systems*, T2005/2, April 2005, TRAIL Thesis Series, The Netherlands

Zuidgeest, M.H.P., *Sustainable Urban Transport Development: a Dynamic Optimization Approach*, T2005/3, April 2005, TRAIL Thesis Series, The Netherlands

Hoogendoorn-Lanser, S., *Modelling Travel Behaviour in Multimodal Networks*, T2005/4,

May 2005, TRAIL Thesis Series, The Netherlands

Dekker, S., *Port Investment - Towards an integrated planning of port capacity*, T2005/5, June 2005, TRAIL Thesis Series, The Netherlands

Koolstra, K., *Transport Infrastructure Slot Allocation*, T2005/6, June 2005, TRAIL Thesis Series, The Netherlands

Vromans, M., *Reliability of Railway Systems*, T2005/7, July 2005, TRAIL Thesis Series, The Netherlands

Oosten, W., *Ruimte voor een democratische rechtsstaat. Geschakelde sturing bij ruimtelijke investeringen*, T2005/8, September 2005, TRAIL Thesis Series, Sociotext, The Netherlands

Le-Duc, T., *Design and control of efficient order picking*, T2005/9, September 2005, TRAIL Thesis Series, The Netherlands

Goverde, R., *Punctuality of Railway Operations and Timetable Stability Analysis*, T2005/10, October 2005, TRAIL Thesis Series, The Netherlands

Kager, R.M., *Design and implementation of a method for the synthesis of travel diary data*, T2005/11, October 2005, TRAIL Thesis Series, The Netherlands

Boer, C., *Distributed Simulation in Industry*, T2005/12, October 2005, TRAIL Thesis Series, The Netherlands

Pielage, B.A., *Conceptual Design of Automated Freight Transport Systems*, T2005/14, November 2005, TRAIL Thesis Series, The Netherlands

Groothedde, B., *Collaborative Logistics and Transportation Networks, a modeling approach to network design*, T2005/15, November 2005, TRAIL Thesis Series, The Netherlands

Valk, J.M., *Coordination among Autonomous Planners*, T2005/16, December 2005, TRAIL Thesis Series, The Netherlands

Krogt, R.P.J. van der, *Plan Repair in Single-Agent and Multi-Agent Systems*, T2005/17, December 2005, TRAIL Thesis Series, The Netherlands

Bontekoning, Y.M., *Hub exchange operations in intermodal hub-and-spoke networks. A performance comparison of four types of rail-rail exchange facilities*,T2006/1, February 2006, TRAIL Thesis Series, The Netherlands

Lentink, R., *Algorithmic Decision Support for Shunt Planning*, T2006/2, February 2006, TRAIL Thesis Series, The Netherlands

Ngoduy, D., *Macroscopic Discontinuity Modeling for Multiclass Multilane Traffic Flow Operations*, T2006/3, April 2006, TRAIL Thesis Series, The Netherlands

Vanderschuren, M.J.W.A., *Intelligent Transport Systems for South Africa. Impact assessment through microscopic simulation in the South African context*, T2006/4, August 2006, TRAIL Thesis Series, The Netherlands

Ongkittikul, S., *Innovation and Regulatory Reform in Public Transport*, T2006/5, September 2006, TRAIL Thesis Series, The Netherlands

Yuan, J., *Stochastic Modelling of Train Delays and Delay Propagation in Stations*, T2006/6,

October 2006, TRAIL Thesis Series, The Netherlands

Viti, F., *The Dynamics and the Uncertainty of Delays at Signals*, T2006/7, November 2006, TRAIL Thesis Series, The Netherlands

Huisken, G., *Inter-Urban Short-Term Traffic Congestion Prediction*, T2006/8, December 2006, TRAIL Thesis Series, The Netherlands

Feijter, R. de, *Controlling High Speed Automated Transport Network Operations*, T2006/9, December 2006, TRAIL Thesis Series, The Netherlands

Makoriwa, C., *Performance of Traffic Networks. A mosaic of measures*, T2006/10, December 2006, TRAIL Thesis Series, The Netherlands

Miska, M., *Microscopic Online Simulation for Real time Traffic Management*, T2007/1, January 2007, TRAIL Thesis Series, The Netherlands

Chorus, C., *Traveler Response to Information*, T2007/2, February 2007, TRAIL Thesis Series, The Netherlands

Weijermars, W.A.M., *Analysis of Urban Traffic Patterns Using Clustering*, T2007/3, April 2007, TRAIL Thesis Series, The Netherlands

Zondag, B., *Joined Modeling of Land-use, Transport and Economy*, T2007/4, April 2007, TRAIL Thesis Series, The Netherlands

Bok, M.A. de, *Infrastructure and Firm Dynamics: A micro-simulation approach*, T2007/5, May 2007, TRAIL Thesis Series, The Netherlands

Fiorenzo-Catalano, M.S., *Choice Set Generation in Multi-Modal Transportation Networks*, T2007/6, June 2007, TRAIL Thesis Series, The Netherlands

Vlist, P. van der, *Synchronizing the retail supply chain*, T2007/7, June 2007, TRAIL Thesis Series, The Netherlands

Joksimovic, D., *Dynamic Bi-Level Optimal Toll Design Approach for Dynamic Traffic Networks*, T2007/8, September 2007, TRAIL Thesis Series, The Netherlands

Warffemius, P.M.J., *Modeling the Clustering of Distribution Centers around Amsterdam Airport Schiphol. Location Endowments, Economies of Agglomeration, Locked-in Logistics and Policy Implications*, T2007/9, September 2007, TRAIL Thesis Series, The Netherlands

Driel, C.J.G. van, *Driver Support in Congestion: an assessment of user needs and impacts on driver and traffic flow*, T2007/10, November 2007, TRAIL Thesis Series, The Netherlands

Gietelink, O.J., *Design and Validation of Advanced Driver Assistance Systems*, T2007/11, November 2007, TRAIL Thesis Series, The Netherlands

Nuttall, A.J.G., *Design Aspects of Multiple Driven Belt Conveyors*, T2007/12, November 2007, TRAIL Thesis Series, The Netherlands

Nederveen, A.A.J., *Ruimtelijke Inpassing van Lijninfrastructuur. Een onderzoek naar de geschiktheid van inspraakreacties voor het ontwerpen van lijninfrastructuur*, T2007/13, December 2007, TRAIL Thesis Series, The Netherlands

Negenborn, R.R., *Multi-Agent Model Predictive Control with Applications to Power Networks*, T2007/14, December 2007, TRAIL Thesis Series, The Netherlands

Samenvatting

Multi-Agent Modelgebaseerd Voorspellend Regelen met Toepassingen in Elektriciteitsnetwerken

Transportnetwerken, zoals elektriciteitsnetwerken, verkeersnetwerken, spoornetwerken, waternetwerken, etc., vormen de hoekstenen van onze moderne samenleving. Een soepele, eff ciënte, betrouwbare en veilige werking van deze netwerken is van enorm belang voor de economische groei, het milieu en de leefbaarheid, niet alleen wanneer deze netwerken op de grenzen van hun kunnen moeten opereren, maar ook onder normale omstandigheden. Aangezien transportnetwerken dichter en dichter bij hun capaciteitslimieten moeten werken, en aangezien de dynamica van dergelijke netwerken alsmaar complexer wordt, wordt het steeds moeilijker voor de huidige regelstrategieën om adequate prestaties te leveren onder alle omstandigheden. De regeling van transportnetwerken moet daarom naar een hoger niveau gebracht worden door gebruik te maken van nieuwe geavanceerde regelstrategieën.

Elektriciteitsnetwerken vormen een specif eke klasse van transportnetwerken waarvoor nieuwe regelstrategieën in het bijzonder nodig zijn. De structuur van elektriciteitsnetwerken is aan het veranderen op verschillende niveaus. Op Europees niveau worden de elektriciteitsnetwerken van individuele landen meer en meer geïntegreerd door de aanleg van transportlijnen tussen landen. Op nationaal niveau stroomt elektriciteit niet langer alleen van het transmissienetwerk via het distributienetwerk in de richting van bedrijven en steden, maar ook in de omgekeerde richting. Daarnaast wordt op lokaal niveau regelbare belasting geïnstalleerd en kan energie lokaal gegenereerd en opgeslagen worden. Om minimumeisen en -serviceniveaus te kunnen blijven garanderen, moeten *state-of-the-art* regeltechnieken ontwikkeld en geïmplementeerd worden.

In dit proefschrift stellen wij verschillende regelstrategieën voor die erop gericht zijn om de opkomende problemen in transportnetwerken in het algemeen en elektriciteitsnetwerken in het bijzonder het hoofd te bieden. Om het grootschalige en gedistribueerde karakter van de regelproblemen te beheersen gebruiken wij *multi-agent* aanpakken, waarin verschillende regelagenten elk hun eigen deel van het netwerk regelen en samenwerken om de best mogelijke netwerkbrede prestaties te behalen. Om alle beschikbare informatie mee te kunnen nemen en om vroegtijdig te kunnen anticiperen op ongewenst gedrag maken wij gebruik van modelgebaseerd voorspellend regelen (MVR). In de regelstrategieën die wij in dit proefschrift voorstellen, combineren wij multi-agent aanpakken met MVR. Hieronder volgt een overzicht van de regelstrategieën die wij voorstellen en de regelproblemen uit de specif eke klasse van elektriciteitsnetwerken, waarop wij de voorgestelde regelstrategieën toepassen.

Multi-agent modelgebaseerd voorspellend regelen

In een multi-agent regeling is de regeling van een systeem gedistribueerd over verschillende regelagenten. De regelagenten kunnen gegroepeerd worden aan de hand van de autoriteitsrelaties die tussen de regelagenten gelden. Een dergelijke groepering resulteert in een gelaagde regelstructuur waarin regelagenten in hogere lagen meer autoriteit hebben over regelagenten in lagere lagen en waarin regelagenten in dezelfde laag dezelfde autoriteitsrelaties met betrekking tot elkaar hebben. Gebaseerd op de ideeën van MVR bepalen in multi-agent MVR de regelagenten welke actie zij nemen aan de hand van voorspellingen. Deze voorspellingen maken zij met behulp van voorspellingsmodellen van die delen van het algehele systeem die zij regelen. Daar waar de regelagenten in hogere lagen typisch minder gedetailleerde modelen en langzamere tijdschalen beschouwen, beschouwen regelagenten op lagere regellagen typisch meer gedetailleerde modelen en snellere tijdschalen. In dit proefschrift worden de volgende regelstrategieën voorgesteld en bediscussieerd:

- Voor de coördinatie van regelagenten in een regellaag wordt een nieuw serieel schema voor multi-agent MVR voorgesteld en vergeleken met een bestaand parallel schema. In de voorgestelde aanpak wordt aangenomen dat de dynamica van de deelnetwerken alleen uit continue dynamica bestaat en dat de dynamica van het algehele netwerk gemodelleerd kan worden met verbonden lineaire tijdsinvariante modellen, waarin alle variabelen continue waarden aannemen.

- In de praktijk komt het regelmatig voor dat deelnetwerken hybride dynamica vertonen, veroorzaakt door zowel continue als discrete dynamica. We bediscussiëren hoe discrete dynamica gevat kan worden in modellen bestaande uit lineaire vergelijkingen en ongelijkheden en hoe regelagenten dergelijke modellen kunnen gebruiken bij het bepalen van hun acties. Daarnaast stellen wij een uitbreiding voor van de coördinatieschema's voor continue systemen naar systemen met continue en discrete variabelen.

- Voor een individuele regelagent die richtpunten bepaalt voor regelagenten in een lagere regellaag wordt het opzetten van object-georiënteerde voorspellingsmodellen bediscussieerd. Een dergelijk object-georiënteerd voorspellingsmodel wordt dan gebruikt om een MVR-regelprobleem te formuleren. Wij stellen voor om de optimalisatietechniek *pattern search* te gebruiken om het resulterende MVR-regelprobleem op te lossen. Daarnaast stellen wij omwille van de eff ciëntie een MVR-regelstrategie voor die gebaseerd is op een gelineariseerde benadering van het object-georiënteerde voorspellingsmodel.

- Regelmatig worden deelnetwerken gedef nieerd op basis van reeds bestaande netwerkregio's. Dergelijke deelnetwerken overlappen meestal niet. Als deelnetwerken echter gebaseerd worden op bijvoorbeeld invloedsgebieden van actuatoren, dan kunnen de deelnetwerken overlappend zijn. Wij stellen een regelstrategie voor voor het regelen van overlappende deelnetwerken door regelagenten in een hogere regellaag.

Multi-agent regelproblemen in elektriciteitsnetwerken

Elektriciteitsnetwerken vormen een specif eke klasse van transportnetwerken waarvoor de ontwikkeling van geavanceerde regeltechnieken noodzakelijk is om adequate prestaties te

behalen. De regelstrategieën die in dit proefschrift worden voorgesteld worden daarom aan de hand van toepassing op specif eke regelproblemen uit elektriciteitsnetwerken geëvalueerd. In het bijzonder worden de volgende regelproblemen besproken:

- We beschouwen een gedistribueerd *load-frequency* probleem, wat het probleem is van het dicht bij nul houden van frequentie-afwijkingen na verstoringen. Regelagenten regelen elk hun eigen deel van het netwerk en moeten samenwerken om de best mogelijke netwerkbrede prestaties te behalen. Om deze samenwerking te bewekstellingen gebruiken de regelagenten de seriële of de parallele MVR-strategieën. We beschouwen zowel samenwerking gebaseerd op voorspellingsmodellen die alleen continue variabelen bevatten, als met gebruikmaking van voorspellingsmodellen die zowel continue als ook discrete variabelen bevatten. Met behulp van simulaties illustreren we de prestaties die de schema's kunnen behalen.

- In de nabije toekomst zullen huishoudens de mogelijkheid hebben om hun eigen energie lokaal te produceren, lokaal op te slaan, te verkopen aan een energie-aanbieder en mogelijk uit te wisselen met naburige huishoudens. We stellen een MVR-strategie voor die gebruikt kan worden door een regelagent die het energiegebruik in een huishouden regelt. Deze regelagent neemt in zijn regeling verwachte energieprijzen, voorspelde energieconsumptiepatronen en de dynamica van het huishouden mee. We illustreren de prestaties die de regelagent kan behalen voor een gegeven scenario van energieprijzen en consumptiepatronen.

- Spanningsinstabiliteiten vormen een belangrijke bron van elektriciteitsuitval. Om te voorkomen dat spanningsinstabiliteiten ontstaan is lokaal bij generatielokaties een laag van regelagenten geïnstalleerd. Een dergelijke lokale regeling werkt onder normale omstandigheden goed, maar levert ten tijde van grote verstoringen geen adequate prestaties. In dergelijke situaties moeten de acties van de lokale regelagenten gecoördineerd worden. Wij stellen een MVR-regelagent voor die tot taak heeft deze coördinatie te realiseren. De voorgestelde MVR-strategie maakt gebruik van ofwel een object-georiënteerd model van het elektriciteitsnetwerk ofwel van een benadering van dit model verkregen na linearisatie. We illustreren de prestaties die behaald kunnen worden met behulp van simulaties op een dynamisch 9-bus elektriciteitsnetwerk.

- Regeling gebaseerd op *optimal power flow* (OPF) kan gebruikt worden om in transmissienetwerken de *steady-state* spanningsprof elen te verbeteren, het overschrijden van capaciteitslimieten te voorkomen, en vermogensverliezen te minimaliseren. Een type apparaat waarvoor met behulp van OPF-regeling actuatorinstellingen bepaald kunnen worden zijn *flexible alternating current transmission systems* (FACTS). Wij beschouwen een situatie waarin verschillende FACTS-apparaten aanwezig zijn en elk FACTS-apparaat geregeld wordt door een regelagent. Elke regelagent beschouwt als zijn deelnetwerk dat deel van het netwerk dat zijn FACTS-apparaat kan beïnvloeden. Aangezien de deelnetwerken gebaseerd zijn op beïnvloedingsregio's kunnen verschillende deelnetwerken overlappend zijn. Wij stellen een coördinatie- en communicatieschema voor dat kan omgaan met een dergelijke overlap. Via simulatiestudies op een aangepast elektriciteitsnetwerk met 57 bussen illustreren we de prestaties.

Rudy R. Negenborn

Summary

Multi-Agent Model Predictive Control with Applications to Power Networks

Transportation networks, such as power distribution and transmission networks, road traffic networks, water distribution networks, railway networks, etc., are the corner stones of modern society. A smooth, efficient, reliable, and safe operation of these systems is of huge importance for the economic growth, the environment, and the quality of life, not only when the systems are pressed to the limits of their performance, but also under regular operating conditions. As transportation networks have to operate closer and closer to their capacity limits and as the dynamics of these networks become more and more complex, currently used control strategies can no longer provide adequate performance in all situations. Hence, control of transportation networks has to be advanced to a higher level using novel control techniques.

A class of transportation networks for which such new control techniques are in particular required are power networks. The structure of power networks is changing at several levels. At a European level the electricity networks of the individual countries are becoming more integrated as high-capacity power lines are constructed to enhance system security. At a national level power does not any longer only flow from the transmission network in the direction of the distribution network and onwards to the industrial sites and cities, but also in the other direction. Furthermore, at the local level controllable loads are installed, energy can be generated locally with small-scale generators, and energy can be stored locally using batteries. To still guarantee basic requirements and service levels and to meet the demands and requirements of the users while facing the changing structure of power networks, state-of-the-art control techniques have to be developed and implemented.

In this PhD thesis we propose several new control techniques designed for handling the emerging problems in transportation networks in general and power networks in particular. To manage the typically large size and distributed nature of the control problems encountered, we employ multi-agent approaches, in which several control agents each control their own part of the network and cooperate to achieve the best possible overall performance. To be able to incorporate all available information and to be able to anticipate undesired behavior at an early stage, we use model predictive control (MPC).

Next we give a summary of the control techniques proposed in this PhD thesis and the control problems from a particular class of transportation networks, viz. the class of power networks, to which we apply the proposed control techniques in order to assess their

performance.

Multi-agent model predictive control

In multi-agent control, control is distributed over several control agents. The control agents can be grouped according to the authority relationships that they have among each other. The result is a layered control structure in which control agents at higher layers have authority over control agents in lower layers, and control agents within a control layer have equal authority relationships. In multi-agent MPC, control agents take actions based on predictions that they make using a prediction model of the part of the overall system they control. At higher layers typically less detailed models and slower time scales are considered, whereas at lower layers more detailed models and faster time scales are considered.

In this PhD thesis the following control strategies for control agents at various locations in a control structure are proposed and discussed:

- For coordination of control agents within a control layer a novel serial scheme for multi-agent MPC is proposed and compared with an existing parallel scheme. In the approach it is assumed that the dynamics of the subnetworks that the control agents control are purely continuous and can be modeled with interconnected linear discrete-time time-invariant models in which all variables take on continuous values.

- In practice, the dynamics of the subnetworks may show hybrid dynamics, caused by both continuous and discrete dynamics. We discuss how discrete dynamics can be captured by systems of linear equalities and inequalities and how control agents can use this in their decision making. In addition, we propose an extension of the coordination schemes for purely continuous systems that deals with interconnected linear time-invariant subnetworks with integer inputs.

- For an individual control agent that determines set-points for control agents in a lower control layer, creating object-oriented prediction models is discussed. Such an object-oriented prediction model is then used to formulate an MPC control problem. We propose to use the optimization technique pattern search to solve the resulting MPC control problem. In addition, for eff ciency reasons, we propose an MPC control strategy based on a linearization of the object-oriented prediction model.

- Commonly, subnetworks are def ned based on already existing network regions. Such subnetworks typically do not overlap. However, when subnetworks are based on, e.g., regions of inf uence of actuators, then the subnetworks may be overlapping. For multiple control agents in a higher control layer, at which it can be assumed that the behavior of the underlying control layers is static, we propose an MPC strategy for control of overlapping subnetworks.

Multi-agent control problems in power networks

Power networks are a particular class of transportation networks and are subject to a changing structure. This changing structure requires the development of advanced control techniques in order to maintain adequate control performance. The control strategies proposed

in this PhD thesis are applied to and assessed on specif c power domain control problems. In particular, we discuss the following power network problems and control approaches:

- We consider a distributed load-frequency control problem, which is the problem of maintaining frequency deviations after load disturbances close to zero. Control agents each control their own part of the network and have to cooperate in order to achieve the best possible overall network performance. The control agents achieve this by obtaining agreement on how much power should f ow among the subnetworks. The serial and parallel MPC strategies are employed for this, both when the prediction models involve only continuous variables, and when the prediction models involve both continuous and discrete variables. In simulations we illustrate the performance that the schemes can obtain.

- In the near future households will be able to produce their own energy, store it locally, sell it to an energy supplier, and perhaps exchange it with neighboring households. We propose an MPC strategy to be used by a control agent controlling the energy usage in a household. This control agent takes into account expected energy prices, predicted energy consumption patterns, and the dynamics of the household, including dynamics of local energy generation and storage devices. For a given scenario of energy prices and consumption patterns, the performance that the control agent can achieve are illustrated.

- Voltage instability is a major source of power outages. To prevent voltage instability from emerging, a lower layer of control agents is installed in power networks at generation sites. These agents locally adjust generation to maintain voltage magnitudes. Such local control works well under normal operating conditions. However, under large disturbances such local control does not provide adequate performance. In such situations, the actions of the local control agents have to be coordinated. We propose an MPC control agent that has the task to coordinate the local control agents. The MPC strategy that the agent uses is based on either an object-oriented model of the power network or on a linearized approximation of this model. The object-oriented model includes a model of the physical network and the local control agents. We illustrate the performance of the MPC control agent using the object-oriented model or the linearized approximation via simulations on a dynamic 9-bus power network.

- Optimal power f ow control is commonly used to improve steady-state power network security by improving the voltage prof le, preventing lines from overloading, and minimizing active power losses. Using optimal power f ow control, device settings for f exible alternating current transmission systems (FACTS) can be determined. We consider the situation in which there are several FACTS devices, each controlled by a different control agent. The subnetwork that each control agent considers consists of a region of inf uence of its FACTS device. Since the subnetworks are based on regions of inf uence, the subnetworks of several agents may be overlapping. We propose a coordination and communication scheme that takes this overlap into account. In simulation experiments on an adjusted 57-bus IEEE power network the performance of the scheme is illustrated.

Rudy R. Negenborn

Curriculum vitae

Rudy R. Negenborn was born on June 13, 1980 in Utrecht, The Netherlands. He finished his pre-university education (*VWO*) in 1998 at the Utrechts Stedelijk Gymnasium, Utrecht, The Netherlands. After this, Rudy Negenborn started his studies in Computer Science at the Utrecht University, Utrecht, The Netherlands. He received the title of *doctorandus* (comparable with Master of Science) in Computer Science, with a specialization in Intelligent Systems, *cum laude* from this university in 2003. For his graduation project, he performed research on Kalman filtering and robot localization. The research involved in this project was carried out during a one-year visit to the Copenhagen University, Denmark, and was supervised by Prof.Dr.Phil. P. Johansen and Dr. M. Wiering.

Since 2004, Rudy Negenborn has been working on his PhD project at the Delft Center for Systems and Control of Delft University of Technology, The Netherlands. The research of his PhD project has been on multi-agent model predictive control with applications to power networks, and has been supervised by Prof.dr.ir. B. De Schutter and Prof.dr.ir. J. Hellendoorn. During his PhD project, Rudy Negenborn obtained the DISC certificate for fulfilling the course program requirements of the Dutch Institute for Systems and Control. Furthermore, he cooperated with and spent time at various research groups, including the Hybrid System Control Group of Supélec, Rennes, France, and the Power Systems Laboratory and Automatic Control Laboratory of ETH Zürich, Zürich, Switzerland.

Rudy Negenborn's more fundamental research interests include multi-agent systems, hybrid systems, distributed control, and model predictive control. His more applied research interests include applications to transportation networks in general, and power networks in particular.

Since 2004, Rudy Negenborn has been a member of the DISC and of The Netherlands Research School for Transport, Infrastructure, and Logistics (TRAIL). Moreover, from 2004 until 2007, Rudy Negenborn fulfilled the positions of public relations representative and treasurer in the board of Promood, the representative body of the PhD candidates at Delft University of Technology.