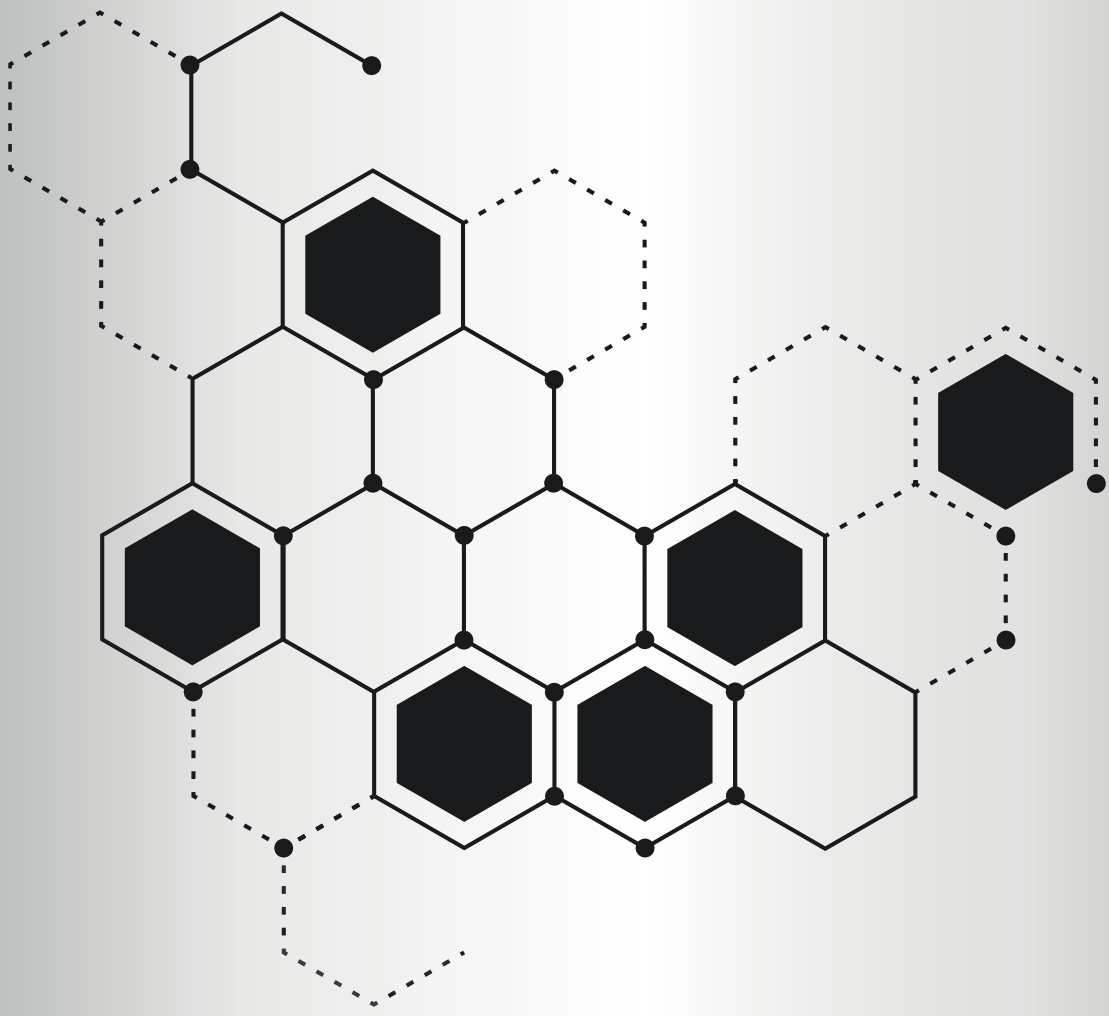
DISTRIBUTED AND MULTI-LEVEL PREDICTIVE CONTROL: PARTITIONING AND ABSTRACTION



Alessandro Riccardi

Distributed and Multi-Level Predictive Control: Partitioning and Abstraction

Distributed and Multi-Level Predictive Control: Partitioning and Abstraction

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus,
Prof. dr. ir. H. Bijl,
chair of the Board for Doctorates
to be defended publicly on
Friday, 22 May 2026, 12:30

by

Alessandro RICCARDI

This thesis has been approved by the promotors and the copromotor.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. ir. B. De Schutter	Delft University of Technology, Netherlands, *promotor*
Dr. L. Laurenti	Delft University of Technology, Netherlands, *copromotor*

Independent members:

Prof. dr. ir. J.W. van Wingerden,	Delft University of Technology, Netherlands
Prof. dr. R.R. Negenborn,	Delft University of Technology, Netherlands
Prof. dr. A. Parisio,	The University of Manchester, United Kingdom
Prof. dr. J.M. Maestre Torreblanca,	University of Seville, Spain
Dr. M. Lazar,	Eindhoven University of Technology, Netherlands

This thesis has been completed in fulfillment of the requirements of the Dutch Institute of Systems and Control (DISC) for graduate studies.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant agreement No. 101018826) – Project CLariNet.

Keywords: Model Predictive Control, Hybrid Systems, Partitioning, Abstraction, Multi-Agent Systems, Multi-Level Control, Distributed Control

Printed by: www.ridderprint.nl

Style: TU Delft House Style, with modifications by Moritz Beller
https://github.com/Inventitech/phd-thesis-template

ISBN: 978-94-6518-315-2

An electronic version of this dissertation is available at
https://repository.tudelft.nl/.

To my family, for the immovable roots;
To my friends, for the impalpable lightness;
To my partner, for the unconditional love.

Contents

Summary

The evolution of communication and computing technologies of the recent decades has enabled the rapid development and scaling of networks of systems. Consequently, modern networks of systems present complexities and geographical extents for which traditional monitoring, planning, and control paradigms based on centralized, or even human-driven, operation are not sufficient anymore to guarantee efficient and safe operation. For such systems, more sophisticated control strategies are required for nominal functioning and further extension. While the availability of information, given by real-time communication, and the computing power, generally accessible for large applications, are no longer a fundamentally limiting factor in modern networks, the same cannot be stated for the control technologies behind their operation. The achievement of complete non-centralization of control decisions and actions, as well as the satisfaction of complex requirements for safe network operations, preservation, and restoration, are among the main drivers of the future development of networks. Pursuing the achievement of these advanced specifications, of profound societal relevance, and the necessity of improving performance and efficiency are at the basis of current research in the field of systems and control of networks. This thesis approaches some of these topics and consists of two main parts.

In the first part, we consider the problem of network partitioning for non-centralized Model Predictive Control (MPC), which is of great relevance to current network operations. An MPC control strategy is based on the online solution of an optimization program that provides the best sequence of future control actions for a given performance indicator and accounting for the prediction of the future behavior of the system given by a mathematical model. Despite its flexibility and advantages, conventional centralized MPC control is difficult to scale for large applications. Non-centralized MPC approaches are a solution for such an issue, and they are always based on a partitioning of the network. In its essence, the partitioning problem consists in finding the best subdivision of a network into fundamental subsystems for which local controllers can be defined for non-centralized operation. The control community has devoted great efforts in finding clever solutions to this problem, which is, by its own nature, too computationally expensive and complex to be solved exactly in real time, even for relatively small networks. The present work proposes a general strategy for solving the partitioning problem for non-centralized MPC control. We start by developing in Chapter 2 a benchmark for electricity networks for testing non-centralized strategies and showing the limitations of centralized MPC. Then, in Chapter 3, we propose our generalized partitioning strategy, which is based on two steps: the selection of fundamental systems of the network, followed by an aggregation procedure that minimizes a global performance metric. We show through control simulations on networks of various topologies and complexity that our approach allows for achieving superior computational or control performance. In Chapter 4, we provide a systematization of the various partitioning approaches for non-centralized MPC that have been developed in the literature. Together with this systematization, we also provide the new theoretical

concepts of multi-topological network representations and predictive partitioning that can guide future research in this field, and lead to the development of partitioning strategies able to proactively adapt to topological changes in the networks.

In the second part of this thesis, we approach the problems related to the satisfaction of complex specifications for systems in which performance is also of primary relevance. We achieve this objective with the introduction of a multi-level control framework that is discussed in Chapter 5. In particular, we consider nonlinear stochastic systems that have to meet safety-critical requirements while optimizing a performance criterion. One approach for obtaining actions that satisfy complex specifications is based on the abstraction of the system through interval Markov decision processes (IMDPs), and of robust dynamic programming for obtaining a policy in the abstracted domain. In this context, we introduce a novel IMDP formulation that, instead of using sampled points of the input space, allows for the definition of optimization regions in the form of intervals of the input space. We reconstruct this policy from the abstract level to the real-time optimization level using an MPC controller based on hybrid dynamical systems. We show how, with this novel multi-level approach, we are able to obtain superior control performance for nonlinear stochastic systems while guaranteeing the satisfaction of temporal specifications within a predefined probability lower bound.

We conclude the discussion about these topics in non-centralized and multi-level control in Chapter 6, where we also offer suggestions for further development of each individual research topic and the integration of the two lines of research. In summary, this thesis addresses two key problems in the MPC-based control of complex large-scale systems: partitioning for non-centralized MPC, with the derivation of a generalized strategy and a systematization of the existing work in the literature; and multi-level control for nonlinear stochastic systems that have to satisfy temporal specifications, integrating abstraction-based policy synthesis with model-based online optimization through hybrid systems.

Samenvatting

De evolutie van communicatie- en computertechnologieën in de afgelopen decennia heeft een snelle ontwikkeling en schaalvergroting van netwerken van systemen mogelijk gemaakt. Als gevolg daarvan vertonen moderne netwerken van systemen een complexiteit en geografische omvang waardoor traditionele monitoring-, planning- en controleparadigma's op basis van gecentraliseerde of zelfs door mensen aangestuurde operaties niet langer toereikend zijn om een efficiënte en veilige werking te garanderen. Voor dergelijke systemen zijn meer geavanceerde controlestrategieën nodig voor een nominale werking en verdere uitbreiding. Hoewel de beschikbaarheid van informatie, dankzij realtime communicatie, en de rekenkracht, die over het algemeen toegankelijk is voor grote toepassingen, niet langer een fundamentele beperkende factor zijn in moderne netwerken, geldt dat niet voor de controletechnologieën achter hun werking. Het bereiken van volledige decentralisatie van controlebeslissingen en -acties, evenals het voldoen aan complexe eisen voor veilige netwerkoperaties, behoud en herstel, behoren tot de belangrijkste drijfveren voor de toekomstige ontwikkeling van netwerken. Het nastreven van deze geavanceerde specificaties, die van groot maatschappelijk belang zijn, en de noodzaak om de prestaties en efficiëntie te verbeteren, vormen de basis van het huidige onderzoek op het gebied van systemen en besturing van netwerken. Dit proefschrift behandelt een aantal van deze onderwerpen en bestaat uit twee hoofddelen.

In het eerste deel bekijken we het probleem van netwerkpartitionering voor niet-gecentraliseerde Model Predictive Control (MPC), wat van groot belang is voor de huidige netwerkactiviteiten. Een MPC-regelstrategie is gebaseerd op de online oplossing van een optimalisatieprogramma dat de beste reeks toekomstige regelacties biedt voor een bepaalde prestatie-indicator en rekening houdt met de voorspelling van het toekomstige gedrag van het systeem, gegeven door een wiskundig model. Ondanks zijn flexibiliteit en voordelen is conventionele gecentraliseerde MPC-regeling moeilijk op te schalen voor grote toepassingen. Niet-gecentraliseerde MPC-benaderingen bieden een oplossing voor dit probleem en zijn altijd gebaseerd op een partitionering van het netwerk. In wezen bestaat het partitioneringsprobleem uit het vinden van de beste onderverdeling van een netwerk in fundamentele subsystemen waarvoor lokale regelaars kunnen worden gedefinieerd voor niet-gecentraliseerde werking. De besturingsgemeenschap heeft zich enorm ingespannen om slimme oplossingen te vinden voor dit probleem, dat vanwege zijn aard te rekenintensief en complex is om in realtime exact op te lossen, zelfs voor relatief kleine netwerken. In dit werk wordt een algemene strategie voorgesteld voor het oplossen van het partitioneringsprobleem voor niet-gecentraliseerde MPC-besturing. We beginnen met het ontwikkelen van een benchmark voor elektriciteitsnetwerken in Hoofdstuk 2 om niet-gecentraliseerde strategieën te testen en de beperkingen van gecentraliseerde MPC aan te tonen. Vervolgens stellen we in Hoofdstuk 3 onze gegeneraliseerde partitioneringsstrategie voor, die gebaseerd is op twee stappen: de selectie van fundamentele systemen van het netwerk, gevolgd door een aggregatieprocedure die een globale prestatiemaatstaf minimaliseert. We

tonen aan door middel van controlesimulaties op netwerken met verschillende topologieën en complexiteit dat onze aanpak superieure reken- of controleprestaties mogelijk maakt. In Hoofdstuk 4 bieden we een systematisering van de verschillende partitioneringsbenaderingen voor niet-gecentraliseerde MPC die in de literatuur zijn ontwikkeld. Samen met deze systematisering bieden we ook de nieuwe theoretische concepten van multi-topologische netwerkrepresentaties en voorspellende partitionering die als leidraad kunnen dienen voor toekomstig onderzoek op dit gebied en kunnen leiden tot de ontwikkeling van partitioneringsstrategieën die zich proactief kunnen aanpassen aan topologische veranderingen in de netwerken.

In het tweede deel van dit proefschrift behandelen we de problemen die verband houden met het voldoen aan complexe specificaties voor systemen waarin prestaties ook van primair belang zijn. We bereiken dit doel met de introductie van een meerlagig controlekader dat wordt besproken in Hoofdstuk 5. We kijken met name naar niet-lineaire stochastische systemen die moeten voldoen aan veiligheidskritische eisen en tegelijkertijd een prestatiecriterium moeten optimaliseren. Een benadering voor het verkrijgen van acties die voldoen aan complexe specificaties is gebaseerd op de abstractie van het systeem door middel van interval Markov-beslissingsprocessen (IMDP's) en robuuste dynamische programmering voor het verkrijgen van een beleid in het geabstraheerde domein. In deze context introduceren we een nieuwe IMDP-formulering die, in plaats van gebruik te maken van bemonsterde punten van de invoerruimte, de definitie van optimalisatieregio's in de vorm van intervallen van de invoerruimte mogelijk maakt. We reconstrueren dit beleid van het abstracte niveau naar het realtime optimalisatieniveau met behulp van een MPC-controller op basis van hybride dynamische systemen. We laten zien hoe we met deze nieuwe meerlaagse benadering superieure regelprestaties voor niet-lineaire stochastische systemen kunnen verkrijgen en tegelijkertijd kunnen garanderen dat aan de temporele specificaties wordt voldaan binnen een vooraf gedefinieerde ondergrens voor de waarschijnlijkheid.

We sluiten de discussie over deze onderwerpen in niet-gecentraliseerde en meerlagige besturing af in Hoofdstuk 6, waar we ook suggesties doen voor de verdere ontwikkeling van elk afzonderlijk onderzoeksonderwerp en de integratie van de twee onderzoekslijnen. Samenvattend behandelt dit proefschrift twee belangrijke problemen in de MPC-gebaseerde besturing van complexe grootschalige systemen: partitionering voor niet-gecentraliseerde MPC, met de afleiding van een gegeneraliseerde strategie en een systematisering van het bestaande werk in de literatuur; en meerlagige besturing voor niet-lineaire stochastische systemen die moeten voldoen aan temporele specificaties, waarbij abstractiegebaseerde beleidssynthese wordt geïntegreerd met modelgebaseerde online optimalisatie via hybride systemen.

1

Introduction

"Evolution brings human beings.
Human beings, through a long and painful process, bring humanity."

Dan Simmons, Hyperion, 1989

The scale and complexity of modern networks of systems present remarkable challenges for the development of their controllers. Such challenges include, among others, the computational complexity associated with the solution of large optimization problems, the real time communication over vast geographical distances, the preservation of the privacy of the users, and the secure and resilient operation of the network. The properties of such networks characterize them as complex large-scale multi-agent systems. Non-centralized control architectures are emergent solutions for large-scale systems. However, in addition to traditional regulation and tracking objectives, multi-agent and complex systems require the ability to comply with logic-driven and temporal specifications, which reside within the domain of hybrid systems and automata. The aim of this dissertation is to address specific challenges associated with non-centralized and complex control of modern networks. This chapter provides a description of the problems studied in this work and of the strategies developed to address them, and it is subdivided as follows: In Sec. 1.1 we introduce the concept of complex large-scale multi-agent networks, in Sec. 1.2 the research lines developed throughout the thesis are motivated, and the associated objectives defined; in Sec. 1.3 the main contributions of the research are presented; and in Sec. 1.4 the overall structure of the thesis is outlined.

1

1.1 Complex Large-Scale Multi-Agent Networks

Networks appear in many forms, from the most notorious ones, e.g. communication networks (Internet, World Wide Web) [86, 111], electricity networks [152, 262, 269] and transportation networks [201, 308], to the most futuristic and advanced, such as constellations of satellites [79] or swarms of aerial robots [357]. The maturity of the field, and of the technology that concerns these systems, is such that some of the challenges associated with network control today appear to be solved or close to a solution. In fact, concepts such as plug-and-play operation, autonomous formation management, and smart electricity production seem to be part of our daily lives. While the complete realization of our wishes for network autonomy in a short time is desirable, the reality is that many technical and conceptual challenges remain to be solved, and lively scientific research delivers novel solutions and open problems to address every day [160, 338, 352].

A network, in its simplest form, is a collection of systems interconnected through a specific topology, i.e. a graph [351]. For example, a basic electricity network is composed of systems such as generators and loads connected by transmission lines that carry electricity among them [42]. When the scale of these networks, even if constituted by simple elements, grows beyond the concrete possibility of controlling them in real time using a single central unit, we talk about large-scale networks [172]. For them, it is necessary to deploy non-centralized control strategies [295, 303], a field of control research that has been among the most active in the past decades [260]. With the evolution of computing and telecommunication technologies, the basic systems in the network have been gradually equipped with advanced communication and coordination abilities oriented at improving the performance and stability of the overall architecture. The fundamental systems in a network are now considered agents capable of exerting advanced abilities, and the resulting networks are large-scale multi-agent networks [87]. For them, the development of advanced non-centralized strategies that leverage the ability of the local controllers to adapt to topological and structural changes in the network, potentially in real time, is a field of active research. The next step of network evolution resides in the ability to perform complex logical tasks, not only based on pure control specification but also on logical conditions or temporal specifications [186]. Here is where the line between the past and future of network control lies, and where the research in this thesis takes place. Consequently, the present work is articulated into two main parts that are discussed in the next section.

1.2 Motivation of the Research and Objectives

The first part of this thesis is devoted to the study and advancement of the broad field of network partitioning for non-centralized control, with a specific focus on distributed predictive control. Centralized control, specifically centralized Model Predictive Control (MPC), is not a viable option for large-scale systems because, for such systems, it cannot deliver real-time control actions due to the scale of the associated optimization program. In the literature, known solutions that address this limitation are non-centralized MPC architectures, i.e. decentralized, distributed, and hierarchical MPC. In decentralized MPC, a local controller is developed independently for each system in the network, and the effect of neighboring systems is either neglected or compensated for using robustness arguments. In

distributed MPC, a communication protocol is introduced to achieve coordination between the local controllers of the independent subsystems. In hierarchical MPC, local controllers are coordinated through a supervisory layer that has access to local information and defines the global control objective. In this context, the role of partitioning is to aggregate local systems into composite units for which coordinated control actions allow achieving superior non-centralized performance or lower computational complexity. The network partitioning problem has been extensively studied, both in general network theory and control theory. However, the partitioning problem for non-centralized MPC has never been approached from a general perspective, thus considering the development of strategies that are applicable across different techniques, in real time, and for different network architectures, e.g. agent-oriented or monolithic. These aspects are deeply investigated in the present work, allowing the introduction of novel concepts for partitioning networks of multi-agent systems and new algorithmic and optimization-based partitioning techniques. In addition, due to the interdisciplinary nature of the topic of network partitioning for non-centralized predictive control, there is no specific reference in the literature serving as a fixed basis for the development of the strategies. This fact motivated us to provide a comprehensive systematization of the field in the form of a survey, which also introduces novel theoretical concepts.

The second part of the research concerns complex systems and their associated specifications. In particular, we focus on nonlinear stochastic systems that have to satisfy safety-critical specifications in the form of temporal logic requirements. In the literature, one popular approach for obtaining a control law for such systems is based on the abstraction framework, which allows for deriving a coarse description of the system in the form of a Markov Decision Process (MDP). Once the system has been abstracted, either on the basis of a model obtained through first principles or driven by data, temporal logic specifications can be converted into an automaton, which directly interfaces with the MDP, and robust dynamic programming can be used to obtain a policy that satisfies the specification with a guaranteed probability bound. This framework, despite its maturity and expressive power, does not consider performance optimization while defining the policy. Our objective is to develop a multi-level control strategy that enables the integration of abstraction-based policy systems with online model-based optimization by leveraging the logic capabilities characteristic of hybrid systems.

1.3 Contributions

The contributions of this thesis are based on the two main research objectives presented in the previous section and are detailed in the following:

1. **Development of a real-world inspired benchmark for load-frequency control of power networks**: In control theory, the applicability of the mathematical results is of fundamental relevance. Therefore, we developed a benchmark for the load-frequency control problem of power networks using real-world data to test our research hypothesis and systematically compare our results. This benchmark is based on one of the most relevant control problems of modern society, i.e. the load frequency control problem in the presence of renewable energy sources with the support of energy storage systems.

2. **Unification of top-down and bottom-up partitioning approaches for large-scale systems**: In non-centralized control, the different strategies either assume that a partition of the network in terms of control agents is given a priori, or that the network is a monolithic system to be decomposed. These diametrically opposite assumptions define the top-down and bottom-up partitioning strategies. We introduce a unification of these approaches through the concepts of fundamental system units and composite system units.

3. **Development of a generalized partitioning strategy for the application of non-centralized MPC control**: Leveraging the concept of fundamental system units, we develop a partitioning strategy based on their allocation into bigger entities, the composite system units, for which local controllers can be developed. The generalized strategy we develop is based on the optimization of a novel global network metric. The resulting partitions show significant advantages in terms of computational cost for networks of linear systems, and remarkable improvements in control performance for networks of hybrid systems with different topologies.

4. **Systematization of the field of partitioning for non-centralized MPC**: We develop a systematic survey of the works concerning partitioning focused on non-centralized predictive control. In this context, we provide a classification of the works under several interpretation frameworks. We categorize them in terms of the approach they use to achieve the decomposition, e.g. based on an algorithmic procedure or guided by the definition of an optimization problem; according to the objective of the strategy, e.g. the optimization of a criterion or the pursuit of a particular network structure; and on the basis of control-oriented criteria. In addition, we introduce a classification based on the non-centralized MPC architecture used, and the domain of application, thereby completing the discussion by reporting relevant benchmarks when available.

5. **Novel theoretical insights for partitioning in non-centralized MPC**: Guided by the comprehensive classification of the works in this domain, we propose new theoretical concepts for the future development of partitioning strategies. We propose the concept of multi-topological network representations using hybrid systems to characterize dynamic topologies driven by logical conditions, either generated by events or by changes in the system conditions (state). In addition, we introduce the concept of predictive partitioning for control, which may lead in the future to the development of adaptive strategies that, using a model of network dynamics, can adapt the definition of the agents in the network for improved performance or reliability of the network. Finally, we formalize a set of metrics and a systematic procedure for the assessment of the quality of a partition.

6. **Definition of the abstraction-driven MPC controller**: To address the problem of controlling stochastic non-linear systems that require the satisfaction of complex specifications, we propose a new multi-level framework for the integration of a policy satisfying temporal logic specification with online optimization. We achieve this integration through the use of interval MDPs (IMDPs) and hybrid dynamical systems. The resulting architecture allows for achieving superior control performance for

stochastic nonlinear systems while ensuring the satisfaction of complex specifications within a certain probability bound.

1.4 Structure of the Thesis

In this section, the structure of the thesis is presented, and its graphical representation is provided in Fig. 1.1, where the current connections among the topics discussed are represented by dashed arrows.

In Chapter 2, we develop a benchmark that considers the Load Frequency Control (LFC) problem for a power network that represents an abstraction of the European economic area's network, namely the EEA-ENB [280]. This is a large-scale network where each electrical area is an individual agent, i.e. the overall structure is a multi-agent system. We show how, with real-world data, a centralized MPC controller cannot deliver real-time control actions. Additionally, we have used the system to show the benefits of a partitioning strategy in [278], to develop an integrated optimization- and learning-based controller in [317], and we have extended the benchmark to hybrid systems in [279].

We consider the problem of partitioning for non-centralized MPC in Chapter 3, and develop a partitioning strategy in two phases. First, we introduce an algorithmic procedure that allows the definition of fundamental system units for monolithic networks. In the second phase, we aggregate the fundamental units into composite units using a global metric to assess the quality of the partition. We achieve this decomposition using both an optimization-based approach and an algorithmic procedure. The composite units are then used for the definition of local controllers. We show the difference in the quality of the resulting partitions by implementing for them a distributed MPC control approach in networks of hybrid dynamical systems with different topologies [282]. A previous genetic-algorithm-based implementation of the work has been deployed on the EEA-ENB [278].

In Chapter 4, we develop a systematic review of the field of partitioning for non-centralized predictive control [283]. We categorize the strategies based on the partitioning methodology, objective, target control strategy, and application domain. We complement the study by introducing novel concepts for the future development of the field, including a systematic procedure to assess the quality of the partition and its relevant metrics, multi-topological graph representations to represent logic-driven behaviors in dynamic topologies, and the novel concept of predictive partitioning for control.

We present, in Chapter 5, the integration of abstraction-based policy synthesis and online performance optimization [275]. The scope is to define a control strategy for nonlinear stochastic systems that allows for the simultaneous satisfaction of temporal logic specifications and the optimization of a desired cost function. The multi-level framework is based on the abstraction of nonlinear stochastic systems using a modified formulation of IMPDs, which introduces input intervals in the model. The policy resulting from the application of robust dynamic programming on this iMDP guarantees the satisfaction of the specification for every input selected from the input intervals considered. Such intervals are the basis for the definition of a mixed-integer optimization program on the lower level of the control architecture. Then, the optimizer selects, across the possible model-based future system trajectories, the best sequence of input intervals and the best inputs within the intervals for performance optimization. This logic-driven choice is possible thanks to

1

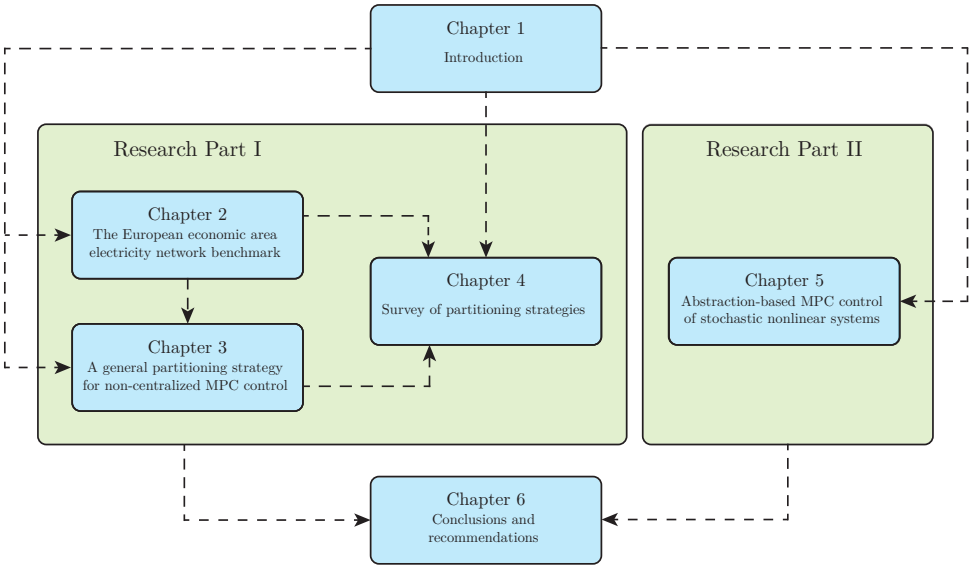


Figure 1.1: Structure of this thesis

the hybrid nature of the resulting architecture.

We complete the discussion about the research topics introduced in Chapter 6, where we present the conclusions of the present work, and provide recommendations for the future development and integration of the techniques we derived.

2

The European Economic Area Electricity Network Benchmark

"The real world was full of larger structures, smaller structures, simpler and more complex structures than the tiny portion comprising sentient creatures and their societies, and it required a profound myopia of scale and similarity to believe that everything beyond this shallow layer could be ignored."

Greg Egan, Diaspora, 1997

The European Economic Area Electricity Network Benchmark (EEA-ENB) is a multi-area power system representing the European network of transmission systems for electricity to facilitate the application of distributed control techniques. In the EEA-ENB we consider the Load Frequency Control (LFC) problem in the presence of Renewable Energy Sources (RESs), and Energy Storage Systems (ESSs). RESs are known to cause instability in power networks due to their inertia-less and intermittent characteristics, while ESSs are introduced as a resource to mitigate the problem. In the EEA-ENB, particular attention is dedicated to Distributed Model Predictive Control (DMPC), whose application is often limited to small and homogeneous test cases due to the lack of standardized large-scale scenarios for testing, and due to the large computation time required to obtain a centralized MPC action for performance comparison with DMPC strategies under consideration. The second problem is exacerbated when the scale of the system grows. To address these challenges and to provide a real-world-based and control-independent benchmark, the EEA-ENB has been developed. The benchmark includes a centralized MPC strategy providing performance and computation time metrics to compare distributed control within a repeatable and realistic simulation environment.

This chapter is based on [280].

2.1 Introduction

2.1.1 The Origin of the Benchmark: Motivation and Challenges

The European Economic Area Electricity Network Benchmark (EEA-ENB) is a benchmark designed for the implementation and testing of distributed control strategies for large-scale power networks. The idea behind the benchmark is to build an abstract model of the European network of transmission systems for electricity. We represent each country of the European economic area as an independent electrical area connected to others through tie lines according to a predefined electricity network topology. The result is a real-world oriented benchmark that accounts for the presence of renewable generation and Energy Storage Systems (ESSs) in the Load-Frequency Control (LFC) problem of the power network.

The development of the EEA-ENB is essential because no established control model for the European electricity transmission system consistently serves as a reference for distributed control techniques, especially with energy storage systems and renewable energy sources. Additionally, the use case for the EEA-ENB is not restricted only to the pure development of control strategies. With minimal modifications, it can also be used for other applications, such as the economic optimization of network operation, the study of network expansion strategies, testing of security and privacy features, and simulation of emergency situations such as cascading blackouts and network restoration.

To assess the time and computation requirements for the implementation of a distributed control strategy we implement centralized Model Predictive Control (MPC) on the network. Together with the value of the cost function of centralized MPC developed, this provides the user metrics to evaluate the advantages and disadvantages in the implementation of a specific distributed control technique. The EEA-ENB is formulated with a modular approach such that extensions can be implemented if needed, allowing for various application scenarios as mentioned before. The stability of the network is assessed through the study of LFC problem. Moreover, another application that is particularly relevant for this benchmark is the economical optimization of energy trading among network agents. The EEA-ENB can also be employed to formulate Distributed MPC (DMPC) techniques in the presence of hybrid dynamics thanks to a modified ESS dynamics reported. Additional extensions, not included in this work, include the characterization of each electrical area according to the deregulated energy market through the modeling of generation plants, the auction system for scheduling energy production across the various generation companies, and the market of power exchanges between different electrical areas [42].

The main challenge in controlling the EEA-ENB has to be sought in its scale: 26 electrical areas are considered, each subject to distinct variations in load requests and renewable generation. When using a growing number of control agents the computation time of a centralized control action becomes increasingly prohibitive, thus, distributed control approaches are required.

2.1.2 Load Frequency Control in Modern Power Networks

The LFC problem is a crucial challenge in power systems, and it has a particular socio-economic interest [177]. The LFC problem gained interest in the research community in

the 1970s [103] after some major systems events led to cascading blackouts [124]. These problems typically arise when unexpected changes in the load of a power system occur, with consequent shifts in the operating frequency of the electrical area under consideration, and the propagation of this effect to neighboring areas. In the last decades cascading blackouts have been exacerbated by the increasing diffusion of renewable energy sources, which are posing new challenges for LFC of interconnected power grids due to their intermittent and stochastic nature, and inertia-less generation [270].

Nowadays, new strategies to increase network robustness are constantly sought [42]. This is the reason why ESSs are fundamental in modern energy grids: they allow for more efficient use of energy, optimizing its usage based on the demand, and they can be used to counteract the inertia-less properties of renewable energy sources. Therefore, part of the modeling section of this chapter is dedicated to ESSs, from the simplest dynamical formulation to more complex hybrid formulations.

Formally, the main control problem solved in the EEA-ENB is the regulation to zero of the frequency deviation of the network from the nominal value. This problem is solved in the presence of unexpected changes in the load, renewable generation, and ESSs. Early approaches to the solution were mainly based on PID control theory. With the progression of technology, more advanced techniques have been implemented, such as variable gain scheduling, fuzzy logic control, artificial neural networks, and optimal control [177, 270].

In this chapter, we propose MPC as reference control technique for the benchmark, and DMPC as its natural extension. The choice of MPC is related to the fact that it provides the optimal control action according to a certain cost function defined by the user, while incorporating constraints on the evolution of the state and control. Nevertheless, the EEA-ENB is designed to be control-independent, and virtually all control techniques can be implemented on it. For a detailed list of control approaches for the LFC problem, we refer to [270].

2.2 Problem Description

2.2.1 System Description

The EEA-ENB is composed of 26 interacting electrical areas connected through tie lines and uses real-world data acquired from the European Network of Transmission System Operators for Electricity (ENTSO-E) transparency platform accessible from [92]. Each area represents an equivalent electrical machine aggregating the inertia and dispatchable capacity of generators in that specific area, a modeling technique commonly used in the context of LFC [177]. The electrical topology of the network is derived from the grid map also provided by ENTSO-E [92]. The benchmark is constituted by 26 control areas due to considerations about the availability and scale of the data about the 31 members of the EEA. The electrical topology of the resulting network is reported in Fig. 2.1, where each country is labeled with the respective ISO code. Positions of the areas in the space are selected according to their geographical centroids, and, on this basis, the lengths of the tie lines are defined using the Euclidean distance as reported in Table 2.1. In this graph representation, each node is associated with dynamics incorporating generation, storage, consumption, and interaction behaviors of the considered electrical area and of its neighborhood. In particular, an electrical area is composed of a multiplicity of autonomous

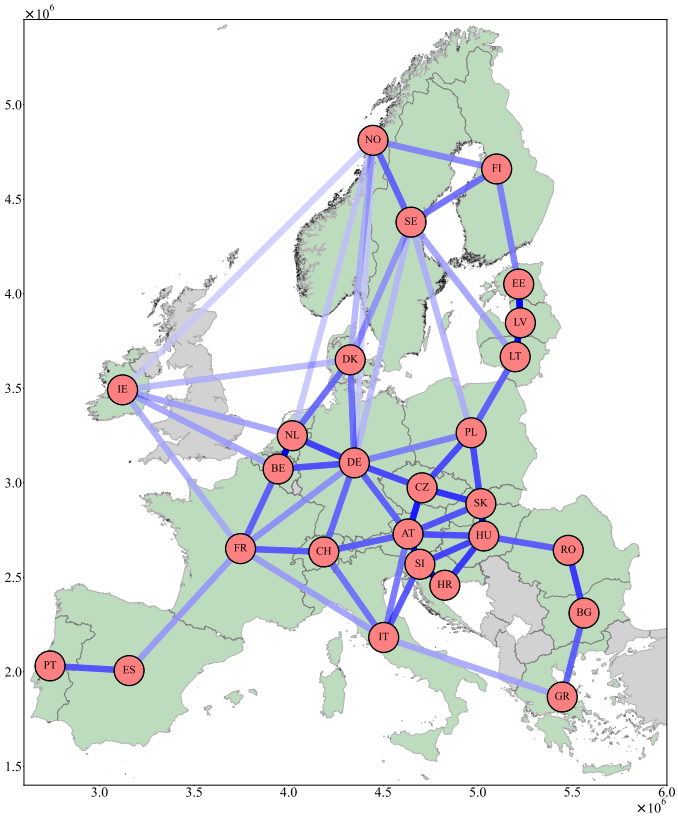


Figure 2.1: Electrical topology of the EEA-ENB. Each node represents an electrical area, coinciding with a single country, whereas each edge is a transmission line. The transparency of the edges represents the strength of interaction among the areas.

Table 2.1: Lengths of the tie lines expressed in 10^3 [km].

	AT	BE	BG	HR	CZ	DK	EE	FI	FR	DE	GR	HU	IE	IT	LV	LT	NL	NO	PL	PT	RO	SK	SI	ES	SE	CH
AT	0	0	0	0	2.65	0	0	0	0	4.82	0	5.61	0	4.76	0	0	0	0	0	0	0	5.88	1.85	0	0	5.58
BE	0	0	0	0	0	0	0	0	4.69	5.77	0	0	13.19	0	0	0	1.75	0	0	0	0	0	0	0	0	0
BG	0	0	0	0	0	0	0	0	0	0	4.02	0	0	0	0	0	0	0	0	0	3.01	0	0	0	0	0
HR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2.12	0	0	0
CZ	2.65	0	0	0	0	0	0	0	0	5.13	0	0	0	0	0	0	0	0	4.67	0	0	4.33	0	0	0	0
DK	0	0	0	0	0	0	0	0	0	5.03	0	0	17.82	0	0	0	5.44	11.56	0	0	0	0	0	0	10.22	0
EE	0	0	0	0	0	0	0	6.37	0	0	0	0	0	0	2.2	0	0	0	0	0	0	0	0	0	0	0
FI	0	0	0	0	0	0	6.37	0	0	0	0	0	0	0	0	0	0	8.99	0	0	0	0	0	0	8.89	0
FR	0	4.69	0	0	0	0	0	0	0	9.35	0	0	12.38	11.19	0	0	0	0	0	0	0	0	0	8.58	0	6.09
DE	4.82	5.77	0	0	5.13	5.03	0	0	9.35	0	0	0	0	0	0	0	4.98	15.23	9.06	0	0	0	0	0	13.41	4.84
GR	0	0	4.02	0	0	0	0	0	0	0	0	0	0	10.94	0	0	0	0	0	0	0	0	0	0	0	0
HU	5.61	0	0	3.61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5.87	1.48	4.63	0	0	0
IE	0	13.19	0	0	0	17.82	0	0	12.38	0	0	0	0	0	0	0	13.84	27.51	0	0	0	0	0	0	0	0
IT	4.76	0	0	0	0	0	0	0	11.19	0	10.94	0	0	0	0	0	0	0	0	0	0	0	3.81	0	0	5.84
LV	0	0	0	0	0	0	2.2	0	0	0	0	0	0	0	0	1.69	0	0	0	0	0	0	0	0	0	0
LT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.69	0	0	0	5.55	0	0	0	0	0	10.14	0
NL	0	1.75	0	0	0	5.44	0	0	0	4.98	0	0	13.84	0	0	0	0	16.99	0	0	0	0	0	0	0	0
NO	0	0	0	0	0	11.56	0	8.99	0	15.23	0	0	27.51	0	0	0	16.99	0	0	0	0	0	0	0	2.28	0
PL	0	0	0	0	4.67	0	0	0	0	9.06	0	0	0	0	0	5.55	0	0	0	0	0	3.37	0	0	10.93	0
PT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4.34	0	0
RO	0	0	3.01	0	0	0	0	0	0	0	0	5.87	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SK	5.88	0	0	0	4.33	0	0	0	0	0	0	1.48	0	0	0	0	0	0	3.37	0	0	0	0	0	0	0
SI	1.85	0	0	2.12	0	0	0	0	0	0	0	4.63	0	3.81	0	0	0	0	0	0	0	0	0	0	0	0
ES	0	0	0	0	0	0	0	0	8.58	0	0	0	0	0	0	0	0	0	0	4.34	0	0	0	0	0	0
SE	0	0	0	0	0	10.22	0	8.89	0	13.41	0	0	0	0	0	10.14	0	2.28	10.93	0	0	0	0	0	0	0
CH	5.58	0	0	0	0	0	0	0	6.09	4.84	0	0	0	5.84	0	0	0	0	0	0	0	0	0	0	0	0

subsystems working together to guarantee the satisfaction of the setpoints assigned by the area-level controller. The aggregation of those subsystems allows one to define an equivalent electrical machine for each area. Specifically, each area may comprehend the following:

- A *dispatchable generator* used to model all sources of energy that can be actively controlled to balance the load. Conventional power sources are hydroelectric turbines, nuclear power plants and gas, oil, or coil turbines. Those sources are associated with an aggregated power generation that we can allocate at each time step according to the production limits of each area.

- *Non-dispatchable generation* associated with renewable energy production, such as wind and solar generation, which have intermittent and stochastic nature. We assume that data are available both for the exact value of the produced power and for day-ahead forecasts.

- An *ESS* used to accumulate and supply energy at the best convenience and according to the control strategy implemented. In general, energy storage systems can be classified into three macro-categories: electrical storage (e.g. ultracapacitors), electrochemical storage (e.g. batteries), and mechanical storage (e.g. water reservoirs). However, this distinction is not considered in the benchmark, but it is suggested as a possible extension. Following the same approach used for dispatchable generation, we consider the aggregated storage and power of all the ESSs in the electrical area.

- A *load demand* for which measurements and day-ahead forecasts are available.

Those components contribute to the internal load-frequency balance of the electrical area. Moreover, a power exchange among areas is present over the tie lines reported in the electrical topology. This interaction must also be accounted for in the overall power balance.

2.2.2 System Dynamics

The topology of the power system is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each node $v_i \in \mathcal{V}$ is associated with an independent electrical area i, and each undirected edge $\epsilon_{ij} = \epsilon_{ji} = (v_i, v_j) \in \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a tie line connecting adjacent areas i and j, allowing for bidirectional power flow. In our case, we have 26 nodes, one for each electrical area. The presence of an edge represents the existence of a power connection. For each node v_i, we define its neighborhood as $\mathcal{N}_i = \{v_j \in \mathcal{V} \,|\, (v_i, v_j) \in \mathcal{E}\}$, i.e. the set of nodes connected to the node v_i. To each node v_i, $i = 1, \dots, 26$ an equivalent electrical machine is associated according to the schematic in Fig. 2.2. Each electrical area i is always characterized by at least three states: the angle δ_i [deg] of the rotor, the operating frequency f_i [Hz] of the equivalent machine, and the energy e_i [GWh] stored in the ESS. The control inputs for the i-th area are the deviation ΔP_i^{disp} [GW] of dispatchable power production w.r.t. the scheduled value, and the power $P_i^{\text{ESS, c}}$ [GW] supplied to or $P_i^{\text{ESS, d}}$ [GW] withdrawn from the ESS. Additionally, each area is subjected to the influence of external inputs: the variation in the load request ΔP_i^{load} [GW], renewable energy production ΔP_i^{ren} [GW], and the power transmitted over the tie lines ΔP_i^{tie} [GW] connected to area i.

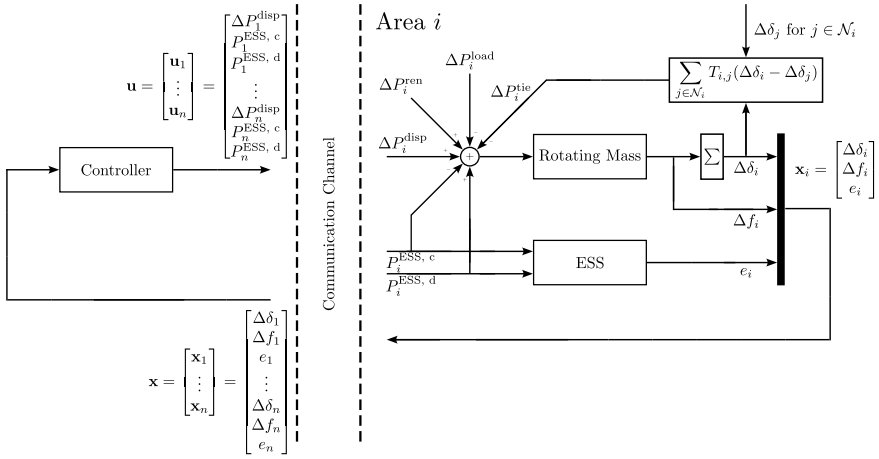


Figure 2.2: Schematic of the network of equivalent machines, with details of the i-th electrical area.

For this system, it is common to assume [42, 177] a linearized discrete-time model around an operating point $(\delta_{0,i}, f_{0,i})$ for the power angle and frequency dynamics for each area i. For all $i \in \{1,\dots,26\}$, we assume $\delta_{0,i} = 30$ [deg], but this depends on the scheduled power exchanges among electrical areas as specified later, with limits $\delta_0 \in (0, 90)$ [deg]; moreover, the operating frequency of the European power network is $f_{0,i} = 50$ [Hz] [177]. Regarding the ESS of the i-th area, the simplest model capturing the charging and discharging characteristics of a storage system is the linear representation also reported in [307]. Extensions of this model, and alternative formulations of the ESS dynamics, as the PWA description in [259], are discussed in Section 2.2.4. For the tie lines interaction, we also use a linearized equation [177] under the assumption that machine angle deviations are small enough, which will be guaranteed through operating constraints in the control formulation.

To summarize, the state, input, and external input of the i-th area are the vectors:

$$\mathbf{x}_i = \begin{bmatrix}\Delta\delta_i & \Delta f_i & e_i\end{bmatrix}^\top \quad \mathbf{u}_i = \begin{bmatrix}\Delta P_i^{\text{disp}} & P_i^{\text{ESS, c}} & P_i^{\text{ESS, d}}\end{bmatrix}^\top \quad \mathbf{w}_i = \begin{bmatrix}\Delta P_i^{\text{load}} & \Delta P_i^{\text{ren}} & \Delta P_i^{\text{tie}}\end{bmatrix}^\top, \tag{2.1}$$

and their aggregation provides the respective definition of the state, input, and external input vectors for the overall network:

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_1^\top & \cdots & \mathbf{x}_{26}^\top\end{bmatrix}^\top \quad \mathbf{u} = \begin{bmatrix}\mathbf{u}_1^\top & \cdots & \mathbf{u}_{26}^\top\end{bmatrix}^\top \quad \mathbf{w} = \begin{bmatrix}\mathbf{w}_1^\top & \cdots & \mathbf{w}_{26}^\top\end{bmatrix}^\top. \tag{2.2}$$

Assuming that the discrete-time dynamics obtained through forward Euler discretization

has sampling time τ, the dynamics of the i-th electrical area $\mathcal{S}_i$ has the form:

$$\begin{aligned}
\Delta\delta_i(k+1) &= \Delta\delta_i(k) + \tau 2\pi \Delta f_i(k) \\
\Delta f_i(k+1) &= \left(1 - \frac{\tau}{T_{\mathrm{p},i}}\right)\Delta f_i(k) + \tau \frac{K_{\mathrm{p},i}}{T_{\mathrm{p},i}}\left(\Delta P_i^{\mathrm{disp}}(k) - \Delta P_i^{\mathrm{load}}(k) + \Delta P_i^{\mathrm{ren}}(k) + \right. \\
&\qquad \left. - \Delta P_i^{\mathrm{tie}}(k) - P_i^{\mathrm{ESS,\,c}}(k) + P_i^{\mathrm{ESS,\,d}}(k)\right) \\
e_i(k+1) &= e_i(k) + \tau\left(\eta_i^{\mathrm{c}} P_i^{\mathrm{ESS,\,c}}(k) - \frac{1}{\eta_i^{\mathrm{d}}} P_i^{\mathrm{ESS,\,d}}(k)\right) \\
\Delta P_i^{\mathrm{tie}}(k) &= \sum_{j \in \mathcal{N}_i} T_{ij}(\Delta\delta_i(k) - \Delta\delta_j(k)),
\end{aligned} \tag{2.3}$$

where $K_{\mathrm{p},i}$ and $T_{\mathrm{p},i}$ are respectively the gain and the time constants of the dynamics of the rotating mass; η_i^{c} and η_i^{d} are charging and discharging rates of the battery with $0 < \eta_i^{\mathrm{c}}, \eta_i^{\mathrm{d}} < 1$; and T_{ij} [GW/deg] is the gain associated with the tie line (i, j), i.e. $T_{ij} = k_{ij}/d_{ij}$, which depends on the geographical distance d_{ij} [km] among the electrical areas, and on the gain k_{ij} [km·GW/deg], which is assumed to be equal to 1 for all i, j in this chapter.

2.2.3 Assumptions and Operating Conditions

The electrical angle deviation is constrained as $-30 \leq \Delta\delta_i \leq 30$, so that the electrical angle satisfies $0 \leq \delta_i \leq 60$, with $\delta_{0,i} = 30$ [deg]. For the operating frequency we assume the range $-0.04 \leq \Delta f_i \leq 0.04$, with $f_{0,i} = 50$ [Hz] [42, 177]. For the ESSs, we consider the maximum storage capacity to be equal to the total dispatchable capacity, i.e. $0 \leq e_i \leq e_{i,\max}^{\mathrm{disp}}$, for each area i, with $e_{i,\max}^{\mathrm{disp}} = 1 \cdot P_{i,\max}^{\mathrm{disp}}$ [GWh]. For each electrical area i the following state constraints hold:

$$-30 \leq \Delta\delta_i \leq 30 \qquad -0.04 \leq \Delta f_i \leq 0.04 \qquad 0 \leq e_i \leq P_{i,\max}^{\mathrm{disp}} \tag{2.4}$$

Input limits are selected such that the total available dispatchable or storage capacity can be allocated over one hour:

$$-\frac{P_{i,\max}^{\mathrm{disp}}}{1440} \leq \Delta P_i^{\mathrm{disp}} \leq \frac{P_{i,\max}^{\mathrm{disp}}}{1440} \qquad 0 \leq P_i^{\mathrm{ESS,\,c}},\, P_i^{\mathrm{ESS,\,d}} \leq \frac{P_{i,\max}^{\mathrm{disp}}}{1440} \tag{2.5}$$

The sampling time of the system is $\tau = 2.5$ [s], which is 10 times faster than the time constant $T_{\mathrm{p},i} = 25$ [s]. A variation in the external inputs occurs every 1440 time steps, i.e. every hour. A simulation of $24 \cdot 1440 = 34560$ steps would use 24 hours of real-world data about load and renewable generation, see Section 2.3.1 for additional details.

2.2.4 Extensions and Alternative Formulations

We propose three directions to modify or extend the proposed dynamics: a PWA formulation of the ESS dynamics, an extension to include the behavior of turbines and pumps, and an augmented state representation to describe the energy market. Other possible extensions are reported at the end of this section.

ESS Hybrid Dynamics

Assuming that the ESSs can only be in a charging or discharging state at each time step, their dynamics can be described with the following Piecewise Affine Linear (PWA) equations [259]:

$$e_i(k+1) = \begin{cases} e_i(k) + \tau \eta_i^{\mathrm{c}} \Delta P_i^{\mathrm{ESS}}(k) & \text{if} \quad \Delta P_i^{\mathrm{ESS}}(k) \geq 0 \\ e_i(k) + \tau \frac{1}{\eta_i^{\mathrm{d}}} \Delta P_i^{\mathrm{ESS}}(k) & \text{if} \quad \Delta P_i^{\mathrm{ESS}}(k) < 0 \end{cases} \tag{2.6}$$

In this formulation, the charging and discharging inputs used in (2.3) are substituted by a single input P_i^{ESS}, representing the total power exchange of the electrical area with the ESS. This formulation is completely different from the linear one in (2.3), and it can be demonstrated that the two representations are equivalent only if the ESS is lossless, i.e. is if $\eta^{\mathrm{c}} = \eta^{\mathrm{d}} = 1$.

Turbine and Pump Dynamics Extension

A finer representation of the system would include the presence of a turbine for the generation of the dispatchable power, and of a turbine/pump system for mechanical ESS to allocate and use energy in the water reservoirs (this is not necessary for other types of ESSs). This concept can be applied both to the ESS formulation in (2.3) and (2.6). Additional states are introduced in this new description: the signals previously considered in (2.3) and (2.6) as inputs are now the states of the turbines or pump. Additionally, new inputs u_i^{disp}, $u_i^{\mathrm{ESS,\,c}}$, $u_i^{\mathrm{ESS,\,d}}$ are introduced to control the turbines and pump. Specifically, if we consider the linear formulation (2.3) for the i-th electrical area we have:

$$\mathbf{x}_i = \begin{bmatrix} \Delta\delta_i & \Delta f_i & e_i & \Delta P_i^{\mathrm{disp}} & P_i^{\mathrm{ESS,\,c}} & P_i^{\mathrm{ESS,\,d}} \end{bmatrix}^{\mathsf{T}} \quad \mathbf{u}_i = \begin{bmatrix} u_i^{\mathrm{disp}} & u_i^{\mathrm{ESS,\,c}} & u_i^{\mathrm{ESS,\,d}} \end{bmatrix}^{\mathsf{T}} \tag{2.7}$$

and the dynamics (2.3) is augmented with the update equations:

$$\begin{aligned} \Delta P_i^{\mathrm{disp}}(k+1) &= \left(1 - \frac{\tau}{T_{\mathrm{t},i}}\right) \Delta P_i^{\mathrm{disp}}(k) + \tau \frac{K_{\mathrm{t},i}}{T_{\mathrm{t},i}} u_i^{\mathrm{disp}} \\ P_i^{\mathrm{ESS,\,c}}(k+1) &= \left(1 - \frac{\tau}{T_{\mathrm{c},i}}\right) \Delta P_i^{\mathrm{ESS,\,c}}(k) + \tau \frac{K_{\mathrm{c},i}}{T_{\mathrm{c},i}} u_i^{\mathrm{ESS,\,c}} \\ P_i^{\mathrm{ESS,\,d}}(k+1) &= \left(1 - \frac{\tau}{T_{\mathrm{d},i}}\right) \Delta P_i^{\mathrm{ESS,\,d}}(k) + \tau \frac{K_{\mathrm{d},i}}{T_{\mathrm{d},i}} u_i^{\mathrm{ESS,\,d}} \end{aligned} \tag{2.8}$$

where $T_{\mathrm{t},i}$, $T_{\mathrm{c},i}$, $T_{\mathrm{d},i}$ and $K_{\mathrm{t},i}$, $K_{\mathrm{c},i}$, $K_{\mathrm{d},i}$ are respectively the time constants and gains of the turbine and storage turbine/pump of the i-th electrical area. As good engineering practice, the time constants $T_{\mathrm{t},i}$, $T_{\mathrm{c},i}$, $T_{\mathrm{d},i}$ are selected to be at least 10 times smaller than $T_{\mathrm{p},i}$, and accordingly the sampling time τ has to be at least 100 times smaller than the original one in (2.3), i.e. $\tau = 0.025$ [s]. For further details see [42, 93].

State Augmentation and Total Production Constraints

An aspect usually not considered in LFC systems are the constraints on the total dispatchable production. In this benchmark, we want to constrain the total dispatchable production P_i^{disp} at time step k to be non-negative, and smaller than the overall capacity of a certain area. Moreover, we might require that P_i^{tie}, which is the total energy transmitted over the

tie-lines connected to area i, is between certain limits, or tracks a reference. To this end, the dynamics (2.3) are augmented with the two states P_i^{disp}, P_i^{tie} that evolve as follows:

$$\begin{aligned} P_i^{\text{disp}}(k+1) &= P_i^{\text{disp}}(k) + \tau \Delta P_i^{\text{disp}}(k) \\ P_i^{\text{tie}}(k+1) &= P_i^{\text{tie}}(k) + \tau \Delta P_i^{\text{tie}}(k) \end{aligned} \tag{2.9}$$

In this way, we impose limits on the overall dispatchable generation for each electrical area by adding the constraint $0 \leq P_i^{\text{disp}}(k) \leq P_{i,\max}^{\text{disp}}$ to the control problem. For example, some areas may have a renewable production that exceeds the total load request. This constraint ensures that the excess is stored for later use or transmitted to neighboring areas. As initial condition, we assume to have a dispatchable production that compensates for the total load request and that accounts for the renewable generation:

$$P_i^{\text{disp}}(0) = \max\left\{0; P_i^{\text{load}}(0) - P_i^{\text{ren}}(0)\right\}. \tag{2.10}$$

Additional Extensions

The equivalent machine modeling approach can also be applied to the deregulated energy market [42]. This approach involves defining various actors for electricity production in different regions, each with its dispatchable generation capacity. These actors, known as generation companies, can be represented by individual turbines that aggregate the inertia of all the generators within the same company. Additionally, there are ESSs that aggregate the storage capacities of each company. Thus, in each area i, there is a certain number of dispatchable generators and ESSs. A centralized auction system determines which generation company supplies energy to each area, considering cross-border production, electrical topology, and predefined operational strategies.

We also highlight the fact that each electrical area can be further subdivided into frontier sectors and a central sector, with tie-lines connecting them. This subdivision of the electrical topology can be used for energy trade modeling. The central sector may account for the generation of critical infrastructures and is connected to all the frontier sectors. Each frontier sector is connected to the frontier sector of a neighboring area and to the central sector of the area it belongs to. This further subdivision of the topology can be used to mitigate the effect of power transmission from adjacent areas, to ensure enhanced stability of the central sector, and to define scheduled power transmissions among neighboring areas.

Future research should also consider the exploration of different ESS technologies, the challenges related to their implementation, their feasibility, and economic sustainability, all aspects that can contribute to the further refinement of the EEA-ENB.

2.2.5 Goal of the Control System

The main control goals for the benchmark that are satisfied by the centralized MPC strategy (2.11) are the following:

- *Regulation* of the frequency deviation $\Delta f_i = f_i - f_{i,0}$ of each electrical area to zero, so that the frequency of the network stays at the desired value $f_0 = 50$ [Hz]. This is also the main goal of the control system. Moreover, we also regulate the machine angle deviation $\Delta\delta_i = \delta_i - \delta_{i,0}$ to zero, such that the efficiency of the machine is preserved.

- *Operational constraints satisfaction.* In addition to the regulation goals, the control system should also ensure that the operational constraints of Sections 2.2.3 and 2.2.4 are satisfied. Ensuring that these constraints are satisfied is of primary importance for the stability of the network, and its correct functioning. Similar constraints should be enforced also for the augmented models in Section 2.2.4.

- *Disturbance rejection.* From a control perspective, the external signals of load and renewable generation variations in (2.3) can be interpreted as disturbances to reject.

- *Minimization of the control effort.* The control inputs of the benchmark are the variation in dispatchable generation and power exchange with the ESS for each area i, namely $\mathbf{u}_i = [\Delta P_i^{\text{disp}} \quad P_i^{\text{ESS, c}} \quad P_i^{\text{ESS, d}}]^\intercal$. Minimizing the control effort $\mathbf{u}_i^\intercal \mathbf{u}_i$ for each area i while ensuring the correct functioning of the system is another relevant feature for control design.

In addition to the main control goals, other control objectives can be designed depending on the study that one wants to conduct on the benchmark. For example, if an economic MPC problem is formulated using electricity prices, then the total monetary cost for running the network can be considered, defined in terms of €$\cdot$MW for each agent, for each energy source, and at each time step. With this approach, the least expensive network operation strategy can be defined, trading off the lower operational cost of the network with its stability. In this regard, if soft constraints are implemented to limit the frequency deviation, then the total-time-spent and the average-time-spent outside the optimal operation interval of the frequency can also be considered as a performance indicator. Moreover, other control goals can be considered that are more specific to the technological implementation of the network. For example, those could regard the number of charging and discharging cycles of the batteries, or their average charge level. This means that we might incorporate operational and maintenance costs in the benchmark and consider them as a way to compare control strategies.

2.3 Benchmark Design

2.3.1 Input Data

The network of equivalent machines is modeled using real data about load requests, renewable generation, and dispatchable capacities of the 26 European states selected for the implementation. Data is acquired from the ENTSO-E electricity transparency platform [92]. As an example, data for the 24 hours of January 1, 2022, are reported in Fig. 2.3.

Raw data is available every hour for each area considered, both for measurements and day-ahead forecasts.

2.3.2 Implementation Details

Table 2.2 reports the parameters used in the benchmark. Their selection is done according to the parameters used in similar simulation designs [42, 93, 177]. The sampling time of the systems is selected as $\tau = 2.5$ [s]. It follows that, for each hour, i.e. for each new data sample, 1440 steps of duration τ are considered in the control simulation. We use linear interpolation to compute the external inputs $\Delta P_i^{\text{load, meas}}(k)$, for $k = 0, \dots, 34559$, from the

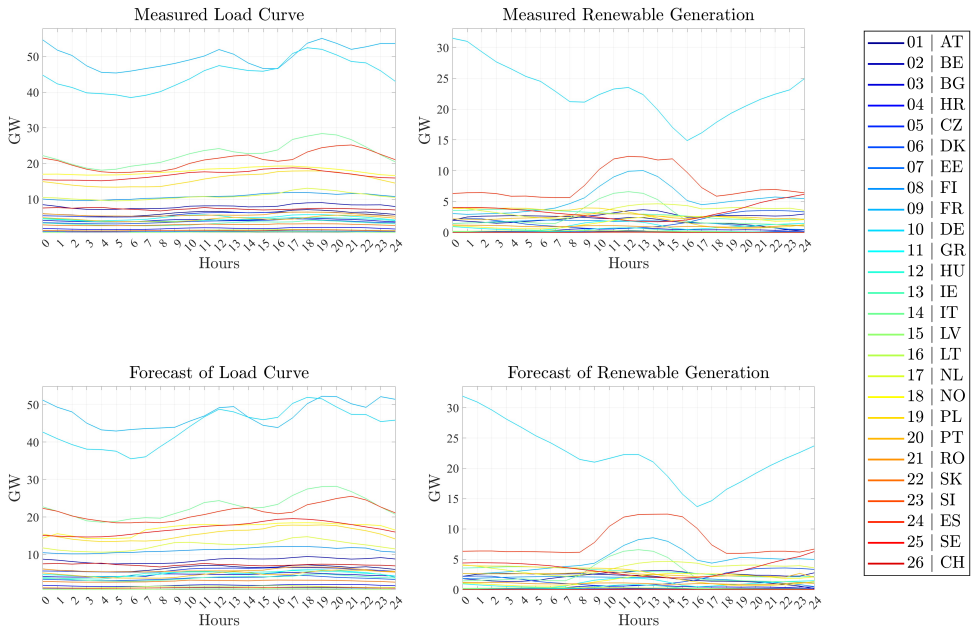


Figure 2.3: Measurements and forecasts of the load request and renewable generation of each electrical area.

Table 2.2: Parameters in the EEA-ENB.

τ	$T_{\mathrm{p},i}$	$K_{\mathrm{p},i}$	η_i^{c}	η_i^{d}
2.5 [s]	25 [s]	0.05 $\left[\frac{\mathrm{Hz}}{\mathrm{GW}}\right]$	0.9	1.1

data of $P_i^{\text{load, meas}}(h)$ available every hour, for $h = 1, \dots, 24$. The same approach is used for renewable generation measurements $P_i^{\text{ren, meas}}$, and for the forecasts $P_i^{\text{ren, for}}$, $P_i^{\text{ren, for}}$. The resulting signal variations are in Fig. 2.4.

2.3.3 Comparison with Other Benchmarks in the Field

Several benchmarks for the simulation of power networks are present in the literature. Among the most popular ones, we can report the various implementations of IEEE buses [142]. Those benchmarks are oriented towards the simulation of power networks for electrical engineering applications. The benchmark we propose is instead oriented to the implementation of distributed control techniques, with predictive control as primary objective. To the best of our knowledge, an LFC-oriented benchmark modeled using data from the EEA is not present in the literature. A similar study was performed in [93] for the simulation of the Northern European network, but without considering renewable generation and ESSs.

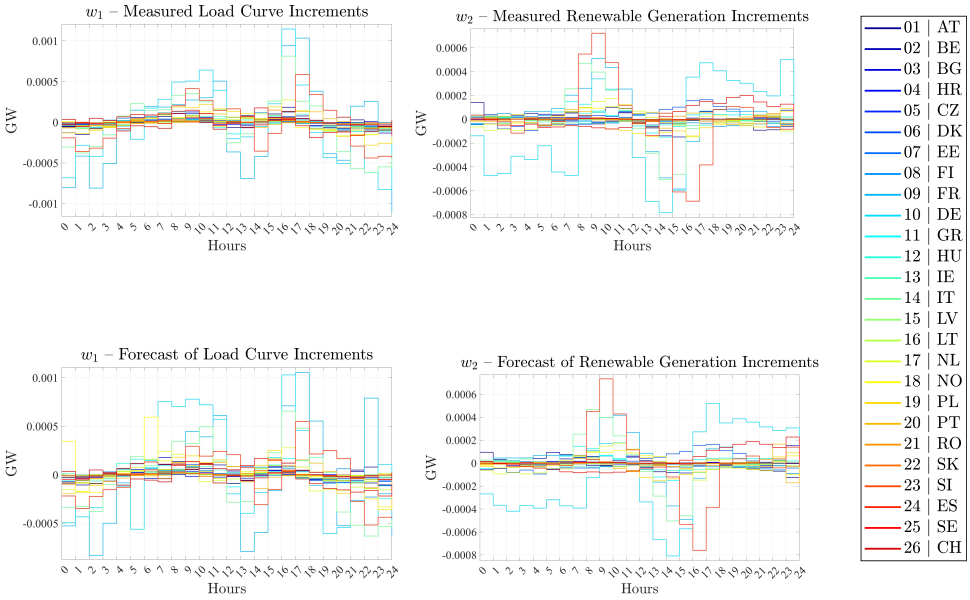


Figure 2.4: Measurements and forecasts for the external signals.

2.3.4 Performance Metrics

The performance of the control system are measured using the value of the quadratic cost function that will be formally introduced in the control problem (2.11) of Section 2.5.1. Since the EEA-ENB is structured for the implementation of distributed control techniques, the overall cost over one day of simulation for the network controlled with the centralized MPC architecture (2.11) represents the optimality target for every alternative control formulation.

Another performance indicator of the control strategy is the computation time required to obtain the control action. Specifically, centralized MPC might not be suited for real-time control of the EEA-ENB due to the excessive computation time required to obtain the optimal control action. Distributed architectures are usually faster in obtaining the control law since they distribute the computational burden among the control agents, but they are more complex to implement.

As a part of the benchmark we provide the data and results for a control simulation of the network using the centralized MPC scheme (2.11) for one day of operation of the network. Both the stage cost at each step, the overall cost for one day, and the total computation time are provided as a reference for alternative control strategies. Further details are in Section 2.5. The end-user can consider these indices as provided for a direct comparison, or perform a centralized MPC simulation with a different cost function to use personalized metrics.

2.3.5 Alternative Test Cases

We propose the simulation for a single day of operation of the network. However, data is available at [92] for every day of the year. Seasonality plays an important role in power generation from renewable sources. For example, solar production can increase or decrease depending on the presence of clouds, the temperature, and the length of the day. Load data is also affected by seasons. Evaluating the network with the average data about load and renewable generation of the four different seasons will give a clear view of the effectiveness of the control strategy considered over an entire year, with a mitigated variability introduced by a single day selection.

2.3.6 Output Data

Executing the benchmark will provide data about the electrical machine angles deviations, their frequency deviations, the energy stored in each area, the total power production and exchange with the ESSs, and the power transmitted over the tie lines. Those quantities are used to compute the performance metrics, and to evaluate the control strategy. Thus, both the evolution of the states of the system and the control actions can be collected and stored.

2.3.7 Essential Properties

The constraints provided for the frequency operation are essential for the stability of the network. Any prolonged deviation from the intervals provided will lead to emergency operation modes or failure of components, which in turn may generate cascading blackouts in the network. The implementation of soft constraints on the state can allow for this deviation outside safety margins, but always considering the stability of the operation of the network and the economic cost of such deviations. For more information, we refer the reader to [93, 124].

Regarding the MPC implementation, both the feasibility and stability properties should be met [37, 227]. Moreover, for the robustness of the system to the disturbances, which are the variations of the load and the renewable generation, an in-depth analysis of their evolution over an extended time window should be performed to characterize them correctly. Then, robust MPC synthesis methods could be used to guarantee this property [37].

2.3.8 Benchmark Scope and Limitations

The EEA-ENB is designed as a control-oriented benchmark for the comparative evaluation of distributed control strategies, and several modeling simplifications have been introduced to develop a tractable case study. In particular, in choosing the scale and complexity of the benchmark, we sought a balance between its real-world adherence and level of detail, and the tractability of the control strategy to deploy, both in terms of design and computational complexity. This benchmark is therefore real-world-oriented in the sense that it is based on real-world data, on a plausible network topology, and that it addresses with a reasonable level of complexity an actual control problem present in the European power grid. On the other hand, this benchmark is not a high-fidelity model, which would be too complex to test control strategies on standard hardware. Despite this intrinsic limitation, the EEA-ENB can be developed further to incorporate higher-order dynamics, heterogeneous phenomena,

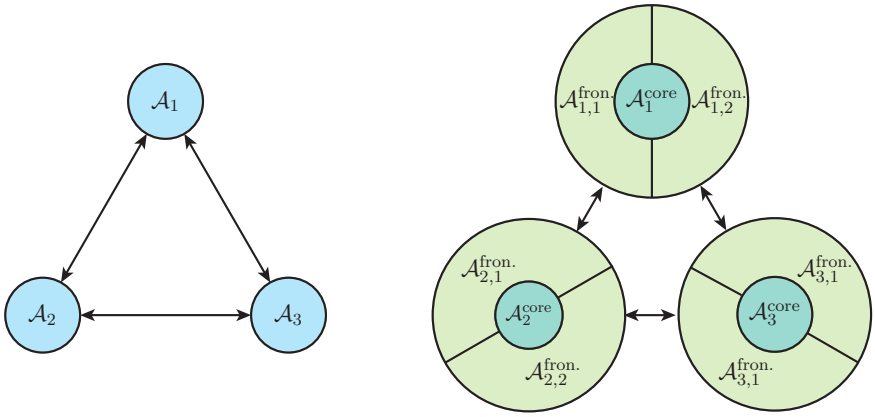


Figure 2.5: On the left, a conventional network topology, where each electrical area is connected to the others. On the right, a network topology with subdivision into core and frontier sub-areas: each core sub-area is connected only with frontier sub-areas within the same electrical area; each frontier sub-area is then connected to the frontier sub-area of a neighboring electrical area. In this way, a core portion of each electrical area is isolated from other electrical areas, thereby potentially improving grid stability.

and its scale can be extended using the same modeling and engineering principles presented in this chapter.

In particular, we report some relevant modeling simplifications that have been introduced in the benchmark. Some of them find a relatively easy extension to the more complex case, while others present greater generalization complexity. We discuss them briefly in the following:

- Each country is modeled as one aggregated generator concentrating the total inertia and dispatchable capacity of all generation units in that area, which is a standard modeling approximation in the LFC literature [42, 177]. While this choice is reasonable for the study of inter-area electricity dynamics, it intrinsically simplifies the heterogeneity of the dynamics within the electrical area. Moreover, the EEA-ENB is geographically vast, and therefore the LFC dynamics in the various regions within the same area might be quite different. A first advisable extension in this direction that would allow developing a realistic model without incurring in the complexity of high-fidelity representations would be dividing each electrical area into one core sub-area surrounded by frontier sub-areas as represented in Fig. 2.5: the core sub-area would be only connected with frontier sub-areas within the same regions, while each frontier sub-area would be connected externally exclusively with a single frontier sub-area of a neighboring electrical area. Moreover, a data-driven approach might be used to estimate the generator inertia of the sub-areas to realistically capture their LFC dynamics. Such an approach might be further developed till a very fine-grained level of representation, closely approximating the real network.

- In each electrical area, we have defined a single generator, a single load, a single renewable source, and a single ESS. The first step to extend such simplification is to incorporate sources, loads, renewables, and ESSs with different characteristics based

on the information about that area to enrich the description of the behavior of the generation and consumption mix.

- We used linear models and a linearization of electrical couplings, and linear ESS dynamics. For the generators, such linear approximations are not far from reality at operating points, especially if saturations are handled by constraints in the MPC. Regarding the power transmitted across different electrical areas, the linearization assumption also holds within small deviations of the electrical angles of the generators w.r.t. the current generation point, which affect the amount of power transmitted. In the benchmark, we also included an extension of the linear ESS model using a hybrid, more realistic model with nonlinear efficiency rates [279]. Other than using more complex, nonlinear models for the generators and ESSs, a potential direction for extending the benchmark is to introduce stochastic phenomena or uncertainty in network parameters. Also, network topology dynamics, e.g. represented by multi-topological network representations as defined in [284] might be used to represent tie-line dynamics or network failures.

A Natural extension of the benchmark therefore goes in the direction of incorporating nonlinear stochastic phenomena and using more complex network topologies with sub-areas. Moreover, we might introduce further functionalities on top of the control layer related to network security and resilience, and economic optimization of network management.

2.4 Accessing the Benchmark

2.4.1 Links to Sources, Limitations, Costs, and Licensing

The benchmark is implemented in Matlab (r2023b), and the necessary files to execute it are available at [276]. Gurobi Optimizer [123] is required for the computation of the centralized MPC strategy. Alternatively, the Matlab Optimization Toolbox [220] can be used with minimal modifications.

The benchmark is provided for free as it is under MIT the license. Data from [92] used for the construction of the benchmark is publicly available under the Creative Commons Attribution 4.0 International License (CC-BY 4.0).

2.4.2 Documentation

Documentation for the benchmark is available at [276]. In the following, we provide the user with information about the functions used. The data from [92] has been preliminarily checked for integrity, replacing missing entries with linear interpolations, and reported on a consistent scale. This process was performed with specialized scripts reported in the online documentation. The resulting preprocessed data set is also part of the benchmark and provided as .csv files. The benchmark is constituted by the following files:

- `main.m`: this is the principal script to run the control simulation.
- `data_import.m`: this script reads the preprocessed data about load demands and renewable generation measurements and forecasts stored in .csv files, and returns the parameters and signals required for the simulation.

- `state_update_network.m` and `state_update_model.m`, which are identical files in this first formulation, but might be distinguished later to implement model mismatches or parameters inaccuracies. The former is used to simulate the system dynamics, and the latter as a prediction model for the centralized MPC strategy.
- `objective_function.m`: this function takes as inputs the parameters, the current value of the state, and the inputs and external inputs over the prediction window to return the total cost over that window.
- `plot_results.m`: used to produce plots of the simulation results and input data.

2.5 Discussion for Future Comparison

2.5.1 Reference Approach: Centralized Predictive Control

The LFC problem has been extensively studied in the literature. As a source of references to existing approaches for its solution, we refer to the survey [270]. To provide a comparison case for the implementation of distributed control techniques, we have implemented a centralized MPC scheme [93, 259]. Considering the system described in (2.3), we define at each time step k the following centralized control problem:

$$\begin{aligned} \min_{\bar{\mathbf{u}}} \quad & \sum_{j=1}^{N} \mathbf{x}^\top(j|k)\mathbf{R}\mathbf{x}(j|k) + \mathbf{u}^\top(j-1|k)\mathbf{Q}\mathbf{u}(j-1|k) \\ \text{s.t.} \quad & \text{dynamics (2.3)} \quad \forall v_i \in \mathcal{V} \\ & \mathbf{x}(0|k) = \mathbf{x}(k) \\ & \text{constraints (2.4)} \quad \forall v_i \in \mathcal{V} \\ & \text{constraints (2.5)} \quad \forall v_i \in \mathcal{V} \end{aligned} \tag{2.11}$$

were $\bar{\mathbf{u}} = [\mathbf{u}^\top(0|k) \cdots \mathbf{u}^\top(N-1|k)]^\top$, $\mathbf{R}$ and $\mathbf{Q}$ are diagonal cost matrices of appropriate dimensions such that, for each area i, $\mathbf{R}_i = \text{diag}[100; 10; 1]$ and $\mathbf{Q}_i = \text{diag}[1; 1; 1]$, and $N = 30$ is the prediction horizon. For the external signals $\mathbf{w}$ we use the current measurement for $j = 1$, and their forecast for the remaining time steps. According to the receding horizon logic of MPC, we apply $\mathbf{u}(0|k)$ to the system, discard the remaining control actions, and iterate. Problem (2.11) is a quadratic optimization problem, hence efficient algorithms for solving it exist in the literature. For example, the problem can be solved using an active-set or an interior point algorithm.

For the benchmark, the optimization is performed with Gurobi Optimizer using the barrier algorithm. The simulation required 206 [h], 15[m], and 24[s][1] to be completed using a processor Intel Xeon E5-2637v3, with a base clock of 3.5 GHz, and coupled with 128 GB of RAM. The solution of the optimal control problem is the vector of inputs $\mathbf{u}(k)$, for $k = 0, ..., 34559$, (corresponding to 24 hours of simulation) reported in Fig. 2.6. In the same

[1] The amount of time required to run the simulation is only indicative, and serves to understand the complexity of the system. The end-user should re-run the benchmark on their own system to obtain results that are comparable with the control architectures at study.

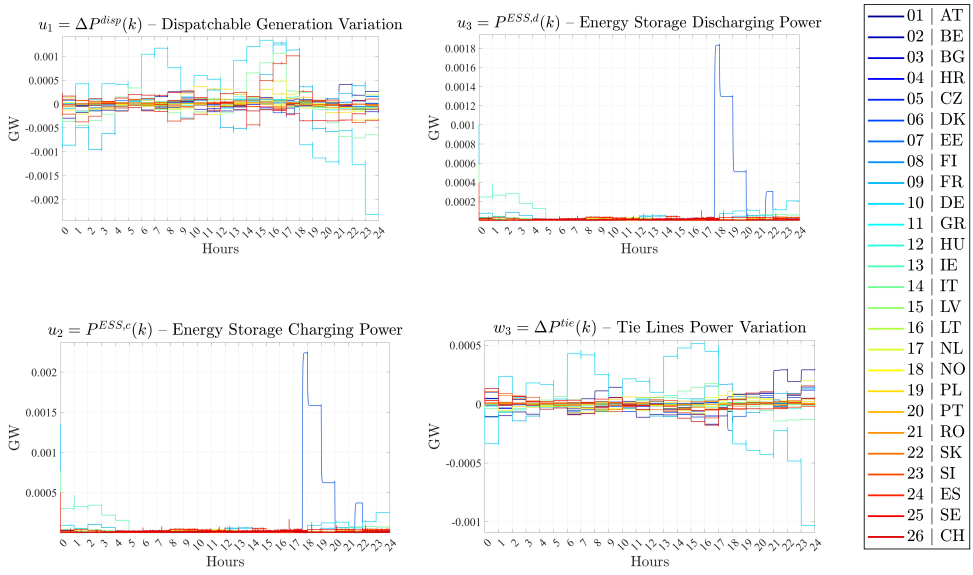


Figure 2.6: Evolution of the inputs and of the power transmitted over the tie lines.

figure, the power transmitted over the tie lines connecting electrical areas is reported too, which allows quantifying the power trade necessary across electrical areas for the optimal operation of the network from a centralized and a cooperative perspective. Interactions among electrical areas, or agents in a generalized setting, is indeed one of the critical aspects of a distributed control strategy [95, 208], and often one of the aspects characterizing the control strategy itself or the definition of the sub-networks in cooperation/competition.

The evolution of the states resulting from the sequence of inputs obtained through MPC is reported in Fig. 2.7. For the overall power balance of each country, please consult the online repository [276].

The evolution of the cost function is presented in Fig. 2.8. Its value is the metric to evaluate distributed control techniques w.r.t. the proposed centralized approach. In the application of distributed control, the performance of the system usually decreases to achieve auxiliary objectives such as improvement in the computation time required to obtain the control action, reduction in the volume of information exchanged across the network (with advantages in the sector of security too), or reliability and redundancy of control in the presence of faults or unforeseen events.

2.5.2 Other Possible Control Approaches

To develop a deeper insight into the topics of LFC, MPC, and distributed control some references are reported in the following. In the LFC literature, see also [270], many articles focus on PI control strategies where tuning of the parameters is performed through various optimization techniques [122, 176, 268]. Despite their promising validations and a general

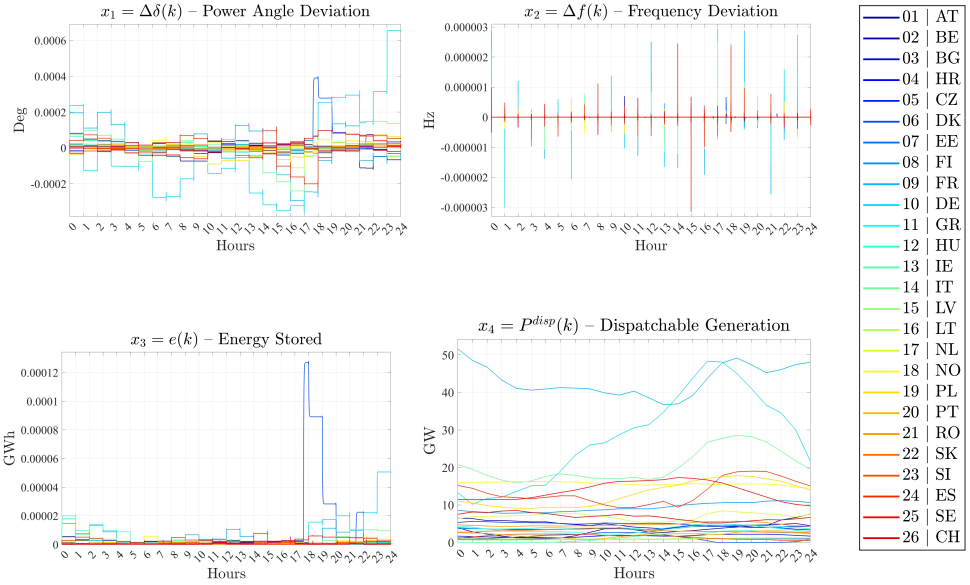


Figure 2.7: Evolution of the states.

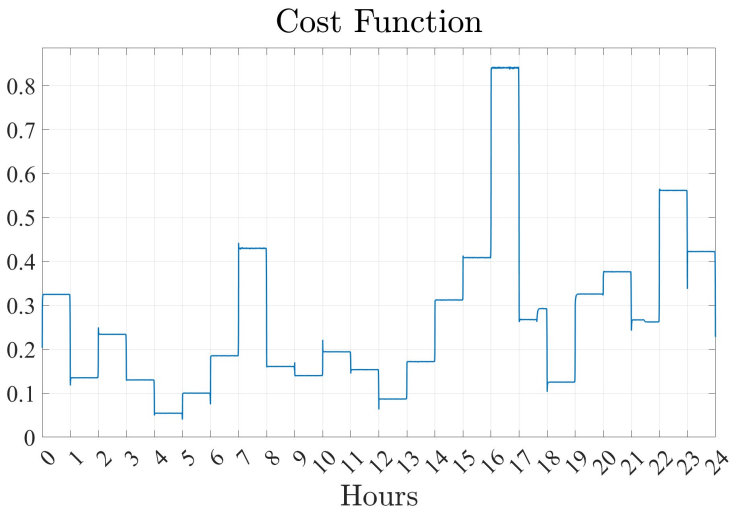


Figure 2.8: Evolution of the cost function.

increase in performance w.r.t. conventional PI-based LFC, all these strategies still lack the fundamental advantages of model-based control: optimization of performance indices, incorporation of constraints into the control problem, ability to compensate for known external signals, and multi-objective optimization. On the other hand, PI control is easier

to implement and has a faster computation speed.

A similar optimization-based tuning approach is used in [91], this time to tune the parameters of an MPC controller.

In [93] a centralized MPC approach similar to the one proposed in (2.11) is presented, but in [93], data of the Nordic transmission system is used to build an electricity network model. Power plant models are used to characterize the generation dynamics of each electrical area, but ESSs and renewable generation are not considered. Economic MPC is addressed too in [93] and the results of centralized MPC are compared with a PD controller.

The use of hydro-pumped storage for LFC of microgrids including renewable generation is explored in [74], where a control architecture based on a decentralized PD controller tuned using a Quasi-Netwon optimization method is implemented.

A characterization of an electricity network using hybrid dynamics is provided in [259]. There, PWA dynamics and binary decision variables are used, leading to a hybrid MPC problem formulated as a Mized Integer Linear Program (MILP), and solved using a branch-and-bound optimization algorithm. This approach is used to model three different aspects: the ESS, the operational mode of the microgrid, and the operational mode of the generation plants.

In [336] a distributed Linear Quadratic Regulator (LQR) is implemented to tackle the LFC problem. Methods for the distributed computation of the LQR control action are used in [336] to increase the modularity and the scalability of the control architecture. The advantages of this approach rely on the ease of implementation, and on the low computation time needed to compute the control action. The disadvantages are that the applicability of the approach is limited to linear unconstrained control problems.

A way to address Economic MPC can be found in [154]. There, both the LFC and economic load dispatch problems for power networks are considered. Those problems are usually approached using hierarchical control structures, where the economic load dispatch is at the upper level, and LFC at a lower level. Instead, in [154] the two problems are considered simultaneously, to improve the economic performance of the systems.

Another Economic MPC strategy is reported in [113]. There, a multi-objective genetic algorithm predictive control technique is used to simultaneously optimize the conflicting objectives of LFC and security-constrained economic dispatch.

2.5.3 Summary

The LFC problem has been widely investigated in the literature, and we have proposed a benchmark to evaluate distributed control strategies for solving it. The challenges arising in recent years are often related to the ever-growing use of distributed energy sources which are inertia-less and can affect the frequency of the network. The use of ESSs can mitigate this effect, and we have proposed strategies to model their dynamics, extensions, and future directions for their exploration. Accordingly, future control strategies of electricity networks should account for the presence of distributed energy sources, and the implementation of distributed control strategies for efficient and resilient operation of the network. Those aspects are indeed part of the centralized MPC formulation that we have proposed as a benchmark scheme. An efficient distributed control strategy is expected to perform worse than the centralized one proposed here if only the value of the cost function is considered, but has additional properties such as a reduced computation

complexity and computation time, a lower shared volume of information, or enhanced privacy, security and resilience properties.

3

A General Partitioning Strategy for Non-Centralized Control

"He had a suspicion of plausible answers; they were so often wrong."

Arthur C. Clarke, Rendezvous with Rama, 1973

Partitioning is a fundamental challenge for non-centralized control of large-scale systems, such as hierarchical, decentralized, distributed, and coalitional strategies. The problem consists of finding a decomposition of a network of dynamical systems into system units for which local controllers can be designed. Unfortunately, despite its critical role, a generalized approach to partitioning applicable to every system is still missing from the literature. This chapter introduces a novel partitioning framework that integrates an algorithmic selection of fundamental system units (FSUs), considered indivisible entities, with an aggregative procedure, either algorithmic or optimization-based, to select composite system units (CSUs) made of several FSUs. A key contribution is the introduction of a global network metric, the partition index, which quantitatively balances intra- and inter-CSU interactions, with a granularity parameter accounting for the size of CSUs, allowing for their selection at different levels of aggregation. The proposed method is validated through case studies in distributed model predictive control (DMPC) for linear and hybrid systems, showing significant reductions in computation time and cost while maintaining or improving control performance w.r.t. conventional strategies.

This chapter is based on [282].

3.1 Introduction

The technological and scientific progress of the last century allowed the realization of large-scale infrastructures in several domains. These networks of systems are commonly found in the fields of transportation [10, 69], power generation and distribution [57, 234, 255, 276, 280], and telecommunications [352], among [55, 78, 249]. These are considered large-scale systems (LSSs) due to their geographical distribution, heterogeneity of components, and complexity of design and operation [172]. With such characteristics, large-scale networks represent a stimulating challenge for control.

Significant improvements in performance, safety, and user experience can be achieved by applying control methodologies. Of particular interest are non-centralized control techniques [295, 303], solutions developed for the application of real-time control to large-scale networks. The more traditional centralized approach to control is often unsuitable in this case. Limitations arise from the computational complexity introduced by the size of the problem, delays in data communication, operational constraints, and privacy limitations. With non-centralized control, such issues can be mitigated or translated into a more tractable class of problems [208].

A fundamental task at the core of every non-centralized control strategy is the definition of the sub-networks of the system that will be considered as individual operational units, i.e. suitable for the application of a local controller part of a non-centralized architecture. The act of defining such sub-networks is defined *partitioning* [278]. From this abstract formulation, it immediately follows that partitioning a network is a complex task that often has to be tailored for the specific application considered. As a matter of fact, a generalized approach to partitioning is currently missing in the literature. On one side, this issue is related to the fact that often when a novel control strategy is being developed, a network partition is assumed to be given a priori; on the other, partitioning might depend on requirements non exclusively related to control objectives. The present article addresses some of these problems, mainly in the former class, laying the foundation for developing generalized partitioning strategies for control, and showing how it is a central problem to address when deploying a non-centralized control strategy.

In the literature, the partitioning problem is usually approached in one of the two following ways: i) using the *top-down* aggregation, where we decompose a monolithic system into components, which define the control units; ii) through the *bottom-up* approach, where predefined subsystems are merged into larger control units. However, an approach that works for all categories of systems following a systematic procedure, i.e. a generally valid technique, is not present yet in literature. Additionally, the problem of defining the number of control units in the partition is often solved by introducing assumptions or heuristics, whereas the role of the size of these units and their hierarchy is not usually a point of investigation.

In this work, we propose a general partitioning strategy for control, a novel approach expanding and improving on our preliminary results [277, 278] with stronger mathematical foundations, refined algorithms, generalized metrics, and new strategies to approach the problem. The main contributions we propose are: i) a formal definition of graph equivalence, which allows the definition of the concepts of fundamental system units (FSUs), composite system units (CSUu), and control partition; ii) a novel algorithm for the selection of the FSUs for any graph equivalent to a dynamical system; iii) a global metric, the *partition index*,

accounting for the interactions in the network, and the size of CSUs through a *granularity* parameter α; iv) optimization-based and algorithmic partitioning approaches, for which the partition index can be specifically characterized, and returning the control partition in terms of groups of FSUs depending on the desired level of granularity.

The chapter is organized as follows. In the remainder of this section, we present the current partitioning approaches in literature, and we briefly recapitulate some graph theory concepts and notation. In Sec. 3.2 we propose the concept of graph equivalent to a dynamical system. Then, the novel core concepts of FSU, CSU, control partition, aggregation operation, and the proposed structure of the generalized partitioning strategy are presented in Sec. 3.3. The algorithm we propose for the selection of the FSUs is detailed in Sec. 3.4; while the generalized partitioning strategy, including the new concept of partition index, and the optimization-based and algorithmic approaches are presented in Sec. 3.5. We conclude with some application examples for partitioning in Sec. 3.6, and a case study illustrating the role of partitioning for the distributed model predictive control of networks of linear and hybrid systems in Sec. 3.7. All the experiments and software developed for this article are available at the open-source long-term access repository [281].

3.1.1 Literature Survey

Despite the absence of a generalized approach, partitioning is a relevant topic for the control [64]. Existing partitioning techniques can be classified into the two fundamental approaches presented below.

Top-Down Approaches

These methods address the partitioning problem by decomposing a system into smaller, manageable sub-networks based on network structure, interaction strength, or physical properties. These approaches primarily leverage graph-theoretic metrics [134, 296]. In this context, the most widely accepted partitioning metric is modularity [101], allowing the definition of partitioning techniques referred to as community detection strategies [210, 298, 324]. The modularity metric can be used to capture both the information in the state transition matrix of a linear dynamical system, and the topological information about the network. The scope of modularity-based techniques is to maximize the modularity. Maximization of modularity is known to be an NP-hard problem [45, 296]; therefore, polynomial-time algorithmic procedures have been developed for partitioning according to modularity maximization.

Top-down partitioning approaches have also been developed in specialized contexts for selected applications. For example, in [306] a wind farm is partitioned into control units according to the coupling given by the wake-effect that influences down-stream turbines. In [29], flow-based distribution systems are considered, focusing on water distribution networks. One relevant feature of the approach proposed in that work is multi-criteria optimization for partitioning.

Bottom-Up Approaches

These methods are aggregative and start from the assumption that a group of autonomous and indivisible FSUs is given a priori. This assumption will generally hold for cooperative groups of mobile agents, such as coalitions of mobile robots, fleets of aerial vehicles, and

platoons of autonomous driving cars. However, bottom-up approaches are also used in transportation and power networks.

One recent distributed control setting is coalitional control [94, 95] where control theory and game theory are combined. In coalitional control, FSUs aggregate into bigger groups, denoted coalitions, to pursue a common control objective. For each coalition, i.e. collection of FSUs, a performance index is computed, which is a function of the aggregated state and input values of the coalition. This index can be interpreted as a transferable utility that can be reallocated among the agents. Partitioning is achieved through an algorithmic procedure, where agents are exchanged among coalitions to maximize the differential gain in the coalitional cost until no further improvement is possible.

Bottom-up approaches are generally more suited for situations where the definition of the sub-networks is non-static, which can occur in the case of non-stationary networks, where a partitioning technique is required to produce a non-stationary definition of the groups. An example of this approach is in [11, 12], which considers linear switching systems and event-driven systems. In these cases, conditions triggering the re-partition of the network are defined, together with ad-hoc strategies to perform it. The effects of partitioning on the properties of a control architecture are explored less frequently. An exception is [27], where feasibility regions and robust invariant sets are defined for distributed tube-based MPC.

Mixed Strategies

In [67] coalitional control, a bottom-up approach, is coordinated by a supervisory layer introducing a clustering problem for the selection of the size and composition of the coalitions, which is a top-down decision. Partitioning is achieved through a binary quadratic program where the decision variables are the states of the links between agents, and the cost is a composition of the gradient vectors of the control objective. However, such approach inherits the computation complexity of the general clustering problem, thus requiring the solution of an NP-hard problem.

3.1.2 Preliminary Concepts and Notation

A graph [46] is an ordered pair of sets $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, \dots, n\}$ is the set of n vertices, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of the edges. The edges are associated to the vertices through a binary adjacency matrix A_{adj}, where $A_{\text{adj},(i,j)} = 1$ if and only if an edge $\epsilon_{ij} = (i, j) \in \mathcal{E}$ exists. Therefore, the topology of the graph is specified by the adjacency matrix A_{adj}, and the set of the edges can also be written as $\mathcal{E} = \{(i, j) \,|\, i, j \in \mathcal{V} \wedge A_{\text{adj},(i,j)} = 1\}$. A subgraph of $\mathcal{G}$ is a graph $\mathcal{S}_i = (\mathcal{V}_i, \mathcal{E}_i)$ representing a part of $\mathcal{G}$. The set of vertices $\mathcal{V}_i$ is a subset of $\mathcal{V}$, i.e. $\mathcal{V}_i \subseteq \mathcal{V}$, and the set of the edges is $\mathcal{E}_i = \{(i, j) \,|\, i, j \in \mathcal{V}_i \wedge A_{\text{adj},(i,j)} = 1\}$, where the topology is still specified by the relevant entries of A_{adj}. For a directed graph $\mathcal{G}$, an edge $\epsilon_{ij} = (i, j)$ denotes an arrow starting from node i and ending in node j. A graph $\mathcal{G}$ is weighted if a weighting matrix W_{adj} assigning to each edge a number is specified in addition to A_{adj}. For each vertex $i \in \mathcal{V}$ we denote by d_i its degree, i.e. the number of edges entering or exiting that vertex. In directed graphs, we can specify an in-degree ($d_{i,\text{in}}$) and an out-degree ($d_{i,\text{out}}$), if the edge is respectively ending or starting in the vertex i. For a vertex i, the neighborhood of i is the set of all vertices connected to it, and we denoted it by $\mathcal{N}_i = \{j \in \mathcal{V} \,|\, (i, j) \vee (j, i) \in \mathcal{E}\}$. For a

subgraph $\mathcal{S}_i = (\mathcal{V}_i, \mathcal{E}_i)$, the frontier is its set of nodes that are connected to nodes outside the subgraph, i.e. $\mathcal{F}_i = \{i \in \mathcal{V}_i \mid (i,j) \vee (j,i) \in \mathcal{E}, j \in \mathcal{V} \setminus \mathcal{V}_i\}$.

3.2 The Equivalent Graph of a Dynamical System

In this section, we introduce the definition of the equivalent graph of a dynamical system for a general class of nonlinear discrete-time systems. Then, we characterize the definition for two important classes of dynamical systems: linear and piecewise affine (PWA) systems. For both classes, we highlight the properties that can be established from their topology, which are relevant for defining a partitioning strategy.

3.2.1 General Definition of Equivalent Graph

Consider a discrete-time system of the form:

$$x(k+1) = f(x(k), u(k)) + g \tag{3.1}$$

with $x \in \mathcal{X} \subseteq \mathbb{R}^n$, $u \in \mathcal{U} \subseteq \mathbb{R}^p$, with f differentiable over $\Omega = (\mathcal{X}, \mathcal{U})$ almost everywhere (i.e. except for a Lebesgue set of measure zero), and g is a constant. Moreover, assume that $f(0,0) = 0$. The equivalent graph representation of (3.1) is a time-dependent graph $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k, w_k, \tilde{g})$, where $\mathcal{V}$ is the set of the vertices, $\mathcal{E}_k \subseteq \mathcal{V} \times \mathcal{V}$ is the set of the edges, w_k is a weighting function for the edges that implicitly characterizes the topology of the graph, and $\tilde{g}$ is a labeling function for the vertices. To construct the equivalent graph, first, we define the set of vertices: a new vertex is defined for each input and state variable of (3.1). These vertices belong respectively to two subsets of vertices, which are $\mathcal{V}_u = \{u_1, \dots, u_p\}$ and $\mathcal{V}_x = \{x_1, \dots, x_n\}$. The set of vertices of the equivalent graph representation is thus the union $\mathcal{V} = \mathcal{V}_u \bigcup \mathcal{V}_x$. Then, we define the weighting function w_k, from which we can also extract the edges of the graph. This function w_k assigns to each couple of vertices a function defined as:

$$w_k(i,j) = \begin{cases} \frac{\partial f_j(k)}{\partial i} & \text{for} \quad i \in \mathcal{V}, j \in \mathcal{V}_x \\ 0 & \text{for} \quad i \in \mathcal{V}, j \in \mathcal{V}_u \end{cases} \tag{3.2}$$

From w_k, it is possible to distinguish between the sets of input-to-state and state-to-state edges, which are respectively:

$$\mathcal{E}_{ux,k} = \{(i,j) \mid i \in \mathcal{V}_u, j \in \mathcal{V}_x, w_k(i,j) \neq 0\} \tag{3.3a}$$

$$\mathcal{E}_{xx,k} = \{(i,j) \mid i, j \in \mathcal{V}_x, w_k(i,j) \neq 0\} \tag{3.3b}$$

The set of the edges of the equivalent graph is then obtained as the union $\mathcal{E}_k = \mathcal{E}_{ux,k} \bigcup \mathcal{E}_{xx,k}$. We will use an analogous notation for the weighting function referring to the above sets, using $w_{ux,k}$ and $w_{xx,k}$ respectively. Finally, the labeling $\tilde{g}$ of the vertices is selected to coincide with the constants g of (3.1) for state vertices, and zero for input vertices:

$$\tilde{g}(i) = \begin{cases} g_i & \text{for} \quad i \in \mathcal{V}_x \\ 0 & \text{for} \quad i \in \mathcal{V}_u \end{cases} \tag{3.4}$$

The equivalent graph $\mathcal{G}$ thus obtained is a variable structure that reflects the nonlinear nature of system (3.1). Specifically, for each pair $(x(k), u(k))$, the topology and the weighting of the graph may change, while the set of the vertices is time-independent. We now show how to build the equivalent graph for linear and piecewise linear systems.

3.2.2 Equivalent Graph Equivalent of a Linear System

Consider the system:

$$x(k+1) = Ax(k) + Bu(k) \tag{3.5}$$

where A and B are matrices of appropriate dimensions. The structure of the graph $\mathcal{G}$ is now time-independent and presents no labeling. Consequently, the sets of the edges are:

$$\mathcal{E}_{ux} = \left\{(i,j) | i \in \mathcal{V}_u, j \in \mathcal{V}_x, B_{(j,i)} \neq 0\right\} \tag{3.6a}$$

$$\mathcal{E}_{xx} = \left\{(i,j) | i,j \in \mathcal{V}_x, A_{(j,i)} \neq 0\right\} \tag{3.6b}$$

3.2.3 Equivalent Graph of a Piecewise Affine Linear (PWA) System

Consider a PWA system of the form:

$$x(k+1) = A^q x(k) + B^q u(k) + g^q \quad \text{for} \quad \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in C^q, \tag{3.7}$$

for operational modes $q = 1, \ldots, N$, with $C^1, \ldots, C^N$ convex polyhedra in the input-state space with non-overlapping interiors [129]. For such systems, the weighting function w_k (3.2), the set of the edges $\mathcal{E}$, and the labeling $\tilde{g}$ can change according to the convex polyhedron C^q. Considering the set of the edges $\mathcal{E}$, depending on the transition, the change can occur in $\mathcal{E}_u$, in $\mathcal{E}_x$, in both simultaneously, or not occur. According to this description, the sets of the edges (3.3) for PWA systems assume the characterization depending on the current mode $q = q(x(k), u(k))$:

$$\mathcal{E}^q_{ux} = \left\{(i,j) | i \in \mathcal{V}_u, j \in \mathcal{V}_x, B^q_{(j,i)} \neq 0\right\} \tag{3.8a}$$

$$\mathcal{E}^q_{xx} = \left\{(i,j) | i,j \in \mathcal{V}_x, A^q_{(j,i)} \neq 0\right\} \tag{3.8b}$$

and the set of the edges of the equivalent graph is $\mathcal{E}^q = \mathcal{E}^q_{ux} \bigcup \mathcal{E}^q_{xx}$. Therefore, we have a finite set of at most N topologies that can vary according to the operational mode.

3.2.4 Number of Distinct Topologies for a Dynamical System

The topology of the equivalent graph of a dynamical system is determined by the set of the edges $\mathcal{E}$. As stressed above, for linear systems this topology is static; thus a linear system only has one possible configuration for the edges for a given pair (A, B) of state space matrices. For PWA systems, the possible number of topologies is equal to the number N of different operating modes at most. However, some operating modes may share the same topology and only differ in the formulation of the dynamics. For a general dynamical system of the form (3.1), we can define an upper bound on the number of distinct topologies:

Lemma 3.1. *The graph associated with system (3.1) has at most $2^{n(n+p)}$ distinct topologies. If self-edges are not considered, the graph has at most $2^{n(n+p-1)}$ distinct topologies.*

Proof. This statement is shown by considering the definition (3.3), where for each pair of vertices (i, j), we are interested only in the cases where the derivative is zero or non-zero.

Thus, we have two possible combinations for each of the n functions and p or n variables, which provides the result. If self-edges are not considered, the statement holds without accounting for the edges $(i,i) \in \mathcal{E}$ with $i \in \mathcal{V}_x$. □

3.3 The Concepts of Composite System Unit and Control Partition, and the Structure of the Generalized Partitioning Strategy

3.3.1 Composite System Unit and Control Partition

One of the scopes of a partitioning strategy applied to a dynamical system is to select subsystems that can be considered as individual control units for a specified application. In this section, we first define a special type of subsystem, the *composite system unit*, which allows the definition of a generalized partitioning strategy. Subsystems of this type can be systematically represented as graphs. Then, we show how to aggregate these subsystems. Finally, we provide the general definition of partitioning based on the concepts above. First, we consider the abstract definition of subsystem:

Definition 3.1 (Subsystem)**.** For a dynamical system of the form (3.1), a subsystem is a group of state dynamics collected into a vector $x^{[i]}$ and whose behavior is defined by $x^{[i]}(k+1) = f^{[i]}(x(k), u(k)) + g^{[i]}$, with $x^{[i]} \in \mathcal{X}^{[i]} \subseteq \mathbb{R}^{n_i}$.

This definition is not sufficiently specialized to define a generalized partitioning strategy. To this end, we introduce the definition of composite system unit, which allows the systematic selection of input-state vector pairs $(x^{[i]}, u^{[i]})$. The choice of these pairs is not unique; however, our characterization is motivated by the scope of defining a generalized partitioning strategy.

Definition 3.2 (Composite System Unit)**.** A composite system unit (CSU) is a nonautonomous dynamical system whose inputs affect only the state dynamics of the CSU itself. All the dynamic relations the CSU has with other CSUs occur through dynamical coupling among the states, or direct input-output connections. For the dynamics (3.1), a CSU is a subsystem of the form $x^{[i]}(k+1) = f^{[i]}(x(k), u^{[i]}(k)) + g^{[i]}$ with $x^{[i]} \in \mathcal{X}^{[i]} \subseteq \mathbb{R}^{n_i}$, $u^{[i]} \in \mathcal{U}^{[i]} \subseteq \mathbb{R}^{p_i}$, and such that $w(k,l) = 0$ for all $l \in \mathcal{V}_{i,x}$, $k \in \mathcal{V}_{j,u}$.

From this definition, it follows immediately that a CSU is a subsystem in the sense of Definition 3.1. We denote a CSU i associated to the input-state pair $(x^{[i]}, u^{[i]})$ with the symbol $\mathcal{S}_i$. The graph equivalent to the CSU in Definition 3.2 is constituted by the set of vertices $\mathcal{V}_i = \mathcal{V}_{i,x} \cup \mathcal{V}_{i,u} \subseteq \mathcal{V}$, by the weighting function $w(i,j)$ with $i,j \in \mathcal{V}_i$, by the set of edges $\mathcal{E}_{i,k} = \left\{(i,j) \,|\, i,j \in \mathcal{V}_i, w_{i,k}(i,j) \neq 0\right\}$, and by the labeling $\tilde{g}_i$. This is the graph $\mathcal{S}_{i,k} = (\mathcal{V}_i, \mathcal{E}_{i,k}, w_{i,k}, \tilde{g}_i)$.

If we consider linear discrete-time time-invariant systems of the form (3.5), then from Definition 3.2 a CSU indexed by i has the form:

$$x^{[i]}(k+1) = A^{[i]}x^{[i]}(k) + B^{[i]}u^{[i]}(k) + \omega^{[i]}(k) \tag{3.9a}$$

$$\omega^{[i]}(k) = \sum_{j \in \mathcal{N}_i} A^{[ij]}x^{[j]}(k) \tag{3.9b}$$

where $x \in \mathcal{X}^{[i]} \subseteq \mathbb{R}^{n_i}$, $u \in \mathcal{U}^{[i]} \subseteq \mathbb{R}^{p_i}$, and $A^{[i]}$, $B^{[i]}$, $A^{[ij]}$ are matrices of appropriate dimensions. The signal $\omega_k^{[i]}$ is the coupling effect that CSU i experiences with its neighboring states $j \in \mathcal{N}_i$.

In the definition of a general partitioning strategy, a fundamental task is the selection of the smallest possible CSUs constituting the network. We specify this concept through the definition of a fundamental system unit.

Definition 3.3 (Fundamental System Unit)**.** Given a network of dynamical systems, a fundamental system unit (FSU) is the smallest CSU definable through any network decomposition.

FSUs represent the smallest individual components that can be selected in a network without losing the nonautonomous property of the dynamics, i.e. the smallest subsystem that has at least one control input and one state variable. Before proceeding with the general definition of partitioning, we introduce the aggregation operation for two CSUs, which allows the merging of two distinct dynamics to form a bigger subsystem.

Definition 3.4 (Aggregation Operation)**.** Consider the equivalent graph representation $\mathcal{G}_k$ of the dynamical system (3.1), and two subsystems of the same dynamical system described by the equivalent graphs $\mathcal{S}_{1,k}$, $\mathcal{S}_{2,k}$. We define the aggregation of the two subsystems over the graph $\mathcal{G}$ as $\mathcal{S}_{(1,2),k} = \left(\mathcal{S}_{1,k} \uplus \mathcal{S}_{2,k}\right)\Big|_{\mathcal{G}_k}$:

$$\mathcal{S}_{(1,2),k} = \left(\mathcal{V}_1 \cup \mathcal{V}_2, \mathcal{E}_{1,k} \cup \mathcal{E}_{2,k} \cup \mathcal{E}_{(1,2),k}, w_{(1,2),k}, (\tilde{g}_1, \tilde{g}_2)\right)$$

where $w_{(1,2),k} = \{w_k(i,j) \,|\, i,j \in \mathcal{V}_1 \cup \mathcal{V}_2\}$, $\mathcal{E}_{(1,2),k} = \{(i,j) \,|\, ((i \in \mathcal{V}_1, j \in \mathcal{V}_2) \vee (i \in \mathcal{V}_2, j \in \mathcal{V}_1)) \wedge w_k(i,j) \neq 0\}$.

The following proposition allows the construction of CSUs as the result of the aggregation of other CSUs:

Proposition 3.1. *The aggregation of multiple CSUs is a CSU.*

Proof. The statement is verified by construction. Consider two CSUs, $\mathcal{S}_{1,k} = (\mathcal{V}_1, \mathcal{E}_{1,k}, w_{1,k}, \tilde{g}_1)$, $\mathcal{S}_{2,k} = (\mathcal{V}_2, \mathcal{E}_{2,k}, w_{2,k}, \tilde{g}_2)$, with $\mathcal{V}_1 = \mathcal{X}_1 \cup \mathcal{U}_1$, $\mathcal{V}_2 = \mathcal{X}_2 \cup \mathcal{U}_2$. By definition of CSU, for both $\mathcal{S}_{1,k}$ and $\mathcal{S}_{2,k}$, it holds that no edges are present from the set of nodes $\mathcal{U}_i$ to the set of states $\mathcal{X}_j$, for $i,j = 1,2$, and $i \neq j$. Now we define the CSU resulting from their aggregation as $\mathcal{S}_{(1,2),k} = \left(\mathcal{S}_{1,k} \uplus \mathcal{S}_{2,k}\right)\Big|_{\mathcal{G}_k} = \left(\mathcal{V}_1 \cup \mathcal{V}_2, \mathcal{E}_{1,k} \cup \mathcal{E}_{2,k} \cup \mathcal{E}_{(1,2),k}, w_{(1,2),k}, (\tilde{g}_1, \tilde{g}_2)\right)$, where $\mathcal{E}_{(1,2),k}$ contains the edges in $\mathcal{E}_k$ linking the two sets of nodes, but that are not present in the individual sets of edges. The operation does not add any edge connecting the set of input nodes $\mathcal{U}$ to the aggregated set of states $\mathcal{X}_1 \cup \mathcal{X}_2$. Thus, the CSU resulting from the aggregation is a CSU according to Definition 3.2. □

We conclude this section by providing the general definition of partitioning for a dynamical system:

Definition 3.5 (Control Partition of a Dynamical System)**.** For a dynamical system of the form (3.1) with equivalent graph $\mathcal{G}_k$, a control partition $\mathcal{P}_k$ is defined as a collection of CSUs:

$$\mathcal{P}_k = \{\mathcal{S}_{1,k}, \ldots, \mathcal{S}_{m,k}\} \quad \text{s.t.} \quad \left(\biguplus_{i=1}^{m} \mathcal{S}_{i,k}\right)\Big|_{\mathcal{G}_k} = \mathcal{G}_k. \tag{3.10}$$

This definition of control partition comprehends both non-overlapping and overlapping partitionings thanks to the definition of aggregation operation.

3.3.2 The Structure of the Generalized Partitioning Strategy

In this section, we present our novel generalized partitioning technique[1]. This technique applies to all types of dynamical systems that can be represented in form (3.1). Here, we describe how the algorithms, metrics, and optimization problem presented in the next sections merge to construct the generalized partitioning strategy. A generalized partitioning technique consists of two main steps:

1. Selection of the FSUs, performed through an algorithmic procedure.
2. Partitioning by aggregation, achieved either by solving an integer optimization problem, or through an algorithm.

For step 1), we describe the algorithmic approach in Section 3.4. This algorithm is designed to work over the equivalent graph representation of the dynamical system.

Once the FSUs are defined, we are presented with a collection of subsystems that can not be further divided without losing the non-autonomicity characteristic. By aggregating these subsystems, we obtain the desired partition of the system. To this end, first, we introduce a novel metric in Section 3.5.1, which we call the partition index. This is a generalized global metric that needs information about the entire structure of the network and can be characterized according to the partitioning procedure that one wants to deploy. Then, we formulate the integer optimization problem to obtain the best value of the partition index in Sec. 3.5.4. Alternatively, two algorithmic approaches to solve the same problem with a different characterization of the partition index are presented in Sec. 3.5.2 and 3.5.3. All strategies have advantages and disadvantages, which will be discussed in the respective sections.

3.4 Algorithm for the Selection of FSUs

Here, we present the algorithm for the selection of the FSUs. This algorithm is general and can be applied to any network with structure (3.1). Given the time-dependence of (3.1), this algorithm is potentially applied at each time step k, but in the following we omit the subscript k for simplicity. Also, the algorithm can be specialized to exploit the specific structure of the dynamics for linear and PWA systems. Before proceeding, we introduce some additional notation that will be used in the algorithm. We will indicate the FSUs as subsystems $\mathcal{A}_i$, for $i = 1,\dots,N_{\text{FSU}}$, where $N_{\text{FSU}} \leq |\mathcal{V}|$, and the collection of all the FSUs as $\mathcal{A} = \{\mathcal{A}_1,\dots,\mathcal{A}_{N_{\text{FSU}}}\}$. Also, we will use the additional set $\mathcal{L} \subset \mathcal{X}$, indicating the state nodes that remain to be assigned to the FSUs after preliminary operations are complete.

The general algorithm for selecting FSUs consists of three main steps that we present in the following in detail. These steps will be performed in a specific order as a part of the iterative procedure illustrated in Alg. 3.1:

[1]The optimization problem and algorithms, as well as the examples and case studies presented below, can be accessed through the repository [277].

1. *Selection of the roots of the FSUs.* To select the FSUs, we first select their anchor points, one for each FSU. We call these anchor points the roots of the FSUs. These roots consist of at least one input node and one state node directly connected by an edge. The selection of the roots is performed on the subgraph $(\mathcal{V}, \mathcal{E}_{ux}, w_{ux}, \tilde{g})$, and works as follows. We start by assuming to have one FSU $\mathcal{A}_i \in \mathcal{A}$ for each input node $i \in \mathcal{U}$, where the indexing of the FSUs in the collection $\mathcal{A}$ may vary as the procedure progresses. Then, we consider for each input node in $\mathcal{U}$ the set of the edges connecting them to the state nodes: for $i \in \mathcal{U}$, if $w_{ij} \neq 0$, with $j \in \mathcal{X}$, we assign the state node j to the FSU $\mathcal{A}_l$ to which the input node i belongs. If we encounter a state node $j \in \mathcal{X}$ connected to two or more input nodes in $\mathcal{U}$, the entire group constitutes an indivisible root, and all are merged into the same FSU. Consequently, we update the indexing of the FSUs in the collection $\mathcal{A}$. We can have a maximum number of roots, and therefore of FSUs, equal to $N_{\text{FSU}} \leq |\mathcal{U}|$. All the unassigned state nodes remaining from the selection of the roots are collected into the list $\mathcal{L}$ that will be used in the next steps.

2. *Forward assignment of the state nodes.* In this step, we perform the assignment of state nodes not belonging to the roots of the FSUs already, and for which a directed edge from the state nodes in the roots exists. We call this procedure *forward assignment of the state nodes*. Here we consider the graph $(\mathcal{X}, \mathcal{E}_{xx}, w_{xx}, \tilde{g})$. For each unassigned node $j \in \mathcal{L}$, we consider the edge w_{ij} starting from a state node $i \in \mathcal{X}$ in the roots and ending in j that has the largest absolute value, i.e. $\max_i |w_{ij}|$. If this w_{ij} exists, we assign the node $j \in \mathcal{L}$ to the FSU to which the state node $i = \arg\max_i |w_{ij}|$ belongs, and we update the set $\mathcal{L}$. If such w_{ij} does not exist, we proceed with the assignment procedure for the next state node in $\mathcal{L}$ until no further forward assignments are possible. We call this phase *forward assignment* since we look at the edges starting from the states in the roots of the FSUs and that are directed toward the periphery of the equivalent graph.

3. *Backward assignment of the state nodes.* This step of the algorithm assigns the state nodes in $\mathcal{L}$ that have no forward connection from the roots. Therefore, we call this step *backward assignment* because we consider the connections from the nodes belonging to the periphery of the graph toward the direction of the roots. For each remaining $i \in \mathcal{L}$, following the same principle in step 2), we select the node j belonging to an FSU such that $j = \arg\max_j |w_{ij}|$. We repeat the procedure for the next state node in $\mathcal{L}$ until we have checked all the nodes once.

Remark 3.1. If the system considered is time-varying, its topology can change at each time step. For such systems, the algorithm for selecting the FSUs has to be applied each time there is a variation in the topological structure, i.e. in the edges or their weights.

Remark 3.2. If outputs are also specified in the dynamics of the system, the procedure defined above still holds, but it is necessary to assign the output nodes to each CSU in the same way as done for the state nodes.

Algorithm 3.1 Selection of FSUs

Part 1 - Selection of the roots
data: $(\mathcal{V}, \mathcal{E}_{ux}, w_{ux}, \tilde{g})$
perform 1) in Sec. 3.4: selection of the roots
return $\left(\mathcal{A}^{[0]}, \mathcal{L}^{[0]}\right)$
Part 2 - Selection of the FSUs
data: $(\mathcal{X}, \mathcal{E}_{xx}, w_{xx}, \tilde{g})$
$k \leftarrow 0$
while $\mathcal{L}^{[k]}$ not empty $\vee$ $\mathcal{L}^{[k]}$ connected **do**
    Perform 2) in Sec. 3.4: forward assignment
    $\left(\mathcal{A}^{[k+1]}, \mathcal{L}^{[k+1]}\right) \leftarrow \left(\mathcal{A}^{[k]}, \mathcal{L}^{[k]}\right)$
    Perform 3) in Sec. 3.4: backward assignment
    $\left(\mathcal{A}^{[k+2]}, \mathcal{L}^{[k+2]}\right) \leftarrow \left(\mathcal{A}^{[k+1]}, \mathcal{L}^{[k+1]}\right)$
    $k \leftarrow k+2$
end while

3.5 The Partitioning Strategy

In this section, we present a novel metric that will be used to address the partitioning problem in a generalized setting. We call this metric the *partition index*. Then, we propose an algorithmic procedure to solve the problem, which returns the partition, i.e. a set of CSUs, for the required granularity. This approach provides a sub-optimal solution but requires polynomial time to be computed. In Sec. 3.6.2, we empirically show how this first approach offers a good trade-off between optimality of the solution and computational complexity. Moreover, we show that when the partition index is a quadratic function, the partitioning problem can be formulated as an integer quadratic program (IQP). This will require a novel characterization of the partition index, which must be a quadratic function. In Sec. 3.6.2, we show how this approach provides the best solution to the problem, but the downside is that it is an NP-hard problem.

3.5.1 The Partition Index

For a specific partition $\mathcal{P}$ we denote the *partition index* by $p^{\text{idx}}(\mathcal{P})$. We define this metric to account for both the interactions that occur inside a CSU, and among the CSUs. In this sense, the partition index is a global metric since it requires global information about the system to be computed.

Consider a partition $\mathcal{P} = \{\mathcal{S}_1, \dots, \mathcal{S}_{N_{\text{CSU}}}\}$, with $N_{\text{CSU}} \leq N_{\text{FSU}} \leq p$ where each $\mathcal{S}_i$ is a CSU in the sense of Definition 3.2. For CSU $\mathcal{S}_i$ we define following measures: the intra-CSU interaction $W_{\mathcal{S}_i}^{\text{intra}}$, the inter-CSU interaction $W_{\mathcal{S}_i}^{\text{inter}}$, and the size of the CSU $W_{\mathcal{S}_i}^{\text{size}}$, i.e. the number of FSUs belonging to $\mathcal{S}_i$. The components $W_{\mathcal{S}_i}^{\text{inter}}, W_{\mathcal{S}_i}^{\text{intra}}, W_{\mathcal{S}_i}^{\text{size}}$ of the partition index (3.13) can be constructed through the equivalent graph representation. For this, consider a non-linear system of the form (3.1), with equivalent graph representation $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w, \tilde{g})$ where the dependence on k is omitted. A CSU $\mathcal{S}_i$ belonging to such system has an equivalent graph $\mathcal{S}_i = (\mathcal{V}_i, \mathcal{E}_i, w_i, \tilde{g}_i)$. For CSU $\mathcal{S}_i$ we can define the intra-CSU, and

the inter-CSU interactions respectively as

$$W_{\mathcal{S}_i}^{\text{intra}} = \sum_{s,t \in \mathcal{V}_i} |w_i(s,t)| \tag{3.11}$$

$$W_{\mathcal{S}_i}^{\text{inter}} = \sum_{s \in \mathcal{F}_{\mathcal{S}_i}} \sum_{j \in \mathcal{N}_{\mathcal{S}_i}} \sum_{t \in \mathcal{N}_s \cap \mathcal{V}_j} |w_i(s,t)| + |w_j(t,s)| \tag{3.12}$$

where $\mathcal{F}_{\mathcal{S}_i}$ denotes the frontier of the CSU $\mathcal{S}_i$, $\mathcal{N}_{\mathcal{S}_i}$ the set of neighbor CSUs of $\mathcal{S}_i$, and $\mathcal{N}_s$ the set of neighbor nodes of node s. Note that $W_{\mathcal{S}_i}^{\text{size}}$ is equal to the square of the number of FSUs $\mathcal{A}$ belonging to the CSU $\mathcal{S}_i$, i.e. $W_{\mathcal{S}_i}^{\text{size}} = |\mathcal{V}_i|^2$.

The partition index is defined as:

Definition 3.6 (Partition Index)**.** The partition index of a network of CSUs for a specified set-valued function h and a given partition $\mathcal{P} = \{\mathcal{S}_1, \dots, \mathcal{S}_m\}$ is a global metric given by:

$$p^{\text{idx}}(\mathcal{P}) = h\left(\sum_{i=1}^{m} W_{\mathcal{S}_i}^{\text{inter}}, \sum_{i=1}^{m} W_{\mathcal{S}_i}^{\text{intra}}, \sum_{i=1}^{m} W_{\mathcal{S}_i}^{\text{size}}, \alpha\right) \tag{3.13}$$

where $\alpha > 0$ is a scalar parameter allowing for the selection of the granularity of the partitioning

In the general case of the nonlinear dynamics (3.1), $p^{\text{idx}}(\mathcal{P})$ is time-dependent, because is is based on a time varying graph. The framework for partitioning that we propose is based on the optimization of $p^{\text{idx}}(\mathcal{P})$, and works for every characterization of h. In the following we propose one nonlinear form for h that we will use for algorithmic partitioning in Sec. 3.5.2; and one quadratic for the optimization-based strategy in Sec. 3.5.4. Other strategies using different characterizations of h can be defined by the same approach.

3.5.2 Algorithmic Partitioning

For the first characterization of the partition index (3.13), we assume a function that accounts for the ratio between the intra- and inter-CSU interactions. Since the algorithmic approach will try to maximize the partition index, then we assume that terms $W_{\mathcal{S}_i}^{\text{intra}}$ are at the numerator, and $W_{\mathcal{S}_i}^{\text{inter}}$ at the denominator. This approach will favor merging strongly coupled FSUs into the same CSU, while assigning weakly coupled FSUs to different CSUs. This approach is inspired by the modularity metric [51, 243], and the motivation for using it in non-centralized control is that such approaches modular partitioning usually improve performances of the architecture by minimizing the interaction among subsystem. Also, advantages in communication costs and privacy are present. However, it is important to stress that this partitioning approach will not provide the optimal partitioning in absolute terms (as we will show in Sec. 3.7), and which motivates the introduction of the abstract formulation (3.13). Moreover, we include in the (3.13) a novel term that accounts for the size of the resulting CSUs, according to the parameter α allowing for selecting the desired granularity of the partitioning:

$$p^{\text{idx}}(\mathcal{P}) = \frac{\sum_{i=1}^{m} W_{S_i}^{\text{Intra}}}{1+\sum_{i=1}^{m} W_{S_i}^{\text{Inter}}} + \frac{\alpha}{1+\sum_{i=1}^{m} W_{S_i}^{\text{size}}} \tag{3.14}$$

This index grows when the interactions among FSUs in the same CSU increase, the interactions among CSUs decrease, and the sizes of the CSUs decrease. A greedy algorithm maximizing this index is reported in Alg. 3.2. The time complexity of this algorithm is $O(N_{\text{FSU}}^3)$ because for each of the N_{FSU} iterations it is necessary to evaluate the immediate gain in (3.14) of the assignment of every unassigned FSU to each CSU, which can be at most N_{FSU}, and this operation has to be performed N_{FSU} times, until no more FSUs have to be assigned.

In the algorithm, we start by creating a list of unassigned FSUs $\mathcal{L}$, and an empty partition $\mathcal{P}$, which is a list with N_{FSU} entries, where the i-th entry is itself a sub-list denote by $\mathcal{P}_i$, also initialized as empty. The partition index corresponding to the empty allocation at the first step is $p^{\text{idx}}(\mathcal{P}) = 0$. At each iteration, the algorithm tries all possible assignments of the elements in $\mathcal{L}$ to all the sub-lists $\mathcal{P}_i$. For each assignment, it evaluates which element of $\mathcal{L}$ maximizes the immediate return given by the difference in (3.14) evaluated for $\mathcal{P}$ and for $\mathcal{P}^{\text{New}}$, where the latter is the candidate next partitioning under evaluation for an element of $\mathcal{L}$. This operation provides the maximum immediate gain Δp^{max}, which is re-initialized to minus infinity at each iteration of the while-loop. If the assignment of FSU $\mathcal{A}_i$ to the sub-list $\mathcal{P}_j$ gives a $\Delta p^{\text{idx}} > \Delta p^{\text{max}}$, then we store the candidate best allocation in the variables $\mathcal{A}_{\text{Next}}$, j_{Next}. When all possible evaluations in $\mathcal{L}$ are performed, the resulting $\mathcal{A}_{\text{Next}}$ is assigned to $\mathcal{P}_{j_{\text{Next}}}$, and removed from $\mathcal{L}$. The algorithm stops when $\mathcal{L}$ is empty, returning the partition $\mathcal{P}$.

3.5.3 Improved Algorithmic Partitioning

We improve the partition quality returned by the greedy Algorithm 3.2 by introducing a local refinement step at the expense of higher computational complexity. Specifically, once the assignment of an FSU to a CSU is performed, we can run a nested evaluation of the partition, going through all the FSUs already allocated and checking whether relocating them to a different CSU improves the value of the partition index, and iterating this procedure until no further improving by swapping is possible. A similar idea was proposed in [45], where an analogous procedure constitutes the overall partitioning strategy. The local refinement routine is reported in Alg. 3.3, and the improved algorithm is obtained by performing the steps in Alg. 3.3 right after the last step in the while loop of Algorithm 3.2, i.e. after $\mathcal{L}$.remove($\mathcal{A}_{\text{Next}}$) in line 11. Since in the worst case Alg. 3.3 re-runs three times through all the FSUs in a while loop, the overall computational complexity of the improved algorithm is $O(N_{\text{FSU}}^4)$.

3.5.4 Optimization-Based Partitioning

The characterization of the partition index proposed in (3.14) can be interpreted as a nonlinear mixed-integer metric, for which, in general, it is not possible to formulate programming problems allowing for a global optimum. To address this limitation, in this

Algorithm 3.2 Greedy algorithm for partitioning

data: $\mathcal{A} = \{\mathcal{A}_1, \dots, \mathcal{A}_{N_{\text{FSU}}}\}$
result: $\mathcal{P}$
$\mathcal{L} \leftarrow \mathcal{A}$, $\mathcal{P} \leftarrow \{\mathcal{P}_1, \dots, \mathcal{P}_{N_{\text{FSU}}}\}$, $p^{\text{idx}}(\mathcal{P}) \leftarrow 0$
while $length(\mathcal{L}) > 0$ **do**
    $\Delta p^{\max} \leftarrow -\infty$, $\mathcal{A}_{\text{Next}} \leftarrow$ None, $j_{\text{Next}} \leftarrow$ None
    for $i = 1, \dots, length(\mathcal{L})$ **do**
        $\mathcal{A}_i \leftarrow \mathcal{L}_i$
        for $j = 1, \dots, length(\mathcal{P})$ **do**
            $\mathcal{P}^{\text{New}} \leftarrow \mathcal{P}$ s.t. $\mathcal{P}_j$.append($\mathcal{A}_i$)
            $\Delta p^{\text{idx}} \leftarrow p^{\text{idx}}(\mathcal{P}^{\text{New}}) - p^{\text{idx}}(\mathcal{P})$
            if $\Delta p^{\text{idx}} > \Delta p^{\max}$ **then**
                $\mathcal{A}_{\text{Next}} \leftarrow \mathcal{A}_i$, $j_{\text{Next}} \leftarrow$ j, $\Delta p^{\max} \leftarrow \Delta p^{\text{idx}}$
            end if
        end for
    end for
    $\mathcal{P} \leftarrow \mathcal{P}$ s.t. $\mathcal{P}_{j_{\text{Next}}}$.append($\mathcal{A}_{\text{Next}}$), $\mathcal{L}$.remove($\mathcal{A}_{\text{Next}}$)
end while

section we propose a characterization of the partition index (3.13) that is a mixed-integer quadratic function. With this choice, we will define a multi-objective optimization problem that will try to balance the difference between $W_{\mathcal{S}_i}^{\text{inter}}$ and $W_{\mathcal{S}_i}^{\text{intra}}$, while accounting for the size of the resulting CSUs through the product of α and $W_{\mathcal{S}_i}^{\text{size}}$. The approach is conceptually similar to the one used to derive (3.14), but provides a mixed-integer quadratic metric. Consequently, the optimization-based strategy for partitioning is based on the solution of an Integer Quadratic Programming (IQP) problem, for which an optimal solution can be obtained.

Under these considerations, we introduce the binary variables $\delta_{ij} \in \{0, 1\}$ to specify whether or not an FSU $\mathcal{A}_i$ belongs to a CSU $\mathcal{S}_j$:

$$\delta_{ij} = 1 \iff \mathcal{A}_i \in \mathcal{S}_j \tag{3.15}$$

All decision variables δ_{ij} are collected into the vector δ. The total number of entries of δ is $N_{\text{FSU}}^2 \leq p^2$, since the range of i, j is $1, \dots, N_{\text{FSU}}$. The number of non-empty CSUs $\mathcal{S}_i$ that the optimization procedure will generate is unknown a priori, but it cannot exceed the number of FSUs by definition. Using the binary variables δ, we can rewrite the terms in (3.12)-(3.13) respectively as:

Algorithm 3.3 Partition refinement

data: $\mathcal{P}$
result: $\mathcal{P}$
$\Delta p^{\max} \leftarrow 0$, Exit_Condition $\leftarrow$ False
while Exit_Condition = False **do**
    $\mathcal{A}_{\text{Next}} \leftarrow$ None, $j_{\text{Next}} \leftarrow$ None, $j_{\text{Prev}} \leftarrow$ None
    for $i = 1, \ldots, length(\mathcal{P})$ **do**
        $\mathcal{A}_{\text{Test}} \leftarrow \mathcal{P}_{i,j}$ $\mathcal{P}^{\text{New}} \leftarrow \mathcal{P}$ s.t. $\mathcal{P}_i$.remove($\mathcal{A}_{\text{Test}}$)
        for $k = 1, \ldots, length(\mathcal{P})$ **do**
            if k != i **then**
                $\mathcal{P}^{\text{New}} \leftarrow \mathcal{P}^{\text{New}}$ s.t. $\mathcal{P}_k$.append($\mathcal{A}_{\text{Test}}$)
                $\Delta p^{\text{idx}} \leftarrow p^{\text{idx}}(\mathcal{P}^{\text{New}}) - p^{\text{idx}}(\mathcal{P})$
                if $\Delta p^{\text{idx}} > \Delta p^{\max}$ **then**
                    $\mathcal{A}_{\text{Next}} \leftarrow \mathcal{A}_{\text{Test}}$
                    $j_{\text{Next}} \leftarrow k$, $j_{\text{Prev}} \leftarrow i$
                end if
            end if
        end for
    end for
    if $\mathcal{A}_{\text{Next}}$!= None **then**
        $\mathcal{P} \leftarrow \mathcal{P}$ s.t. $\mathcal{P}_{j_{\text{Prev}}}$.remove($\mathcal{A}_{\text{Next}}$)
        $\mathcal{P} \leftarrow \mathcal{P}$ s.t. $\mathcal{P}_{j_{\text{Next}}}$.append($\mathcal{A}_{\text{Next}}$)
    else
        Exit_Condition $\leftarrow$ True
    end if
end while

$$W^{\text{inter}}(\delta) = \sum_{m=1}^{N_{\text{FSU}}} \sum_{i=1}^{N_{\text{FSU}}} \sum_{\substack{j=1 \\ j\neq i}}^{N_{\text{FSU}}} \sum_{\substack{l=1 \\ l\neq m}}^{N_{\text{FSU}}} \delta_{i,m}\delta_{j,l}\left(|w(i,j)| + |w(j,i)|\right) \tag{3.16}$$

$$W^{\text{intra}}(\delta) = \sum_{m=1}^{N_{\text{FSU}}} \sum_{i=1}^{N_{\text{FSU}}} \sum_{j=1}^{N_{\text{FSU}}} \delta_{i,m}\delta_{j,m}\left(|w(i,i)| + |w(i,j)| + |w(j,i)| + |w(j,j)|\right) \tag{3.17}$$

$$W^{\text{size}}(\delta) = \sum_{m=1}^{N_{\text{FSU}}} \left(\sum_{i=1}^{N_{\text{FSU}}} \delta_{i,m} \right)^2 \tag{3.18}$$

The network partition is thus expressed as a function of vector δ, i.e. $\mathcal{P}(\delta)$. Additionally, we include a set of constraints on assigning the FSUs to the CSUs so that we get a non-overlapping partitioning. This requirement is codified by imposing that the sum of variables $\delta_{i,j}$ over the index j is equal to one, i.e. each FSU must belong only to one CSU. The resulting IQP problem is:

$$\begin{aligned} \min_{\delta} \quad & p^{\text{idx}}(\mathcal{P}(\delta)) = W^{\text{inter}}(\delta) - W^{\text{intra}}(\delta) + \alpha W^{\text{size}}(\delta) \\ \text{s.t.} \quad & \sum_{j=1}^{N_{\text{FSU}}} \delta_{ij} = 1 \qquad \forall i \\ & \delta_{ij} \in \{0,1\} \end{aligned} \tag{3.19}$$

Solving the IQP problem (3.19) provides the partition minimizing the partition index (3.13) for a given choice of the granularity parameter α. Such optimization problems are known to be NP-hard [76]; therefore, their scalability is limited, as will be shown in the examples in Sec. 3.6.1.

Remark 3.3. For the problem definition (3.19), and for the algorithmic approach in Sec. 3.5.2, it is possible to find a range for the selection of α. In this range, for large values of α, the resulting partitioning will be the set of individual FSUs, i.e. $\mathcal{P} = \{\mathcal{A}_1, \ldots, \mathcal{A}_{N_{\text{FSU}}}\}$. Conversely, for small values of α, the resulting partitioning will be a single CSUs comprehending all FSUs, which is the entire system, i.e. $\mathcal{P} = \{\mathcal{S}_1\}$ with $\mathcal{S}_1 = \{\mathcal{A}_1, \ldots, \mathcal{A}_{N_{\text{FSU}}}\}$. Other choices of α in the range will lead to different levels of aggregation of the FSUs.

3.6 Examples: Application of the Generalized Partitioning Strategy

3.6.1 Selection of the FSUs

To show the features of Algorithm 3.1, we consider a dynamical system with 100 states and 20 inputs. The system has an equivalent graph represented in Fig. 3.1a. This graph can be interpreted as a snapshot of the dynamical relationships occurring through system variables at a certain time instant. The complexity of such graph makes the expert selection of FSUs, i.e. FSUs, a long and challenging task that quickly becomes intractable as the group size grows. In practice, no clear structure can be identified in such graph at first

sight. The application of Algorithm 3.1 reveals the existence of 17 FSUs, as shown in Fig. 3.1b. For such FSUamental units, all the couplings with their neighbors are guaranteed to occur through dynamic relationships, and no inputs are shared among them. As shown in Fig. 3.1b, this property might require the aggregation of some input nodes together to form single FSUs. It is interesting to notice that the existence of a forward path from an input to a state node inside an FSU implies, for discrete-time systems, that the input will affect that state in a number of time steps equal to the length of the path (an analogous consideration is possible for continuous-time systems in the number of their time derivatives [80]). Also, certain states, or groups of them, only exhibit a backward effect on the FSU they belong to. For such states, no forward path exists from any input node, making them unreachable states. Still, we assign them to their closest FSU according to their coupling strength.

3.6.2 Partitioning a Modular Network

We now show the application of our partitioning strategy to a modular network [243][52] of FSUs reported in Fig. 3.2. Such network is characterized by a basic pattern of four FSUs strongly coupled, but weakly coupled with similarly configured groups. This type of connection structure is repeated throughout the network at different levels of aggregation. We propose this example to show three important aspects of the proposed approach: 1) the use of the partition index (3.13) allows to obtain partitions that aggregate together strongly connected components of the network while minimizing the coupling with the others; 2) the function of the granularity parameter α defined in (3.13), which allows selecting the relevance of the level of aggregation of the components of the partition, thus returning different configurations; 3) the respective advantages and limitations of the optimization-based and algorithmic partitioning approaches.

In the network in Fig. 3.2, a basic pattern of four FSUs that are strictly connected can easily identified by inspection. This pattern repeats at a coarser level if we consider a group of 16 FSUs connected by their corner links. Further, the structure repeats if we consider the entire network of 64 FSUs. To partition this network, we first apply the optimization-based strategy of Section 3.5.4. To this end, we need to select a value for the granularity parameter α. In our testing, we used a value related to the weights of the edges of the graphs, i.e. $\alpha = \left(\frac{\kappa}{w^{\min}}\right)^2$, where κ is a scalar free to select by the user, and $w^{\min}$ is the minimum weight of the edges of the graph. By varying κ, we obtain different α and consequently different partitions. We tried a broad spectrum of values α; however, we retrieved only four distinct partitions, which actually cover all the possible aggregations of the FSUs in Fig. 3.2 at different granularities that we can identify by inspection. Specifically, for $\kappa = 1, 0.1, 0.01, 0.001$ we have $\alpha = 10^6, 10^4, 10^2, 1$. The partitions obtained, which we denoted respectively as partitions $\mathcal{P}_1, \ldots, \mathcal{P}_4$, are represented in the corresponding Figs. 3.2a-3.2d. As expected, the larger the value of α is, the higher the relevance of the size of the resulting CSU in the partition. If the value of α becomes sufficiently large, then the resulting partition is constituted by all individual FSUs. Conversely, if α is small enough, the network partition is one individual CSU aggregating all the FSUs together. The other two selections of α return groups of 4 or 16 FSUs, representing the other two optimal aggregations of FSUs. When applying algorithmic partitioning to the same problem, we have retrieved two of the four optimal results obtained with optimization-based partitioning. In particular, using the metric (3.14), with a choice of $\alpha = 25$ we get the partition in Fig. 3.2a, where

every FSU is allocated into a distinct CSU. By reducing α to 1, we get the partition in Fig. 3.2a, with 16 CSUs containing 4 FSUs each. For α smaller than 1, we obtained various combinations of the basic groups of four FSUs, but never the exact partition in Fig. 3.2c

The last aspect to consider in this example is the computational cost of the two approaches. We solved the optimization-based problem using the Gurobi Optimizer software [123], running with parallel computing on a CPU Intel XEON E5-6248R with 24 cores at 3GHz. In the worst case scenario, that is for the partition in Fig. 3.2a with 64 individual FSUs, we let the optimizer run for 3 hours and 34 minutes before prematurely stopping it while reaching a solution gap of 7.07%. For the other cases, the solver could get the solution faster, with computation times as small as 15 minutes. The algorithmic approach is executed as a single-threaded task, given its sequential nature. The solution has been retrieved consistently in less than 600 seconds, with a clear computational advantage compared to the optimization-based approach.

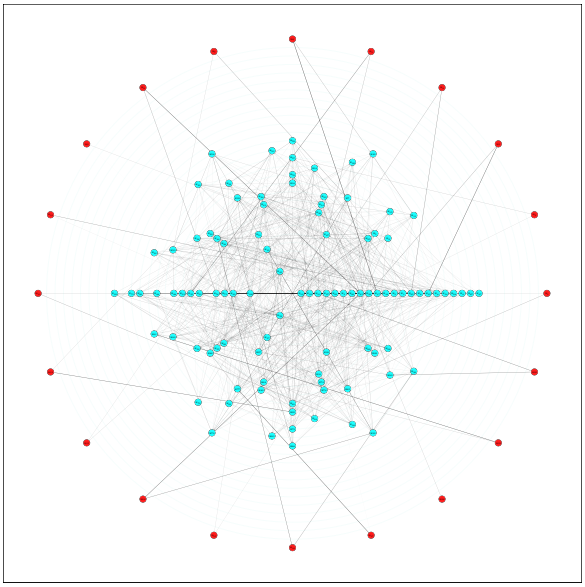


Fig. 3.1a

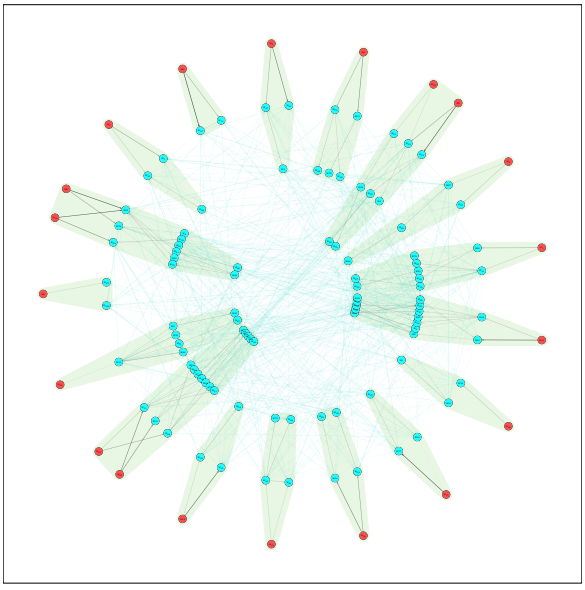
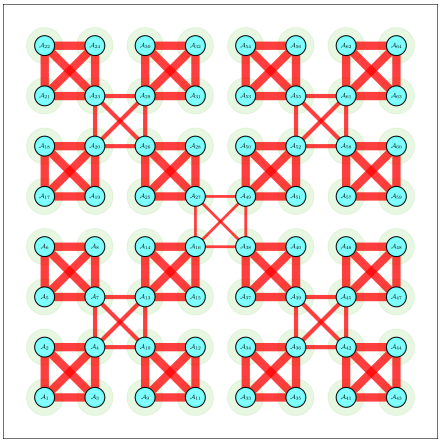
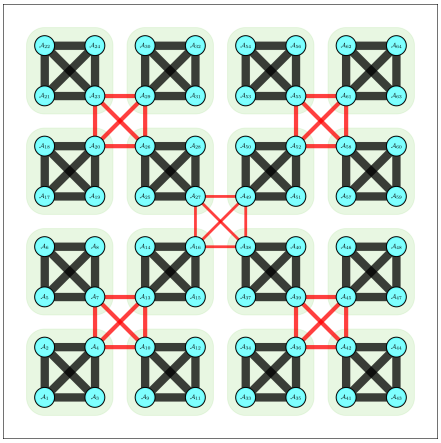


Fig. 3.1b

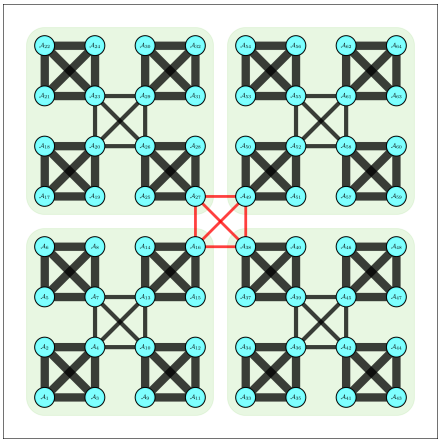
Figure 3.1: Representations of the equivalent graph of a system with 20 input (in red) and 100 states (in cyan) variables. Fig. 3.1a: the concentric degree-based representation, where input nodes are positioned in the most outer part of a circle, while state nodes on concentric rings representing their degree, with the most outer being degree one, and growing of one each ring toward the center. Fig. 3.1b: the representation of the FSUs after the application of the selection procedure. Each of the 17 FSU is represented by the green area, where dynamic relationships among the variables in the same FSU are represented by with black arrows, and among the FSUs by green arrows.



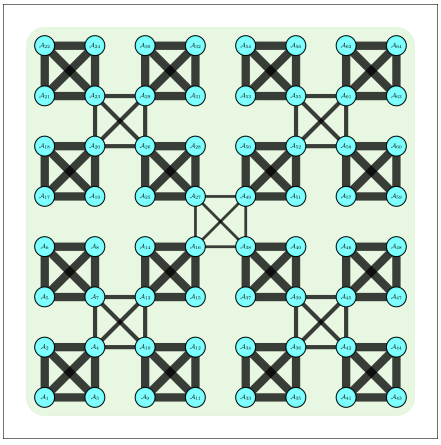
Partition $\mathcal{P}_1$



Partition $\mathcal{P}_2$



Partition $\mathcal{P}_3$



Partition $\mathcal{P}_4$

Figure 3.2: Graphs related to the partitioning of a modular network with 64 FSUs. The strength of the connection is represented by the thickness of the links. Partitions $\mathcal{P}_1$-$\mathcal{P}_4$ are obtained for values $\alpha = 10^6, 10^4, 10^2, 1$ respectively.

3.7 Case Studies: The Role of Partitioning in Distributed Predictive Control

In this section, we illustrate the applicability of the partitioning techniques developed throughout the chapter to the distributed MPC (DMPC) control of networks. The DMPC architecture we will use for this aim is based on the alternating direction method of multipliers (ADMM) [49], which is a well-established framework [288, 321]. We will show how our partitioning technique can improve the performance of DMPC-ADMM, both in terms of stage and computation cost. This will be achieved first by applying DMPC-ADMM on the modular network of 64 FSUs presented in Sec. 3.7.1 and assuming that each FSU is a linear system. Then in Sec. 3.7.2, for the same network, we assume that each FSU is a piecewise affine (PWA) system [129, 314, 323], for which, at each ADMM iteration, we deploy the MPC technique formulated in [36]. The performance of DMPC-ADMM improved w.r.t. the conventional formulation, where all FSUs are considered independently, while retaining the same pattern in computation times and costs obtained for linear systems. Finally, in Sec. 3.7.3, we propose a case study of a random network of 50 FSUs for which we will compare optimization-based and algorithmic partitioning.

3.7.1 DMPC-ADMM for a Modular Network of Linear FSUs

We consider each FSU a linear systems with matrices A and B such that the dynamics is stable and the network is controllable. We consider $A_{(i,i)} = 0.5$, $B_{(i,i)} = 1$, and values $A_{(i,j)} = 0.1$ or 0.01 for strong and weak couplings respectively, as represented by the thickness of the edges in Fig. 3.2. For this system, we consider a prediction horizon of 30 in time steps. Centralized MPC [227, 295] is applied to partition $\mathcal{P}_4$, while DMPC-ADMM is used for the others, according to the formulations [288, 321], and selecting parameters $\epsilon = 0.001$ (consensus threshold), $\rho = 0.01$ (penalty parameter). The control objective is to track a sinusoidal reference of unit amplitude and with frequency $w = 2\pi/15$, with bounds $u^{[i]} \in [-0.5; 0.5]$, $x^{[j]} \in [-0.9; 0.9]$ $\forall i, j$. We assume a zero initial condition. Given the maturity of the MPC framework, it is not surprising that the controller achieves excellent performance for CMPC, and without any substantial variation for the DMPC-ADMM alternatives. Control objectives are met easily; thus, we omit them, but they are accessible in the repository [281]. Instead, we are interested in evaluating the results through the performance metrics reported in Tab. 3.1, where in the first column, each row corresponds to control applied using one of the partitions derived in Sec. 3.6.2. In the second column, we report the number of CSUs in the partition, corresponding to the number of computing cores necessary to deploy a fully distributed control architecture. Next, we have the value of the cumulative stage cost at the final step of the simulation. We see that the performance deteriorates minimally with a growing number of CSUs, not significantly enough to prefer the selection of a strategy w.r.t. another, but also indicating that each distributed approach based on the partitions obtained is effective in controlling the network. We also evaluate the different approaches through parallel computation time, and computational cost, and for the latter we use core seconds, i.e. the total amount of seconds the computing cores operate to deploy a fully distributed strategy. We compute core seconds by picking at each ADMM iteration the computation time of the slowest core and multiplying it by the number of cores, then summing all these values throughout the simulation. If we consider

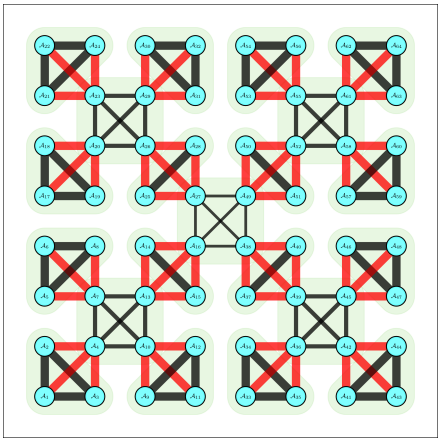


Figure 3.3: A partition selected to maximize the interaction among CSUs used in Sec. 3.7.1

the parallel computation time in the fourth column, we see that the higher the number of CSUs is, the faster the DMPC-ADMM strategy is. To put the numbers in perspective, in the fifth column we consider the time required by fastest architecture $\mathcal{P}_1$ as a reference unit, and compute the ratio with the other approaches. It follows that the partition $\mathcal{P}_2$ is only 1.35 times slower than $\mathcal{P}_4$, while the centralized MPC 4.66 times. Therefore, even if $\mathcal{P}_1$ is much faster than CMPC, there is a diminishing return w.r.t. $\mathcal{P}_2$. This fact can also be observed in Fig. 3.4a, where we report the computation times. Finally, we compare the computational costs in the last two columns of Tab. 3.1. Accordingly, considering CMPC in $\mathcal{P}_1$ the least expensive strategy, the computation cost increase trend is almost linear in the number of cores. This fact can be seen in Fig. 3.4b, where we plot the computation costs. The analysis of the results shows that while the computation cost increases almost linearly with the number of CSUs, the computation time decreases exponentially, highlighting a pattern of diminishing returns. Accordingly, it is comparatively more expensive to increase the number of CSUs from the computation perspective. In this case, partition $\mathcal{P}_2$ is 1.35 times slower than $\mathcal{P}_1$ while being 2.94 cheaper, and requiring 48 computing cores less to be executed. Considerations of this nature can vastly differentiate the selection of different approaches in practice. While DMPC-ADMM performance as a function of a selected partition is not an issue for linear systems, computation times can be. As an example, we considered a partition manually selected to maximize the inter-CSU coupling, shown in Fig. 3.3, and accessible at [281]. As a result of our testing of the same control problem of the other cases, even if the ADMM approach can converge, we get an estimated parallel computation time of 160 hours for the same simulation compared to less than 400 seconds for the other approaches, a clear demonstration of the value of proper selection of partitions.

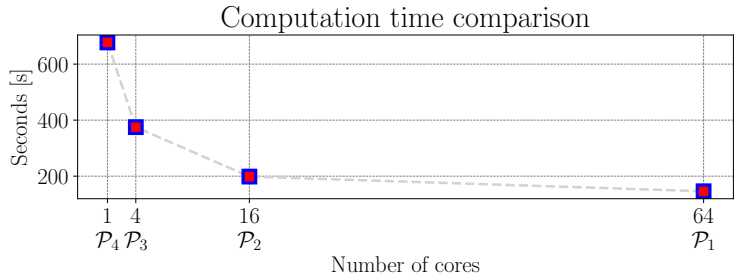


Fig. 3.4a

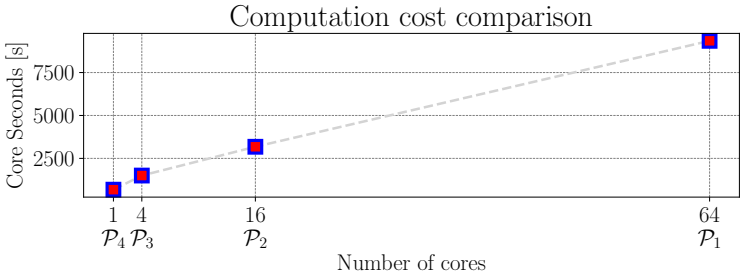


Fig. 3.4b

Figure 3.4: Comparison of the computation times (Fig. 3.4a), and computation costs (Fig. 3.4b) for the different partitions. For the former, the gray trend line highlights a diminishing return in computation speed as the number of cores increases, while for the latter the almost linear relation between the number of cores and their associated computation cost.

3.7.2 DMPC-ADMM for a Modular Network of Hybrid FSUs

We now assume that each CSU in the network Fig. 3.2 is a hybrid system, specifically a PWA system of the form:

$$x^{[i]}(k+1) = 0.5x^{[i]}(k) + u^{[i]}(k) + \sum_{j\in\mathcal{N}_i} A_{(i,j)}x^{[j]}(k) \qquad \text{if} \quad x^{[i]}(k) \geq 0 \tag{3.20}$$

$$x^{[i]}(k+1) = -0.5x^{[i]}(k) + u^{[i]}(k) + \sum_{j\in\mathcal{N}_i} A_{(i,j)}x^{[j]}(k) \qquad \text{if} \quad x^{[i]}(k) < 0$$

where, as in the previous case, $A_{(i,j)} = 0.1$ or 0.01 as represented in Fig. 3.2. Also, we retain the bounds $u^{[i]} \in [-0.5; 0.5]$, $x^{[j]} \in [-0.9; 0.9]$ $\forall i, j$. The objective is to track a sinusoidal

Table 3.1: Comparison of DMPC-ADMM performance applied to a network of linear systems for different partitioning strategies

Partition	Cores	Cost function value	Computation time [s]	Computation time ratio	Core seconds [s]	Core seconds ratio
$\mathcal{P}_4$	1	465.4703	680.88	4.6619	680.88	1.0000
$\mathcal{P}_3$	4	465.4709	369.56	2.5303	1478.23	2.1710
$\mathcal{P}_2$	16	465.5153	198.52	1.3592	3176.39	4.6651
$\mathcal{P}_1$	64	465.5158	146.05	1.0000	9347.37	13.7283

Table 3.2: Comparison of DMPC-ADMM performance applied to a network of hybrid systems for different partitioning strategies

Partition	Cores	Cost function value	Computation time [s]	Computation time ratio	Core seconds [s]	Core seconds ratio
$\mathcal{P}_4$	1	24949.16	849.57	2.7691	849.57	1.0000
$\mathcal{P}_3$	4	24945.76	632.35	2.0611	2529.41	2.9772
$\mathcal{P}_2$	16	24945.77	488.05	1.5907	7808.80	9.1914
$\mathcal{P}_1$	64	24946.04	306.80	1.0000	19635.58	23.1123

reference with unit amplitude and frequency $w = 2\pi/20$ from a zero initial condition. Here, we reduce the prediction horizon to 6 time steps to reduce the complexity of the problems. In this case we first convert the system into Mixed Logical Dynamical (MLD) form [129], and then we apply MPC as described in [36] for centralized control, and use the same procedure in combination with the DMPC-ADMM strategy [49, 288, 321]. In this case, at each step we are required to solve a Mixed Integer Quadratic Program (MIQP) [76] to obtain the control action. Even if for nonconvex and possibly nonsmooth optimization it is not always guaranteed that the ADMM algorithm will converge [342], similar approaches have been deployed successfully in literature [202]. We use the same hardware specified in Sec. 3.7.1. We report the results of the simulations in Figs. 3.5a, 3.5b and in Tab. 3.2. The results show a marginally improved cumulative stage cost. Also, the patterns found in the previous section for linear systems regarding computation times and costs repeat here.

3.7.3 DMPC-ADMM for a Random Network of 50 Hybrid Systems

We consider a network with 50 FSUs with hybrid dynamics (3.20) that are connected through a randomly generated topology, reproducible through the software in [281]. The network is in Fig. 3.7. We apply both optimization-based and algorithmic partitioning, obtaining different configurations of CSUs for which we will test CMPC and DMPC-ADMM control strategies as above. All the partitions obtained are reported in Fig. 3.7a - 3.7i. For both partitioning approaches, the endpoints of the range of α return two partitions that we denote by $\mathcal{P}^{\text{ADMM}}$ (with 50 individual CSUs) and $\mathcal{P}^{\text{CMPC}}$ (with one CSU for the whole network). The superscript notation refers to the fact that we deploy the conventional DMPC-ADMM strategy for the former partition, and for the latter CMPC. Selecting the weights as described in Sec. 3.6.2, for optimization-based partitioning we obtain five partitions $\mathcal{P}_i^{\text{Opt}}$, $i = 0,\dots,4$, for α varying, which are in Figs. 3.7a, 3.7b, 3.7e, 3.7h, 3.7i. For algorithmic partitioning we have the six partitions $\mathcal{P}_i^{\text{Alg}}$, $i = 0,\dots,5$, shown in Figs. 3.7a, 3.7c, 3.7d, 3.7f, 3.7g, 3.7i. Respective weights are indicated below the figure, where the CSUs are represented by the colored areas collecting together FSUs. Given such partitions, we apply the same DMPC strategy deployed in Sec. 3.7.2. This time, we assume a different initial condition for each CSU and a different sinusoidal reference to track, both randomly generated. Data resulting from the simulations and scripts to reproduce them can be accessed at [281]. Reporting the graphs of each simulation is prohibitive. Therefore, here we propose in Figs. 3.6a, 3.6b respectively the evolution of the maximum error and the maximum absolute value of the states across all FSUs for each partition, showing that the controller can achieve the objectives. Results for all partitions are in Tab. 3.3. Here we notice that, while for $\mathcal{P}^{\text{CMPC}}$ we still get the best performance at the expense of the highest computation time, for $\mathcal{P}^{\text{ADMM}}$ we have a different situation w.r.t. what we found

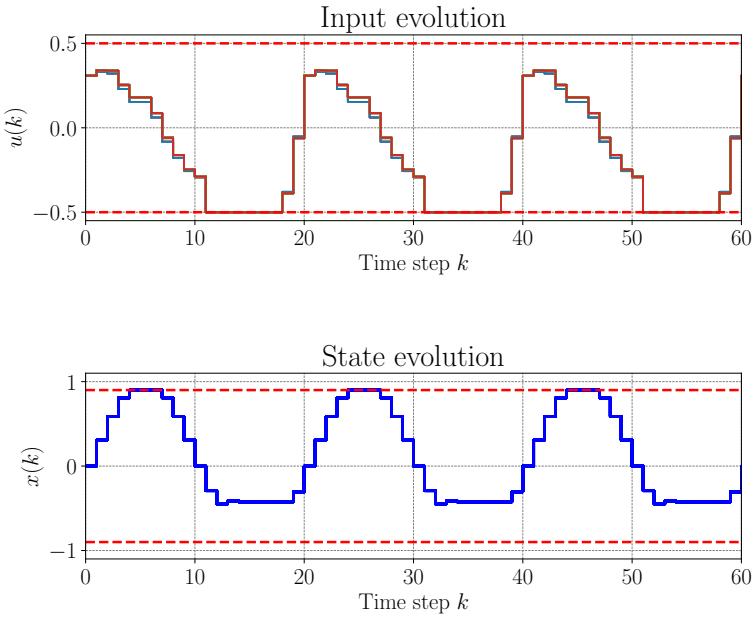


Figure 3.5: Evolution of the inputs (Fig. 3.5a) and of the states (Fig. 3.5b) for the network of PWA FSUs. For the former, all signals are approximately the same and they respect the constraints.For the latter, when the value of a state crosses the threshold at zero, we have a change of dynamics, which is reflected in a different behavior. Accordingly, the input signal changes too, but reaches saturation before the state is able to reproduce the reference.

previously in Sec. 3.7.2. For $\mathcal{P}^{\text{ADMM}}$, there is now a substantial difference in performance w.r.t. the other partitions, which poses the question of the impact of topology in network control. The best-performing strategies are now in order $\mathcal{P}_3^{\text{Alg}}$ and $\mathcal{P}_2^{\text{Opt}}$, with a difference in the cost value of 0.16%, 0.24% respectively, but a significantly lower computation time for $\mathcal{P}_2^{\text{Opt}}$, which can be considered as the most suitable partition in this case. To further compare the results, Figs. 3.8a, 3.8b show the graphs relating computation time and cost to the number of CSUs in the network, and consequently of computing cores necessary for the implementation of the distributed architecture, as done in Sec. 3.7.2. Here, we see that for optimization-based partitions, we still get the same trend in the scaling of the computation w.r.t. the number of CSUs as in Fig. 3.4a, and an almost linear scaling in the computation cost as in Fig. 3.4b. However, for algorithmic partitioning, we get two outliners with $\mathcal{P}_4^{\text{AOpt}}$ and $\mathcal{P}_5^{\text{AOpt}}$, both relatively high in the computation time and cost, but different in the overall performance. This analysis suggests that partitions obtained with the optimization-based approach tend to provide more reliable performances according to these evaluation criteria. However, algorithmic partition remains a valid alternative, often providing good results with low computational cost.

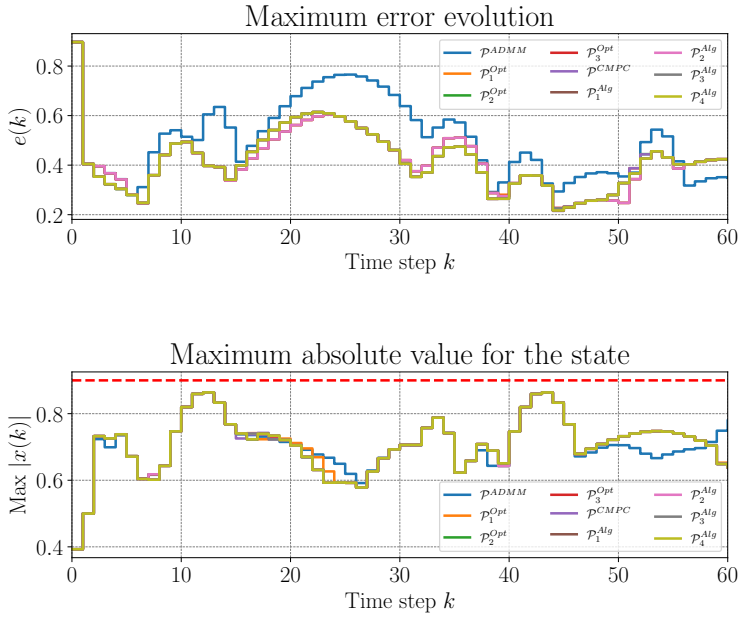
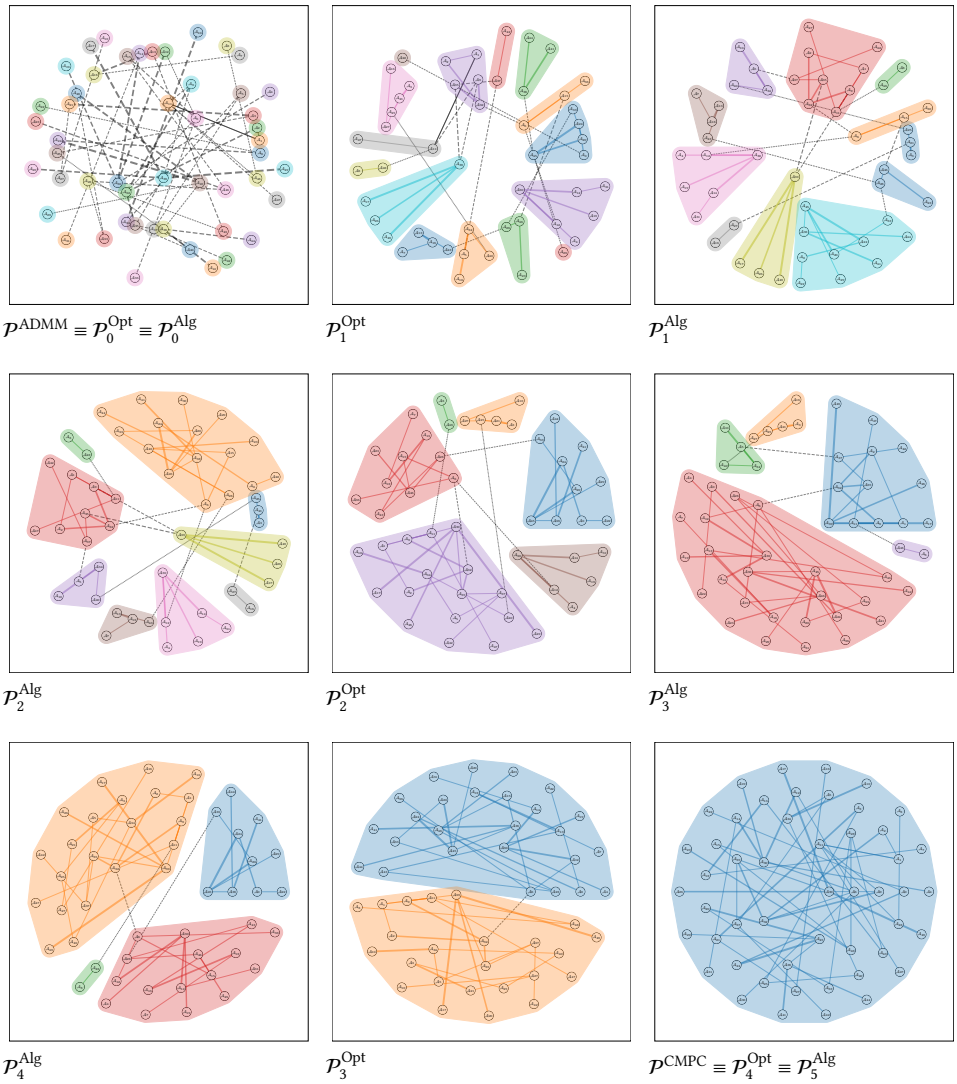


Figure 3.6: Evolution of the maximum error (Fig. 3.6a) and absolute value (Fig. 3.6b) across all CSUs for different partitions.

Table 3.3: Comparison of DMPC-ADMM performance applied to a random network of hybrid systems for different partitioning strategies

Partition	Cores	Cost fun. value	Opt. loss %	Comp. time [s]	Comp. time ratio	Core seconds [s]	Core seconds ratio
$\mathcal{P}^{\text{CMPC}}$	1	6899.9750	0.00	2628.04	26.4870	2628.04	1.3736
$\mathcal{P}^{\text{ADMM}}$	50	7749.2102	12.31	99.22	1.0000	4960.99	2.5930
$\mathcal{P}_1^{\text{Opt}}$	15	6976.9122	1.12	279.25	2.8145	4188.73	2.1893
$\mathcal{P}_2^{\text{Opt}}$	6	6916.7114	0.24	436.29	4.3972	2617.71	1.3682
$\mathcal{P}_3^{\text{Opt}}$	2	6918.2608	0.27	1368.36	13.7918	2736.72	1.4304
$\mathcal{P}_1^{\text{Alg}}$	11	6982.5798	1.20	173.93	1.7530	1913.28	1.0000
$\mathcal{P}_2^{\text{Alg}}$	9	6975.5149	1.09	353.81	3.5660	3184.27	1.6643
$\mathcal{P}_3^{\text{Alg}}$	5	6911.0475	0.16	2818.69	28.4085	14093.47	7.3661
$\mathcal{P}_4^{\text{Alg}}$	4	6923.4294	0.34	1681.59	16.9481	6726.37	3.5156



$\mathcal{P}^{\text{ADMM}} \equiv \mathcal{P}_0^{\text{Opt}} \equiv \mathcal{P}_0^{\text{Alg}}$ $\mathcal{P}_1^{\text{Opt}}$ $\mathcal{P}_1^{\text{Alg}}$

$\mathcal{P}_2^{\text{Alg}}$ $\mathcal{P}_2^{\text{Opt}}$ $\mathcal{P}_3^{\text{Alg}}$

$\mathcal{P}_4^{\text{Alg}}$ $\mathcal{P}_3^{\text{Opt}}$ $\mathcal{P}^{\text{CMPC}} \equiv \mathcal{P}_4^{\text{Opt}} \equiv \mathcal{P}_5^{\text{Alg}}$

Figure 3.7: Partitions of a random network with 50 FSUs. In the following, for each partition it is indicated the number of resulting CSUs, and the granularity parameter α used in optimization-based and algorithmic partitioning. $\mathcal{P}^{\text{ADMM}} \equiv \mathcal{P}_0^{\text{Opt}} \equiv \mathcal{P}_0^{\text{Alg}}$: 50 CSUs, $\alpha_0^{\text{Opt}} = 4.3 \cdot 10^6$, $\alpha_0^{\text{Alg}} = 100$; $\mathcal{P}_1^{\text{Opt}}$: 15 CSUs, $\alpha_1^{\text{Opt}} = 4.3 \cdot 10^4$; $\mathcal{P}_1^{\text{Alg}}$: 11 CSUs, $\alpha_1^{\text{Alg}} = 30$; $\mathcal{P}_2^{\text{Alg}}$: 9 CSUs, $\alpha_2^{\text{Alg}} = 20$; $\mathcal{P}_2^{\text{Opt}}$: 6 CSUs, $\alpha_2^{\text{Opt}} = 4.3 \cdot 10^3$; $\mathcal{P}_3^{\text{Alg}}$: 5 CSUs, $\alpha_3^{\text{Alg}} = 10$; $\mathcal{P}_4^{\text{Alg}}$: 4 CSUs, $\alpha_4^{\text{Alg}} = 1$; $\mathcal{P}_3^{\text{Opt}}$, 6 CSUs, $\alpha_3^{\text{Opt}} = 4.3 \cdot 10^2$; $\mathcal{P}^{\text{CMPC}} \equiv \mathcal{P}_4^{\text{Opt}} \equiv \mathcal{P}_5^{\text{Alg}}$, 1 CSU, $\alpha_4^{\text{Opt}} = 4.3 \cdot 10^1$, $\alpha_5^{\text{Alg}} = 10^{-13}$.

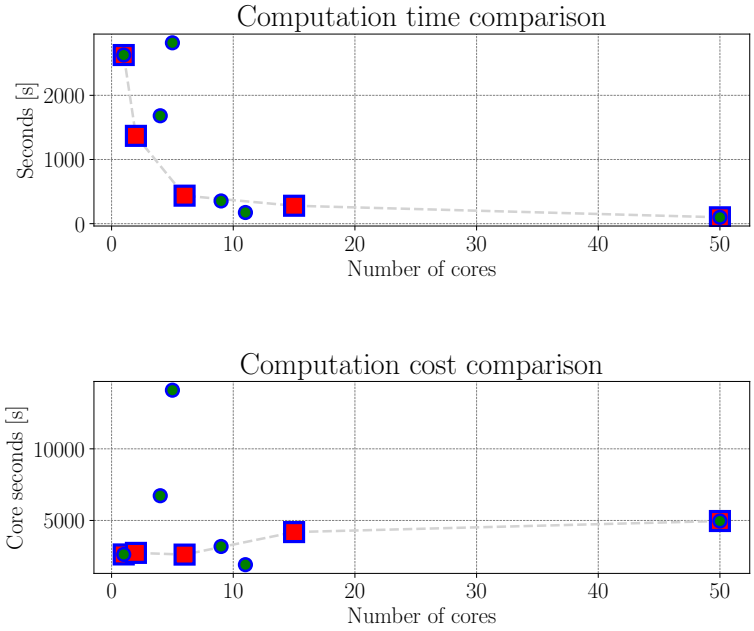


Figure 3.8: Computation time in [s] (Fig. 3.8a), and computation cost in core-seconds (Fig. 3.8b) required to perform the simulation related to the number of required cores. The results for each strategy are reported by a marker, square-red for optimization-based partitioning, circle-green for algorithmic.

3.8 Conclusions and Future Work

This work establishes foundations for the systematic partitioning of networks of control systems. We have pursued this goal by formalizing the concept of equivalent graph of a dynamical system, extending analogous previous propositions; and introducing the novel concepts of fundamental system unit (FSU), composite system unit (CSU), aggregation operation, and control partition. This has allowed us to propose a generalized partitioning technique consisting of two steps: first we select FSUs; then we aggregate them to obtain a collection of CSUs, which is the control partition. The former step is achieved through a selection algorithm; while for the latter we propose both an optimization-based and an algorithmic approach. These new approaches have been validated through examples, and case studies on networks of linear and hybrid systems of different topologies. Empirical evidence shows that, through partitioning, we can achieve superior distributed control performance, computation efficiency, and lower costs.

Future work could focus on improving the real-time applicability of these techniques and exploring the impact of the topology in distributed network control in terms of performance and stability. Other areas for extensions are the inclusion in the equivalent graph of external signals or disturbances through additional nodes, and applying generalized partitioning to control strategies different from predictive control.

4

Partitioning Techniques for Non-Centralized Predictive Control: A Systematic Review and Novel Theoretical Insights

"Men must fumble awhile with error to separate it from truth, I think– as long as they don't seize the error hungrily because it has a pleasanter taste."

Walter M. Miller Jr. , A Canticle for Leibowitz, 1959

The partitioning problem is of central relevance for designing and implementing non-centralized Model Predictive Control (MPC) strategies for large-scale systems. These control approaches include decentralized MPC, distributed MPC, hierarchical MPC, and coalitional MPC. Partitioning a system for the application of non-centralized MPC consists of finding the best definition of the subsystems, and their allocation into groups for the definition of local controllers, to maximize the relevant performance indicators. The present survey proposes a novel systematization of the partitioning approaches in the literature in five main classes: optimization-based, algorithmic, community-detection-based, game-theoretic-oriented, and heuristic approaches. A unified graph-theoretical formalism, a mathematical re-formulation of the problem in terms of mixed-integer programming, the novel concepts of predictive partitioning and multi-topological representations, and a methodological formulation of quality metrics are developed to support the classification and further developments of the field. We analyze the different classes of partitioning

This chapter is based on [283].

techniques, and we present an overview of their strengths and limitations, which include a technical discussion about the different approaches. Representative case studies are discussed to illustrate the application of partitioning techniques for non-centralized MPC in various sectors, including power systems, water networks, wind farms, chemical processes, transportation systems, communication networks, industrial automation, smart buildings, and cyber-physical systems. An outlook of future challenges completes the survey.

4.1 Introduction

4.1.1 Motivation

Modern systems are increasingly characterized by architectural scales and implementation complexities that challenge the implementation of centralized control strategies [172, 304]. This trend is supported by the advancements and availability of information transmission networks, as well as by the wide accessibility of computing resources [160]. When the scale of a system grows, it is common and advisable to structure it as a collection of autonomous interconnected components (subsystems). These subsystems should coordinate or be coordinated to achieve a common goal. To this aim, these entities necessitate local computing power, and communication and negotiation abilities: this is why, when these features are available, these advanced subsystems are usually defined as control agents. A schematic representation of a network of control agents is proposed in Fig. 4.1. Consequently, modern systems constituted by multiple agents having scales that exceed specific (hardware) operational thresholds are commonly referred to as large-scale multi-agent systems (LS-MASs) [87]. Examples of LS-MASs can be found in infrastructural systems such as power generation and distribution networks [152, 177, 262, 269]; urban and freeway networks [308]; railway and subway networks [201]; water distribution networks [33]; oil and gas distribution networks; large groups of mobile robots such as swarms of UAVs [357], or of terrestrial and maritime autonomous vehicles; large plants for chemical processing [232], which might also integrate autonomous energy generation; large industrial networks [106]; and satellite constellations [79]; where this list of applications keeps growing and evolving with the introduction of new technologies.

Conventional control methodologies such as proportional-integral control and pole placement [253], loop-shaping and h-infinity synthesis, [310], or feedback linearization [168] are not directly applicable to LS-MASs because of the presence of a large number of input-output channels and the large spatial distribution of such networks, which complicate centralized controller design and parameter tuning. Therefore, deployment of non-centralized control strategies [25, 303] is necessary for LS-MASs, and the level of sophistication of such approaches is tightly related to the availability of reliable communication channels and local computing power.

4.1.2 Non-Centralized MPC: Control Architectures

One of the most advanced modern control strategies is model predictive control (MPC) [271], which integrates the use of a mathematical model of the system dynamics with optimal control methodologies to compute predictive control actions that optimize performance while guaranteeing the stability of the controlled system, as well as the respect of operational constraints [204, 230], according to the receding horizon paradigm. The MPC

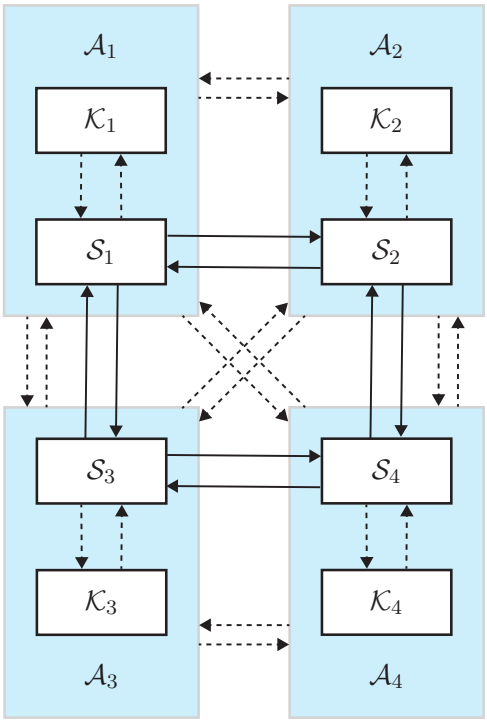


Figure 4.1: A network of control agents. Subsystems are indicated by $\mathcal{S}$, local controllers by $\mathcal{K}$, and control agents by $\mathcal{A}$. The solid lines represent the interactions at the physical level, i.e. the dynamical couplings; instead, the dashed lines represent interactions at the information level.

framework has also significantly evolved thanks to its design flexibility, which allows a relatively easy development of non-centralized predictive control strategies (NCen-MPC) [72, 208], i.e. of MPC strategies in which the computation of the control action for the overall system is not performed by a single central unit, but divided across control agents. The traditional classification of these strategies [295] comprehends decentralized MPC (Dec-MPC), distributed MPC (DMPC), and hierarchical MPC (HMPC). A conceptual representation of these architectures is proposed in Fig. 4.2. More recently, a novel NCen-MPC methodology incorporating concepts from game theory has emerged, called coalitional predictive control (Coal-MPC) [95]. In this survey, we will abbreviate centralized MPC as CMPC, to distinguish it from NCen-MPC. A list of these abbreviations is reported in Tab. 4.1. The single common characteristic of all NCen-MPC approaches is that they assume to operate in a network of agents, where, for each individual subsystem, a local optimization problem is solved. Then, the various techniques are distinguished according to how they handle communication and coordination of the local control actions.

When referring to NCen-MPC techniques, the simplest coordination technique is Dec-MPC, in which there is no communication among agents, but the effect of neighboring subsystems on local dynamics is generally assumed to be contained in invariant sets, thus allowing stable operation of such networks while preserving privacy, security, and

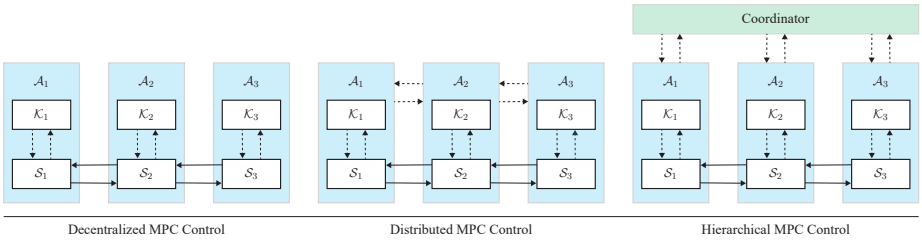


Figure 4.2: Main categories of non-centralized control architectures. In decentralized control, there is no information-level interaction among control agents. In distributed control, the information-level interaction is horizontal, i.e. each control agent can communicate with the others. In hierarchical control, the information-level interaction among control agents is vertical, i.e. they should, in principle, communicate with the coordinator. Mixed approaches are also possible.

resilience since there is no information sharing. A communication and coordination protocol is instead at the basis of DMPC approaches, where the agents in the network usually share their measurements or predicted evolution of local variables with neighbors, thus allowing for iterative or non-iterative adjustments of local control actions. In the context of linear systems, this distributed control approach can achieve global performance close to CMPC while drastically reducing computation times, and allowing real-time operation of the networks where centralized predictive control would not be possible. In HMPC, the control architecture is structured across multiple vertical layers, with at least the presence of a global coordinator and a set of local controllers. These strategies pose as an alternative to DMPC, and can enhance global coordination, as well as network resilience, introduce privacy features, or allow for multi-time-scale operation of different network models at different aggregation layers. Finally, the Coal-MPC strategy arises as the result of the combination of predictive control with game theory. In fact, in Coal-MPC, the network is seen as a collection of agents that participate in a cooperative game with the objective of maximizing the global collective outcome, which is the global operation cost of the network.

In conclusion, NCen-MPC strategies allow for the introduction of complex control features, such as advanced algorithmic coordination procedures, plug-and-play capabilities, and privacy and security preservation strategies, into LS-MASs. At the same time, NCen-MPC strategies can ensure stable real-time control of LS-MASs while preserving the optimality of their operation as much as possible.

4.1.3 The Partitioning Problem

The underlying assumption of the above discussion about NCen-MPC of LS-MASs is that the network is provided as a collection of agents with full autonomy. While this assumption may seem simple to satisfy, this is not always true in practice. In fact, the definition of the agents themselves may be challenging, especially for large and interconnected networks. Additionally, even if the network is given as a collection of individual agents, it might be more convenient for network operation to aggregate them into bigger entities. These two distinct classes of problems, i.e. the definition of the agents of the network and the problem of their aggregation, fall both into the category of network partitioning [64, 303].

Formally speaking, the partitioning problem consists of finding the optimal allocation of a group of elements into given sets according to a given metric. If the network $\mathcal{N}$ is provided as a collection of agents $N_{\mathcal{A}}$, i.e. $\mathcal{N} = \{\mathcal{A}_1, \dots, \mathcal{A}_{N_{\mathcal{A}}}\}$, and we have a number $N_{\mathcal{C}}$ of possible sets for the allocation, whose quality is defined by a cost function $h(\cdot)$, then the optimal partitioning problem consists in finding the set $\mathcal{P}$ (i.e. the partition) defined as $\mathcal{P} = \{\mathcal{C}_1, \dots, \mathcal{C}_{N_{\mathcal{C}}}\}$, where the elements $\mathcal{C}_i$ are groups of agents $\mathcal{A}_j$, such that the quality measure $h(\mathcal{P})$ is optimized. On the other hand, if the network $\mathcal{N}$ is provided as a monolithic system that does not show any natural decomposition, the partitioning problem consists of selecting several subsystems $N_{\mathcal{A}}$ for which control agents can be defined, which allows to interpret the network as a collection of agents $\mathcal{N} = \{\mathcal{A}_1, \dots, \mathcal{A}_{N_{\mathcal{A}}}\}$. Also in this case, the subsystem selection is generally guided by a cost function $h(\cdot)$. Both these problems are known to be NP-hard [51, 162, 293].

When the partitioning problem is applied to NCen-MPC, several further features can be developed and extended for both the partitioning and the MPC. Many questions may arise, such as: What is the best definition for the individual agents? How can agents be allocated optimally into sets to maximize the performance of the NCen-MPC architecture? How can the partitioning strategy handle topological changes in the network or different operating conditions? These are a few examples of profound technical challenges that researchers in this field have encountered in the last decades, finding answers and new open problems.

Many of the partitioning strategies that will be presented in this survey are borrowed from other scientific sectors, such as network and graph theory, machine learning, or computer science in general. A general overview of clustering methodologies applied to distributed network control can be found in [64], which can serve as a general reference for these methods, while the current survey is tailored specifically for NCen-MPC. We also refer to the work [346] to explore further general clustering methodologies such as k-means, fuzzy c-means, and hierarchical clustering. Other general approaches that have been applied to partitioning for NCen-MPC are community detection methodologies [101, 102], such as modularity maximization and spectral algorithms; and coalition formation approaches [14], which have led to the development of game-theory-based MPC architectures.

4.1.4 Survey Objectives and Contributions

Under these considerations, the present survey has two main overarching goals:

1. Unifying in a common framework all the results currently present in the literature addressing the partitioning problem for NCen-MPC.

2. Laying foundations for further systematic developments of this field.

These two objectives are achieved through the following series of steps: a systematization of fundamental notions for graph representation of dynamical systems and networks; the introduction of precise key performance indicators that are comparable across strategies and application domains, as well as a precise assessment methodology of the quality of a partition; a categorization of the known partitioning strategies for NCen-MPC in terms of methodology, partitioning objective, and relative control strategy; a discussion of the main partitioning methodologies to highlight their strengths and limitations; a brief technical discussion of each partitioning technique found in the literature; and a classification of the current application domains of the partitioning techniques.

MPC	Model Predictive Control
CMPC	Centralized Model Predictive Control
NCen-MPC	Non-Centralized Model Predictive Control
Dec-MPC	Decentralized Model Predictive Control
DMPC	Distributed Model Predictive Control
HMPC	Hierarchical Model Predictive Control
Coal-MPC	Coalitional Model Predictive Control
NLin-MPC	Nonlinear Model Predictive Control
LS-MAS	Large-Scale Multi-Agent System
MIMO	Multiple-Input Multiple-Output

Table 4.1: List of abbreviations

Further, we extend the analysis and classification of the partitioning techniques with novel theoretical insights, which are: the introduction of multi-topological graph representations to model variable topologies, and their link to hybrid systems; a formal definition of the partitioning problem for performance optimization in terms of a bi-level mixed-integer program (MIP); and a re-definition of the problem of time-varying partitioning, introducing the concept of predictive partitioning for control.

Given the extension of this survey, and the amount of different topics explored in detail, we provide an overview of its organization in Sec. 4.2 below, briefly describing the contents and the objectives of each section.

4.2 Organization of the Survey

In this section, we present the structure of the survey, briefly describing the content of each section. This will provide the reader with an organic view of the material presented, and will help to navigate the content, having a general knowledge of all the topics that will be discussed throughout the survey.

Graph representations Most partitioning techniques, both involving NCen-MPC or other control strategies, are based on abstract representations of the underlying system dynamics [303]. This representation is generally provided in the form of a graph [83]; therefore, it is natural to start the discussion about partitioning techniques by introducing graph representations in Sec. 4.3. In this section, we classify the graph representations used in partitioning and presented in Fig. 4.3. This classification is supported by a technical discussion of each type of representation in dedicated subsections.

Partitioning for predictive control Once the abstract representation of the network is available, the partitioning problem for NCen-MPC can be formally introduced and discussed in Sec. 4.4. In this section, we discuss the general problem definition and its common characteristics usually present in the partitioning techniques. In addition, we

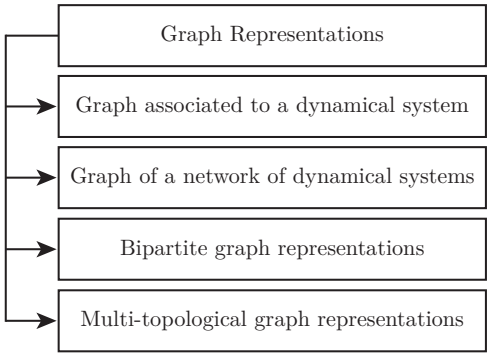


Figure 4.3: Graph representations used in partitioning for NCen-MPC.

provide metrics and an evaluation methodology to assess the quality of a partition, and we complete the discussion by introducing the novel concept of predictive partitioning as a component of the MPC formulation.

Classification of the partitioning techniques In Sec. 4.5, we will provide a classification of the partitioning methodologies for the application of NCen-MPC according to three criteria: 1) the general partitioning class; 2) the subclass defined by the main structure of the method or by its objective; and 3) the control architecture to which it has been applied. The classification performed according to the first two criteria is proposed in Fig. 4.4, where the first level of the classification tree defines the main class, and the second level defines the subclass. The main theoretical characteristics as well as the strengths and limitations of the five main partitioning classes are discussed in Sec. 4.5.1, for the subclasses in Sec. 4.5.2, and for the methodologies in Sec. 4.5.3. Finally, in Sec. 4.5.4 we classify the techniques according to the control methodology for which they have been designed.

Analysis of the individual partitioning techniques Once the classification of the partitioning strategies has been presented, and the main characteristics of each class and subclass have been highlighted, we deepen the technical discussion by providing further details about the methods in each class in Fig. 4.4. Therefore, an extensive analysis of the individual methodologies in the literature can be found in the dedicated sections, which are: Sec. 4.6 for optimization-based partitioning; Sec. 4.7 for algorithmic partitioning; Sec. 4.8 for community-detection-based partitioning; Sec. 4.9 for game-theory-based partitioning; and Sec. 4.10 for heuristic partitioning.

Applications In Sec. 4.11, we discuss the main case studies that have been used in the literature about partitioning for NCen-MPC. These are divided by application sector, and, when possible, we also provide reference systems with further details about the systems considered.

Conclusions and future work The overall discussion of the main topic of the survey is completed in Sec. 4.12 with final considerations about the state of this research field, and

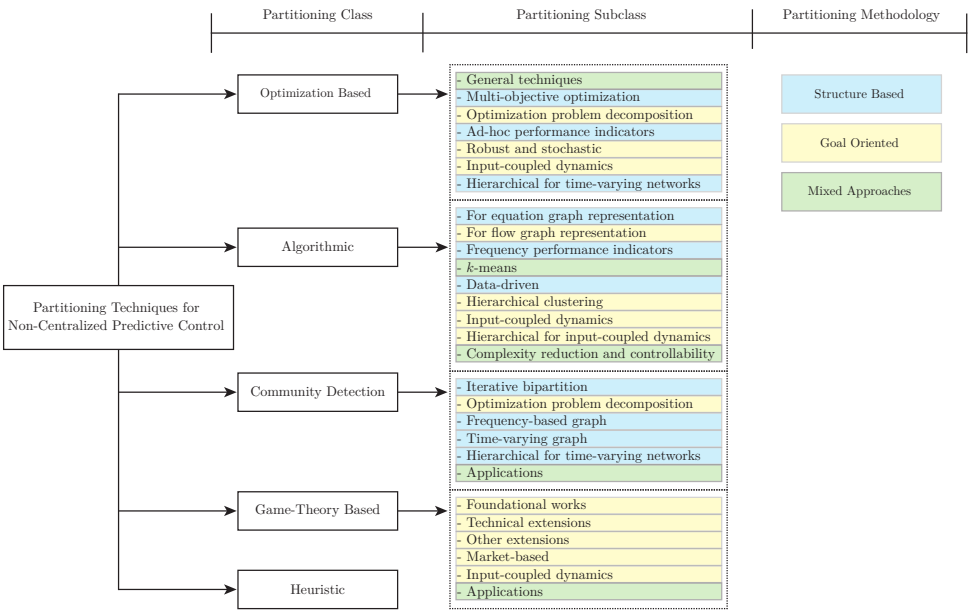


Figure 4.4: Categorization of the partitioning techniques in classes and subclasses. The methodologies in each subclass can be further distinguished between the approaches based on the structure of the network, and the ones oriented at achieving a given objective, whether it is a control or another functional specification.

with recommendations for future work, identifying the current research gaps and potential new directions to explore.

4.3 Graph Representations

At the basis of almost all partitioning approaches, there is a graph representation of the system to be decomposed. Accordingly, specific graph representations can be deployed when defining a partitioning strategy for applying an NCen-MPC method. These representations belong to three main categories: 1) graphs equivalent to dynamical systems; 2) graphs representing networks of dynamical systems; and 3) graph representations of an optimization problem. In this section, we first introduce graph theory terminology that will be used throughout the article. Then, we present the classes of graphs introduced above. We close the section by conceptually reformulating the graph representation of a network of dynamical systems linking multi-topological graphs and hybrid systems.

4.3.1 Fundamentals of Graph Theory

A graph [83] is an ordered pair of sets $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, \dots, n\}$ is the set of n vertices (or nodes), and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of the edges (or arcs, links). The edges are associated to the vertices through an $n \times n$ binary adjacency matrix A^{adj}, where $A^{\text{adj}}_{(i,j)} = 1$ if and only if an edge $\epsilon_{ij} = (i, j) \in \mathcal{E}$ exists. Therefore, the topology of the graph is specified by the adjacency matrix A^{adj}, and the set of the edges can also be written as $\mathcal{E} = \{(i, j) \,|\, [i, j \in \mathcal{V}] \wedge [A^{\text{adj}}_{(i,j)} = 1]\}$.

A subgraph of $\mathcal{G}$ is a graph $\mathcal{S}_\ell = (\mathcal{V}_\ell, \mathcal{E}_\ell)$ representing a part of $\mathcal{G}$. The set of vertices $\mathcal{V}_\ell$ is a subset of $\mathcal{V}$, i.e. $\mathcal{V}_\ell \subseteq \mathcal{V}$, and the set of the edges is $\mathcal{E}_\ell = \{(i,j) \,|\, [i,j \in \mathcal{V}_\ell] \wedge [A^{\text{adj}}_{(i,j)} = 1]\}$, where the topology is still specified by the relevant entries of A^{adj}. For a directed graph $\mathcal{G}$, an edge $\epsilon_{ij} = (i,j)$ denotes an arrow starting from node i and ending in node j. A graph $\mathcal{G}$ is weighted if a weighting matrix W^{adj} assigning to each edge a number is specified in addition to A^{adj}. For each vertex $i \in \mathcal{V}$ we denote by d_i its degree, i.e. the number of edges entering or exiting that vertex. In directed graphs, we can specify an in-degree (d_i^{in}) and an out-degree (d_i^{out}), if the edge is respectively ending or starting in the vertex i. For a vertex i, the neighborhood of i is the set of all vertices connected to it, and we denoted it by $\mathcal{N}_i = \{j \in \mathcal{V} \,|\, [(i,j) \vee (j,i)] \in \mathcal{E}\}$. For a subgraph $\mathcal{S}_\ell = (\mathcal{V}_\ell, \mathcal{E}_\ell)$, the frontier is its set of nodes that are connected to nodes outside the subgraph, i.e. $\mathcal{F}_\ell = \{i \in \mathcal{V}_\ell \,|\, [(i,j) \vee (j,i)] \in \mathcal{E}, j \in \mathcal{V} \setminus \mathcal{V}_\ell\}$. These fundamental concepts will be extended throughout the survey for specific topics when necessary.

4.3.2 Graph Associated to a Dynamical System

The most direct and intuitive graph representation of a dynamical system is the so-called associated graph. According to [303], the earliest formulation of this type of graph representation for linear systems can be traced back to [194]. We start by presenting associated graph representations for linear discrete-time systems, where the same formulation proposed in [303] for the continuous-time version holds. Consider the dynamics:

$$\mathcal{S} : \begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases} \tag{4.1}$$

where $x \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$, $y \in \mathbb{R}^{n_y}$ are respectively the state, input, and output of the system; and A, B, C are matrices of appropriate dimensions. The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated to (4.1) is constructed by first defining one node for each variable, which provides the set of vertices $\mathcal{V} = \{x_1, \dots, x_{n_x}, u_1, \dots, u_{n_u}, y_1, \dots, y_{n_y}\}$, where this set can be considered as the union of the sets for the individual state, input, and output variables, i.e. $\mathcal{V} = \mathcal{V}_x \cup \mathcal{V}_u \cup \mathcal{V}_y$, $|\mathcal{V}| = n_x + n_u + n_y$. Then, the set of edges $\mathcal{E}$ is built looking at the nonzero entries of matrices A, B, C, and as before, it can be thought of as the union of three different sets $\mathcal{E} = \mathcal{E}_{ux} \cup \mathcal{E}_{xx} \cup \mathcal{E}_{xy}$. These sets of edges define the interactions among variables, and are derived respectively as $\mathcal{E}_{ux} = \{(i,j) \,|\, i \in \mathcal{V}_u, j \in \mathcal{V}_x, B_{(i,j)} \neq 0\}$, $\mathcal{E}_{xx} = \{(i,j) \,|\, i,j \in \mathcal{V}_x, A_{(i,j)} \neq 0\}$, $\mathcal{E}_{xy} = \{(i,j) \,|\, i \in \mathcal{V}_x, j \in \mathcal{V}_y, C_{(i,j)} \neq 0\}$. This graph $\mathcal{G}$ associated with the dynamics (4.1) is static because the dynamical system is time-invariant. Moreover, the graph represents the interactions among the variables in the system. A measure of this interaction is provided by the weighting matrix that can be constructed considering the entries of matrices A, B, C:

$$W^{\text{adj}} = \begin{bmatrix} A & B & 0 \\ 0 & 0 & 0 \\ C & 0 & 0 \end{bmatrix} \tag{4.2}$$

A more recent evolution in the associated graph representation is found in [282], where the following nonlinear dynamics is considered:

$$\mathcal{S} : \begin{cases} x(k+1) = f(x(k), u(k)) \\ y(k) = h(x(k)) \end{cases} \tag{4.3}$$

The scope in [282] is to obtain a weighted and time-varying representation $\mathcal{G}(k) = (\mathcal{V}, \mathcal{E}(k))$ of the system (4.3). To this aim, using the same vertices definition introduced for (4.1), the following weighting function is defined:

$$w_{(i,j)}(k) = \begin{cases} \frac{\partial f_j(x(k),u(k))}{\partial i} & \text{for} \quad i \in \mathcal{V}_u \cup \mathcal{V}_x, j \in \mathcal{V}_x \\ 0 & \text{for} \quad i \in \mathcal{V}, j \in \mathcal{V}_u \\ \frac{\partial h_j(x(k))}{\partial i} & \text{for} \quad i \in \mathcal{V}_x, j \in \mathcal{V}_y \end{cases} \tag{4.4}$$

Accordingly, a time-varying set of edges $\mathcal{E}(k)$ is defined as:

$$\mathcal{E}(k) = \{(i,j) \mid i,j \in \mathcal{V}, w_{(i,j)}(k) \neq 0\} \tag{4.5}$$

4

This time-varying graph can capture the instantaneous interactions among the system variables at each time step. In the most general case, a different topological representation exists at each time step. Accordingly, a different choice of graph partition might be the best option for non-centralized predictive control. However, such an approach is computationally demanding.

Example 4.1. We consider the following linear discrete-time system to show how to construct the graph associated with a dynamical system. Consider the system:

$$x(k+1) = Ax(k) + Bu(k) \tag{4.6}$$

with $x \in \mathcal{X} \subseteq \mathbb{R}^{10}$, $u \in \mathcal{U} \subseteq \mathbb{R}^3$, where the matrices A and B are defined by the entries

$$\begin{array}{llll} a_{2,1} = 0.5 & a_{6,1} = 0.1 & a_{8,2} = 0.84 & a_{9,2} = 0.57 \\ a_{8,4} = 0.54 & a_{9,5} = 0.91 & a_{2,6} = 0.98 & a_{3,6} = 0.96 \\ a_{5,6} = 0.8 & a_{6,7} = 0.6 & a_{2,8} = 0.31 & b_{4,1} = 0.04 \\ b_{9,1} = 0.6 & b_{10,1} = 0.63 & b_{2,2} = 0.02 & b_{4,2} = 0.6 \\ b_{10,2} = 0.11 & b_{1,3} = 0.19 & b_{2,3} = 0.03 & \end{array} \tag{4.7}$$

and zero elsewhere. According to the definition of a graph $\mathcal{G}$ associated with a dynamical system, we define the set of vertices $\mathcal{V} = \{u_1, \ldots, u_3, x_1, \ldots, x_{10}\}$, while the nonzero entries of matrices A, B define the edges in $\mathcal{E}$ of the graph and their weights in the matrix W^{adj}. The representation of this graph is given in Fig. 4.5. This example will be continued in Sec. 4.4.1 to show how to select subsystems for constructing control agents.

4.3.3 Graph Representation of a Network of Systems

A different type of graph representation is considered when the dynamical system is a network admitting a natural decomposition into fundamental subsystems interacting through their dynamics. In this case, the network admits a graph representation $\mathcal{G}$ where the individual subsystems constitute the elements of the set of vertices $\mathcal{V} = \{\mathcal{S}_1, \ldots, \mathcal{S}_{N_S}\}$. The set of edges $\mathcal{E}$ is defined by state-to-state interactions. Accordingly, to each subsystem $\mathcal{S}_i$ are associated a local state $x_{\mathcal{S}_i} \in \mathbb{R}^{n_{x_{\mathcal{S}_i}}}$ and input $u_{\mathcal{S}_i} \in \mathbb{R}^{n_{u_{\mathcal{S}_i}}}$. The neighbors of a node of the network, i.e. of a subsystem $\mathcal{S}_i$, is the set $\mathcal{N}_{\mathcal{S}_i} = \{\mathcal{S}_j \mid (i,j) \in \mathcal{E}\}$. The definition of an output vector $y_{\mathcal{S}_i} \in \mathbb{R}^{n_{y_{\mathcal{S}_i}}}$ can also be included, but it will be omitted in the following for

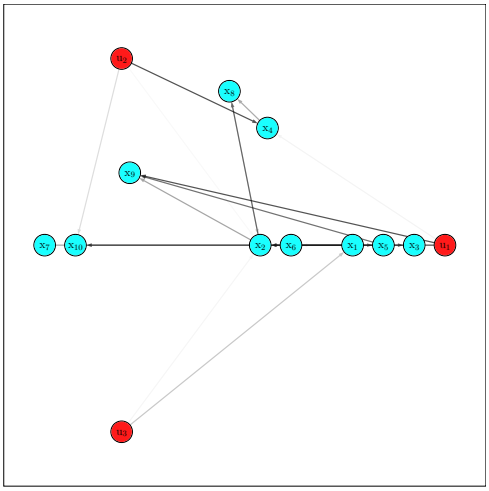


Figure 4.5: Graph associated with the dynamical system (4.7). The vertices are the system variables and are colored in red if they are inputs and cyan if they are states. The arrows represent the edges, and their opacity the strength of interaction, i.e. the weight, defined by the entries of matrices A and B.

simplicity. In other words, for a general nonlinear system of the form (4.3), there exists a natural subdivision of the state and input vectors such that every individual subsystem is described by:

$$\mathcal{S}_i : x_{\mathcal{S}_i}(k+1) = f_{\mathcal{S}_i}(x_{\mathcal{S}_i}(k), (x_{\mathcal{S}_j}(k))_{\mathcal{S}_j \in \mathcal{N}_{\mathcal{S}_i}}, u_{\mathcal{S}_i}(k)) \tag{4.8}$$

This type of representation has been extensively used in partitioning for non-centralized predictive control, especially in the form of linear interacting systems, where each subsystem takes the form:

$$\mathcal{S}_i : \begin{cases} x_{\mathcal{S}_i}(k+1) = A_{\mathcal{S}_i} x_{\mathcal{S}_i}(k) + B_{\mathcal{S}_i} u_{\mathcal{S}_i}(k) + w_{\mathcal{S}_i}(k) \\ w_{\mathcal{S}_i}(k) = \sum_{\mathcal{S}_j \in \mathcal{N}_{\mathcal{S}_i}} A_{\mathcal{S}_{ij}} x_{\mathcal{S}_j}(k) \end{cases} \tag{4.9}$$

Each subsystem $\mathcal{S}_i$ is affected only by its local input, and is coupled to its neighbors through dynamic interactions defined by matrices $A_{\mathcal{S}_{ij}}$. This coupling is seen by subsystem $\mathcal{S}_i$ as an exogenous signal $w_{\mathcal{S}_i}$ whose nature is determined by the coordination protocol used in the control strategy, i.e. it is considered a disturbance in decentralized control, or it is known or measurable for cooperative strategies. Further details about this topic are given in Sec. 4.3.5 where multi-topological representations are introduced.

Remark 4.1. From the discussion above, it is clear that each subsystem defined by (4.8) can itself be seen as a graph as described in Sec. 4.3.2. A possible algorithmic approach to link the graph associated with a dynamical system and the graph associated with a network of dynamical systems is proposed in [282].

Remark 4.2. In the definition of subsystem (4.8), we assumed that each $\mathcal{S}_i$ is driven only by its local input $u_{\mathcal{S}_i}$. There is, however, the mathematical possibility that dynamics (4.8)

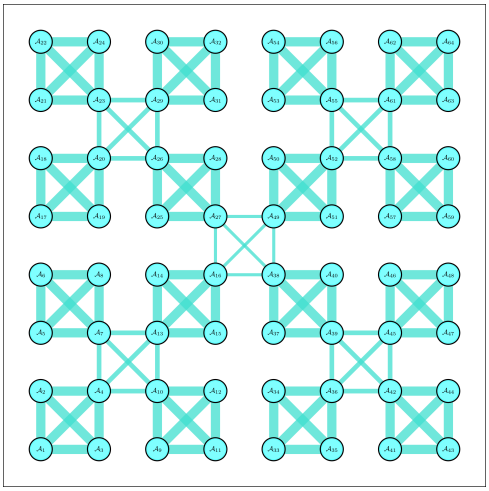


Figure 4.6: Graph representation of a modular network with 64 agents. The width of the edges represents the strength of the interaction among the agents. This network exhibits a repeating modular pattern.

may be driven also by $u_{\mathcal{S}_j}$ with $\mathcal{S}_j \in \mathcal{N}_{\mathcal{S}_i}$. The resulting networks are constituted by input-coupled subsystems. We decided to treat these networks in separate subsections.

Example 4.2. In this example, we propose two different network representations of control agents, one having a modular topology, the other having a random one. According to the discussion above, a control agent will incorporate subsystem dynamics and all the control, communication, coordination, and algorithmic requirements for deploying an NCen-MPC strategy.

A network can be considered modular if it exhibits a high level of modularity, which can be quantified using the modularity metric, but also visually because it will present recurring patterns. An example of such a network with 64 control agents is reported in Fig. 4.6, where the recurring structure of 4 and 16 agents is evident. The topology of this network is defined as follows: from the thickest to the thinnest lines, the bidirectional interactions have a strength of $w_{i,j} = 0.1, 0.01, 0.001$.

The second network has 50 control agents and a randomly generated topology, which is reported in Tab. 4.2, and which shows the presence of directed arcs. The network representation is proposed in Fig. 4.7.

We will use these modular and random networks in Sec. 4.4.4 to show an application of optimization-based and algorithmic partitioning approaches and the evaluation methodology for the quality of a partition.

4.3.4 Bipartite Graph Representations

In a bipartite graph [83], the set of the nodes is divided into two groups $\mathcal{V} = \mathcal{V}_a \cup \mathcal{V}_b$, $\mathcal{V}_a \cap \mathcal{V}_b = \emptyset$, and all the edges start in one group and end in another. This type of graph representation has two main use cases in partitioning for non-centralized control. In the first case, a bipartite graph is used to represent the relations between the variables

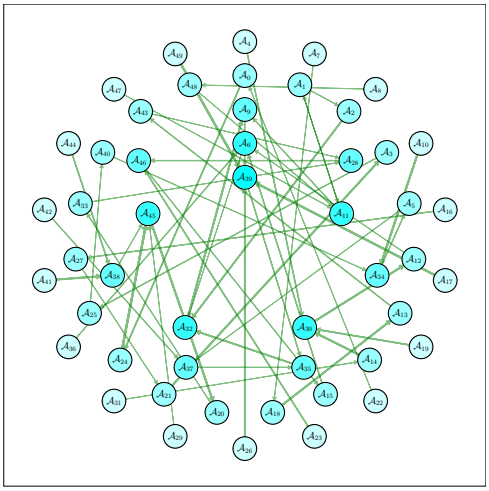


Figure 4.7: Graph representation of a random network with 50 agents. The nodes are sorted according to their degree, which is also reflected in the strength of their color. The randomly generated topology is detailed in Tab. 4.2.

$w_{1,25} = 0.53$	$w_{2,3} = 0.36$	$w_{2,12} = 0.01$	$w_{3,33} = 0.60$	$w_{4,26} = 0.41$	$w_{5,31} = 0.47$	$w_{6,38} = 0.32$	$w_{7,33} = 0.24$	$w_{8,19} = 0.24$	$w_{9,49} = 0.20$
$w_{10,40} = 0.36$	$w_{11,35} = 0.72$	$w_{12,2} = 0.01$	$w_{12,10} = 0.42$	$w_{13,7} = 0.17$	$w_{14,44} = 0.44$	$w_{15,31} = 0.67$	$w_{16,7} = 0.46$	$w_{17,28} = 0.42$	$w_{18,40} = 0.76$
$w_{19,14} = 0.67$	$w_{20,31} = 0.55$	$w_{21,34} = 0.37$	$w_{22,4} = 0.66$	$w_{23,1} = 0.20$	$w_{24,47} = 0.51$	$w_{25,46} = 0.78$	$w_{26,41} = 0.10$	$w_{27,40} = 0.60$	$w_{28,22} = 0.35$
$w_{29,47} = 0.43$	$w_{30,46} = 0.16$	$w_{31,13} = 0.68$	$w_{32,15} = 0.34$	$w_{33,10} = 0.66$	$w_{34,29} = 0.19$	$w_{35,6} = 0.43$	$w_{36,33} = 0.60$	$w_{37,7} = 0.41$	$w_{38,36} = 0.40$
$w_{39,46} = 0.23$	$w_{40,36} = 0.44$	$w_{41,35} = 0.31$	$w_{42,39} = 0.66$	$w_{43,38} = 0.39$	$w_{44,29} = 0.19$	$w_{45,39} = 0.49$	$w_{46,21} = 0.69$	$w_{47,16} = 0.40$	$w_{48,12} = 0.29$
$w_{49,12} = 0.13$	$w_{50,40} = 0.77$								

Table 4.2: Randomly generated topology of the network in Fig. 4.7. The entries $w_{i,j}$ are the i-th row and j-th column of the weighted adjacency matrix W^{adj}.

and the constraints of an optimization problem, e.g. as done in [324]. This approach is used to decompose the optimization problem by minimizing the number of complicating[1] constraints that are removed in the distributed solution of the problem. In the second case, a bipartite graph is used to represent the input-output paths of the network, as done in [326, 339]. In these approaches, the relationships between output and input variables are made explicit. Then, among all possible paths between each pair, the shortest is chosen. Accordingly, partitioning is used to minimize the interactions between input and output dynamics, an approach that is conceptually similar to the quantification of input-output interactions in MIMO systems using an RGA matrix [310].

Example 4.3. In this example, we propose the use of a bipartite graph representation for a network subject to complicating constraints[2]. Consider the following optimization problem representing MPC optimization at a generic time step k for a network of linear systems, each with two states and one input, with no dynamical coupling, but subject to complicating

[1] Complicating constraints are those that introduce an interdependence into subproblems, thus affecting (complicating) the separability of the original problem. In this discussion, complicating constraints are those that involve variables of different subsystems.

[2] The reader can refer to Tab. 4.4 for examples of bipartite representations used to capture input-output interactions in MIMO systems.

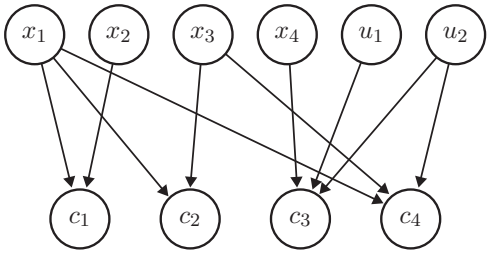


Figure 4.8: The bipartite graph representation of a set of complicating constraints. The graph is constituted by two sets of node, one for the optimization variables, one for the problem constraints. The arcs represents the participation of a variable in a constraint. The graph is bipartite because it is formed by two sets of nodes for which arcs only go from one set to another.

constraints:

$$\begin{aligned} \min_{\tilde{x},\tilde{u}} \quad & J(\tilde{x},\tilde{u}) \\ \text{s.t.} \quad & x(k+1) = Ax(k) + Bu(k) \\ c_1(k): \quad & x_1(k) + x_2(k) \leq 0 \\ c_2(k): \quad & x_1(k) + x_3(k) \leq 0 \\ c_3(k): \quad & u_1(k) + u_2(k) + x_4(k) + 5 \leq 0 \\ c_4(k): \quad & u_2(k) + x_1(k) + x_3(k) \leq 0 \end{aligned} \tag{4.10}$$

where $\tilde{x}, \tilde{u}$ represent state and input sequences over an optimization horizon N, and $k = 0, \dots, N$. The constraints c_i introduce interactions among the subsystems of the network, which can be, e.g. interpreted as a set of specifications on shared resources. This coupling can be captured by the bipartite graph in Fig. 4.8, where the nodes in one set are the variables, the nodes in the other set are the constraints, and the arcs represent the participation of variables in constraints. To partition a network subject to complicating constraints, it is possible to develop algorithmic procedures to maximize the effect of constraint couplings among cooperating agents in the same coalition, and to minimize the coupling between agents in different coalitions. These inter-coalition constraints can be ignored in solving local problems at first, and they can then be accounted for in a later step of the network optimization.

4.3.5 Multi-Topological Network Representations

Consider a network of dynamical systems where the connections are determined by a variable topology whose nature will be specified later in this section. The presence of a link introduces a directed relationship between two subsystems that represents a dynamic coupling as described in Fig. 4.9. Concerning the representation of a network of systems in Sec. 4.3.3, here we consider only macro links connecting subsystems, thus omitting the subscript notation related to the topological representation of the interactions among variables of different subsystems. Moreover, we index each subsystem S_i with the letter i, so that the network of subsystems is made by the set of nodes $\mathcal{V} = \{1, \dots, N_S\}$. These choices simplify the presentation of the following concepts. The existence of a link between the subsystem i and j at a time step k can be represented by the binary variable $\epsilon_{ij}(k)$ such

that:

$$\epsilon_{ij}(k) = \begin{cases} 1 & \text{if } \mathcal{S}_i \text{ is connected to } \mathcal{S}_j \text{ at time step } k \\ 0 & \text{otherwise} \end{cases} \tag{4.11}$$

The collection of these links determines the topology of the network. In the context of control systems, a link representing a dynamical coupling in this network can have three different natures:

- The existence of the link depends on the input-state configuration of the network, i.e. the network has an input-state-dependent topology. This happens when the dynamical coupling is determined by the regions of the input-state configuration of the system, such as in PWA dynamics [36, 129].

- The link can be activated or deactivated as a part of the control strategy of the network, i.e. it is a decision variable.

- The link activation is driven by an external function, either known or unknown.

A possible topology can co-exist for each of the above-mentioned link classes. Consequently, the overall topology of the network will result from the composition of these superposed topologies, i.e. a multi-topological network representation, as in Fig. 4.9.

In the general case, we assume a number of N_ϵ distinct topological levels characterizing the network. We associate a binary variable $\epsilon_{ij}^q(k)$ representing the connection between areas i and j in the topological level q at time step k. According to the nature of the topology with which this variable is associated, it can be an input-state-dependent variable, a decision variable, or a signal. Since all binary variables must be equal to one for a connection to exist, the state of variable $\epsilon_{ij}(k)$ is directly determined by the product:

$$\epsilon_{ij}(k) = \prod_{q=1}^{N_\epsilon} \epsilon_{ij}^q(k) \tag{4.12}$$

Incorporating binary variables $\epsilon_{ij}(k)$ in the network description is straightforward. For this, consider the network of nonlinear systems:

$$x(k+1) = f(x(k), u(k)) \tag{4.13}$$

and assume it admits a decomposition in $N_\mathcal{S}$ subsystems according to the discussion in Sec. 4.3.3. Then, their time-varying topological dynamics is:

$$x_i(k+1) = f_i(x_i(k), u_i(k), \omega_i(k)) \tag{4.14}$$

$$\omega_{ij}(k) = \epsilon_{ji}(k) x_j(k) \qquad \forall j \in \mathcal{N}_i \tag{4.15}$$

where $x_i \in \mathbb{R}^{n_{x_i}}$, $u_i \in \mathbb{R}^{n_{u_i}}$ are the state and input vectors of subsystem i; and the vector ω_i constituted by the elements ω_{ij} incorporates all topologically defined dynamical couplings of subsystem i with the its neighborhood $\mathcal{N}_i$.

4.3.6 Multi-Topological Representations and Hybrid Systems

When applying the concept of multi-topological time-varying representations to networks of linear systems, the result is a hybrid network system [323]. For the sake of simplicity, and without any loss of generality, in what follows, we consider the case of three topological levels of different nature, but the more general case of $N_\epsilon > 3$ topological levels follows similarly. In particular, the network is described as:

$$x_i(k+1) = A_{ii}x_i(k) + B_{ii}u_i(k) + \omega_i(k) \tag{4.16}$$

$$\omega_i(k) = \sum_{j\in\mathcal{N}_i} \epsilon_{ij}(k)A_{ij}x_j(k) \tag{4.17}$$

$$\epsilon_{ij}(k) = \epsilon^1_{ij}(k)\epsilon^2_{ij}(k)\epsilon^3_{ij}(k) \tag{4.18}$$

$$\text{s.t.} \quad \epsilon^1_{ij}(k) = 1 \Leftrightarrow \begin{bmatrix} x_j(k) \\ u_j(k) \end{bmatrix} \in \Omega^{\text{a}}_j \tag{4.19}$$

where $A_{ii} \in \mathbb{R}^{n_{x_i}\times n_{x_i}}$, $B_{ii} \in \mathbb{R}^{n_{x_i}\times n_{u_i}}$, $A_{ij} \in \mathbb{R}^{n_{x_i}\times n_{x_j}}$; ϵ^1 is the logical variable related to the input-state-dependence of a link; Ω^{a} is the convex polyhedron for which the link ϵ^1 is activated; ϵ^2 is a control action; and ϵ^3 an external signal affecting the topology.

This multi-topological network description admits a reformulation into Mixed-Logical Dynamical (MLD) form [36], allowing the direct application of MPC control. To this, assume that the directed dynamical coupling of the j-th system is defined over the polytope $\Omega^{\text{a}}_j = \left\{ \left[x_j^\intercal; u_j^\intercal\right]^\intercal : S^{\text{a}}_j x_j + R^{\text{a}}_j u_j \le T^{\text{a}}_j \right\}$, and we compute the constant $M^*_j \triangleq \max_{\Omega_j} S^{\text{a}}_j x_j + R^{\text{a}}_j u_j - T^{\text{a}}_j$. Then, we introduce auxiliary variables z^1, z^2, z^3 for each edge of the graph, with $i, j \in \mathcal{V}$:

$$A_{ij}\epsilon^1_{ij}(k)x_j(k) = z^1_{ij}(k) \tag{4.20}$$

$$\epsilon^2_{ij}(k)z^1_{ij}(k) = z^2_{ij}(k) \tag{4.21}$$

$$\epsilon^3_{ij}(k)z^2_{ij}(k) = z^3_{ij}(k) \tag{4.22}$$

and the set of constraints that ensure the satisfaction of the logical conditions, and the correct definition of auxiliary variables:

$$S^{\text{a}}_j x_j(k) - T^{\text{a}}_j \le M^*_j(1-\epsilon^1_{ij}(k)) \tag{4.23}$$

$$z^1_{ij}(k) \le M_j\epsilon^1_{ij}(k) \tag{4.24}$$

$$z^1_{ij}(k) \ge m_j\epsilon^1_{ij}(k) \tag{4.25}$$

$$z^1_{ij}(k) \le A_{ij}x_j(k) - m_j(1-\epsilon^1_{ij}(k)) \tag{4.26}$$

$$z^1_{ij}(k) \ge A_{ij}x_j(k) - M_j(1-\epsilon^1_{ij}(k)) \tag{4.27}$$

$$z^\ell_{ij}(k) \le M_j\epsilon^\ell_{ij}(k) \tag{4.28}$$

$$z^\ell_{ij}(k) \ge m_j\epsilon^\ell_{ij}(k) \tag{4.29}$$

$$z^\ell_{ij}(k) \le z^{\ell-1}_{ij}(k) - m_j(1-\epsilon^\ell_{ij}(k)) \tag{4.30}$$

$$z^\ell_{ij}(k) \ge z^{\ell-1}_{ij}(k) - M_j(1-\epsilon^\ell_{ij}(k)) \tag{4.31}$$

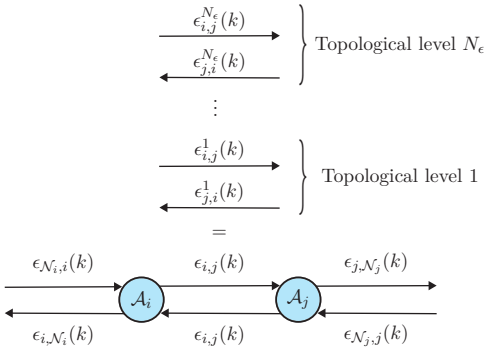


Figure 4.9: General representation of the connections between agents i and j at a time-step k. The topology describing the dynamical coupling among agents at a time step k results from multiple topological levels all acting simultaneously on the network.

for $\ell = 2,3$, allowing the definition of constraints related to the second and third variables; and $M_j = -m_j = \max_{\Omega_j} A_{ij}x_j$ are constants. The resulting system dynamics is then:

$$x_i(k+1) = A_{ii}x_i(k) + B_{ii}u_i(k) + \omega_i(k) \tag{4.32}$$

$$\omega_i(k) = \sum_{j\in\mathcal{N}_i} z^3_{ij}(k) \tag{4.33}$$

The equations (4.23)-(4.33) constitute the MLD form of (4.16)-(4.19).

Remark 4.3. The procedure to obtain multi-topological representations presented in this section is also valid for more complex classes of systems, other than the linear ones. However, for a general nonlinear system, obtaining an MLD representation might not be possible, and more complex approaches to incorporate variable topologies into the dynamics could be required.

Remark 4.4. The existence of an input-state-dependent link between two areas can also be based on the configuration of both areas. In this case, the condition (4.19) must include variables of both areas.

4.4 Partitioning for Predictive Control

This section introduces the main ideas behind the partitioning problem for non-centralized predictive control. To this aim, we first define a specific terminology for the network components, then we introduce the metrics and the evaluation methodology to assess the quality of a partition, and finally, we present a formulation of the partitioning problem for the maximization of the performance of the control architecture. This section aims to provide the reader with a clear perspective of what partitioning optimally means, and the consequent effect on the non-centralized control architecture.

4.4.1 The General Partitioning Problem

Consider a network described by the nonlinear dynamics (4.3), and denoted by $\mathcal{N}$. The state, input, and output vectors are respectively $x \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$, and $y \in \mathbb{R}^{n_y}$. The act of

partitioning consists in finding a subdivision of the vectors x, u, y into a number N_C of subvectors $x_i \in \mathbb{R}^{n_{x_i}}$, $u_i \in \mathbb{R}^{n_{u_i}}$, and $y_i \in \mathbb{R}^{n_{y_i}}$ for $i = 1, \dots, N_C$, and of the respective vector fields into f_i, h_i, which describe the local subsystem dynamics:

$$\mathcal{C}_i : \begin{cases} x_i(k+1) = f_i(x_i(k), u_i(k), w_i(k)) \\ y_i(k) = h_i(x_i(k), w_i(k)) \end{cases} \tag{4.34}$$

where $w_i(k)$ represents the coupling of subsystem i with its neighboring subsystems $j \in \mathcal{N}_i$. The partition of the network is thus constituted by the set of subsystem dynamics:

$$\mathcal{P} = \{\mathcal{C}_1, \dots, \mathcal{C}_{N_C}\} \tag{4.35}$$

Depending on the context, we call these groups $\mathcal{C}_j$ sets or collections of subsystems, clusters, or coalitions. This general formulation of the partitioning problem is generally too broad to be considered directly in defining a partitioning strategy. Instead, this setting has several simplified reformulations, most notably the ones reported next.

Complete non-overlapping partitioning In (4.34), there is no limitation on the structure of local vectors and dynamics. However, the prevalent setting in partitioning for non-centralized predictive control is to assume that the partitioning is complete and non-overlapping, and it covers the entirety of the original dynamics. Using set notation, a complete non-overlapping partitioning $\mathcal{P}$ is such that:

$$\bigcup_{i=1}^{N_C} \mathcal{C}_i = \mathcal{P} \quad \text{and} \quad \bigcap_{i=1}^{N_C} \mathcal{C}_i = \emptyset \quad \text{with} \quad \mathcal{C}_i \neq \emptyset \; \forall i \tag{4.36}$$

A complete non-overlapping partitioning allows the straightforward definition of local controllers and coordination protocols, making it the preferred choice in non-centralized control.

Overlapping partitioning In the literature of partitioning for NMPC, most attention has been dedicated to non-overlapping approaches, while overlapping techniques are [140]. In an overlapping partition, some nodes are assigned to multiple agents simultaneously, forming a common subnetwork over which agents must reach agreement. Such a situation can occur when two distinct subnetworks are influenced by a set of topologically close actuators, in which case the boundaries of their regions of influence may not admit a clean separation. Coordination over such structures requires dedicated mechanisms for handling shared objectives and constraints, as done in [141, 242], which can be the starting point for further development of the considered NCen-MPC approach in this direction.

Coupling through state dynamics The coupling term $w_i(k)$ in (4.34) can, in general, comprehend both state and input interactions with neighbors, i.e.

$$w_i(k) = [(x_{S_j}(k))_{S_j \in \mathcal{N}_{S_i}}; (u_{S_j}(k))_{S_j \in \mathcal{N}_{S_i}}]. \tag{4.37}$$

However, in most settings, only state couplings are considered, yielding the vector field $f_i(x_i(k), (x_{S_j}(k))_{S_j \in \mathcal{N}_{S_i}}, u_i(k))$. This approach is the most intuitive and represents most

real-world scenarios in which a local controller would be designed to steer local dynamics through the input channel u_i without directly interfering with the neighbor dynamics x_j. Moreover, it is often assumed that the output function depends only on the local state, thus taking the form $h_i(x_i(k))$. However, even if this is the most used setting in the partitioning literature, we acknowledge the presence and relevance of studies for input-coupled subsystems. We decided to treat these approaches separately in Sec, 4.6.6, 4.7.7, 4.7.8, and 4.9.6. In fact, studies for input-coupled subsystems generally consider a small number of subsystems, or neglect the existence of delays in the input coupling that would introduce dynamics in the interaction among subsystems.

Fundamental subsystems A common assumption in partitioning for non-centralized predictive control is that the network $\mathcal{N}$ in (4.3) admits a natural decomposition into a number N_S of atomic or fundamental subsystems that cannot be further divided for the definition of local controllers. Moreover, fundamental subsystems are coupled exclusively by state dynamics interactions, as formalized in [282]. Therefore, the network is given as a collection of subsystems $\mathcal{N} = \{S_1, \dots, S_{N_S}\}$. In this network setting, partitioning consists in grouping the subsystems S_i into a number $N_C \leq N_S$ of bigger units C_j, i.e. using the notation (4.35) in defining the partition $\mathcal{P} = \{C_1, \dots, C_{N_C}\}$. Two extreme partitions are possible, one where each group is an individual subsystem, i.e. $\mathcal{P} \equiv \mathcal{N}$, $N_C = N_S$, and one that comprises the entire network, i.e. $\mathcal{P} = \{C_1\}$, $N_C = 1$.

Example 4.4. We continue Ex. 4.1 by showing a possible selection of the fundamental subsystems for that network. To this aim, we apply the algorithm for selecting fundamental subsystems defined in [282], which iterates over network nodes allocating them to subsystems according to coupling strengths. The resulting definition of the subsystems is given in Fig. 4.10. This is one definition of the subsystems, and others are possible.

Top-down and bottom-up approaches From the discussion above, it is clear that the problem of partitioning a network can be approached from two different directions: a top-down and a bottom-up approach. In the top-down approach, a network $\mathcal{N}$ is considered a monolithic system (generally without any natural decomposition) that must be divided into smaller units. This approach is generally considered when complex nonlinear plants have to be decomposed for non-centralized control. In the bottom-up approach, instead, the problem is solved by aggregating fundamental subsystems that are given a priori, i.e. the network is assumed as $\mathcal{N} = \{S_1, \dots, S_{N_S}\}$. Both top-down and bottom-up strategies are generally valid approaches, and the preferred direction is usually dictated by the application considered.

Remark 4.5. When referring to a group of subsystems, we can also call it a set, cluster, or coalition. All these terms are necessarily used interchangeably throughout the survey because they all represent the same concept of a group of objects. There are subtle distinctions between the terms that will be remarked on in the specific sections. In general, the term cluster is used in the machine learning literature to indicate a group of objects that are strongly connected [346], while coalition is a term used in cooperative game theory to denote a group of players [14].

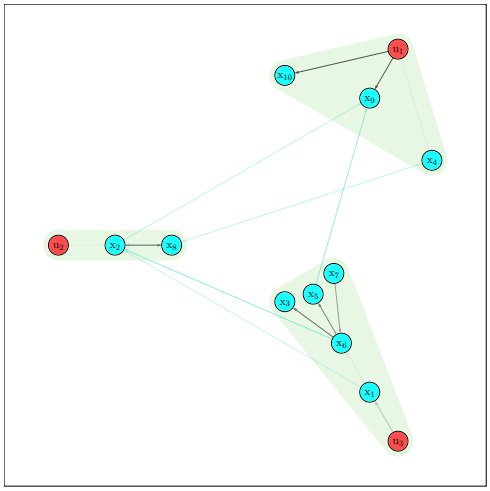


Figure 4.10: A possible selection of the subsystems for the network in Ex. 4.1. The green areas indicate the subsystems and comprehend several input and state variables. The arrows that go from one subsystem to the other can be interpreted as the dynamical coupling among control agents.

4.4.2 Metrics and Evaluation Methodology

The fundamental question that each partitioning strategy in this survey tries to answer is: What is the best partition for non-centralized predictive control? This question only admits a posterior quantitative answer independently from the control strategy considered. A formal motivation for this fact is given in Sec. 4.4.3; instead, in this section, we focus on the metrics that allow us to assess the quality of a partition, and on the methodologies to do it. First of all, the best partition for a selected evaluation criterion must be defined for a specific non-centralized predictive control, i.e. Dec-MPC, DMPC, HMPC, or Coal-MPC, and w.r.t. CMPC. Throughout the section, we assume that the partition associated with CMPC is denoted by $\mathcal{P}^{\text{CMPC}}$ (this is the entire network), the one under evaluation by $\mathcal{P}^{\text{NMPC}}$, i.e. the partitioning for the application of a desired NCen-MPC strategy, and one generic partition by $\mathcal{P}^{\text{gen}}$. In this section, we first present the main metrics used to assess the quality of a partition, and then we briefly discuss the evaluation methodologies.

Metrics In the literature, four main key performance indicators are used to assess the quality of a partition: 1) the cumulative stage cost J^{stage}; 2) the computation time J^{time}; 3) the computational cost $J^{\text{comp.}}$; and 4) the communication cost $J^{\text{comm.}}$. To validate the partition, it is necessary to simulate the system using both CMPC and NCen-MPC using the desired partitioning strategy. Then, the key performance indicators are computed as follows.

Cumulative stage cost Assume that the stage cost for CMPC at the time step k is defined by the cost function $J(x(k), u(k-1))$. Moreover, assume a simulation horizon of N_{sim} time steps. At time step k, the optimal control problem for CMPC is solved over a horizon N, yielding a solution control sequence $\tilde{u}^*_{\text{CMPC}}(k)$, of which the first element $u^*_{\text{CMPC}}(k)$ is

applied to the system, providing the next step value for the state $x_{\text{CMPC}}(k)$. Consequently, the cumulative stage cost for CMPC is

$$J^{\text{stage}}(\mathcal{P}^{\text{CMPC}}) = \sum_{k=1}^{N_{\text{sim}}} J(x_{\text{CMPC}}(k), u^*_{\text{CMPC}}(k-1)) \tag{4.38}$$

The cumulative stage cost for the non-centralized strategy and a selected partitioning $\mathcal{P}^{\text{NMPC}}$ is obtained similarly. However, in this case, a number $N_C = |\mathcal{P}^{\text{NMPC}}|$ of local problems is solved in parallel, providing local solutions $u^*_{\text{NMPC},i}(k)$ for $i = 1, \ldots, N_C$. Then, a global vector $u^*_{\text{NMPC}}(k)$ is obtained by grouping local solutions, and is applied to the plant to compute the global state transition $x_{\text{NMPC}}(k)$. This procedure allows the computation of the cumulative stage cost for NMPC, i.e. $J^{\text{stage}}(\mathcal{P}^{\text{NMPC}})$, as done in (4.38) but using the non-centralized vectors. In general, for cost minimization it holds that $J^{\text{stage}}(\mathcal{P}^{\text{CMPC}}) \leq J^{\text{stage}}(\mathcal{P}^{\text{NMPC}})$. There are exceptions if the dynamics is nonlinear and the solution is obtained for a linearized version around an operating point, or if the network is subject to external uncertain signals. However, the centralized solution of the optimization problem is the reference to assess the optimality of the selected partition (and of the partitioning methodology) for a given non-centralized strategy.

An approach to compare the cumulative stage cost of different architectures consists in normalizing these results such that, for a given partition $\mathcal{P}^{\text{gen}}$ under evaluation, the normalized cumulative stage cost is given by:

$$J^{\text{stage}}_{\text{norm.}}(\mathcal{P}^{\text{gen}}) = \frac{J^{\text{stage}}(\mathcal{P}^{\text{gen}})}{J^{\text{stage}}(\mathcal{P}^{\text{CMPC}})} \tag{4.39}$$

It holds that $J^{\text{stage}}_{\text{norm.}}(\mathcal{P}^{\text{CMPC}}) = 1$, and in general $J^{\text{stage}}_{\text{norm.}}(\mathcal{P}^{\text{gen}}) \geq 1$, so various partitions can be evaluated easily according to a metric that is valid across all possible strategies and applications.

Computation time This metric is straightforward to obtain. It is sufficient to measure the execution time in seconds necessary to execute the simulation over a horizon N_{sim}. For CMPC, one CPU core is used to execute this task[3], and the time in seconds to perform the simulation constitutes the computation time cost $J^{\text{time}}(\mathcal{P}^{\text{CMPC}})$. For NMPC, local optimization problems are solved in parallel at each time step, which requires N_C CPUs[4] The time required for this parallel execution constitutes the computation time cost for NMPC, i.e. $J^{\text{time}}(\mathcal{P}^{\text{NMPC}})$. For any well-designed non-centralized strategy and good choice of partition, it holds that $J^{\text{time}}(\mathcal{P}^{\text{NMPC}}) < J^{\text{time}}(\mathcal{P}^{\text{CMPC}})$. The gain in computation time is often one of the main reasons for deploying a non-centralized strategy. In fact, centralized

[3] In some cases, parallel computing can also be used for CMPC. An example is when the network is constituted by hybrid systems. In this case, the MPC problem requires mixed-integer optimization, for which parallel execution algorithms are available. In such cases, instead of using one CPU for CMPC, it is possible to use any available number, given that each set of subsystems in the non-centralized strategy has such CPUs available at each time step.

[4] The analysis of the computation time can be easily extended to the case in which the number of CPUs is time-varying, i.e. for $N_C(k)$. This case occurs either when there is a time-varying partitioning $\mathcal{P}(k)$, or when the computational resources can change over time. Such extension also applies to other performance indicators.

computations may be prohibitive in several settings. For a partition $\mathcal{P}^{\text{gen}}$ under evaluation, the normalized version of the computation time is:

$$J^{\text{time}}_{\text{norm.}}(\mathcal{P}^{\text{gen}}) = \frac{J^{\text{time}}(\mathcal{P}^{\text{gen}})}{J^{\text{time}}(\mathcal{P}^{\text{CMPC}})} \tag{4.40}$$

where $J^{\text{time}}_{\text{norm.}}(\mathcal{P}^{\text{CMPC}}) = 1$, and for a well-designed non-centralized strategy and partition $J^{\text{time}}_{\text{norm.}}(\mathcal{P}^{\text{gen}}) < 1$.

Computation cost We discussed how, to assess the computation time in non-centralized control, it is necessary to deploy the strategy in parallel, or alternatively, perform a simulation replicating such a situation. The computation cost is a metric that quantifies the cost associated with the usage of CPUs for these parallel operations, and was introduced in [278, 282] for the evaluation of different partitions of the same network in DMPC. The best way to do so is to look at the CPU usage time, which translates immediately into power and monetary requirements once a specific technology is selected. Consequently, the unit measure of the computation cost is [core · seconds], i.e. how much CPU time in parallel is required to perform the distributed computation. For a generic predictive control strategy, being it centralized or non-centralized, the computation cost is thus assessed by computing for the simulation horizon N_{sim} the sum over the number of CPUs of the active CPUs usage time for that time step, which for a CPU i we denote by $\tau_i(k)$. If we assume that, in the non-centralized control strategy considered, one CPU is available for each agent in the partition $\mathcal{P}^{\text{NMPC}}$, then it holds that $N_{\text{CPU}} = N_{\mathcal{C}}$, and the computation cost can be written as:

$$J^{\text{comp.}}(\mathcal{P}^{\text{NMPC}}) = \sum_{k=1}^{N_{\text{sim}}} \sum_{i=1}^{N_{\mathcal{C}}} \tau_i(k) \tag{4.41}$$

It is possible to simplify this expression further if we assume that at each time step N_{sim} all local controllers will wait and idle for the slowest controller to obtain its result without performing any operation. Then, the computation cost can be written as $J^{\text{comp.}}(\mathcal{P}^{\text{NMPC}}) = \sum_{k=1}^{N_{\text{sim}}} N_{\mathcal{C}}\, \tau^{\text{slowest}}(k)$. For both definitions of $J^{\text{comp.}}$, the normalized version of the metric for a generic partition $\mathcal{P}^{\text{gen}}$ is given by:

$$J^{\text{comp.}}_{\text{norm.}}(\mathcal{P}^{\text{gen}}) = \frac{J^{\text{comp.}}(\mathcal{P}^{\text{gen}})}{J^{\text{comp.}}(\mathcal{P}^{\text{CMPC}})} \tag{4.42}$$

where $J^{\text{comp.}}_{\text{norm.}}(\mathcal{P}^{\text{CMPC}}) = 1$. In general $J^{\text{comp.}}_{\text{norm.}}(\mathcal{P}^{\text{gen}}) > 1$, but very efficient strategies can also achieve $J^{\text{comp.}}_{\text{norm.}}(\mathcal{P}^{\text{gen}}) < 1$.

Remark 4.6. In literature, to the authors' best knowledge, the only a priori assessment of the computational cost associated with a specific non-centralized predictive control strategy has been performed in [15]. However, in that work, the determination is rather qualitative since it is performed through a Big-O analysis of the computational complexity of the algorithm for non-centralized predictive control. In practice, such an approach cannot always establish which is better among algorithms with the same Big-O complexity, as in iterative schemes.

Communication cost The communication cost assesses the impact of information transmission in different non-centralized control architectures. In its original formulation, see e.g. [95, 97, 205, 213] among others, the communication cost is a function of the information topology defining how coalitions in a network share information to achieve a coordinated control action. Therefore, to the non-centralized control architecture an information graph $\mathcal{G}_{\text{info}}^{\text{NMPC}} = (\mathcal{V}_{\text{info}}^{\text{NMPC}}, \mathcal{E}_{\text{info}}^{\text{NMPC}})$ is assigned, where the set of the nodes is constituted by the coalitions in the network, and the set of the edges by the active communication links. Then, to each link $\epsilon_{ij} \in \mathcal{E}_{\text{info}}^{\text{NMPC}}$ a cost is assigned, i.e. $\nu(\epsilon_{ij})$, and the communication cost is therefore computed as:

$$J^{\text{comm}}(\mathcal{P}^{\text{NMPC}}) = \sum_{\epsilon_{ij} \in \mathcal{E}_{\text{info}}^{\text{NMPC}}} \nu(\epsilon_{ij}) \tag{4.43}$$

This formulation of the communication cost has been used consistently in deriving coalitional control strategies, leading to partitions of the network minimizing the information sharing. The communication cost of CMPC is obtained by considering the cost associated with each possible active link in the network. The value of the cost of communication can be quantified using distance-based criteria, or the operational costs of the lines. Additionally, we stress that this approach in defining the communication cost can be used to obtain a partition, i.e. it is available a priori since it is a pure topological metric, whereas the other costs introduced before are only available a posteriori after the simulation.

While this formulation of the communication cost is direct and straightforward, it can be insufficient to establish the cost associated with iterative non-centralized control strategies. In fact, if the coordination protocol relies on the iterative sharing of information among agents to achieve an agreement about the control action to deploy, then a static topological metric can only be used to quantify the maximum amount of information shared once the maximum number of iterations of the coordination protocol is given. Posterior measurement of the true amount of information shared is, therefore, a more precise way to assess communication cost in this case. For example, assume that for an NMPC iterative strategy with information topology $\mathcal{G}_{\text{info}}^{\text{NMPC}} = (\mathcal{V}_{\text{info}}^{\text{NMPC}}, \mathcal{E}_{\text{info}}^{\text{NMPC}})$, at each time step k several iterations $N_{\text{iter}}(k)$, and at every iteration a sequence of state-input predictions of length N_{seq} is shared among the controllers. Then, for a simulation horizon N_{sim}, and assuming that each state and input variables vectors have an information transmission cost $\nu(x_i)$, $\nu(u_i)$, $i \in \mathcal{V}_{\text{info}}^{\text{NMPC}}$, then the communication cost can be defined as:

$$J^{\text{comm}}(\mathcal{P}^{\text{NMPC}}) = \sum_{k=1}^{N_{\text{sim}}} N_{\text{iter}}(k) \sum_{i \in \mathcal{V}_{\text{info}}^{\text{NMPC}}} \sum_{j \in \mathcal{N}_i} N_{\text{seq}}(\nu(x_i) + \nu(u_i)) \tag{4.44}$$

where the cost ν associated with the information transmission can then be directly translated into network operation or economic requirements. The CMPC strategy does not need any iteration; only variables at the current time step are shared. Therefore, its communication cost is:

$$J^{\text{comm}}(\mathcal{P}^{\text{CMPC}}) = N_{\text{sim}} \sum_{i \in \mathcal{V}_{\text{info}}^{\text{CMPC}}} \nu(x_i) + \nu(u_i) \tag{4.45}$$

For both formulations of the communication cost, a normalization assessment is possible. Therefore, for a given partition $\mathcal{P}^{\text{gen}}$ associated with a non-centralized MPC strategy, its

normalized version is:

$$J^{\text{comm}}_{\text{norm}}(\mathcal{P}^{\text{gen}}) = \frac{J^{\text{comm}}(\mathcal{P}^{\text{gen}})}{J^{\text{comm}}(\mathcal{P}^{\text{CMPC}})} \tag{4.46}$$

where $J^{\text{comm}}_{\text{norm}}(\mathcal{P}^{\text{CMPC}}) = 1$, for decentralized MPC or non-iterative strategies usually holds $J^{\text{comm}}_{\text{norm}}(\mathcal{P}^{\text{gen}}) \leq 1$, while for iterative strategies $J^{\text{comm}}_{\text{norm}}(\mathcal{P}^{\text{gen}}) \geq 1$.

Evaluation methodology From the above discussion about metrics, it is clear that assessing the quality of a partition is mainly a task performed after a simulation or experiment is completed. This fundamental fact, i.e. the impossibility of establishing the best partitioning prior to the deployment of the strategy, is one of the main limiting factors in developing partitioning strategies for non-centralized predictive control. In fact, once a partition is selected, computationally intensive simulations involving often large (in number or size) optimization problems have to be performed. Once the metrics of interest are selected for a specific problem and control strategy, the only effective way to determine the best partition is by complete enumeration, see e.g. [17]. However, enumerating and testing all possible partitioning quickly becomes intractable once the number of subsystems grows by more than a few units, due to a combinatorial explosion in the number of possible partitions. Therefore, most partitioning strategies have either developed paradigms for the topological a priori evaluation of partitions, or approached the problem by maximizing the immediate gain of a performance criterion by iterative exchange of agents. A definitive statement about what is the best approach cannot be formulated yet with the current literature, which leaves open many directions for future research. In practice, there might not even be a single partition minimizing simultaneously all four indicators J^{stage}, J^{time}, $J^{\text{comp.}}$, and $J^{\text{comm.}}$. Therefore, the desired partition should be selected according to control requirements among the most promising ones.

4.4.3 Optimal Partition for Performance Maximization

An agent $\mathcal{A}_i$ in the network is a structure with autonomy constituted by a group of subsystems $\mathcal{C}_i$, a local controller $\mathcal{K}_i$, and further devices allowing communication with other agents, or other digital features, such as the execution of algorithmic procedures.

The problem of partitioning consists of finding an allocation of the agents of the network into groups such that a set of specifications is satisfied. Different criteria, including geographical distribution, communication and coordination effort, operational constraints, security and privacy guarantees, and design choices, can guide the selection of these groups. Often, these criteria are application-dependent and, in almost all cases, are related to the control strategy to deploy. Consequently, there is no common rationale underlying all the different partitioning approaches. However, when the partitioning problem is considered in the context of non-centralized predictive control, it assumes a more precise connotation, and an optimal version can be formulated.

Assume to have a network with $N_{\mathcal{A}}$ agents, i.e. a collection $\mathcal{N} = \{\mathcal{A}_1, \dots, \mathcal{A}_{N_{\mathcal{A}}}\}$. A set $\mathcal{C}_i$ of $N_{\mathcal{C}_i}$ agents is defined as $\mathcal{C}_i = \{\mathcal{A}_{i,1}, \dots, \mathcal{A}_{i,N_{\mathcal{C}_i}}\}$. We introduce a matrix of binary variables $\delta \in \mathbb{M}_{N_{\mathcal{A}}}(0,1)$[5] s.t. $\delta_{ij} = 1 \Leftrightarrow \mathcal{A}_i \in \mathcal{C}_j$. In general, we can assume δ to be time-dependent, i.e. $\delta(k)$, but time-dependence is omitted in the following for simplicity, and only used

[5]The class of square binary matrices of dimension $N_{\mathcal{A}}$.

when essentially required. For a given choice δ, we denote a partition of network $\mathcal{N}$ into $N_{C(\delta)}$ sets of agents by $\mathcal{P}(\delta) = \{\mathcal{C}_1, \ldots, \mathcal{C}_{N_{C(\delta)}}\}$. Now we consider the control performance of the network that is measured through a cost function $J(x, u, \delta)$, where x is the state of the network, u is the applied control action, and δ is the selected partitioning, a set of binary decision variables. Once the non-centralized predictive control strategy is selected, the cost J is minimized iteratively at each time step over a selected horizon N. For this, we use the vector notation $\tilde{x}_k = [x^\intercal(1|k), \ldots, x^\intercal(N|k)]^\intercal$, $\tilde{u}_k = [u^\intercal(0|k), \ldots, u^\intercal(N-1|k)]^\intercal$, $\tilde{\delta}_k = [\delta^\intercal(0|k), \ldots, \delta^\intercal(N-1|k)]^\intercal$ to define state and input sequences over the horizon N. The global control problem is then defined as:

$$\begin{aligned} \min_{\tilde{x}_k, \tilde{u}_k, \tilde{\delta}_k} \; & J(\tilde{x}_k, \tilde{u}_k, \tilde{\delta}_k) = \sum_{i=1}^{N-1} J_s(x(i|k), u(i-1|k), \delta(i-1|k)) + J_f(x(N|k), u(N-1|k), \delta(N-1|k)) \\ \text{s.t.} \quad & x(k+1) = f(x(k), u(k)) \\ & x(0|k) = x(k) \\ & g(\tilde{x}_k, \tilde{u}_k, \tilde{\delta}_k) \leq 0 \end{aligned} \tag{4.47}$$

where J_s is the stage cost, J_f the terminal cost, and g a set of inequality constraints. This formulation of the optimal partitioning problem assumes that it is possible to simultaneously select the variables in matrix δ, and perform the steps to deploy the non-centralized control strategy. Conceptually, this contemporaneous optimization is not always possible for non-centralized control, especially if communication and coordination protocols are involved, i.e. in all cases except for purely decentralized MPC. This limitation can be overcome with a nested reformulation of (4.47). Specifically, the outer level is an integer optimization problem for the selection of δ, and the inner level is associated with the solution of the non-centralized control problem:

$$\begin{aligned} \min_{\tilde{\delta}_k} \quad & J^*(\tilde{\delta}_k) \\ \text{s.t.} \quad & g_{\text{out}}(\tilde{\delta}_k) \leq 0 \end{aligned} \tag{4.48}$$

$$\begin{aligned} J^*(\tilde{\delta}_k) = \min_{\tilde{x}_k, \tilde{u}_k} \; & \sum_{i=1}^{N-1} J_s(x(i|k), u(i-1|k))|_{\tilde{\delta}_k} + J_f(x(N|k), u(N-1|k))|_{\tilde{\delta}_k} \\ \text{s.t.} \quad & x(k+1) = f(x(k), u(k)) \\ & x(0|k) = x(k) \\ & g_{\text{in}}(\tilde{x}_k, \tilde{u}_k) \leq 0 \end{aligned}$$

where, at the inner level, algorithmic procedures that ensure coordination among the agents might be present. In this formulation we assumed that the inequalities g in (4.47) can be split in an outer g_{out} and inner g_{in} sets depending on variables $\tilde{\delta}_k$, and $(\tilde{x}_k, \tilde{u}_k)$ respectively. This assumption usually holds since once variables $\tilde{\delta}_k$ are fixed, they do not affect further the non-centralized control strategy. Moreover, the set of constraints g_{out} can be used to impose desired properties on the partitioning. One common choice is to assume that sets $\mathcal{C}_i$ are non-overlapping, which can be codified with the constraints

$$\forall i \quad \sum_{j=1}^{N_A} \delta_{ij} = 1 \tag{4.49}$$

The complexity of the nested optimization problem (4.48) is NP-hard due to the outer mixed-integer layer. Moreover, from an implementation perspective, the time requirements to find the optimal partitioning and the optimal control action with this approach can quickly become prohibitive with a growing number of agents because, for each choice of $\tilde{\delta}$, the inner non-centralized predictive control strategy might be required to perform many iterative steps involving optimization.

Remark 4.7. Optimal partitioning is intended for performance, but partitioning can be done according to other criteria, for which the optimal solution can be different. See Sec. 4.4.2 for a list of common metrics that can be used.

4.4.4 Solution Methodologies

Partitioning approaches in current literature usually do not consider the level of complexity of the problem formulation (4.48). Instead, simplified formulations, often application-oriented, are considered. These solution approaches can be broadly categorized into the following four methodologies:

- *Static partitioning*: this is the case in which the selection of δ is made prior to the deployment of the non-centralized strategy, and the partitioning $\mathcal{P}(\delta)$ is fixed at all instants. Most approaches follow this logic due to its simplicity and the fact that the partitioning can be computed offline. The disadvantage is that changes in the network's topology cannot be compensated for with this method, making it a viable option only for stationary networks.

- *Event driven partitioning*: it is the first extension of static partitioning. When a topological change is detected, a new network partitioning is deployed. This strategy is reactive since network alterations can be detected, but no assumptions or predictions about their future behavior are made. Suppose the number of possible different topologies of the network is known a priori. In that case, all the associated partitionings of the network can be computed offline and only deployed when necessary. In other cases, the new partitioning is computed as soon as the topological change is detected, implying that the partitioning method is fast enough to be executed between two distinct MPC computations. For large networks, this is not usually suitable through optimization-based approaches. Therefore, algorithmic solutions can be considered to perform local adjustments to partitioning in the neighborhood of the topological change. Also, tabular methods can be implemented to track the topology-partitioning couples, thus avoiding re-computations in known situations.

- *Fixed partitioning over the prediction horizon*: in this case, it is assumed that the topological changes that will occur over the network during the prediction horizon are known at the current time step, either accurately or with some uncertainty. Consequently, before the start of the optimization process in the MPC, a fixed sequence $\tilde{\delta}$ can be established, and the non-centralized MPC is deployed knowing all the changes in topology and partitioning during the prediction horizon. A limited number of techniques of this type are currently available in the literature.

- *Time-varying partitioning*: this is the most complex case, where a potentially different network partitioning is allowed for each time step. In this way, all possible input-

state-dependent topological changes that will occur in the network according to the available prediction model can be compensated, and uncertain topological changes might be accounted for using robustness arguments. This approach is also the only one that might guarantee the stability of the resulting non-centralized predictive control architecture under predictable topological changes. In current literature, no work is present in this category, and future research might consider addressing this problem.

Formally speaking, the last two approaches assume that a *predictive partitioning* of the network can be implemented for the NCen-MPC strategy developed. Such partitioning can assume the network topology to be static, or to change according to known rules or dynamical models. In the first case, the predictive partitioning is performed purely to improve the NCen-MPC approach. For the other two cases, there is no known approach in the literature, making predictive partitioning using models of the network topology dynamics an open problem.

We conclude this section by showing two examples of how to obtain the partition of two networks with different structures in Ex. 4.5, and of how to perform the posterior assessment of the performance of an NCen-MPC strategy applied to different partitionings of the same network in Ex. 4.6.

Example 4.5. We continue the examples started in Ex. 4.2 by showing possible partitions of the modular and random networks.

We start by considering the modular network with 64 agents, and we apply the optimization-based partitioning technique developed in [282]. This methodology returns different optimal partitions according to a selected value for the granularity parameter, which balances coupling strengths with the size of the resulting sets of agents. Applying this partitioning methodology to the modular network returns four different partitions: the one constituted by individual agents, two partitions aggregating groups of four agents according to their modules, and the grand coalition accounting for all the agents. The examples of the two intermediate partitions are shown in Fig. 4.11.

We also show the application of partitioning procedures defined in [282] to the random network with 50 agents. The use of an algorithmic approach here is advised because the previously deployed optimization-based strategy has a slow convergence rate, which is a consequence of the NP-hard nature of the problem. The algorithmic approach is instead known to have a computational complexity of at most $O(n^4)$, where n is the number of nodes of the graph, after which improvements in the partitioning quality are usually marginal, and it can be potentially optimized and parallelized as commonly done in clustering procedures [346]. However, which method provides the best partitions cannot be established a priori, and the results should be validated through control experiments, which we show in the next example. Two different network partitions, one obtained through the optimization-based approach, the other through the algorithmic approach, are shown in Fig. 4.12.

Example 4.6. For this example, we consider again the random network with 50 agents. We

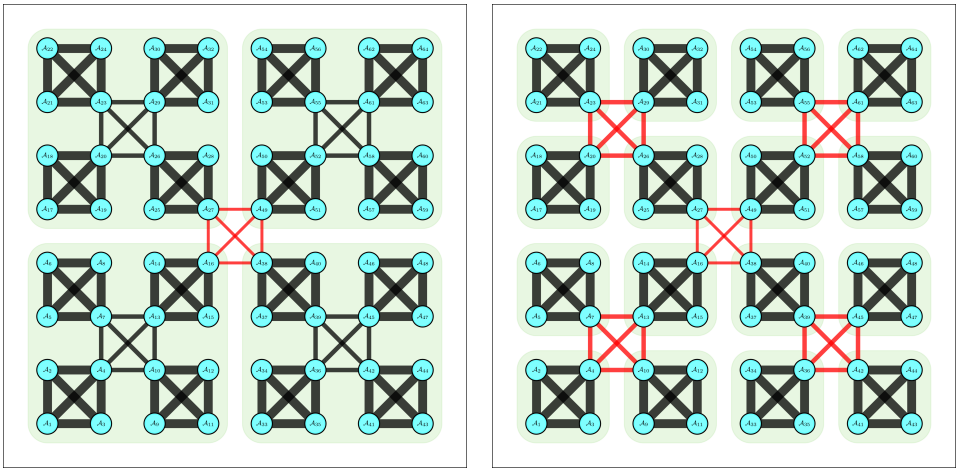


Figure 4.11: Two different partitions for the modular network. The green areas represent the control agents, the black links are the interactions inside the same control agent, while the links in red represent the interactions among the control agents.

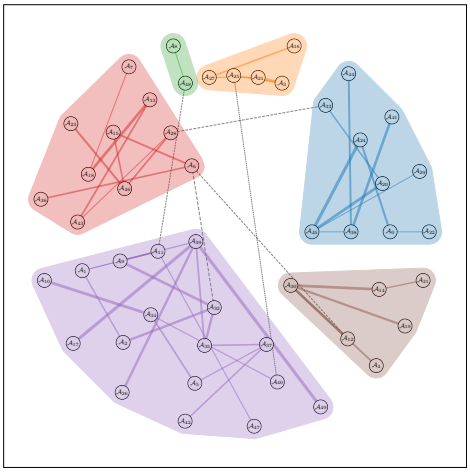
Table 4.3: Comparison of DMPC-ADMM performance applied to a random network of hybrid systems for different partitioning strategies

Partition	Cores	Cost fun. value	Opt. loss %	Comp. time [s]	Comp. time ratio	Core seconds [s]	Core seconds ratio
$\mathcal{P}^{\text{CMPC}}$	1	6899.9750	0.00	2628.04	26.4870	2628.04	1.3736
$\mathcal{P}^{\text{ADMM}}$	50	7749.2102	12.31	99.22	1.0000	4960.99	2.5930
$\mathcal{P}^{\text{Opt}}$	6	6916.7114	0.24	436.29	4.3972	2617.71	1.3682
$\mathcal{P}_1^{\text{Alg}}$	11	6982.5798	1.20	173.93	1.7530	1913.28	1.0000
$\mathcal{P}_2^{\text{Alg}}$	9	6975.5149	1.09	353.81	3.5660	3184.27	1.6643
$\mathcal{P}_3^{\text{Alg}}$	5	6911.0475	0.16	2818.69	28.4085	14093.47	7.3661

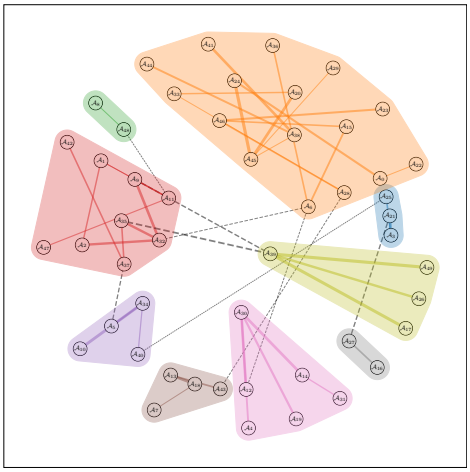
further assume that each agent $\mathcal{A}_i$ controls a subsystem with hybrid dynamics, defined as:

$$x_i(k+1) = 0.5x_i(k) + u_i(k) + \sum_{j \in \mathcal{N}_i} w_{i,j} x_j(k) \quad \text{if} \quad x_i(k) \geq 0$$
$$x_i(k+1) = -0.5x_i(k) + u_i(k) + \sum_{j \in \mathcal{N}_i} w_{i,j} x_j(k) \quad \text{if} \quad x_i(k) < 0$$

Thus, subsystem $\mathcal{S}_i$ is coupled through state interactions to its neighboring subsystems $\mathcal{S}_j$ with $j \in \mathcal{N}_i$, and is subject to local constraints $u_i \in [-0.5; 0.5]$, $x_j \in [-0.9; 0.9]$ $\forall i, j$, but not to coupling constraints or objectives. The dynamical coupling occurs through the weights $w_{i,j}$, which define the topology of the network and are reported in Tab. 4.2. We want to deploy a DMPC strategy based on the alternating-direction method of multipliers (ADMM). We use hybrid dynamics because these are nonlinear systems, for which the effect of partitioning on pure network control performance is evident. Additional technical details about the case study are in [282]. Here we focus on the results of control simulations to show how the metrics and the evaluation methodology developed in Sec. 4.4.2 can be



Optimization-based partition.



Algorithmic partition.

Figure 4.12: Two possible different partitions of the random network for selected levels of the granularity parameter obtained with different strategies. These partitions are obtained with the scope of minimizing the strength of the interaction among control agents in different sets, while maximizing the interaction among control agents in the same set. While these two partitions appear to be very similar, the effects they have on network control can be quite different, as shown in Ex. 4.6.

used to assess the quality of the partitions, and to select the most appropriate partitioning strategy for the considered application. To this end, we compare CMPC and the respective coalition denoted by $\mathcal{P}^{\text{CMPC}}$, which is made by all agents; conventional DMPC, where each agent acts independently, denoted by $\mathcal{P}^{\text{ADMM}}$; one of the partitions obtained using the optimization-based method $\mathcal{P}^{\text{Opt}}$ and reported in Fig. 4.12; and three partitions $\mathcal{P}_i^{\text{Alg}}$ obtained with the algorithmic partitioning procedure. We propose only one optimization-based partition because they produce control simulations that are generally similar w.r.t. the algorithmic approaches that have more interesting aspects to show. The results of the control simulations are reported in Tab. 4.3. The CMPC approach has the best control performance, and is used as a reference in this category, while conventional ADMM presents a noticeable gap in performance, above the 12%. However, it is the fastest control approach, more than 26 times faster than CMPC, which can be the determining factor for selecting a specific partition in many applications. On the other hand, the computational cost in terms of core seconds w.r.t. CMPC is approximately double. The optimization-based and algorithmic control approaches provide a trade-off regarding performance gain, computation time, and cost. The strategy based on $\mathcal{P}^{\text{Opt}}$ has a negligible loss in terms of optimality, while being 6 times faster than CMPC and having approximately the same computational cost. Therefore, if these are a priority over speed, $\mathcal{P}^{\text{Opt}}$ is preferable w.r.t. conventional DMPC. Algorithmic partitioning approaches have mixed results. The strategy based on $\mathcal{P}_3^{\text{Alg}}$ will give the best results in terms of optimality gap, but it is also slower and more computationally expensive than CMPC; therefore, it is undoubtedly an option to discard. The approach that uses $\mathcal{P}_1^{\text{Alg}}$ has a relatively small loss in optimality, but it is also

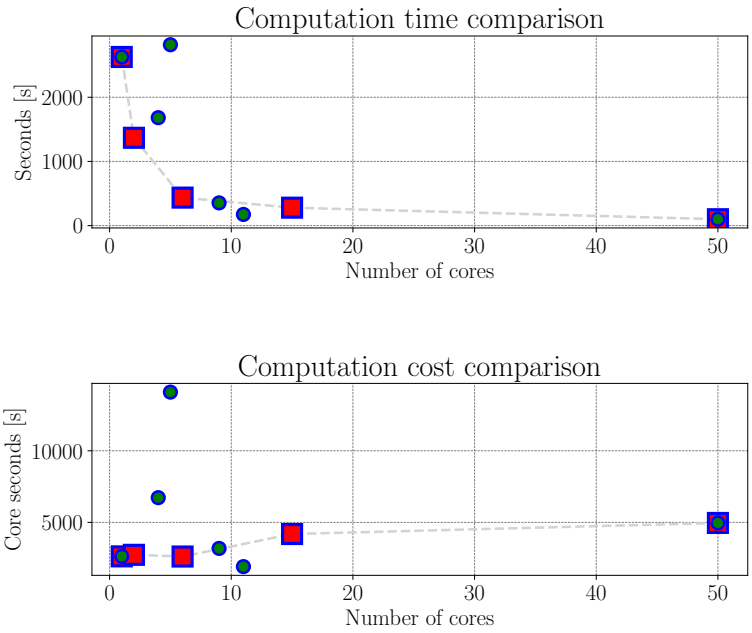


Figure 4.13: Computation times and costs for solving the same NCen-MPC problem using different partitions. The data points represented with squares refer to partitions obtained through an optimization-based technique, and they approximately follow well-defined exponential or linear patterns, represented by the dashed lines. The data points using circles refer instead to the results of simulations using partitions obtained through algorithmic approaches. We see that the latter have a less clear evolution across the number of sets, which might be related to the suboptimality of algorithmic approach.

the least expensive in terms of computational cost, while retaining a good computation time. It is thus a good alternative to $\mathcal{P}^{\text{Opt}}$. The partition $\mathcal{P}_2^{\text{Alg}}$, which is the one reported in Fig. 4.12, offers similar results, and can also be considered. In the end, the most appropriate partition to use will depend on the requirements for the specific application, and can be selected among the listed options with a clear indication of the gains and tradeoffs. A possible way to visualize computational time and costs for different partitions, which can help guide such decisions, is reported in Fig. 4.13.

This illustrative example shows how posterior evaluation of operational performance for different partitions is fundamental in NCen-MPC. In particular, for the same partitioning strategy, variations in the parameters to perform the partition can lead to very different control results. This fact motivates using a solid methodological assessment of control performance under different partitions.

Partitioning subclass	Partitioning class				
	Optimization-based	Algorithmic	Community detection	Game-theory-based	Heuristics
Unique techniques	[31, 247, 345]	[15, 161, 249, 250, 286, 356]; k-means: [66, 68, 180, 195, 354, 355]	[155, 156, 263]	[26, 240, 292]	[139, 150, 263]
Hierarchical		[70, 341]	[120]		
Time-varying		[161, 286, 343]	[15, 340]	[94, 95, 205]	[11, 199]
Hierarchical time-varying	[282]	[66]	[282]	[63, 65, 215–218, 292]	[348]
Problem decomposition	[167]		[298, 324]		
Input coupling	[67]	[341, 343]		[212, 214, 290, 291]	
Frequency-based		[327]	[339]		
Applications	[17, 306]		[120, 128, 233, 264, 325, 340]	[63, 97, 221, 222, 224, 225]	

Table 4.4: Categorization of the partitioning techniques according to class and subclass.

4.5 Analysis and Classification of the Partitioning Techniques for Non-Centralized Predictive Control

In this section, we analyze and classify the partitioning techniques for NCen-MPC that we found in the literature. The analysis we perform here is oriented toward the definition of the main characteristic of each methodology, highlighting their strengths and limitations, which generally apply to all techniques belonging to that category. For a detailed technical discussion both about the general methodologies and the specific papers presented, we developed instead the sections from Sec. 4.6 to Sec. 4.10.

Regarding the classification of the partitioning techniques, we propose and discuss in the following three different perspectives:

1. A categorization according to the general partitioning class, i.e. optimization-based, algorithmic, community-detection-based, game-theory-based, and heuristic.

2. A categorization in subclasses of the partitioning methods based on specific structures in the problem, or objectives to achieve through its deployment.

3. A classification according to the NCen-MPC control architecture used in the strategy.

The classification tables for the techniques in this survey are provided in Tab. 4.4 and Tab. 4.5. In the first table, we collocate the works found in the literature according to class and subclass. In the second table, we classify them according to the control approach used.

4.5.1 Classification According to the Partitioning Class

Optimization-based partitioning As introduced in Sec. 4.4, the problem of partitioning can be seen, in an abstract way, as the problem of assigning a set of objects to several given

sets. This type of problem can be naturally formulated as an MIP, see e.g. Sec. 4.4.3, whose solution will provide the optimal network partitioning according to the selected metric. At the basis of this formulation, there is a binary decision variable δ_{ij} that equals 1 if the object i belongs to the set j. All partitioning methodologies based on this descriptive approach using binary variables fall into the category of optimization-based partitioning and are discussed in Sec. 4.6. When considering an optimization-based partitioning technique, it is essential to consider the fact that the associated MIP is NP-hard [51, 162, 293]. Consequently, their scalability is limited, and optimization-based partitioning is suitable only for relatively small problems and static network topologies. This also means that online re-partitioning of a network using optimization-based approaches is prohibitive. Approximate solutions of mixed-integer problems can be found using, e.g. the genetic algorithm [112, 315], which does not guarantee global optimality, and still suffers from considerable computational complexity.

Algorithmic partitioning Partitioning approaches based on algorithmic procedures are a faster and computationally less intensive alternative to optimization-based ones. The trade-off for these gains is that, unless extensive search is performed, their results are suboptimal w.r.t. the alternative optimization-based strategies, which constitutes their main disadvantage. However, for large problems or in time-varying settings, algorithmic partitioning approaches result to be the only viable option thanks to their scalability. Additionally, through algorithmic procedures, it is possible to obtain partitions according to more sophisticated requirements, such as the satisfaction of control properties, more directly and straightforwardly than through optimization-based strategies. All the approaches discussed in Sec. 4.7 fall in this category of algorithmic partitioning. However, we also stress that the works based on the community detection method reported in Sec. 4.8 are algorithmic procedures. Despite this fact, we decided to discuss community detection methods separately because: 1) it represents by itself a branch of graph and network methods, in this case applied to partitioning for NCen-MPC control; 2) a rich body of studies and approaches has been developed in partitioning for NCen-MPC control exclusively through this method; 3) in this survey, almost every community detection methodology is based on a metric called modularity. Considering these characteristics, we dedicate Sec. 4.7 to all the algorithmic methods in the literature that do not belong to the community detection approaches, and are not based on the modularity metric or its extensions. A similar consideration holds for the game-theoretic oriented partitioning approaches of Sec. 4.9. In fact, these approaches are also mainly based on algorithms; however, the fundamental presence of game-theoretic arguments in the selection of the partitions, as well as the extensive development of the coalition control methodology rooted in this technique, deserves a separate discussion in a dedicated section.

Community-detection-based partitioning As mentioned in the discussion for algorithmic approaches, community detection methodologies have been developed in graph and network theory for the identification of strongly connected components of a graph for various applications [102]. Among all the techniques, great attention has been devoted to community-detection-based partitioning to methods based on the maximization of the *modularity* metric [243]. Most of the techniques in this section are conceptually based

on this approach. The maximization of modularity can be either sought through the solution of an optimization problem, an NP-hard problem, or with a heuristic or greedy algorithm, where the latter approach will, in general, provide a suboptimal result. All the techniques presented here are based on the aforementioned algorithmic approaches, thus allowing for scalability and real-time applicability for time-varying partitioning. The maximization of modularity and other derived metrics will provide groups of agents that exhibit weak inter-group coupling strengths, and, potentially, strong intra-group coupling. The unproven paradigm at the basis of modularity maximization for control problems is that a partition maximizing modularity will also provide optimal NCen-MPC performance. While this statement has not been proven true or false yet, a large body of studies, presented in Sec. 4.8, has shown that partitions maximizing modularity will, in general, improve control performance w.r.t. heuristic, expert, or random partitions.

Game-theory-based partitioning The partitioning approaches based on game-theoretic methodologies find their roots in the theory of coalition formation [272]. Agents in such networks participate in a game in which they seek their best allocation in a coalition to maximize the collective outcome, which, in this context, corresponds to the global cost function of the MPC problem. Most of the partitioning strategies developed in this field are based on algorithmic procedures; however, the prominent presence of game-theoretic techniques, and the fact that a whole body of literature has been developed about the resulting control strategy, i.e. Coal-MPC, motivate the treatment of these methodologies in a dedicated section. Game-theoretic partitioning methodologies are, in general, more complex to develop w.r.t. other algorithmic approaches, and require the clear definition of cooperative games and the associated cost functions. However, these approaches also allow for obtaining interpretable performance gains in the deployment of the Coal-MPC strategy, a point often missing in most algorithmic approaches.

Heuristic partitioning In this class, we include all the partitioning strategies for NCen-MPC that we found in the literature based on heuristic methodologies, which have not been developed originally to be extended to other applications. While the scope and generalizability of these strategies may appear limited, they can still be highly effective in specific contexts, and may offer inspiration for developing more broadly applicable methods.

4.5.2 Classification According to the Partitioning Subclass

As it is possible to see in Tab. 4.4, there are common features shared among partitioning techniques across different general partitioning strategies.

First, we can identify hierarchical strategies, in which we collocate approaches that either have multiple aggregation levels for the resulting partition, or are developed using a partitioning layer distinguished from the control layer. All purely hierarchical approaches presented in Tab. 4.4 belong to the first category. Among these, we find works that use a hierarchy to introduce a sequential decision-making ordering into the NCen-MPC strategy, or works with multi-level partitioning approaches, generally used for partition refinement. The former approaches allow obtaining coordinated actions prioritizing the performance of the controllers at the highest level of the hierarchy, and sacrificing the others; the

latter generally use purely topological metrics, thus not being directly oriented toward performance optimization.

Time-varying approaches include the techniques that allow for a reconfiguration of the network, either online during the execution of the control strategy or offline through the derivation of look-up tables. These methods are developed to react to topological changes in the network with the objective of maximizing the global operation cost. While real-time adaptability of the partition is advisable (when possible) to improve performance, the computational complexity of the partitioning problem can make it prohibitive if the network has fast dynamics. On the other hand, the offline computation of pre-defined partitions will surely allow for fast online reaction to topological changes, but on the other hand, it assumes either that it is possible to compute all these desired partitions, or there is a trade-off between performance and quality of the partition according to heuristics.

Hierarchical time-varying strategies are obtained by combining the two previous concepts. The most common setting is the following: a partitioning layer generally operates at a higher hierarchy level and a slower time scale w.r.t. a control layer. This approach has been extensively explored because the execution of a partitioning strategy cannot generally be performed in real time according to the control sampling time. Therefore, a slower time scale is used for the partitioning layer, allowing either periodic or event-driven network reconfiguration. Hierarchical time-varying strategies allow to obtain enhanced control performance, generally adapting the partitioning (reactively) w.r.t. network performance; however, two main aspects deserve some attention: 1) these are complex strategies, and therefore they require a higher level of coordination and communication w.r.t. more direct approaches 2) operating at different time scales allows for online re-partitioning, but assumes that the performance degradation during the partitioning intervals is acceptable, and eventual topological changes between re-partitioning intervals will not harm network operation.

Partitioning for input-coupled dynamics has been addressed separately because the underlying dynamics lead to strategies that present unique features, such as the definition of private and public control actions and related negotiation strategies, which are usually not considered when the dynamics present coupling through state interactions. In theory, most of the techniques defined for dynamical coupling among network subsystems can be extended to input-coupled dynamics with the necessary care. The most critical aspect for these systems is their limited applicability to real-world problems, which is also reflected in the limited amount of related studies.

Frequency-based approaches are defined based on the network's transfer functions that link input-output channels. These approaches find their roots in the MIMO decoupling approaches [310] for selecting control channels. Frequency-based approaches are generally developed for linear or linearized systems, and instead of using a direct performance assessment for partitioning, they use frequency-based performance metrics.

A range of approaches in the literature can be seen as applicative work of previously developed strategies, or as prototype techniques that have been extended later. These works can be used to develop comparative case studies for future developments.

Finally, a range of techniques has been uniquely defined in each partitioning methodology. These works do not share their direct scope with others; thus, we have placed them in a separate category. However, their features can potentially be extended to other

techniques, and direct comparisons might be possible.

4.5.3 Classification According to the Partitioning Methodology

A further classification of the partitioning techniques can be provided in terms of the methodology they are designed for. Specifically, a partitioning strategy can be either developed to operate on a given structure, or to address a specific problem. This classification is provided in Fig. 4.4 as a coloring scheme to distinguish the methodology to which all the subclass entries belong, where mixed approaches indicate that both methodologies have been used in the same subclass. In the following, we discuss their characteristics.

Structure-based partitioning strategies leverage the presence of a structure in the topology of the network or optimization problem to obtain the partition. Generally speaking, these approaches only require information about the network connections, and can use well-known tools from network and graph theory, such as spectral clustering or k-means. One reason to use such approaches is that for some applications, knowing the dynamics of the network is not essential for the specific partitioning problem, and other factors, such as achieving a particular decomposition for ease of operation, accessibility, or maintenance of the network, must be taken into account. Additionally, structure-based approaches do not generally need any information about the dynamics of the subsystems in the network, which can be advantageous in settings where security and privacy are of main concern. In this context, pairing structure-based partitioning with Dec-MPC approaches can be advisable. In such settings, there will be no requirement for real-time data or a communication infrastructure, and the approach can work well in situations where the network does not change over time, or changes slowly and predictably. The main trade-off in such implementations will be the loss in control performance, and the adaptability of the control structure. However, it is important to stress that structure-based approaches should not be limited to static networks, because they can also be developed for time-varying networks and be used with communication-based NCen-MPC approaches. Their main drawback in this sense is that they do not generally account directly for the dynamics of the subsystems; therefore, their actual impact on the performance should be quantified a posteriori.

Goal-oriented partitioning strategies are, in a sense, oriented toward the opposite direction compared to structure-based ones. In fact, they are developed to achieve a given goal without explicitly accounting for the structure of the problem. Usually, this is a control goal, and often, performance optimization. To this, goal-oriented partitioning must have access to some form of information that can relate to the predictable behaviors of the network, such as subsystem dynamics, time-series predictions from local controllers, or the operation cost of the local optimization problems. Additionally, communication and coordination structures are required to leverage and process such information, which increases development costs and complexity; but also affects the privacy of agents and security of network operation. Additionally, goal-oriented partitioning is naturally suited to work with time-varying networks, because it already requires real-time data about the current operation. It also follows that goal-oriented partitioning can be paired effectively with communication-based NCen-MPC, such as DMPC, HMPC, and Coal-MPC. The advantage here is generally sought in performance optimization, or to achieve particular

configurations of agents for specific tasks.

From this discussion, it is clear that both partitioning methodologies are fundamental in the literature, and research in the field of MPC should keep addressing both themes.

4.5.4 Classification According to the Control Strategy

In Tab. 4.5, we categorize the works in partitioning according to the control architecture to which they have been applied. Other than the more conventional Dec-MPC, DMPC, and HMPC strategies, we report that extensive work has been performed on the Coal-MPC methodology. Instead, few studies involve nonlinear MPC strategies. Finally, we mention the presence of a few mixed control strategies that allow for switching between control architectures according to control necessities. In the following, we briefly discuss each strategy, but for a detailed discussion, we refer the reader to [95, 208, 295].

Starting from the simplest form of NCen-MPC, we have Dec-MPC in which local controllers do not share any information with their neighbors and compute the local control actions either independently, or using some approximated or estimated information about the strength of the incoming dynamical coupling. Robustness arguments are used to ensure the stability of the network under uncoordinated operation. The biggest strength of Dec-MPC, other than the non-centralized computation of the control action, lies in the ability to preserve the privacy of local subsystems during network operation, since there is no information sharing. The main drawback is the loss of performance w.r.t. CMPC, given the conservative nature of local actions.

In the DMPC approach, information about the current state of the local subsystem, the current control action, or even the predicted state-input sequence is shared among local controllers. This communication is supported by a coordination protocol, which allows local controllers to refine the local actions to achieve superior global performance for the network. The communication and coordination strategy can be structured according to different criteria, thus producing different DMPC approaches. In linear settings, DMPC strategies can converge to near CMPC performance, which is the main advantage of DMPC. However, DMPC also has drawbacks: more expensive hardware requirements w.r.t. Dec-MPC, due to the communication infrastructure and the necessity of more advanced abilities for local controllers; complex coordination algorithms, which can also be iterative and must operate within the limits of real-time control; information sharing, which is not always guaranteed to be possible or real-time.

HMPC includes any control strategy having local controllers and a coordination layer in the form of a centralized decision maker. Such approaches are usually designed to achieve performance advantages, while allowing to overcome other technical challenges, such as model complexity reduction, multi-scale network operation, privacy preservation, or optimization of global coordination. Given the flexibility of HMPC approaches, the specific drawbacks of each technique depend on its implementation, but all approaches unquestionably come at the cost of an increased technical complexity and increased hardware requirements w.r.t. simpler NCen-MPC approaches.

The Coal-MPC strategy was born to fuse MPC with game theory in a non-centralized control setting. The result is a control strategy in which local control agents can merge into coalitions according to game-theoretic strategies to achieve superior control performance. Therefore, the Coal-MPC strategy can also be interpreted by itself as a game-theoretic-

Control approach	Partitioning class				
	Optimization-based	Algorithmic	Community detection	Game-theory-based	Heuristics
Decentralized MPC	[17, 247]	[161, 249, 250, 341]	[15, 340]	[26]	[150]
Distributed MPC	[31, 167, 247, 282, 345]	[15, 180, 286, 327, 343, 354, 356]	[15, 120, 155, 233, 263, 264, 282, 298, 324, 325, 339]	[221, 224]	[139, 199, 263]
Hierarchical MPC	[247, 306]	[66, 68, 70, 195, 249, 355]	[128]		[348]
Coalitional MPC	[67]			[63, 94, 95, 97, 205, 212, 214–218, 221, 222, 224, 225, 240, 291, 292]	
Nonlinear MPC		[161, 286]	[324]		
Mixed strategies	[247]			[65, 221, 224]	[11]

Table 4.5: Categorization of the partitioning techniques according to the control strategy deployed.

oriented partitioning strategy for distributed MPC, with dynamic allocation of local controllers into time-varying coalitions. In this regard, the Coal-MPC problem inherits the computational complexity of the general partitioning problem, or coalition formation problem, i.e. it requires the online solution of an NP-hard problem. This main drawback has been solved through different algorithmic procedures, which has led to the development of a large body of literature also discussed in this survey. The main theoretical advantage of Coal-MPC is that it allows for online dynamic partitioning with the objective of global performance optimization in a game-theoretic sense.

Regarding NLin-MPC, the above considerations have to be extended in a setting where the MPC model is nonlinear. This approach can allow for superior operational performance, but has several drawbacks, mainly: the complexity of defining an appropriate nonlinear model, the computational complexity related to nonlinear optimization, the eventual presence of local minima in the cost function, and the difficulty in ensuring stability of operation.

Mixed strategies for NCen-MPC use any combination of the previous techniques, trying to balance their strengths and limitations with online reconfiguration of the controllers' settings and (sometimes) partitions. This fact necessarily implies that such strategies have a high implementation complexity, and a combinatorial number of possible approaches at each time step, which is usually addressed through the use of heuristics.

4.6 Optimization-Based Partitioning

4.6.1 General Techniques

An advanced optimization-based partitioning strategy for implementing NCen-MPC techniques has been presented in [247]. The objective is to design a strategy that can work for every predictive control structure, being decentralized, distributed, or hierarchical. To this aim, a variable communication topology is considered. In particular, a communication

graph is introduced, linking together systems that directly interact on a dynamics level. Then, an integer variable δ_{ij} is introduced for each edge ϵ_{ij} of the information graph, assuming the following values: $\delta_{ij} = 0$ if controller $\mathcal{K}_i$ does not share any information with controller $\mathcal{K}_j$; $\delta_{ij} = 1$ if the control sequence $\tilde{u}_{ij}(k-1)$ at the previous time step is shared from $\mathcal{K}_i$ to $\mathcal{K}_j$; and $\delta_{ij} = 2$ if control sequence $\tilde{u}_{ij}(k)$ at the current time step is shared from $\mathcal{K}_i$ to $\mathcal{K}_j$. The use of these variables for the communication topology in any number of their possible combinations realizes a number of 3^{N_S} possible decentralized, distributed, or hierarchical communication topologies for a number N_S of agents in the network, i.e. a combinatorial explosion. In [247], the number of these configurations is drastically reduced by introducing heuristics. The selection of the optimal communication topology is then achieved by introducing a term into the global cost function of the problem that accounts for the cost of communication. Accordingly, the framework adapts a non-centralized control strategy depending on how information is shared. The methodology is validated on a 16 water tanks system [207], showing how the optimal partitioning is affected by the change in operating conditions.

An original approach to optimization-based network partitioning for DMPC can be found in [345], where an input-output decomposition of large-scale linear systems is sought. In particular, to overcome the limitations of previous techniques such as [250], or system decompositions based on interaction analysis approaches (e.g. RGA [310]), the paper proposes a two-stage procedure based first on input clustering decomposition (ICD), and then on input-output pairing decomposition (IOPD). For the former, ICD consists of finding a matrix $G = (g_{ij}) \in \mathbb{R}^{m \times M}$, where m is the number of inputs and $M \leq m$ the number of subsystems, such that $g_{ij} = 1$ if input i belongs to subsystem j and zero otherwise. Then, a transformation matrix $T(G) \in (R)^{m \times m}$ is built from G, allowing to find the input decoposition of the original vector u into the form $\bar{u} = [\bar{u}_1^\intercal, \dots, \bar{u}_M^\intercal]^\intercal = T(G)u$, and $T(G)$ is an orthogonal matrix allowing easy back transformation. After the ICD, the IOPD is found by minimizing the coupling effect between the subsystems. This is achieved by leveraging the condensed formulation of the MPC problem as a quadratic program [204]: for a prediction horizon N, and a given initial condition $x(0|k) = x_{0,k}$ at a time step k, by defining $\tilde{u}_k$ as the input sequence over N time steps, the cost of MPC is $J_k = \tilde{u}_k^\intercal H \tilde{u}_k + 2x_{0,k}^\intercal F^\intercal \tilde{u}_k$, with matrices H, F constructed using the system model and the weighting matrices (this is a standard derivation, for further details see [204]). Then, applying the ICD transformation for the input decomposition, the cost J is rewritten in terms of the input vector $\bar{u}$, through a new matrix O that is a function of $T(G), H$, and F. The coupling effect among the subsystems is then quantified as:

$$J^{\text{coupling}} = \frac{\|O - \text{diag}(O_{11}, \dots, O_{MM})\|_{\text{F}}}{\|O\|_{\text{F}}} \tag{4.50}$$

The minimization of this cost is achieved by approximately selecting the entries of G, which provides the input clustering sought. Given the binary nature of G, this is a nonlinear integer programming problem. Therefore, the authors approach it using the Genetic Algorithm (GA) [315]. The partitioning approach is further developed for partial observability (for details see [345]). Application of the partitioning methodology is performed on a chemical plant with five operation units known in the literature as the Tennessee Eastman problem [88, 203]. Partitioning of the latter is executed on a linearized version of the plant around

operating points generated through a stabilizing control action [229]. No control validation of the proposed DMPC architecture is performed.

4.6.2 Multi-Objective Optimization in Partitioning

Multi-objective optimization is at the basis of the partitioning strategy proposed in [31]. Specifically, an optimization problem of the following form is considered at each time step k:

$$\begin{aligned} \min_{\mathcal{P}(k)} \ & \sum_{i=1}^{4} \varphi_i \sigma_i(\mathcal{P}(k)) \\ \text{s.t.} \ & \bigcup_i \mathcal{C}_i(k) = \mathcal{P}(k) \\ & \bigcap_i \mathcal{C}_i(k) = \emptyset \end{aligned} \tag{4.51}$$

where the constraints require that the sets $\mathcal{C}_i(k)$ constitute a nonovelapping partitioning, i.e. $\mathcal{P}(k) = \{\mathcal{C}_1(k), \ldots, \mathcal{C}_{N_C}(k)\}$; φ_i are weights; and the four indicators σ_i defined for the sets $\mathcal{C}_i$ account for: the number of links connecting the sets, the difference in the number of nodes between the sets, the distance between the sets, and the relevance of the information shared between the sets. To solve the proposed optimization-based problem, an algorithmic distributed approach based on the Kernighan-Lin algorithm [121] for graph partitioning is proposed. In this algorithm, each node is a decision maker, which selects the set to move it to based on a utility linked to the local cost in (4.51). This partitioning approach is applied to the Barcelona drinking water transport network using a DMPC approach based on density-dependent population games [294].

4.6.3 For Optimization Problem Decomposition

A partitioning technique developed for railway traffic management is presented in [167], where a switching max-plus-linear model is used to describe the Dutch railway network [166]. Given the hybrid nature of the model, optimization of such a system over a prediction horizon provides an MILP, and the aim of [167] is to develop a DMPC strategy to solve it. This strategy is a cooperative iterative approach with a consensus threshold. The centralized optimization problem over a given horizon N is stated as:

$$\begin{aligned} \min_{z(k)} \ & c^\intercal(k) z(k) \\ \text{s.t.} \ & A(k) z(k) \leq b(k) \end{aligned} \tag{4.52}$$

where the state and input sequences for the network are in $z(k) = [\tilde{x}^\intercal(k) \ \tilde{u}^\intercal(k)]^\intercal$. The partitioning of this global optimization problem is performed according to a set of desired properties for the local optimization problems. Specifically, the partitioning has to provide non-overlapping subproblems, such that constraints are decoupled, and the size and number of variables of the problems are approximately the same. This is achieved through an MIQP setup for a selected number n_{sub} of subsystems. Specifically, for each constraint of problem (4.52) (in total n_{T_2}) a number N_C of binary variables δ is introduced. A set of N_C continuous variables S representing the number of constraints of each problem is also introduced,

and an additional variable $M_{\text{MAX}} \geq M_i - M_j$, for $i, j = 1, \ldots, N_{\text{C}}$ represents the maximum difference in the number of constraints. Then, the cost of the partitioning problem is:

$$J = \rho M_{\text{MAX}} - \sum_{j=1}^{n_{\text{T}_2}} \sum_{k=1}^{n_{\text{T}_2}} \sum_{i=1}^{N_C} \delta_{ji} Q_{jk} \delta_{ki} \tag{4.53}$$

where ρ is a tuning parameter, and the weighting matrix Q is constructed assigning to each element Q_{ij} the number of constraints connecting the constraints set i to j. Constraints are added to the MIQP to enforce the properties listed above. Solving this problem provides a decomposition of (4.52) into N_{C} subproblems. The approach is validated on the model of the Dutch railway network [166] against a CMPC implementation. The results show that the distributed implementation is up to 90% faster in computing the predictive control action w.r.t. CMPC with only marginal performance losses.

4

4.6.4 Ad-hoc Performance Indicators

Optimization-based partitioning using as a performance indicator the wake-effect[6] coupling in wind turbines is used in [306]. This work addresses non-centralized hierarchical control of wind farms, where MPC is used for reference point setting at the control partition level, while conventional controllers are used for individual turbines. The definition of the groups in the partition is performed based on the coupling among the turbines induced by the wake effect [164], whose intensity is derived using first-principles modeling of the system [43]. Accordingly, a weighted directed graph is constructed using the intensity of the wakes as labeling [13]. A multi-objective integer optimization program is constructed, where a binary variable δ_{ij} is 1 if turbine S_i belongs to the group C_j and 0 otherwise. Three concurrent costs are considered: the first requires that the wake effect among turbines is maximized, the second involves the minimization of the distance among the turbines, and the third balances the size of the resulting groups, requiring that the difference of elements among them is minimized. The constraints ensure non-overlapping partitioning. Due to the formulation of the costs funciton, this is a nonlinear optimization program, which is transformed into a linear form by introducing auxiliary variables [36]. To overcome the computational complexity of the proposed strategy, the operating conditions of the wind turbines are discretized, and 12 wind directions are considered. Additionally, the strategy is only suited for offline construction due to the computational cost, but a look-up table can be constructed for different operational settings. The approach is validated on a wind farm with 42 NREL-5MW wind turbines [158], and modeled using SimWindFarm [119]. The non-centralized strategy is compared with its centralized counterpart, showing a significant reduction in computation times while ensuring a good level of performance.

4.6.5 Robust and Stochastic Optimization

Robust and stochastic methodologies for partitioning a system have been developed in [17] for Dec-MPC of the thermal zones of a building. A graph representing the thermal interactions among the zones is constructed and weighted by a representative metric depending on the temperature difference between areas, and external disturbances. This

[6] The wake-effect refers to the wind reduction and increased turbulence that downstream turbines experience due to the extraction of wind power from upstream turbines.

representation allows the formulation of a mixed-integer optimization problem for partitioning, where specifications about the resulting clusters are imposed through constraints [48]. In addition, in the stochastic formulation the definition of the thermal interactions is replaced by their expectations in the cost function; while in the robust formulation their maximum values are used to account for the worst-case scenario. The former approach assumes that the probability of the occurrence of the disturbances is known, while the former approach is more conservative. The efficacy of the resulting partitioning is assessed through ad-hoc performance indicators for the specific application. The approach is extensively validated for the Dec-MPC control of a 5 and a 20 zones case study, and compared with the partitioning approach of [62].

4.6.6 Input-Coupled Dynamics

A binary programming approach for partitioning a network of linear systems coupled through input interactions has been proposed in [67], i.e. a dynamics of the following form is considered:

$$\begin{aligned} x_i(k+1) &= A_{ii}x_i(k) + B_{ii}u_i(k) + w_i(k) \\ w_i(k) &= \sum_{j\in\mathcal{N}_i} B_{ij}u_j(k). \end{aligned} \tag{4.54}$$

Using the condensed formulation of the MPC problem [204], and the structure of the input-coupled dynamics, the gradient of the global cost function J for the selected prediction horizon can be approximated [82] for a selected topology Λ by the sum of the gradients for the local contributions g^{local}, and the one for the coupling inputs, thus providing:

$$g^{\Lambda} = \nabla J(\tilde{x}, \tilde{u}) \approx g^{\text{local}} + \sum_{ij\in\Lambda} \Delta g_{ij}^{\Lambda} \tag{4.55}$$

where Δg_{ij}^{Λ} are coupling contributions for Λ. Then, a vector of $|\mathcal{E}|$ binary variables δ is defined to establish if a link $(i,j) \in \mathcal{E}$ belongs to a certain topology. Accordingly, the partitioning of the network is retried by solving a binary quadratic program with cost function $\delta^T Q\delta + R\delta$, where the weighting matrices Q and R are functions of the gradient approximation g^{Λ}. Further details about the problem derivation, and a theorem bounding the performance degradation in the computation of the cost function based on the topology selection are given in [67]. An analysis of the scalability of the approach is also performed, comparing it against genetic algorithm optimization, and a greedy algorithm. The control approach is then validated on an urban transportation network [82] with eight intersections, and performance is compared w.r.t. CMPC. Simulations show how this strategy can reduce the number of active communication links more than the 40% while retaining good levels of performance.

4.6.7 Hierarchical Approaches for Time-Varying Graphs

In [281, 282], a binary quadratic programming (BQP) approach is used to obtain the partitioning of a network at different levels of aggregation through a granularity parameter α, for a potentially time-varying topology. The approach is based on constructing an equivalent graph representation of the network using partial derivatives of the dynamics.

Specifically, for a network characterized by the nonlinear difference equation (4.3), a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is constructed by defining the set of input and state nodes, respectively $\mathcal{V}_x, \mathcal{V}_u$, according to the variables in x and u. The weighting, and therefore the definition, of the edges in $\mathcal{E}$ is given by (4.4). This definition provides a time-varying graph for the general nonlinear case, for which re-partitioning should be performed at each time step. For this equivalent graph, an intermediate algorithmic step, detailed in Sec. 4.8.6, is used to select the fundamental system units (FSUs), denoted by $\mathcal{S}_i$, to be used for the construction of the agents (but these can also be given a priori). Once a number N_{FSU} of FUSs is constructed, a global quadratic metric is used into a BQP to select collections C_i of FSUs that have a strong interaction inside the set, and a weak interaction among the sets. This is done introducing a number N_{FSU}^2 of binary variables δ_{ij} with $\delta_{ij} = 1 \Leftrightarrow \mathcal{S}_i \in \mathcal{C}_j$, and zero otherwise, and defining the intra- and inter-collection weights as:

$$W^{\text{inter}}(\delta) = \sum_{m=1}^{N_{\text{FSU}}} \sum_{i=1}^{N_{\text{FSU}}} \sum_{\substack{j=1 \\ j\neq i}}^{N_{\text{FSU}}} \sum_{\substack{l=1 \\ l\neq m}}^{N_{\text{FSU}}} \delta_{i,m}\delta_{j,l}(|w(i,j)| + |w(j,i)|) \tag{4.56}$$

$$W^{\text{intra}}(\delta) = \sum_{m=1}^{N_{\text{FSU}}} \sum_{i=1}^{N_{\text{FSU}}} \sum_{j=1}^{N_{\text{FSU}}} \delta_{i,m}\delta_{j,m}(|w(i,i)| + |w(i,j)| + +|w(j,i)| + |w(j,j)|) \tag{4.57}$$

Additionally, a weighting term to minimize the size of the resulting sets is defined as:

$$W^{\text{size}}(\delta) = \sum_{m=1}^{N_{\text{FSU}}} \left(\sum_{i=1}^{N_{\text{FSU}}} \delta_{i,m} \right)^2 \tag{4.58}$$

These three weights are used as the cost function in the following BQP that allows finding the optimal partitioning maximizing the interaction strength in the collections for a given value of the parameter α that influences the level of granularity:

$$\begin{aligned} \min_{\delta} \quad & W^{\text{inter}}(\delta) - W^{\text{intra}}(\delta) + \alpha W^{\text{size}}(\delta) \\ \text{s.t.} \quad & \sum_{j=1}^{N_{\text{FSU}}} \delta_{ij} = 1 \qquad \forall i \\ & \delta_{ij} \in \{0,1\} \end{aligned} \tag{4.59}$$

The constraints ensure that non-overlapping sets constitute the resulting partitioning, and varying α allows to obtain collections of different sizes. The partitioning approach thus defined in [282] is applied for partitioning a modular network with 64 agents, and a random network of hybrid systems with 50 agents. The first case shows how varying the granularity α allows to retrieve modules at different aggregation levels, allowing a hierarchical clustering. In the second case, an ADMM-based DMPC approach [321] is deployed for network control. Different simulations are performed: one for CMPC, one for the conventional DMPC-ADMM with 50 agents, and three for partitionings obtained with varying levels of α. The simulation results show how the optimization-based partitioning DMPC controllers have a loss of performance w.r.t. CMPC below 0.3%, while the conventional DMPC-AMM approach with 50 agents has a loss of more than 12%. This performance

advantage is paid in computation time, which is generally higher for partitioned system w.r.t. the conventional ADMM formulation. The approaches are also compared in terms of computational cost by calculating the core seconds for the simulations, i.e., the number of seconds necessary to compute the control action in parallel times the number of agents working in parallel. In this regard, optimization-based partitioning allows a computational cost in line with CMPC, while conventional DMPC-ADMM is at least 2.59 times more expensive. It is important to stress that, even if the framework allows the partitioning for time-varying topologies, the computational cost of a BQP is prohibitive for the online re-partitioning of large systems. This problem is solved if the system only transitions among a given number of known topologies, which allows an offline computation of all the optimal partitionings. Otherwise, an algorithmic modification of the strategy in [282] is proposed in Sec. 4.8.6 to overcome the computational complexity associated with this class of problems.

4.7 Algorithmic Partitioning

4.7.1 Applied to Equivalent Graph-Based Representations

One of the first contributions to graph-based partitioning for the application of non-centralized predictive control is found in [250]. The starting point of the partitioning strategy is a graph-based representation, proposed as a control-oriented representation described by an incidence matrix [47, 351]. The graph is divided into non-overlapping subgraphs according to an algorithm developed starting from the graph-partitioning-based ordering algorithm (GPB) [121], with various modifications and heuristics to adapt it for control of a complex system. One of the core components of the algorithm is the cut size, i.e. the number of links that belong to different subgraphs, which is an indirect measure of the desired subgraph size. The partitioning approach is applied to the case study representing the Barcelona drinking water transport network [252], using the Dec-MPC technique previously developed in [251], obtaining six groups of systems distributed to three level of control hierarchy. Simulation results show a reduction of about 50% in computation time w.r.t. CMPC, while a reduction in the electricity usage is achieved at the expense of a higher water volume required. The overall loss of performance is contained to the 15%. This version of the algorithm is developed to work offline, before the control strategy is deployed.

Partitioning based on nested ϵ-decomposition [299] is proposed in [249] for decentralized predictive control. For a linear causal system, the ϵ-decomposition works as follows. Construct a matrix M using all system variables as nodes of a graph, i.e. build the weighted adjacency matrix:

$$M = \begin{bmatrix} A & B & 0 \\ 0 & 0 & 0 \\ C & 0 & 0 \end{bmatrix} \tag{4.60}$$

Then, for a given threshold ϵ, compute the permutation matrix P that provides a new block decomposed matrix $\tilde{M} = P^\intercal M P$ consisting of N block such that, for the off-diagonal terms, it holds that $\tilde{M}_{ij} \leq \epsilon$. This decomposition transforms the network into N connected subgraphs where interconnections are defined by the off-diagonal terms of $\tilde{M}$ and their strength

constrained by the choice of ϵ. A maximum number of $|M|$ nested ϵ-decompositions is possible for any given M. Further details and stability analysis of this decomposition are presented in [299]. In [249], the ϵ-decomposition is applied to the Barcelona drinking water network [252] incorporating a heuristic selection of ϵ, and a hierarchical Dec-MPC strategy is applied to the resulting three-subsystem network. The architecture is validated against a CMPC controller implementation showing an overall performance loss always smaller than 2%, with a reduction of computation times up to 35%.

An algorithmic approach for nonlinear systems is devised in [161]. This approach is also based on the control-oriented representation and the derivation of the incidence matrix [47]; however, in this case the matrix is constructed accounting for relations among system variables, where each input state and output is considered as a distinct node. A general nonlinear dynamics of the form:

$$\mathcal{S}: \begin{cases} x_{k+1} = f(x_k, u_k, w_k) \\ y_k = g(x_k, u_k, w_k) \end{cases} \tag{4.61}$$

is used to construct the graph; however, this dynamics is linearized around an operating point to derive a weighting of the associated graph; specifically, the matrices (A, B, C, D) resulting from the linearization are used. The algorithmic approach starts by the centers of the clusters as the input variables. Then, a sorting procedure is used to order the state and output vertices according to their degree. A merging phase groups subgraphs based on their number of edges. The procedure is regulated by the cut size, according to [151], but also considering the number of resulting groups. The resulting partitioning is used in the deployment of a Dec-NLin-MPC strategy over an industrial chemical plant constituted by two continuous stirred-tank reactors in cascade [25, 333]. The results show that the decentralized approach proposed in [161] has a performance comparable to C-NLin-MPC, and superior performance w.r.t. the Dec-NLin-MPC approach proposed in [333].

A partitioning approach based on the strength of interaction among subsystems is proposed in [356]. The underlying DMPC strategy in this work is the dual-mode DMPC proposed in [191]. Specifically, for a network composed of subsystems interacting through a dynamical coupling, the approach in [356] requires grouping these subsystems into larger virtual middle-scale subsystem, also called M-subsystems. Then, a cooperative DMPC strategy is deployed, for which, at each time step, iterative optimization is performed within the systems in the M-subsystem, and communication occurs only once between different M-subsystems. The clustering approach requires the determination of these M-subsystems to minimize the coupling strength. Further details about the definition of weakly coupled M-subsystems are detailed in the paper. Once the weakly-coupling condition is established, for a selected threshold δ the variable adjacency matrix $A(\delta) = (a)_{ij}$ defining the M-subsystems is obtained as $a_{ij} = 1$ if $\|A_{ij}\| \geq \delta$, and $a_{ij} = 1$ otherwise. The algorithmic clustering approach consists of finding such δ and a permutation matrix T such that $T^\intercal \tilde{A} T$ is block-diagonal, and the overall system is weakly coupled. The clustering algorithm consists of gradually reducing δ from a given initial value δ_0 until the decomposition into weakly coupled M-subsystems is achieved, where conditions are validated at each time step. Algorithms details are in [356]. The overall approach is validated on a case study for building temperature regulation against CMPC. The system comprehends eight rooms that should keep the temperature variation at zero despite external influences. The DMPC

approach can stabilize the network, as CMPC, but only qualitative results are provided, and some performance degradation is present. The paper also provides theorems for the stability and recursive feasibility of the DMPC strategy.

A framework for algorithmic partitioning of nonlinear systems based on the equivalent graph representation of linearized dynamics around an operating point is proposed in [286]. In this approach, each time a re-linearization of the nonlinear dynamics is performed, the system might be re-partitioned. The re-partitioning trigger is not specified, but it is reasonable to assume that it coincides with a change in the topology of the associated graph. The partitioning algorithm proposed is based on the iterative grouping of input-state-output variables, followed by a controllability check. The algorithm does not guarantee the terminability, or that controllable groups are achievable. The other contributions of the paper are the derivation of two DMPC techniques, cooperative and non-cooperative, working on linearized versions of the models. The viability of the approach is demonstrated for the reactor-separator process [318], with two reactors in series and a separator.

4.7.2 Applied to Flow Graph Representations

Algorithmic partitioning for power networks using a flow-graph representation is considered in [180]. The approach is developed in multiple conceptual steps. First, the power network is divided into sources for generators (a set of nodes $\mathcal{V}^{\text{source}}$), and sinks for the loads (a set of nodes $\mathcal{V}^{\text{sink}}$), thus constructing a flow graph. Then, the optimal power flow problem [104] for the best and worst case scenarios is solved, i.e. treating separately maximum demands and generations to obtain two optimal solutions for transactions of flows between sources and sinks. The average of these two solutions allows defining the average transaction $x^*_{ij}(k)$ between sources $i \in \mathcal{V}^{\text{source}}$, and sinks $j \in \mathcal{V}^{\text{sink}}$. Then, for each $i \in \mathcal{V}^{\text{source}}$ and $j \in \mathcal{V}^{\text{sink}}$ the shortest path $\mathcal{L}_{ij}$ is defined [84], and the value $x^*_{ij}(k)$ is assigned to all edges in $\epsilon \in \mathcal{L}_{ij}$. Finally, the weight of each edge in the network is computed by summing all the values $x^*_{ij}(k)$ of the shortest paths passing by that edge. A weighted flow graph is thus constructed. A partitioning of this graph for a given number of clusters is obtained using the k-way partitioning method minimizing the edge cut using the METIS algorithm [163]. This procedure is performed at the time scale of the clustering procedure, slower than the time scale of the control process. An HMPC approach with local Dec-MPC is then deployed to control local power clusters independently. Local requests of energy activate a supervisory layer if clusters cannot satisfy the demand. Further features, such as using energy storage systems, multiple time scales, and ADMM distributed computations in the supervisory layer, are detailed in [180]. The approach is implemented on the IEEE 118-bus, showing the online clustering capabilities of the approach.

4.7.3 Using Frequency-Based Performance Indicators

The use of the Relative Time-averaged Gain Array (RTGA) as a metric to perform partitioning for DMPC is explored in [327]. The traditional Relative Gain Array (RGA) [54] has been extensively used in control of multiple-input multiple-output (MIMO) systems [310]. It is defined by considering the MIMO transfer function of a system $G(s)$, and computing the RGA matrix $\Lambda = G(0) \cdot G^{\intercal}(0)$. Then, if the element λ_{ij} of Λ is close to 1, the I/O loop of the MIMO system from input u_j to output y_i is well decoupled from other loops in the systems, and the variables should be paired for control design. This feature has been used

for optimal pairing using integer programming in [171]. The authors in [327] propose a new metric called Relative Time-Avaraged Gain Array (RTAGA), based on the step response of the system averaged by an exponential distribution function $f(t,\tau) = (1/\tau)\cdot e^{-t/\tau}$, for a parameter τ characterizing the decay of the exponential. Then, for matrix G of transfer functions, the element $g_{ij}(1/\tau)$ is the intensity of the response y_i for a step input u_i weighted by the distribution $f(t,\tau)$ decaying at time scale τ. Accordingly, the RTAGA matrix is defined as $\Lambda(1/\tau) = G(1/\tau)\cdot G^{\intercal}(1/\tau)$. The authors proposed the RTGA because it provides a better framework for system decomposition w.r.t. RGA. Further details are in [327]. For partitioning the system using the RTGA, the authors of [327] rely on the input-output bipartite graph, and the weighting of each edge (u_j, y_i) representing input-output loops is given by the scalar w_{ij} defined according to the entries of $\Lambda(1/\tau)$ as:

$$w_{ij} = \begin{cases} \lambda_{ij} & 0 \leq \lambda_{ij} \leq 1 \\ 1/\lambda_{ij} & \lambda_{ij} > 1 \\ 0 & \lambda_{ij} < 0 \end{cases} \tag{4.62}$$

Then the modularity Q of the bipartite weighted graph is defined according to [28], and modularity maximization is achieved through the Louvain fast unfolding algorithm [45]. Further, heuristics on the choice of the parameter τ are given. The approach is applied for deploying a noncooperative and iterative DMPC control scheme [196] over a reactor-separator process with two continuously stirred tank reactors in series [263]. Different decompositions of the networks are achieved, and results are compared against CMPC through a quality index normalizing the performance-computation-time product w.r.t. CMPC. This quality index is used to determine the best partitioning of the network.

4.7.4 Using k-Means

One of the most used algorithms for clustering is k-means [346]. At the core of the algorithm there is the problem of organizing N objects, e.g. vectors $x \in \mathbb{R}^d$, into K subsets. This is achieved by using the definition of Euclidean distance, and an algorithm is developed to minimize the squared error between each object and the center of the clusters. The algorithm starts with an initialization of the centers of the K clusters (either random or informed). Then, each object is assigned to the nearest cluster. Accordingly, the prototype matrix, i.e. the matrix containing centroids or the means of the clustering, is updated with the given assignment. The last two steps are iterated until there is no further change in the clusters. The computational cost of the algorithm is $O(N\,K\,d)$. The k-means clustering is well developed, and parallel implementations are available [319] to improve computation times. The interested reader can refer to the survey [346] for further information.

The clustering of a wind farm using k-means has been performed in [68]. The article focuses on the frequency regulation of a double-fed induction generator, which is affected by both the operating conditions of the plant, and the wind orientation and strength. To improve the frequency regulation of the system, a multi-layer control approach is proposed: MPC [3] is used for frequency regulation and power output maximization, whereas k-means clustering [330] based on wake-effect interaction is used to spatially cluster the wind turbines. The clustering thus performed allows the division of wind turbines into minimally coupled clusters. The approach is applied to a 25-turbine farm (1.5 MW), showing its effects on frequency regulation and power output w.r.t. more traditional control approaches.

An improved version of k-means, i.e. crow search [182], is used in [355] to cluster a wind farm. Crow search is used in this approach for its improved clustering accuracy and cluster stability, allowing the authors of [355] to achieve superior cluster quality w.r.t. traditional k-means. The wind farm is partitioned according to four key performance indicators, which are the power characteristic of the turbine, the smooth coefficient, the generation potential coefficient, and the anomaly coefficient [137, 349]. Given this dataset, the algorithmic partitioning is performed for a given number of clusters. The resulting partitioning is used in a HMPC scheme, and the performance w.r.t. CMPC are qualitatively compared in a 12-turbine wind farm case study.

An approach for clustering wind farms based on an approximate linear model of their power tracking [71, 153] is proposed in [195]. Once an estimate of this transfer function is available for each turbine in the farm, [195] proposes to apply a global fuzzy c-means algorithm for clustering the network [133, 309]. The approach is deployed on a farm with 20 NREL 5-MW wind turbines [158], modeled using SimWindFarm [119], and obtaining four clusters. The control approach is hierarchical and employs a proportional controller in the lower layer and an MPC in the upper layer, where in the latter, all the clusters are aggregated into a single performance index. Simulation results show how the proposed strategy outperforms both conventional PD control and CMPC, while reducing computation times.

4.7.5 Data-Driven Decomposition

Partitioning in a data-driven application is discussed in [354], where water distribution networks are used as examples of large-scale nonlinear systems with coupling dynamics. The scope of a data-driven approach is to capture the nonlinear dynamics that might not figure in purely model-based approaches as [249]. Once time series data about inputs $\mathbb{U} = \{\tilde{u}_\ell\}_{\ell=1}^{n_u}$ (outlet pressure at the water pumps), states $\mathbb{X} = \{\tilde{x}_\ell\}_{\ell=1}^{n_x}$ (level of water reservoirs), and outputs inputs $\mathbb{Y} = \{\tilde{y}_\ell\}_{\ell=1}^{n_y}$ (pressure at users' nodes) are collected, a system model is defined as $\mathcal{S}(\mathbb{U}, \mathbb{X}, \mathbb{Y})$. The partitioning problem is then formulated s.t. the WDN is divided into k subsystems $\mathcal{S}_i$, where k is a number defined by inspection depending on the shape of the time-series data in matrix $\mathbb{Y}$. The underlying partitioning procedure is then provided by the k-shape clustering algorithm for time series sequences [258], and canonical correlation analysis to establish the strength of interaction among the groups of variables, which allows to define strong and weakly coupled neighbors according to heuristic thresholds. The algorithmic procedure for partitioning allows to retrieve k groups of strongly coupled non-overlapping subsystems with approximately the same number of variables. The approach is applied to the water distribution network of Shanghai, using 800 samples captured every 10 minutes from 44 sensors in the network. Different partitionings are obtained by varying the parameters of the algorithm, but the one providing 6 groups is selected since it gives the minimum variance. Simulations are performed to compare the proposed enhancing DMPC strategy with the Dec-MPC approach defined in [249, 250]. Overall, the strategy proposed by [354] allows to achieve a reduction in the water pressure of the network, while ensuring stability and robustness, thus reducing leakages and energy requirements.

4.7.6 Hierarchical Clustering

The study [70] introduces a cooperative DMPC framework based on topological hierarchy decomposition, aiming to optimize communication efficiency while maintaining global system performance. The theory at the basis of the approach is interpretive structural modeling [18], which allows to hierarchically structure subsystems based on their coupling strength, ensuring that strongly coupled subsystems are grouped within the same layer, while weakly coupled ones are placed in lower layers. Moreover, it is assumed, not without loss of generality, that only the upper layer influences the lower layer in a sequential cascade. This hierarchical order prioritizes the resolution of the local MPC problems, and their coordination, in the upper-layer subsystems, propagating their optimal control inputs downward, and iterating the process over the fixed down-streamed variables in the lower layer. The update of the input trajectories in the cooperative DMPC is performed through the Gauss-Jacobi distributed optimization method [41]. Proofs of feasibility and stability of the overall architecture are provided. The approach is tested over a walking beam reheating furnace system and a six-area power system, and validated against the DMPC formulation of [335]. In the tests, the hierarchical approach of [70] shows the ability to reduce the communication burden, avoiding the transmission of unnecessary information while ensuring system performance.

The study [66] introduces a hierarchical clustering-based MPC strategy for optimizing heat transfer fluid flow rates in solar parabolic trough plants [90]. In such systems, parabolic installations focus solar radiation on a pipe transporting a fluid, which will then be used for diverse applications. The challenge in these systems is maintaining the heat transfer homogeneous across the loops the pipe performs in the plant, regardless of meteorological conditions, in this case related to the presence of clouds. The architecture proposed has a two-layer structure. The bottom layer consists of local MPC agents controlling coalitions of loops, while the top layer dynamically clusters loops. For this, the k-means clustering algorithm groups loops with similar dynamics, determined by lumped parameters, and recursive least squares estimation [300] adapts system parameters in real-time. Moreover, the top layer accounts for variable weather conditions to assign MPC constraints to local agents so that the flow in the pipe can be restricted or increased in specific loops depending on the presence of clouds. The method allows scalability of the MPC architecture, but is sensitive to parameter estimation errors and relies on fixed cluster numbers. Simulations performed on 10-loop and 80-loop plants show significant effectiveness of the technique and minimal performance loss.

4.7.7 Input-Coupled Systems

An algorithmic partitioning approach for input-coupled systems is proposed in [343], where the objective is to derive a novel iterative DMPC strategy with a dynamic communication topology to improve the communication burden of conventional cooperative DMPC with a static communication topology. The network is assumed to be composed by a number n of coupled linear dynamics of the form

$$x_i(k+1) = A_{ii}x_i(k) + B_{ii}u_i(k) + \sum_{j\in\mathcal{N}_i}\left[A_{ij}x_j(k) + B_{ij}u_j(k)\right]. \tag{4.63}$$

By using the Kalman canonical form, the state coupling can be avoided with an appropriate selection of the new subsystem states [318], providing new input-coupled local dynamics[7] $\bar{x}_i(k+1) = \bar{A}_{ii}\bar{x}_i(k) + \bar{B}_{ii}\bar{u}_i(k) + \sum_{j\in\mathcal{N}_i} \bar{B}_{ij}\bar{u}_j(k)$. A sensitivity analysis is performed to establish the effect of the coupling variables on the optimization problem. On this basis, a threshold triggering communication between local controllers is derived. Accordingly, an algorithmic procedure determines the entries of a communication matrix at each time step, thus obtaining an event-triggered topology change for the communication networks defining the local controllers. The proposed DMPC strategy is validated for the four-tank water system [118] against cooperative DMPC with static topology, effectively reducing the communication burden.

4.7.8 Hierarchical Clustering for Input-Coupled Systems

Hierarchical clustering for input-coupled systems is proposed in [341], where a distance function induced over minimal robust positively invariant sets is used as an underlying metric for the clustering algorithm. Specifically, the hierarchical clustering of [346] is used to to design a robust Dec-MPC, as the one of [329]. The approach is iterative and defined for a given number of hierarchy levels, starting from the network considering each agent as an individual cluster. A tuning parameter $\alpha > 0$ is defined to perform the clustering. At each step, the minimum distance $d^{\min} = \min_{ij} d_{ij}$ is computed. Then, the procedure aggregates together the agents for which $d_{ij} > (1+\alpha)d^{\min}$. Some refinements are performed on the resulting clusters to ensure consistency. Then, the procedure is iterated for the next hierarchy level until one single agent representing the entire network is obtained. The partitioning approach proposed in [341] is validated by computing the size of the resulting minimal robust positively invariant sets for different clustering procedures, showing how it outperforms other strategies in maximizing the sizes of the sets. The case study is a 43 agents flow system [170]. However, the impact of the proposed partitioning on the performance of the robust Dec-MPC strategy has not been explored in the work.

4.7.9 Computational Complexity and Controllability

An algorithmic partitioning approach oriented at the minimization of the computational complexity of the resulting DMPC architecture while ensuring the controllability of the resulting subsystems is developed in [15]. This approach is motivated by previous studies oriented at the minimization of the communication cost in DMPC approaches, such as [31]. To this aim, in [15], the authors develop an algorithm for the reduction of the number of iterations $\bar{r}$ required to retrieve an (approximate) solution of a distributed optimization problem with a desired accuracy ϵ. The idea behind this approach is that by finding the partitioning that minimizes the number of iterations of the DMPC, the amount of information shared among the agents will also be minimized. However, while it is correct that the communication burden grows with the number of iterations of the DMPC strategy, it is not exact to say that minimizing the number of iterations automatically minimizes the communication burden, or computation time. In [15], the desired partitioning is obtained through the minimization of the cost function F dependent by the selected partitioning $\mathcal{P}^j$

[7] Note that this state transformation can be already considered a partitioning of the state of the network.

is defined as:

$$F(\mathcal{P}^j) = \left(\log_{\beta(\mathcal{P}^j)} \frac{\epsilon}{J(\boldsymbol{x}_{(0|k)}, \boldsymbol{u}^0_{(0|k)})} - 1 \right) \sum_{i=1}^{N_{P_a}} g(n_i, m_i, N, n_c) \tag{4.64}$$

where $f(C_i) = g(n_i, m_i, N, n_c)$ is a function of the number of states and inputs of the collection C_k, the prediction horizon N, and of the number of constraints n_c; and $J(\boldsymbol{x}_{(0|k)}, \boldsymbol{u}^0_{(0|k)})$ is the cost function for the first prediction step, evaluated with the first iteration of the control action. The minimization of (4.64) is sought using the Kernighan-Lin algorithm [165], also at the basis of the approach [12], and based on iterative node exchange. The approach is applied to the control of a simplified version of the Richmond water distribution network, Yorkshire, UK [332], using a flow-based graph representation. The simulations show how the DMPC strategy applied to different network partitionings always ensures a negligible loss in performance, while showing computation times that gradually decrease with a higher number of sets in the partition.

4

4.8 Community-Detection-Based Partitioning

4.8.1 Fundamentals and Modularity Metric

Community detection is a fundamental branch of modern network theory, and its scope is the identification of groups of elements in the network that have a higher probability of being strictly connected to each other w.r.t. other member in the network [102]. Among the methodologies for community detection, we find optimization-based, algorithmic, dynamics-based, and consensus-based approaches, as well as methods based on statistical inference, and spectral or hierarchical clustering: an extended discussion about these topics can be found in [101, 102]. Partitioning approaches based on the quality function called *modularity* belong to the broader class of methods for community-detection in graphs [101], i.e. they are clustering methodologies, often algorithmic. In this context, modularity is a metric that has been consistently used to quantify the quality of the resulting clusters, not only in network theory, but also for control systems. Several studies in the field of partitioning for predictive control use modularity as a fundamental metric. Therefore, we treat this topic separately from other partitioning approaches.

In control theory, modularity has been applied to compute the partitioning of the graph associated with a dynamical system. The method to derive this graph has been discussed in Sec. 4.3.2. However, modularity-based partitioning can also be deployed over agent-based representations of the form Sec. 4.3.3, which is a conceptually different use case. In general, for a network with a given adjacency matrix $\mathcal{A}$, and a partition into N communities $\mathcal{P} = \{C_1, \dots, C_N\}$, the modularity Q index is constructed as:

$$Q = \frac{1}{2m} \sum_{ij} \left(\mathcal{A}_{(i,j)} - \frac{k_i^{\text{in}} k_j^{\text{out}}}{2m} \right) \delta_{ij} \tag{4.65}$$

where $\mathcal{A}_{(i,j)}$ is the ij-th element of the adjacency matrix, k_i^{in} and k_i^{out} are respectively the in- and out-degree of node i in the network, m is the total number of edges, and the binary variable δ_{ij} is equal to 1 if nodes i and j are in the same community, and zero otherwise. Modularity-based partitioning approaches all focus on finding the partitioning

$\mathcal{P}$ that maximizes the modularity Q (usually, for a given number N of communities). In the remainder of this section, we will discuss how modularity-based partitioning has been used in predictive control, and provide different examples.

4.8.2 Maximization of Modularity by Iterative Bipartition of the Network

The most used methodology for modularity maximization in control has been presented in [156]. The approach is based on the construction of the modularity matrix $\mathcal{B}$, whose entries are defined as:

$$\mathcal{B}_{(i,j)} = \mathcal{A}_{(i,j)} - \frac{k_i^{\text{in}} k_j^{\text{out}}}{m} \tag{4.66}$$

Then, the partitioning approach iteratively splits the network into two communities. To this aim, a vector $\boldsymbol{s}$ with a size equal to the number of nodes in the network is defined as follows. When a split is performed, the network $\mathcal{N}$ is divided into two communities: $\mathcal{C}_a$ and $\mathcal{C}_b$. Accordingly, the i-th entry of $\boldsymbol{s}$ is defined to be equal to 1 if $i \in \mathcal{C}_a$, and -1 if $i \in \mathcal{C}_b$. The modularity associated with this new partition of the network is then:

$$Q = \frac{1}{4m} \boldsymbol{s}^{\mathsf{T}} \left(\mathcal{B} + \mathcal{B}^{\mathsf{T}}\right) \boldsymbol{s} \tag{4.67}$$

The specific partitioning algorithm used to perform the modularity maximization of the basis of the iterative division is [190], which successively divides the network into communities using approximate spectral optimization for the divisions. Fine-tuning by node shifting [243] is performed at each step to improve the partitioning quality. The limitation of this approach is that it neglects any measure of the strength of interaction among nodes and only accounts for topological information. Also, there is no guarantee that the resulting partitions will be non-autonomous systems because no distinction is made between node variables. Consequently, the algorithm may result in partitions containing only input or state nodes, which has limited applicability from the control perspective. Another aspect to consider is the potential controllability of the resulting partitions, for which structural controllability can be tested by verifying the input reachability and no dilation condition [303], but this is not explicitly performed in [156].

The paper [263] investigates the impact of system decomposition on the performance and computational efficiency of DMPC applied to nonlinear process networks [196]. The study compares the partitioning of a network obtained through community detection with intuitive partitioning given by expert subsystem selection according to energy or technical considerations. The metrics used for comparison are the closed-loop control performance and computational burden. The analysis is conducted on a reactor-separator process, where sequential and iterative DMPC formulations [72] are compared against CMPC. The empirical results obtained by applying the different predictive control strategies show that modularity-based community detection methods perform close to CMPC with significantly reduced computation time. However, the study has some limitations. The optimality of the decomposition, as always for modularity-based approaches, is not explicitly guaranteed, as modularity maximization does not necessarily imply the maximization of control performance. Additionally, the method does not consider adaptive decomposition, meaning the partitioning remains static even under changes in system conditions.

Extension of the partitioning methodology [156] to weighted graphs using the module of the partial derivatives of the dynamics around the operating points for nonlinear systems is proposed in [155]. The partitioning procedure relies on a modified version of the multiway spectral community detection algorithm [353] developed for unweighted graphs.

4.8.3 For Optimization Problem Decomposition

An algorithmic partitioning approach for the optimization problem decomposition using community detection has been proposed in [324]. The optimization problem related to DMPC considered in this work is assumed to be in a "separable" form:

$$\begin{aligned} \min_{v} \; & f_1(v_1) + \ldots + f_n(v_n) \\ \text{s.t. } & c_j(v_1, \ldots v_n) = 0, \; j = 1, \ldots, m \\ & v_i \in \mathcal{V}_i, \; i = 1, \ldots, n \end{aligned} \tag{4.68}$$

where the scalar variables in v belong to decoupled intervals, the objective function is separable, and the coupling in the problem only arises in the equality constraints. This setting is common in MPC, where in a network of agents, each has its own individual objective, and is subject to local constraints in the state and input spaces, but they are dynamically coupled with neighbors. To decompose the problem, the authors of [324] use two different graph representations. In the first, they use a bipartite graph, where variables are linked to constraints according to the existence of their partial derivatives, thus capturing their functional interaction. In the second graph, they use a unipartite representation using the variables as nodes, and the number of coupling constraints as arcs. From these two graphs, it is possible to obtain adjacency matrices, and accordingly find the partitioning of these graphs that minimizes the modularity, both for unipartite [244], and bipartite [28] representations. Modularity optimization is achieved using the Louvain fast unfold algorithm [45]. The approach is validated for control of a reactor-separator process [197, 318], with two reactors in series and a separator. The approach deployed is an ADMM-based DMPC [38], and is validated against nonlinear CMPC. The results show how the DMPC implementation can outperform CMPC for this nonlinear setting while reducing computation time by more than 50%.

Optimization problem decomposition based on modularity optimization is proposed in [298] through the use of optimality condition decomposition (OCD) [75], to overcome the assumption that the cost function of the optimization problem must be separable to decompose it. For a given non-completely-coupled optimization problem:

$$\begin{aligned} \min_{z} \; & f(z) \\ \text{s.t. } & b(z) \leq 0 \end{aligned} \tag{4.69}$$

the OCD allows the ploblem to be decomposed into N subploems, for which a relaxed

formulation [39] takes the form

$$\begin{aligned} \min_{\{z^{(i)}\}_{i=1}^{N}} \sum_{i=1}^{N} f^{(i)}(z^{(i)}) + \lambda^{(i)} h^{(i)}\left(z^{(1)},\dots,z^{(N)}\right) \\ \text{s.t. } h^{(i)}\left(z^{(1)},\dots,z^{(N)}\right) \leq 0 \qquad i \in \{1,\dots,N\} \\ g^{(i)}\left(z^{(i)}\right) \leq 0 \qquad i \in \{1,\dots,N\} \end{aligned} \tag{4.70}$$

where $z^{(i)}$ is the variable of the i-th subproblem, $\boldsymbol{h}$ is a set of complicating constraints without which the subproblems would be independent, $\boldsymbol{g}$ are the constraints resulting from the conversion of $b(\boldsymbol{z}) \leq 0$, and λ are the Lagrange multipliers. To the problem (4.70) can be associated the matrix of first-order Karush-Kuhn-Tucker condition [50]. This matrix can naturally be interpreted as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for which modularity-based community detection can be applied. Specifically, in the work [298], modularity maximization is achieved through the fast unfold algorithm [45], thus providing a decomposition of the optimization problem and consequently a partitioning of the system. The resulting OCD-DMPC approach is qualitatively validated on the quadruple-tank benchmark system [7] against other MPC strategies, and on the Barcelona drinking water transport network.

4.8.4 Frequency-Based Graph Weighting

The use of a frequency-based index to perform partitioning through community detection is explored in [339]. In this paper, the network is represented through an input-output bipartite graph, as in Sec. 4.3.4. The edges connecting I/O variables are weighted through the linearized frequency response between each pair of variables. Specifically, the integral of the magnitude of the transfer function between two variables (u_i, y_j) for a given range of frequencies $[\omega_1, \omega_2]$ is computed as:

$$\beta_{ij} = \int_{\omega_1}^{\omega_2} \frac{|G_{ij}(j\omega)|}{\sqrt{1+|G_{ij}(j\omega)|^2}} d(\omega) \tag{4.71}$$

and then a normalization is used to obtain the weights $w_{ij} = 1 - e^{-\beta_{ij}}$. This allows to retrieve a monotonically increasing weighting in the range $[0,1]$ for all the edges. The computation of the partitioning based on this weighting is performed through a modified version of Barber's algorithm [28]. Moreover, the gap metric [350] is introduced as a way to quantify the stability of the I/O functions, and incorporated into the partitioning algorithm as a quantity to be minimized along with modularity maximization. The resulting partitioning algorithm is used to deploy DMPC over two different case studies, and compared with CMPC and DMPC with partitioning computed using the conventional modularity maximization. The first experiment involves a reactor separator process consisting of two continuously stirred tank reactors and a flash separator [198]; the second is an air separation process. The empirical results show how different decompositions of the network impact the performance of the DMPC, showing that also frequency-based modularity maximization is not always the best choice, which is in line with the concept that modularity maximization does not provide by itself the best partitioning in terms of performance. Additionally, the technique proposed only works with linear systems.

4.8.5 Time-Varying Graph Representations

Exploration of a partitioning algorithm for time-varying systems is proposed in [15] where nonlinear dynamics of the following form are considered:

$$\begin{aligned} \dot{x}(t) &= f(x(t)) + g(x(t), u(t)) \\ y(t) &= h(x(t)). \end{aligned} \tag{4.72}$$

For this class of systems, an associated graph representation is constructed using as weighting for the edges the partial derivatives of the dynamics w.r.t. the variables. Specifically, denoting with an arrow an edge between variables, the corresponding weights are defined as in [173]:

$$u_i \to x_j : \left|\frac{\partial g_j}{u_i}\right|; \quad x_i \to x_j : \left|\frac{\partial f_j}{x_i}\right|; \quad x_i \to y_j : \left|\frac{\partial h_j}{x_i}\right| \tag{4.73}$$

Once all the weights are defined, the corresponding adjacency matrix A^{adj} is constructed, and accordingly, the modularity metric Q can be used for graph partitioning. The algorithm used in this case is the spectral community detection detailed in [353]. It is important to notice that if a change in the structure of the dynamics (4.72) occurs such that the weights in (4.73) are altered, then the graph associated with the network changes. This aspect is explored in the work [15] by considering the same plant in two different operating points. The case study is the benzene alkylation process using four continuous stirred tank reactors and a flash separator controlled through the DMPC strategy developed in [264], which also involves the partitioning of the process using community detection. The strategy developed in [15] shows an improvement in the performance up to 26.9% w.r.t. the one in [264].

4.8.6 Hierarchical Approach for Time-Varying Graphs

A hierarchical algorithmic approach for time-varying topologies is presented in [281, 282]. Starting from the graph representation already introduced in Sec. 4.6.7, the partitioning problem is divided into two parts: first, a selection of fundamental and indivisible systems dynamics, called FSUs, is performed algorithmically; then the FSUs are aggregated into collections, called composite system units (CSUs), for which a controller is designed. The algorithmic procedure for the selection of FSUs is motivated by the fact that the subsystems in the sense of Sec. 4.4.1 might not be given a priori. In this case, a selection of the subsystems is necessary to transform a network into a multi-agent representation. In [282], an algorithmic procedure addressing this problem is proposed. Application of the algorithmic selection of FSUs allows to obtain a network structure $\mathcal{N} = \{\mathcal{S}_1, \dots, \mathcal{S}_{N_{\text{FSU}}}\}$ from any given dynamics of the form (4.3). The second part of the partitioning strategy is an aggregative procedure for merging FSUs into CSUs, the collections constituting the partitioning. To this aim, a modularity-inspired metric is designed to capture the strength of the interaction intra- and inter-CSUs, while balancing their size. These individual

components of the metric are:

$$W_{C_i}^{\text{intra}} = \sum_{s,t \in \mathcal{V}_i} |w_i(s,t)| \tag{4.74}$$

$$W_{C_i}^{\text{inter}} = \sum_{s \in \mathcal{F}_{C_i}} \sum_{j \in \mathcal{N}_{C_i}} \sum_{t \in \mathcal{N}_s \cap \mathcal{V}_j} |w_i(s,t)| + |w_j(t,s)| \tag{4.75}$$

$$W_{C_i}^{\text{size}} = |C_i|^2 \tag{4.76}$$

where $\mathcal{V}_i$ is the set of the nodes in the set C_i, and $\mathcal{F}_i$ its frontier. Using these terms, the global metric for partitioning, named partition index, is defined as:

$$p^{\text{idx}}(\mathcal{P}) = \frac{\sum_{i=1}^{m} W_{S_i}^{\text{Intra}}}{1 + \sum_{i=1}^{m} W_{S_i}^{\text{Inter}}} + \frac{\alpha}{1 + \sum_{i=1}^{m} W_{S_i}^{\text{size}}} \tag{4.77}$$

where α is the parameter affecting the granularity, thus allowing balancing the effect of the size of the collections in the partitioning. A greedy algorithmic procedure is used to iteratively assign the subsystems S_i to the collections C_i such that at each assignment the variation $\Delta p^{\text{idx}} = p^{\text{idx}}(\mathcal{P}^{\text{new}}) - p^{\text{idx}}(\mathcal{P}^{\text{old}})$ is maximized. The approach is validated on the same random network of hybrid dynamical systems described in Sec. 4.6.7 using the same DMPC strategy. The simulation results show how the loss in performance for agents designed with algorithmic partitioning is of an additional 1% w.r.t. the ones obtained with optimization-based partitioning. However, the computation times are comparable to the ones of conventional DMPC-ADMM with 50 agents (1.75 times slower), but having the smallest computational cost among all the approaches in terms of core seconds. Therefore, the algorithmic partitioning strategy proved to be an effective alternative to the optimization-based approach in [282] and detailed in Sec. 4.6.7. The interesting aspect is that the algorithmic approach has a maximum computational complexity of $O(n_{\text{FSU}}^4)$, while the optimization-based is an NP-Hard problem. Accordingly, depending on the size complexity, time constant, and desired quality of the partitioning, the algorithmic approach can be suitable for online re-partition in case of time-varying topologies. We also note that the partition index defined in (4.77) can be used in global search optimization (genetic algorithm) to obtain a non-overlapping partitioning, as similarly proposed in [277, 278].

4.8.7 Applications and Case Studies

Application of the modularity-based partitioning methodology derived in [156] is performed in [233] for iterative DMPC of an Amine gas sweetening plant. The decomposition of the relatively small plant shows how modularity maximization is achieved when two communities are obtained, and further partitioning the system into three communities does not improve the modularity. Modularity maximization also accounts for the structural information about the plant, ensuring the existence of well-posed subsystems (i.e. subsystems for which a controller can be defined, having at least one input and one output of the original plant). No further division of the plant is proposed. The DMPC architecture is compared against CMPC, Dec-MPC, and DMPC for a different partitioning (sub-optimal in

terms of modularity). The modularity-based DMPC is the best-performing non-centralized strategy, approaching CMPC results while reducing computation times. Given the reduced size of the plant, all possible modularity-based partitions of the systems providing well-posed subsystems can be evaluated in this case; however, the procedure still relies on expert knowledge, heuristics, and inspection to be performed accurately.

The approach of [156] is deployed in [264] for both distributed control and estimation of a benzene alkylation process consisting of four continuous stirred-tank reactors, and a flash tank separator. Deploying the DMPC architecture for the selected partition provides a good approximation of CMPC results with a reduced computation burden.

A modularity-based partitioning technique has been used in [120] to deploy a DMPC strategy for perimeter control of urban traffic. The approach is structured to divide urban networks into regions for which traffic control methods based on the macroscopic fundamental diagram [109] can be implemented [9]. To this, a two-layer partitioning method is proposed in [120]. In the upper layer, congested regions are selected using the dynamic modularity metric for urban traffic introduced in [120]. These regions are compact, and a macroscopic fundamental diagram can be identified for them. However, the regions do not cover the entirety of the urban network, i.e. non-congested regions are present at their interconnection, defining a boundary. At the lower layer of the partitioning strategy, the boundary region is divided into multiple areas based on spatial proximity using the Euclidean distance, so that a boundary region exists between each two congested areas. Validation of the partitioning approach is performed by applying the DMPC strategy [169] on the case study of the road network in downtown Jinan, China. The proposed approach is validated against a fixed signal control rate, and the boundary-feedback control strategy [358], demonstrating how the proposed strategy is the most effective in reducing the total time spent on the road by the drivers, and the total accumulated delaye of the vehicles.

Modularity optimization has been used in [340] to partition a power network in the presence of photovoltaic inverters and electric vehicles, with the objective of using the charging/discharging capabilities of the latter to mitigate the curtailment of the former. In [340], a two-step Dec-MPC strategy is developed: in the first phase a modified modularity index is used for partitioning, and in the second step local MPC actions are computed in parallel. The modularity metric is modified to incorporate two ad-hoc performance indicators for power networks. The first is voltage sensitivity, which describes how voltage magnitude changes in nodes after voltage injection in other nodes. The second is the voltage regulation capacity used for reactive power compensation. The modularity is maximized through the Louvain algorithm [111]. The resulting approach is qualitatively validated on the IEEE 123 node test feeder, showing the viability of the strategy.

The paper [128] presents a graph-based hierarchical Lyapunov-based DMPC [196] framework. The control framework is based on the selection of communities performed through the multiway spectral community detection algorithm [353]. This community detection algorithm approaches the modularity maximization problem using spectral methods through a heuristic approach that can work with any number of desired communities. The approach has the same computational complexity of k-means clustering; therefore, it is attractive for its scalability. The method partitions subsystems into a relative leader-follower hierarchy by integrating community detection algorithms. The work is posed as an extension of [70] to nonlinear systems. However, no formal guarantees are given, and

the use of the interpretive structural modeling, as well as the communication strategy, are not entirely clear, contrary to its reference strategy. The proposed architecture minimizes all-to-all communication, requiring only a single inter-layer exchange per sample, reducing the computational burden. The approach is validated over a reactor-separator integrated system developed in [263].

Modularity-based algorithmic partitioning using iterative bisection [243] is also at the basis of the automatic decomposition approach used in the Shell-Yokogawa platform for advanced Control and estimation [325]. In this advanced process control technology, partitioning is performed using an equivalent graph representation of the network, with the usual definition of nodes as variables and arcs as relations. Iterative bisection is performed according to the algorithm of [243], and the resolution parameter [273] is used to limit the size of the resulting clusters. Two post-processing procedures are used to ensure the connectedness of the resulting components, and to re-balance the sets according to their sizes. Heuristics are used to define the number of clusters, and resolution. The partitioning algorithm is applied to three case studies: a crude distillation process for a refinery, a gas-to-liquid process, and a hydrocracking process, all plants with hundreds of nodes. The resulting partitions are used for the application of DMPC showing how the distributed computation of the control action can improve the time required for online optimization up to 5 times. However, the impact on the control performance of this approach w.r.t. CMPC is not assessed.

4.9 Partitioning Based on Game-Theoretical Coalition Formation

Coalitional predictive control is among the most recent formulations of non-centralized predictive control [206]. It consists of a combination of optimization-based control and game theory in which dynamical groups of agents cooperate to achieve a coordinated action to optimize some given performance criteria. At the basis of this strategy, there is the concept of *coalition formation*, explained in detail in [272], according to which agents in a network group themselves into coalitions to improve their collective outcome. In coalitional control this concept is used to obtain a distributed control strategy.

In this section, the main partitioning strategy used in coalitional predictive control will be introduced first, and then details about fundamental alternatives will be given. After, the theoretical properties of coalitional predictive control and their relation to partitioning are discussed. Various extensions and applications are presented in the remainder of the section.

4.9.1 The Concept of Coalitional Control: Predictive Control and Game Theory

Consider a network $\mathcal{N}$ constituted by $N_{\mathcal{A}}$ agents, i.e. a collection $\mathcal{N} = \{\mathcal{A}_1, \ldots, \mathcal{A}_{N_{\mathcal{A}}}\}$. A *coalition* $\mathcal{C}$ is any subset $\mathcal{C} \subseteq \mathcal{N}$ where agents in $\mathcal{C}$ cooperate. To each coalition it is assigned a *characteristic function* $v(\mathcal{C})$, mapping the coalitions into real numbers, i.e. $v : 2^{N_{\mathcal{A}}} \to \mathbb{R}$, $v(\mathcal{C}) \geq 0$. A *coalitional structure* $\mathcal{P}$ is a collection of disjoint coalitions covering the entire network, in other words a non-overlapping partitioning of the network $\mathcal{P} = \{\mathcal{C}_1, \ldots, \mathcal{C}_{N_{\mathcal{C}}}\}$. The value of the coalitional structure is the sum of the individual contributions of each

coalition:

$$V(\mathcal{P}) = \sum_{\mathcal{C}\in\mathcal{P}} v(\mathcal{C}) \tag{4.78}$$

The objective of the *characteristic function game* (CFG) [293] played by the agents, and that is considered in coalitional control, is to find the coalitional structure that maximizes the total welfare:

$$\mathcal{P}^* = \arg\max_{\mathcal{P}\in\mathcal{M}} V(\mathcal{P}) \tag{4.79}$$

where $\mathcal{M}$ is the set of all possible disjoint partitions of $\mathcal{N}$. Various methodologies can be deployed to solve this problem, as it will be presentend in the remainder of the section.

The framework of the CFG is well suited for developing distributed predictive control strategies since it is, at its core, a distributed optimization approach. One of the first works that formalizes the coalitional predictive control strategy is [95], where a large-scale system is assumed to be composed of subsystems of the form:

$$\begin{aligned} x_i(k+1) &= f(x_i(k), u_i(k)) + w_i(k) \\ w_i(k) &= \sum_{j\in\mathcal{N}_i} f(x_j(k), u_j(k)). \end{aligned} \tag{4.80}$$

Each of these subsystems is an agent $\mathcal{A}_i$, and it can participate in a coalition $\mathcal{C}_\ell$, such that $\bigcup_{\ell=1}^{N_C} \mathcal{C}_\ell = \mathcal{N}$, $\bigcap_{\ell=1}^{N_C} \mathcal{C}_\ell = \emptyset$, with N_C the number of coalitions. Each subsystem is associated with a local optimization problem:

$$\begin{aligned} \min_{\tilde{x}_{i,k}, \tilde{u}_{i,k}} \; & J_i = \sum_{j=1}^{N-1} J_s(x_i(j|k), u_i(j-1|k)) + J_f(x_i(N|k), u_i(N-1|k)) \\ \text{s.t.} \quad & x_i(k+1) = f(x_i(k), u_i(k)) + \hat{w}_i(k) \\ & x_i(0|k) = x_i(k) \\ & g_i(\tilde{x}_{i,k}, \tilde{u}_{i,k}) \le 0 \end{aligned} \tag{4.81}$$

where $\hat{w}_i$ is an estimate of the dynamical coupling of x_i with its neighboring subsystems, and $\tilde{x}_k$, $\tilde{u}_k$ are the state and input sequences defined over the prediction horizon N for a time step k. A coalition $\mathcal{C}_\ell$ is formed only if the value of the cost associated with the coalition, i.e. J_ℓ, is lower than the sum of the costs of the individual subsystems. Thus, the coalition formation condition is:

$$J_\ell^* < \sum_{i\in\mathcal{C}_\ell} J_i^* \tag{4.82}$$

In the framework of CFG, the simplest characteristic function associated with a coalition $\mathcal{C}_\ell$ is $v(\mathcal{C}_\ell) = J_\ell^*$. In this case the coalition formation problem consists in finding the optimal coalitional structure $\mathcal{P}^* = \arg\max \sum_{\mathcal{C}_\ell\in\mathcal{P}} v(\mathcal{C}_\ell) = \sum_{\ell=1}^{N_C} J_\ell^*$, with a number N_C of coalitions. This problem is known to be NP-Complete [293], inheriting the same complexity of the general partitioning problem.

The underlying principle of coalition formation described above is shared among all coalitional control strategies, and variations are present in the definition of the characteristic function, the individual payoffs, the implementation of the local MPC controllers, the

computation of ordering maps sorting agents costs, and the aggregation algorithm. In the remainder of the section, we report variations, extensions, and applications of the partitioning approach found in coalitional control literature.

4.9.2 Foundational Works

Coalitional predictive control is effectively formalized in the seminal work [95], where it is applied to energy management in smart grids, specifically to optimize local energy trade among consumer nodes with distributed generation and storage capabilities. In [95], prosumers (producers-consumers) [183] cooperate to reduce power dependence from the main grid while minimizing energy exchange costs and transmission losses among them. To overcome the computation complexity associated with the general coalition formation approach described in the previous section, the partitioning problem is addressed by looking at the coalitional structure $\mathcal{P}$ where the participation preference of each agent $\mathcal{A}$ is sorted according to their Pareto ordering. This is achieved by first using the Shapley value [301] to compute the individual payoffs of each agent $\mathcal{A}$ in each possible subset of agents $\mathcal{S} \subseteq \mathcal{N}$, that for agent $\mathcal{A}_i \in \mathcal{S}$ is defined as:

$$\phi^{\mathcal{S}}_{\mathcal{A}_i} = \sum_{C \subseteq \mathcal{S} \setminus \mathcal{A}_i} \frac{|C|!(|\mathcal{S}| - |C| - 1)!}{|\mathcal{S}|!} [v(C \cup \mathcal{A}_i) - v(C)] \tag{4.83}$$

Using the Shapley value it is possible to build a mapping $\Phi : \mathcal{N} \times 2^{\mathcal{N}} \times \mathbb{Z} \to \mathbb{R}$ for each agent in each possible coalition, i.e. at each time step k a function $\Phi(\mathcal{A}_i, C_j, k)$ is available. The function Φ provides for each agent their preferred participation order into coalitions. Accordingly, agents can autonomously organize into the coalitional structure[8] $\mathcal{P}^{\Phi}$. The dynamic coalition formation is guided by an individual payoff Φ coinciding with the energy exchange with the main grid. Simulation results illustrate how coalitional structures evolve over time, showing how coalitional trade reduces overall costs compared to grid-dependent strategies, as prosumers can access more favorable internal energy prices.

A first extension of [95] is found in [94], which proposes a coalitional predictive control strategy with self-organizing agents. The coalition formation strategy is based on a negotiation protocol allowing agents to autonomously form coalitions based on expected performance improvements and cooperation costs. In particular, the coalition formation problem is framed as a transferable utility game [60, 293, 316], where agents decide to merge or separate dynamically using a bargaining protocol. Specifically, the coalitional benefit is considered under the assumption of individual rationality, described in the following. Consider two coalitions C_1, C_2, and the value of their individual and aggregated characteristic functions, i.e. $v(C_1)$, $v(C_2)$, and $v(C_1 \cup C_2)$. Also, consider the value associated with each of the players in the coalition, denoted by $v(C_1 \cup C_2)|_{(i)}$ for $i = 1, 2$, and defined such that $v(C_1 \cup C_2)|_{(1)} + v(C_1 \cup C_2)|_{(2)} = v(C_1 \cup C_2)$. Then the merger of $v(C_1)$, $v(C_2)$ occurs if and only if the the condition $v(C_1 \cup C_2)|_{(i)} \leq v(C_i)$ holds for both $i = 1, 2$, which is known as individual rationality. The value associated with a player $v(C)$ is then considered as an economic index, a utility that can be transferred. Consequently, a bargaining procedure is designed to merge the coalitions considering that, when aggregating two coalitions, the

[8] The partitioning $\mathcal{P}^{\Phi}$ does not necessarily coincide with the optimal partitioning $\mathcal{P}^*$ in terms of global minimization of the value of the cost function $J(\tilde{x}_k, \tilde{u}_k, \tilde{\delta}_k)$ in (4.47).

value $v(C_1)+v(C_2)-v(C_1\cup C_2)$ is a surplus that can be reallocated between the remaining agents. Further details about the strategy and the stability of the coalitions are given in [94]. This strategy is applied for wide-area control of power networks [61], showing the ability of the architecture to adapt to topological changes that may arise with faults or network extensions.

Another bottom-up aggregative procedure for coalitions has been devised in [205] where a PageRank [53, 145] approach is used as the metric to guide local node exchanges among coalitions. For a graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$, the PageRank associated with each node $i\in\mathcal{V}$ is a scalar $p_i\in[0,1]$, s.t. $\sum_{i\in\mathcal{V}}p_i=1$. Given the neighborhood $\mathcal{N}_i$ of node i, its PageRank value is computed as $p_i=\sum_{j\in\mathcal{N}_i}p_j/n_j$, where p_j is the value associated with node j, and n_j its number of edges.Once the values p are known for all the nodes, a weighting of the links is performed assigning to each ϵ_{ij} a weight $w_{ij}=p_i/n_i$ (undirected arcs are handled summing the weights in both directions). Once the weighting of the graph is available, an algorithm to aggregate nodes into coalition is set up using iterative aid requests, and closed-loop performance evaluations w.r.t. a given threshold to handle the merging. The distributed computation of the PageRank is performed using the algorithm [146]. This coalitional predictive control strategy is deployed over the 16 water tanks system [207], and compared against CMPC, Dec-MPC, and the DMPC scheme [247]. The strategy proposed in [205] is the best performer in terms of optimality gap w.r.t. CMPC, after parameters calibration.

A combination of the methodologies [95] and [205] is found in [240] where a randomized method for the estimation of the Shapley value is applied. Specifically, the Shapley value defined as the vector $\phi(\mathcal{N},v)\ \forall i\in\mathcal{N}$, for the game induced over the set of agents $\mathcal{N}$ and for a characteristic function v (coinciding with the stage cost of the local MPC), is used to introduce a weighting of the links among agents, which is defined for the undirected link $ij\in\mathcal{E}$ as $w_{ij}=\phi_i(\mathcal{N},v)/|\mathcal{E}_i|+\phi_j(\mathcal{N},v)/|\mathcal{E}_j|$. To address the problem of the combinatorial explosion associated with the computation of the Shapley value associated with all possible coalitions, randomized methods [58, 146] are proposed to estimate it. In particular, using the modified definition of the Shapley value given in [58], an estimation of its value is given for a set of q samples of all possible coalitions, giving an approximation of the value, whose efficient estimate is distributed as $\tilde{\phi}_i(\mathcal{N},v)\sim N(\phi_i,\sigma^2_{\phi_i}/q)$, with bounded error. Further details about the algorithmic partitioning approach are given in [240]. The partitioning methodology is validated over the Barcelona drinking water transport network by applying coalitional predictive control and comparing it against CMPC, showing how it can outperform Dec-MPC and other decentralized control architectures.

4.9.3 Technical Extensions: Feasibility, Stability, Robustness

Theorems for the stability and recursive feasibility [227] of a coalitional predictive control formulation have been proposed in [26]. In [26], a DMPC technique [228] relying on tube-based MPC [193] is considered as the underlying control strategy for each coalition. Then, the aggregation of coalitions is achieved through a consensus procedure, where for each coalition C_i in a given state x a consensus optimization problem is defined as:

$$\min_{C_i\in\mathcal{M}} J_i(C_i,C_{-i},x)=J_i^{\text{consensus}}(C_i,C_{-i},x)+\rho J_i^{\text{power}}(C_i,x) \tag{4.84}$$

and $C_{-i}\triangleq\{C_j\}_{j\in\mathcal{N}_i}$ is the set of possible neighboring coalitions. In this optimization problem, the term $J_i^{\text{consensus}}=0$ if coalitions C_i and its neighbors agree on the current arrangement

into coalitions, and the term J_i^{power} weighted by the scalar ρ represents the effect of coalition C_i on neighbors opinions [239]. The consensus optimization is achieved through an algorithm that leverages the theory of finite exact potential games [235]. The approach is successfully validated against CMPC over a four-agent mass-spring-damper planar chain, showing that the coalitional control scheme proposed can reach states that are otherwise not feasible for CMPC.

Another extension is found in [65], where tracking of target sets is achieved. The technique is developed by deploying a tube-based MPC formulation for each coalition, thus obtaining a robust formulation of local controllers. In [65], coalitional control is used in combination with Dec-MPC. Coalitions are formed to enlarge the domain of attraction of MPC, but when sufficient, the decentralized formulation is used. The underlying partitioning strategy is hierarchical, where partitioning is executed at a slower time scale over a heuristic selection of possible communication topologies. The underlying coalitional scheme is defined by [206]. In particular, given the set $\mathcal{M}$ of all possible communication topologies, and for a partitioning $\mathcal{P} \in \mathcal{M}$, the characteristic is defined as:

$$V(\mathcal{P}, x_{\mathcal{P}}) = (x_{\mathcal{P}} - x_\Gamma)^\intercal P_{\mathcal{P}} (x_{\mathcal{P}} - x_\Gamma) + c|\mathcal{E}_{\mathcal{P}}| \tag{4.85}$$

where $x_{\mathcal{P}}$ is the aggregated state of all the coalitions at time step k, x_Γ is the Chebyshev center of the target set, $|\mathcal{E}_{\mathcal{P}}|$ is the number of communication links enabled in the partitioning $\mathcal{P}$, $c > 0$ is a scalar, and $P_{\mathcal{P}}$ is a positive definite matrix. Further details about the methodology are given in [206]. The approach proposed is validated over a 12-tracks system connected through springs and dumpers; an example also used in [285, 329], showing a good performance retention w.r.t. centralized control with significant reductions in communication costs.

4.9.4 Market-Based Partitioning

A market-based coalition formation approach applied to coalitional predictive control is introduced in [216]. The approach optimizes the heat transfer fluid (HTF) flow in a parabolic-trough solar collector field. The strategy is inspired by other market-based approaches proposed for energy networks [313], and results in a hierarchical coalitional control strategy. A parabolic-trough solar collector field is a system composed of many loops of parabolic collectors focusing heat on a trough flowed by the HTF. This fluid is thus heated and can be used for electrical energy generation. The objective of a control strategy applied to this system is to maximize the thermal power output by regulating the flow q of the HTF across the loops, where the dynamics of the plant is nonlinear and subject to disturbances caused, e.g. by the variability in atmospheric conditions. The objective function J of the plant is a quadratic sum of the power output and q contributions for each loop, where q is the control variable. The market-based coalitional strategy is implemented by defining for each agent $i \in \mathcal{N}$ in the plant, i.e. the individual loop, a utility value $U_i(\cdot) = -J_i(\cdot)$ that the agent i can supply or demand to purchase or sell a quantum of flow Δq. Accordingly, the set of agents is split into two disjoint subsets $\mathcal{L}_s$, $\mathcal{L}_d$ of supply and demand loops, with respective utilities. Then, the utility is computed and classified according to the two groups for each agent or coalition. This way, the requests can be sorted in descending and ascending order for demand and supply, and trades are performed according to this matching. The utility gain of each agent participating in a specific trade

is equal to the difference between the demand and supply utilities divided by two. The hierarchical coalition formation procedure is then implemented starting from the coalition formed by individual agents, and runs periodically according to a fixed time step bigger than the control step. Heuristics ensure the terminability of the algorithm. The overall control strategy is validated on the model of the real-world collector field ACUREX, located in Plataforma Solar de Almería [105, 107] composed by 10 loops, and its scaling to 100 loops. Comparison strategies include PI control, two different CMPC strategies, and the control strategy based on loop-pair clustering devised in [215]. According to the simulation results, the market-based coalitional predictive control is the best-performing strategy with a gain of 12.51% w.r.t. PI control, outperforming also the CMPC implementation with 0.37%. Additionally, an analysis of the computational burden is performed. In practice, the CMPC strategy is not deployable because its computation time exceeds the operating time step of the plant. On the contrary, market-based coalitional control is fast enough to be potentially scaled up to a plant of about 300 loops while maintaining the same performance gains.

Feedforward Neural Networks (NNs) [98] are used in [218] to reduce the computational complexity of the coalition control algorithm. The market-based hierarchical formulation introduced in [216] is considered as reference strategy, and the same application to parabolic-trough solar collector field is considered. Specifically, in [218] sets of NNs are used with two different scopes in cascade. The first set of NNs uses information about states and disturbances to approximate the values of the utilities of supply and demand agents. These are used to implement the market-based coalition formation. Then, a second set of NNs, using the same information and considering the coalition obtained, approximate the value of the HTF flow for the coalitions, that can group at most three loops. The coalitional controller is validated on the parabolic-trough solar collector field case study ACUREX. The results are compared with the nonlinear coalitional controller developed in [216]. The NN-based coalitional controller [218] shows a performance that is comparable with the one obtained in the nonlinear implementation [216], but providing a considerable reduction in the computation time needed to compute the control action and the partitioning of the network with a reduction up to the 99% w.r.t. the time required in the NLin-MPC implementation. The drawbacks of this strategy arise from the defining technical characteristics of NNs, which include the necessity of rich enough data to perform the training, the inability to provide suitable outputs when the operating conditions of the plant are distant from the training set, and the lack of guarantees for constraint satisfaction.

4.9.5 Further Extensions

Predicted topology transition is proposed in [217] as an evolution of the work in [95]. The method extends Coal-MPC by incorporating a transition horizon variable, which optimizes the timing of topology changes over the prediction horizon. Unlike previous coalitional control methods that switch coalition structures instantaneously, this approach gradually transitions between topologies, allowing agents to anticipate and optimize their control actions accordingly. The strategy also belongs to hierarchical coalitional control, where the upper layer, working at a lower rate, is designed to obtain the desired coalition and the transition horizon. This problem is formally a mixed-integer optimization program for which the solution space is reduced by defining a heuristic on the possible topology. The strategy in [217] is applied for the control of an eight-tanks water system showing how

the proposed approach can reduce communication and coordination costs of coalitional schemes while maintaining performance close to CMPC.

Pairwise clustering is proposed in [215] where the HTF flow coalitional predictive control for the parabolic-trough solar collector field ACUREX is considered. The control approach is hierarchical, and in the upper layer the partitioning of the network occurs at each time step. Specifically, at each time step k, the measurement of the flow rate in all loops at the previous time step is collected into a vector $\boldsymbol{q}_{k-1}^{\text{measured}}$. This vector is then sorted in ascending order, giving $\boldsymbol{q}_{k-1}^{\text{sorted}}$. Then, the partitioning of the plant is obtained by coupling together the first and last elements of $\boldsymbol{q}_{k-1}^{\text{sorted}}$ and removing them from the vector until no further assignments are possible. This direct approach to partitioning is motivated by the fact that lops with a deficit of flow rate can benefit from those with excess flow. The approach has been proven to outperform Dec-MPC, approaching CMPC performance while significantly reducing computation time.

The problem of resource sharing under partitioning is addressed in [292], which considers as case study the parabolic-trough solar collector field ACUREX. For this plant, a partition partitioning $\mathcal{P} = \{\mathcal{C}_1, \ldots, \mathcal{C}_{N_p}\}$ is assumed to be given, e.g. using one of the techniques in [215, 216, 218]. Then, it is necessary to define the allocation of shared resource, which, for the specific case study, is total HTF flow. The problem of distributing the shared resource is solved using a population-dynamics-assisted resource allocation strategy [32, 211], specifically a Smith population dynamics with carrying capacities [30]. Following the hierarchical coalitional control methodology introduced in [216], the resource allocation (for a fixed partitioning) is performed at a slower time scale. The approach is validated on a 100-loop implementation of the parabolic-trough solar collector field ACUREX and compared with CMPC. The results show a negligible loss in performance while significantly reducing the computation time required to retrieve the control action.

4.9.6 Partitioning for Input-Coupled Systems

The use of coalitional predictive control for systems with coupled input dynamics is found in [214], where the problem of controlling next-generation cellular networks [19] is considered. The dynamics of these system can be formulated as an input-coupled agent representation:

$$x_i(k+1) = A_i x_i(k) + \sum_{j \in \mathcal{C}_i} B_{ij} u_{ij}(k) + \omega_i(k) \tag{4.86}$$

where $u_{ij} = -u_{ji}$, and ω_i is a disturbance. The underlying coalitional formation approach is a hierarchical methodology of the form [95], where in the upper layer a new coalitional structure is assigned according to a fixed time step longer than the control sampling time. In this case, the computational complexity of evaluating the best topology is reduced by considering as candidate successors only the allocations $\mathcal{P}^{\text{next}}$ that have a Hamming distance of one from the current configuration $\mathcal{P}^{\text{current}}$, i.e. they differ from only one link allocation. The proposed strategy is applied to the case of a network of 37 base stations to optimize the number of served users and energy consumption. The approach is validated against the more traditional best-signal-level approach [100], and decentralized and CMPC. Results show significant improvement of all the predictive control strategies

w.r.t. the traditional approach, where coalitional control is the closest to CMPC in terms of performance while reducing the communication burden.

A further extension of coalitional predictive control for coupled input dynamics has been proposed in [290]. In this work, couplings in the inputs among agents are decomposed into private and public variables, a feature detailed in [179]. This approach is used because it allows more flexibility in the computation of the control action w.r.t. robust approaches as tube-based MPC that is more conservative. The resulting architecture is validated using an eight-tank system coupled in the input, showing how varying implementation parameters allows to balance the communication burden with the performance loss.

An extension of [290] is found in [291], where a robust tube-based formulation of the controller is proposed [228]. Additionally, in [291] the presence of communication links is event-driven, i.e. communication links are activated only if scaling factors exceed predefined thresholds that allow to establish a trade-off between performances and communication burden. The approach is validated using an eight-tank water system against centralized and Dec-MPC. The simulation results show that coalitional control can outperform Dec-MPC while approaching CMPC performances with a reduction of 83% in terms of communication cost.

A further advancement in coalitional control for input-coupled dynamics is achieved in [212], where a robust strategy allowing plug-and-play capabilities is devised. The approach is based on an evolution of public and private factors introduced in [329] and already employed in [290]. Validation of the approach is performed through the control of a four-truck system in a coupled chain configuration as also tested in [329]. A fifth truck is added during the simulation to show the plug-and-play capabilities.

4.9.7 Other Applications

In this section, we report applications of the coalitional predictive control schemes discussed above to case studies that have not been presented already, specifically: the control of irrigation canal, freeway transportation, vehicle platooning, and cyber-physical systems.

The first known contribution in coalitional predictive control is [97], where the problem of controlling an irrigation canal is addressed. The aim of the strategy is to optimize water distribution by dynamically adjusting coalitions of control agents to balance control performance and communication cost. The framework is hierarchical: in the top layer, the partition of the system into coalition is achieved through topology optimization, where the optimal topology is selected from a predefined set of possible topologies. Decentralized feedback gains are associated with each topology, and the solution of an LMI problem guides the partition selection. Then, at a lower level, Dec-MPC is applied. Coalition formation and local optimization work at different time scales. The control methodology is validated through the SOBEK hydrodynamic simulator [311] on a model of the Dez irrigation canal [144], and compared against CMPC showing suboptimal but adequate performance, without the need of a complete communication topology.

A hierarchical formulation of coalitional predictive control has also been applied to nonlinear systems in [63]. In particular, this study focuses on freeway traffic control through ramp metering and variable speed limits [256, 257]. The solution proposed in [63] consists of a two-level structure: a top layer forms the coalitions, and at the bottom level, a DMPC strategy is deployed for the resulting coalitions, specifically feasible cooperation-

based MPC [334] with Genetic Algorithm solver (GA) [112]. Moreover, the two layers operate at different time scales, with the top one being slower, allowing more time to solve the coalition formation problem. The study proposes as a potential solution to the coalition formation the bargaining procedure based on the Shapley value [95, 240], or the PageRank method [205]. To simplify the problem, only a limited set of possible coalitions is considered. The approach is extensively validated against Dec-MPC, and feasible cooperation-based MPC on a 15 km freeway segment, with multiple ramps, and speed-limiting devices. The results show a reduction in communication and coordination costs.

An application of coalitional predictive control to cyber-physical systems [85, 188] with chain architecture is proposed in [221]. The key feature of this architecture is that the system first operates according to the non-cooperative DMPC strategy [295], and when the feasibility of the solution fails, the system will switch to the coalitional predictive control formulation [223]. The switch occurs in cascade, triggered by one agent and propagating to its neighbors. Here, coalition formation is purely aggregative. The procedure is applied to a four-agents system, showing that when the local feasibility of non-cooperative DMPC is lost, then the application of coalitional predictive control can still provide satisfactory performance.

Vehicle platooning is the application considered in [222] for the robust coalitional control strategy of [221]. The approach is tested on a four-car platoon detailed in [359], and string stability analysis [89] is performed. The simulation shows how dynamic coalition formation stabilizes the platoon's operation with reduced communication. The work [225] is proposed as an alternative approach to [221, 222] for coalitional control of vehicle platoons, distinguishing itself by the ability of individual agents to aggregate into coalitions autonomously. This objective is achieved by periodical evaluation of the string stability index [89]. The approach is validated on a four identical vehicles platoon under three different testing conditions. The results show that an inversely proportional relationship exists between performance and string stability.

An eight-tank process is used as a case study to perform a comparative performance analysis between DMPC and coalitional control in [224]. In the paper, two non-cooperative DMPC formulations, one using a state-space model and the other an input-output model, are used to validate the performance of the coalitional control strategy based on a matrix gain feedback controller obtained through a gradient-based optimization previously introduced in [226]. The Coal-MPC methodology allows the switch between decentralized and distributed communication topologies according to performance satisfaction. This switching Coal-MPC method shows results that are comparable with the non-cooperative DMPC strategy while allowing for a reduction in the communication burden.

4.10 Heuristic Partitioning

Partitioning for Dec-MPC of wide-area power systems is investigated in [150]. The technique is heuristic and based on the use of the modal participation matrix that highlights the effects of each generation on each dominant mode in low-frequency oscillations. The partitioning technique allows overlapping partitioning, giving both the nature of the dynamical couplings and the use of a DMPC strategy [5]. The approach is applied to the Northeast Power Coordinating Council nonlinear power system model [287], comprehending 48 electrical machines and 140 buses, showing the performance and the resilience of the

network for two different partitionings compared to centralized control.

Partitioning for wind farms is proposed in [348], where a HMPC strategy is proposed. The partitioning strategy is performed on the highest level of the hierarchy every 15 minutes. Based on a forecast of the wind characteristics for the next 20 minutes, an optimization strategy is deployed to cluster the wind turbines in one of 12 categories based on the possible load operating conditions the turbines can experience. The proposed HMPC strategy was validated over a modified version of the IEEE One Area RTS-96 network [117], and compared with conventional dispatch and schedule allocation algorithms, achieving significantly better performance.

A strategy for partitioning vehicles platoons is implemented in [199], with the objective of deploying a noniterative two-level DMPC architecture ensuring closed-loop stability for an optimization problem with coupled cost functions and constraints. The partitioning strategy is based on the assumption that the cooperation set of vehicles $\mathcal{V}$ can be divided into groups that belong to two main conceptual categories, i.e. dominant and connecting clusters. The algorithmic partitioning allows vehicles to perform the operations of joining and leaving a platoon on the basis of this group classification. The DMPC strategy is then designed around this partitioning approach, ensuring stability and feasibility. Validation of the approach is performed for a platoon of four vehicles, and compared against CMPC, showing minimal loss in performance.

A strategy for event-triggered partitioning of microgrids is developed in [11], where the economic dispatch problem for energy production is addressed. The power network is considered to be constituted of microgrids that are considered self-sufficient systems, i.e. they do not exchange energy with their neighbors in nominal operating conditions. However, if this generative autonomy is lost, re-partitioning of the network is triggered, leading to a new definition of the microgrids. This re-partitioning is performed through a communication protocol, which evaluates the best node exchange among the microgrids that minimizes the individual outcomes in economic terms, while ensuring self-sufficiency. The approach is validated on the PG&E 69-bus distribution network. The simulation results show that during peak hours all microgrids should join into a single agent to satisfy the demand, whereas during off-peak hours they can split into multiple coalitions.

The paper [139] proposes a distributed Switching Model Predictive Control (SMPC) strategy for quadrotor UAV swarm aggregation incorporating collision avoidance. Teams of UAVs are selected using a clustering strategy, and local controllers solve the SMPC problem sequentially [72]. The clustering approach is based on the sphere packing problem [77]. A cluster of UAVs is selected according to the positions of UAVs in space [108], assuming these are always available. In the sphere packing problem, the objective is to find the arrangement of non-overlapping spheres so that they occupy the largest possible fraction of space. Solutions are available in the literature for this problem [77]. The approach is validated with a group of 150 UAVs, using both centralized and distributed control strategies for the aggregation. The proposed distributed SMPC can achieve comparable aggregation performance w.r.t. its centralized counterpart while drastically reducing computation time.

4.11 Applications and Case Studies

In this section, we report a classification that relates the partitioning methodologies found in the literature to the systems used for their validation. In the resulting Tab. 4.6, for each

application system, partitioning methodologies are classified according to Fig. 4.4. When possible, we also provide references to more standardized test cases for their direct use in the development of further strategies.

From Tab. 4.6, we note that many works have been developed for power systems. However, if we consider standard generation and transmission systems, no specific case study has been consistently used to derive partitioning techniques. Therefore, it is difficult to quantitatively compare different works. An exception in this sector is the parabolic-trough plant ACUREX [105, 107], for which many Coal-MPC strategies have been developed.

Water network control is another field that has seen extensive application of partitioning strategies. In this case, we distinguish between water-tank systems, which are usually small-scale test cases used to validate the viability of the approaches, and large-scale water distribution networks, among which the Barcelona drinking water transport network [252] is surely the most commonly used test case among different NC-MPC approaches.

Chemical systems have been the subject of deep studies regarding partitioning, given the complexity of the associated dynamics. We report the presence of many system configurations involving CSTRs and separators, e.g. [25, 197, 318, 333] among others. Also, in this case, no single benchmark system has been used consistently in the literature; rather, there are many different similar configurations that complicate the process of direct comparison of partitioning strategies.

For wind farms, we also report the presence of different studies in partitioning, with various topologies, turbine models, and operating conditions.

Several other applications are reported in Tab. 4.6, all used in the development of a specific partitioning methodology for the application of non-centralized control. Especially for transportation networks, we observe a notable lack of studies in partitioning for NCen-MPC of urban traffic, freeway transportation, and railway networks [202]. The other case studies are, in general, smaller systems that can be used for the development of strategies, but do not stress the scalability of the approaches.

Several other large-scale application fields can be considered for studies in partitioning, such as swarms of mobile robots or autonomous maritime vehicles [357], automated agricultural systems, district heating [44], satellite constellations [79], and advanced industrial processes [106]. Some of the applications listed can be found, for example, in the recent work [260] about the design of large-scale systems, or in the set of benchmarks proposed in [209].

Table 4.6: Application fields of the partitioning techniques for NCen-MPC, classified by sector. When available, benchmark systems have been reported.

Sector	Specific application	Partitioning techniques
Power systems	Six-area power system	[70]
	Smartgrids, 8 (check) prosumers: [183, 254]	[95]
	Wide area power network: [61]	[94]
	The EEA-ENB: [276, 280]	[277, 278]
	PG&E 69-bus distribution network	[11]
	IEEE 118-bus	[180]

	IEEE 123 node test feeder	[340]
	Nonlinear power system: [287], 48 machines, 140 buses	[150]
	Parabolic-trough plant: ACUREX model, 100 loops [105, 107]	[66, 215, 216, 218, 292]
Water systems	4-tanks system: [7]	[298, 343]
	8-tanks system	[217, 224, 290, 291]
	16-tanks system: [207]	[205, 247]
	Barcelona drinking water transport network: [252]	[31, 240, 249, 250, 298]
	Shanghai water distribution network	[354]
	Richmond water distribution network: [332]	[15]
	Dez irrigation canal: [144, 311]	[97]
Chemical systems	2 CSTR series: [25, 333]	[128, 161, 327]
	2 CSTR series and flash tank separator: [72, 197, 198, 318]	[263, 286, 324, 339]
	Tennessee Eastman problem: [88, 203], five operation units	[345]
	Benzene alkylation process: 4 CSTR and flash tank separator	[15, 264]
	Amine gas sweetening plant	[233]
	Air separation process	[339]
Wind farms	12-turbine wind farm	[355]
	20-turbine wind farm, NREL 5-MW	[195]
	25-turbine farm, 1.5 MW	[68]
	42-turbine farm, NREL-5MW: [158], SimWindFarm [119]	[306]
	IEEE One Area RTS-96 network: [117]	[348]
Transportation systems	4-vehicles platoon: [359]	[199, 222]
	Urban transportation network: [82], 8 intersections	[67]
	Jinan road network	[120]
	15 km freeway stretch: [231], METANET model	[63]
Mechanical systems	Mass-spring-damper chain, 4 elements	[26]
	(4+1)-tracks, connected with springs and dumpers: [329]	[212]
	12-tracks, connected with springs and dumpers: [285, 329]	[65]
Smart buildings	8 rooms temperature regulation	[356]
	20 thermal zones control: [62]	[17]
Abstract networks	43 agents flow system: [170]	[341]
	Random 50 systems, modular 64 systems, hybrid	[281, 282]
Railway networks	Dutch railway network: [166]	[167]
Telecommunication systems	Next generation cellular networks: [19]	[214]
Industrial plants	Walking beam reheating furnace system	[70]
Process plant	Refinery: gas-to-liquid process, hydrocracking process	[325]
Aerial vehicles	Group of 150 UAVs	[139]

Cyber-physical systems	4-agents chain	[221]

4.12 Conclusions and Future Work

This survey presents the first systematic classification and in-depth analysis of partitioning techniques for non-centralized predictive control. The scope of this work is both to unify the approaches currently present in the literature under a single framework, and to lay solid methodological foundations for future developments.

These objectives are achieved through the novel contributions of this work, which we summarize in the following. First, we introduce a formal reformulation of the partitioning problem in terms of mixed-integer programming, showing how, in the context of predictive control, the problem requires the solution of a bi-level optimization program, where network control performance is the cost functional of the partitioning problem. This aspect is at the basis of the complexity of network partitioning for control. Developing this framework, we introduce the concept of predictive partitioning, which uses predicted topology behavior to obtain the optimal network partitioning over the prediction horizon. Given the inherent NP-hard nature of these problems, their optimization-based solution would be prohibitive in real time; therefore, developing such a framework using greedy or heuristic algorithms or data-driven approaches would be advisable. Moreover, we introduce the concept of multi-topological network representations, which can serve as a basis for applying partitioning methodologies on networks whose topology and dynamical coupling are driven by different factors, such as events, time, network dynamics, or stochastic phenomena. Additionally, we provide a systematization of the key performance indicators to assess the quality of a partitioning for network control. On this basis, we establish an evaluation methodology that allows the direct comparison of different partitioning strategies. Such an approach can be the basis of further systematic development in this field, providing solid quantitative metrics for performance assessment.

In addition, this survey proposes several other ways to analyze and organize the literature in partitioning for predictive control. We start by presenting a systematization of network equivalents based on graphs. Then we introduce a classification of the partitioning techniques based on five main classes: optimization-based, algorithmic, community-detection-based, game-theoretic-oriented, and heuristic partitioning. For each class we discuss its level of optimality, scalability, complexity of computation and implementation, technical requirements, and other specific features it might exert. Further we introduce a functional sub-classification of the partitioning techniques, introducing cross-methodological partitioning objectives. We conclude the survey by discussing the known applications of the partitioning techniques proposing, when possible, reference systems for further developments and comparison.

Future work in the field of partitioning for non-centralized predictive control should focus on the following areas. First, there is space for further practical and theoretical developments in the time-varying (and hierarchical) partitioning approaches, especially considering predictive partitioning. Methodological approaches in this direction should also explore the use of data-driven, evolutionary, or reinforcement learning techniques to obtain the partition, which are strategies rarely deployed so far. Considering instead the theoretical developments, only the framework of coalitional control currently offers solid guarantees of satisfying the properties of feasibility, stability, and robustness when

partitioning is involved, with few studies addressing these issues in general. Therefore, such properties might be established for time-varying partitioning approaches under different non-centralized control frameworks. The use of unified evaluation metrics should be extended to allow for cross-disciplinary and cross-methodological evaluation, with centralized model predictive control as a reference strategy to always include in the study. Finally, we stress that there is a limited selection of large-scale application benchmarks, which include at least 10 000 agents with different connection topologies for the validation and scalability assessment of current strategies. Future work should focus on addressing the indicated aspects to reach a level of sophistication for the partitioning strategies such that they can adapt online to topological changes while ensuring the stability of the network, the feasibility of the control actions, robustness with respect to unexpected events, and minimal losses in terms of global optimality.

4.A Analytical Classification Table

In the following table Tab. 4.7, we report the references presenting the partitioning strategies that have been investigated throughout the survey. They are listed in chronological order, which allows us to further understand the order of development of the techniques. Additionally, we report the control methodology that has been deployed in the study, essential details about the partitioning method developed, and the application considered for the validation of the overall architecture.

Table 4.7: Analytical Classification Table

Work	Year	Control Method	Partitioning method	Application
[250]	2011	H-Dec-MPC	Graph-partitioning-based ordering algorithm (GPB)	Barcelona DWN
[249]	2012	H-Dec-MPC	Nested epsilon decomposition	Barcelona DWN
[97]	2014	H-Coal-MPC	Coalition formation based on topology optimization from a predefined set	Irrigation canal networks
[247]	2015	Dec-, D-, and H-MPC	MI optimization partitioning	16 tanks water system
[161]	2015	Dec-NLin-MPC	Algorithmic partitioning	Two-reactor (CSTR) chain followed by a flash separator with recycle
[345]	2016	DMPC	Genetic algorithm minimization of input-output coupling between subsystems	Chemical plant: Tennessee Eastman problem. Five operation units: a reactor, a condenser, a compressor, a separator, and a stripper.
[167]	2016	DMPC	MIQP optimization for constraints decomposition	Dutch railway network
[263]	2017	CMPC, iterative and sequential DMPC	Community detection through modularity maximization	Reactor-separator process
[95]	2017	Coal-MPC	Game theoretic coalition formation based on Shapley value	Smart grids
[205]	2017	Coal-MPC	Coalition formation based on an algorithm to handle aid requests sorted using distributed PageRank	16 tanks water system
[94]	2018	Coal-MPC	Coalition formation based on bargaining procedure and TU-games	Wide-area control of power grids
[356]	2018	Dual mode DMPC	Algorithmic partitioning based on coupling degree	Building thermal management: eight rooms

[327]	2018	DMPC (noncooperative and iterative)	Relative Time-Averaged Gain Array (RTAGA)-based algorithmic modularity maximization over weighted IO bipartite graph using fast unfold	Reactor-separator process: 2CSTRs
[324]	2018	DMPC-ADMM for nonlinear systems	Community-based decomposition of the optimization problem based on bipartite and unipartite representations, and fast unfold algorithm	Reactor-separator process: 2CSTRs
[286]	2018	Linearized cooperative and non-cooperative DMPC for nonlinear systems	Algorithmic partitioning based on variables matching and controllability check	Reactor-separator process: 2CSTRs
[150]	2018	Dec-MPC	Heuristic partitioning based on ad-hoc performance index (modal participation matrix)	Northeast Power Coordinating Council nonlinear power system model
[233]	2018	DMPC (iterative)	Modularity-based partitioning (iterative division)	Amine gas sweetening plant
[240]	2018	Coal-MPC	Coalition formation based on estimation of Shapley value and randomized methods	Barcelona DWN
[354]	2019	Enhancing DMPC	Data-driven partitioning using k-Shape	Shanghai WDN
[348]	2019	HMPC	Heuristic partitioning (optimization-based)	Modified IEEE One Area RTS-96 network with wind turbines
[199]	2019	HMPC	Heuristic partitioning (algorithmic based on dominant and connecting clusters)	Four vehicles platoon
[264]	2019	DMPC	Modularity-based partitioning (iterative division)	Benzene alkylation process: four continuous stirred-tank reactors, and a flash tank separator
[31]	2019	DMPC based on density-dependent population games	Multiobjective optimization, computed through distributed algorithm for graph partitioning	Barcelona DWN
[120]	2019	DMPC for perimeter control	Modularity-based paritioning based on dynamic traffic estimation	Road network in downtown Jinan, China
[70]	2020	Cooperative DMPC, over a sequential hierarchical down-stream of solutions	Hierarchical interpretive structural modeling (ISM)	Walking beam reheating furnace system, six-area power system
[343]	2020	DMPC (Cooperative)	Algorithmic partitioning based on treshold given by coupling sensitivity analysis	Four-tanks water systems
[306]	2020	H-NCen-MPC	MIP optimization using ad hoc indicator (wake effect)	42 turbines farm (NREL-5MW)
[195]	2020	HMPC	Frequency-based fuzzy c-means algorithmic partitioning	20 turbines farm (NREL-5MW)
[214]	2020	Coal-MPC	Hierarchical time-varying	Next-generation cellular networks with 37 base stations
[26]	2021	Coal-MPC	Coalition formation based on consensus optimization and potential games	Mass-spring-damper planar chain
[63]	2021	H-Coal-MPC	Coalition formation based on bargaining procedure and TU-games, or PageRank method	Freeway transportation network, METANET model
[217]	2021	H-Coal-MPC	Coalition formation based TU-games, and mixed-integer selection of the coalitions with predicted topologies	Eight tanks water system
[221]	2021	Coal-MPC and DMPC	Coalition formation based on cooperative game	Theoretical four agents chain system

[215]	2021	H-Coal-MPC	Loop-pair clustering	Parabolic-trough solar collector fields with 100 loops
[298]	2021	DMPC based on optimality condition decomposition (OCD)	Modularity-based partitioning of the optimization problem	Quadruple-tank benchmark; Barcelona DWN
[17]	2021	Dec-MPC	MI optimization, robust and stochastic	5 and 20 zones thermal buildings
[11]	2021	Dec-MPC for economic dispatch	Heuristic partitioning based on communication protocol (algorithmic)	PG&E 69-bus distribution network
[65]	2022	Coal-MPC and Dec-MPC	Coalition formation based on cooperative game and invariant sets	12 tracks system
[222]	2022	Coal-MPC and DMPC	Coalition formation based on cooperative game	Autonomous vehicle platooning
[290]	2022	Coal-MPC	Coalition formation based on private and public factors	8 tanks input-coupled water system
[216]	2022	H-NLin-Coal-MPC	Market-based coalition formation strategy	Parabolic-trough solar collector fields with 100 loops
[340]	2022	Dec-MPC	Modularity-based partitioning using ad-hoc performance indicators	IEEE 123 node test feeder
[180]	2022	HMPC	k-way partitioning using METIS on a flow graph	IEEE 118-bus
[68]	2022	HMPC	k-means clustering for wake-effect interaction minimization	25 turbines farm (1.5 MW)
[67]	2022	Dec-MPC	Binary quadratic programming (BQP)	Urban traffic network with 8 intersections
[66]	2023	DMPC, ADMM- or ALADIN-based	k-means clustering	Solar parabolic trough plants
[128]	2023	Lyapunov-based DMPC	Hierarchical multiway spectral community detection	Reactor-separator process
[139]	2023	Distributed Switching MPC	Sphere packing clustering combined with MPC	Quadrotor UAV swarm control
[212]	2023	H-Coal-MPC with PnP capabilities	Coalition formation based on invariant sets and dynamic scaling factors	4 + 1 trucks system
[218]	2023	H-NLin-Coal-MPC based on neural networks	Neural-networks-based market-based coalition formation strategy	Parabolic-trough solar collector fields with 100 loops
[224]	2023	Coal-MPC with switching topologies	Coalition formation based on cooperative game	8 tanks water system
[291]	2023	Coal-MPC	Coalition formation based on private and public factors	8 tanks input-coupled water system
[292]	2023	H-Coal-MPC	Arbitrary partitioning	Parabolic-trough solar collector fields with 100 loops
[339]	2023	DMPC	Modularity-based partitioning using frequency metric, and gap metric	Reactor separator process (2CSTR and a flash separator); and air separation process
[68]	2023	HMPC	k-means clustering (crowd search) using a set of key performance indicators	12 turbines farm
[341]	2023	Dec-MPC	Agglomerative hierarchical clustering based on minimal robust positively invariant sets	43 agents flow-based network
[325]	2023	DMPC	Modularity-based partitioning (iterative division)	Crude distillation process for a refinery, gas-to-liquid process, and a hydrocracking process
[225]	2024	Coal-MPC with switching topologies	Coalition formation based on string stability condition	Autonomous vehicle platooning
[15]	2025	DMPC	Algorithmic (Kernighan-Lin) partitioning using computational complexity metric	Richmond water distribution network; Barcelona DWN

[155]	2025	DMPC	Spectral community detection for modularity based on time-varying graph representation	Benzene alkylation process: 4CSTR, and a flash tank separator
[282]	2025	DMPC-ADMM for hybrid systems	Bi-level partitioning; algorithmic selection of system units, and algorithmic or optimization-based (BQP) partitioning; balancing intra- and inter-agent interactions, with granularity parameter	Modular network with 64 agents, random network of hybrid systems with 50 agents

5

Temporal Logic Control of Nonlinear Stochastic Systems with Online Performance Optimization

"In such seconds of decision entire futures are made."

Dan Simmons, Hyperion, 1989

The deployment of autonomous systems in safety-critical environments requires control policies that guarantee satisfaction of complex objectives. These systems are commonly modeled as nonlinear discrete-time stochastic systems. A popular approach to computing provably-correct policies is to construct a finite-state abstraction, often represented as a Markov decision process (MDP) with intervals of transition probabilities (IMDP). However, existing abstraction techniques compute a single policy, thus leaving no room for online cost/performance optimization, e.g., of energy consumption. To overcome this limitation, we propose a novel IMDP abstraction technique in which every abstract action corresponds to a set of inputs for the original system. Hence, an abstract policy leads to a set of provably-correct policies for the concrete system, each of which satisfies the control objective with at least the same probability as the abstract policy. We can thus search through this set of verified policies with an online control algorithm, namely model predictive control (MPC), to minimize a desired cost function online, while retaining the guaranteed satisfaction probability of the entire policy set. Our experiments demonstrate that our approach yields better control performance than state-of-the-art single-policy abstraction techniques, with a small degradation of the guarantees.

This chapter is based on [275].

5.1 Introduction

Modern autonomous control systems, such as unmanned drones and robotic systems, can be generally described by nonlinear stochastic dynamical systems [175, 230, 312]. These systems must meet stringent requirements across multiple facets of their behavior. First, systems must be capable of performing complex tasks expressed as *logical specifications* [24, 34]—for instance, a robot must move an object from location A to B while avoiding obstacles. Second, systems must optimize for other key performance indicators modeled as a *cost* (or *reward*) *function* [16, 99], such as minimizing the energy consumption or time to complete a task. To deploy autonomous systems in real-world environments, automated techniques for synthesizing control policies that provably meet the first type of requirement, whilst optimising the latter, have now become imperative.

Existing policy synthesis techniques, however, largely either focus on satisfying logical specifications or on optimizing a cost function. For example, *abstraction-based approaches* [23, 125, 181, 186] arguably represent the state of the art for computing policies that provably satisfy complex logical specifications—e.g., in *linear temporal logic* (LTL) [261]—under stochastic uncertainty and nonlinear dynamics. However, the resulting policy typically cannot be modified without losing its correctness guarantees. On the other hand, common control methodologies such as *model predictive control* (MPC) [230, 271] excel at online cost minimization but generally cannot provide formal guarantees about the probability of satisfying complex tasks (such as those expressed as logical specifications). As a result, there is a lack of general methodologies that provide guarantees on the satisfaction of complex specifications, while also optimizing a cost function. This gap limits the deployment of high-performance autonomous systems in safety-critical environments.

5

Our approach In this chapter, we address this gap by proposing a novel integration of abstraction-based policy synthesis techniques with online MPC-based optimization. Focusing on systems evolving in discrete time, we consider the following problem:

> **Problem:** Given a stochastic dynamical system, a logical specification with a threshold probability λ, and a cost function J, compute a policy such that the probability of satisfying the specification is at least λ and that minimizes the cost function J.

We split this problem into two parts, as shown in Fig. 5.1. First, we construct a sound finite-state abstraction of the dynamical system, represented as a *Markov decision process* (MDP) with intervals of transition probabilities—called an *interval MDP* (IMDP) [147, 246]. As a novel feature, our abstraction procedure allows computing a *set of policies* for the original system, each of which satisfies the logical specification with the threshold probability λ. (By contrast, existing abstraction procedures yield only a *single* policy.) We then implement an online MPC algorithm that minimizes the cost function J over the subset of control inputs enabled by the set of policies from the abstraction. As a result, the closed-loop dynamical system is guaranteed to satisfy the logical specification with probability at least λ, while also minimizing the cost function J. We now discuss the key features of our abstraction and MPC algorithm.

Existing abstraction methods derive IMDP models in which each abstract action corresponds with a *single* control input for the concrete dynamical system. As a result, selecting an abstract policy determines *all* control choices for the dynamical system, rendering

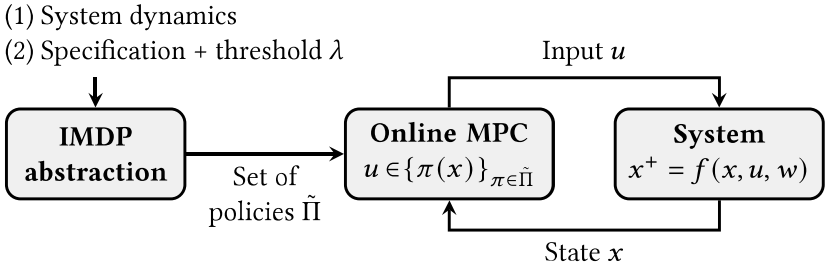


Figure 5.1: Our framework leverages formal abstractions to compute a set of certified policies $\tilde{\Pi}$, together with online MPC to optimize a cost function within this certified set.

these techniques incompatible with online optimization. Instead, we newly associate each abstract action with a *set of control inputs* for the dynamical system, so that every abstract policy corresponds with a *set of policies* for the dynamical system. We show that our abstraction induces a variant of a *probabilistic alternating simulation relation* (PASR) [20, 297]. Furthermore, we formally show that any policy within this set has a satisfaction probability that is *at least as high* as that of its corresponding abstract policy.

In the second step, the online MPC selects control inputs from the subset of inputs allowed by the set of policies obtained from the abstraction. These policies are piecewise-constant over the state-space partition induced by the abstraction, which, together with the (generally) nonlinear dynamics, renders the MPC problem nonconvex. Thus, we leverage reformulations from MPC for hybrid systems to obtain a piecewise-affine approximation of the dynamics [314]. We formulate the resulting MPC problem as a mixed integer quadratic program (MIQP) [36]. Crucially, despite these approximations, the online MPC preserves the certified lower bound λ on the satisfaction probability obtained via the abstraction. Indeed, even if the MIQP is infeasible (which can occasionally occur due to the model approximation used by the MPC controller), choosing *any* input consistent with the set of policies obtained from the abstraction preserves the certified bound on the satisfaction probability.

We empirically evaluate our framework on several benchmarks from the literature [20, 236, 328]. We, in particular, compare our framework against a vanilla IMDP abstraction technique that does not consider cost optimization, as in, e.g., [181]. The experiments confirm that our framework effectively improves a cost function with only a minor and tunable loss in the lower bound on the satisfaction probability (incurred due to considering now sets of policies).

In summary, our main contributions are as follows:

- We propose a novel abstraction that associates each abstract action with a set of control inputs for the original dynamical system, and is thus compatible with online control.
- We develop a tailored MPC scheme that optimizes a given cost function while preserving the lower bound on the satisfaction probability obtained from the abstraction.
- We show that, compared to standard IMDP abstractions, our framework improves the cost function with only a minor loss on the bound on the satisfaction probability.

Structure of the chapter After the related work in Sec. 5.2, we formalize the problem statement in Sec. 5.3 and provide background on IMDPs in Sec. 5.4. We then present our novel abstraction with sets of control inputs Sec. 5.5, which we embed in an IMDP abstraction algorithm in Sec. 5.6. Thereafter, Sec. 5.7 integrates this IMDP abstraction into an MPC scheme for online cost optimization. Finally, we empirically evaluate our abstraction framework in Sec. 5.8.

5.2 Related Work

Policy synthesis for stochastic dynamical systems has largely been addressed via two approaches. The first (which we take in this work) is to construct a—typically finite—abstraction and synthesize a policy on this abstract model [2, 181]. Under an appropriate behavioral relation (e.g., a simulation or bisimulation relation), guarantees on the satisfaction probability for the abstraction carry over to the original system. Abstractions of stochastic systems are commonly represented as MDPs [125, 184, 185, 331], interval MDPs (IMDPs) [22, 23, 59, 181], and variants with more expressive forms of uncertainty [115, 219]. More recently, abstractions constructed from data have also gained significant interest [116, 148, 187, 241].

The second approach to policy synthesis for stochastic systems is to find a *certificate function* that implies the satisfaction of a specification. Certificates have been developed for stability [267], safety [73, 149, 265], reach-avoidance [21, 360], and recently for general (ω-*regular*) temporal logic properties [1, 132]. However, finding certificates is challenging, and developing general methods for computing certificates remains an ongoing research effort [81].

MPC, also known as receding horizon control, is among the most popular modern control techniques [271]. Given a current state, MPC minimizes a cost function over a finite horizon while accounting for the dynamics and constraints on, e.g., inputs. MPC has been effectively deployed in many industrial settings [189, 230, 266]. Current research on MPC focuses on integration with data-based and machine-learning techniques [135, 237, 274], and on large-scale and distributed applications [72, 95, 283, 295]. In this work, we consider MPC for hybrid dynamical systems [129, 323] to connect the abstraction-based policy with online optimization. Specifically, we use logical constraints to select optimization regions, based on the framework for mixed logical dynamical (MLD) systems [36].

Finally, our setting is conceptually related to *safety filters* or *shields* that monitor a policy at runtime and intervene, when necessary, by modifying its intended action [136, 138, 337]. Safety filters have been synthesized, for example, as control barrier functions [8, 174] and as shields using model checking [6, 56]. Nevertheless, as their name suggests, safety filters exclusively guarantee closed-loop safety but not the satisfaction of the richer temporal logic specifications that we consider in this chapter. In addition, our method also differs from the stochastic optimal control methodology for constrained MDPs developed in [245], which uses a learning-based policy gradient approach to ensure safe exploration.

5.3 Problem Setup

In this section, we introduce the models we consider throughout the chapter, and we formally state the problem we address.

Notation The power set (the set of all subsets) over X is written 2^X. For the natural numbers, we write $\mathbb{N}_0 := \{0\} \cup \mathbb{N}$. A probability space $(\Omega, \mathcal{F}, \mathbb{P})$ consists of a sample space Ω, a σ-algebra $\mathcal{F}$, and a probability measure $\mathbb{P} : \mathcal{F} \to [0,1]$. We denote the Borel σ-algebra over a set X by $\mathcal{B}(X)$. The set of all distributions over a set X with a σ-algebra $\mathcal{F}$ is written as $\Delta(X, \mathcal{F})$. For a finite set X, we implicitly use the power set as the σ-algebra and simply write $\Delta(X)$.

5.3.1 Discrete-Time Stochastic Systems

Consider a discrete-time nonlinear stochastic system, denoted by $\mathbf{S}$, described by the difference equation

$$\mathbf{S} : \begin{cases} x_{k+1} = f(x_k, u_k, w_k), & \text{for } k = 0, 1, \dots, \\ x_0 \sim \mu_{x_0}, \end{cases} \tag{5.1}$$

where $x_k \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$, $u_k \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$, are, respectively, the states and control input at time step k (defined over compact subsets $\mathcal{X}$ and $\mathcal{U}$), $w_k \in \mathcal{W}$ is a random variable acting as a stochastic disturbance, $\mu_{x_0} \in \Delta(\mathcal{X}, \mathcal{B}(X))$ is the initial distribution, and $f : \mathcal{X} \times \mathcal{U} \times \mathcal{W} \to \mathcal{X}$ is the nonlinear transition function.

Assumption 5.1. The disturbance $\{w_k\}_{k \in \mathbb{N}_0}$ is a stationary process, where each $w_k \sim \mathbb{P}$ is an independent and identically distributed (i.i.d.) random variable in the probability space $(\mathcal{W}, \mathcal{B}(\mathcal{W}), \mathbb{P})$, where $\mathbb{P}$ is absolutely continuous with respect to the Lebesgue measure.

The probability measure $\mathbb{P}$ being absolutely continuous implies that each w_k admits a density function. As a result, the probability that x_{k+1} lies in a zero-volume set is zero, which will be a desirable property when generating the abstraction (see Footnote 5 in Sec. 5.6).

System $\mathbf{S}$ can equivalently be described using a stochastic kernel $\mathbf{t} : \mathcal{X} \times \mathcal{U} \to \Delta(\mathcal{X}, \mathcal{B}(\mathcal{X}))$, mapping each state-action pair to a probability distribution over states [159, Proposition 11.6]. For all $x_k \in \mathcal{X}$, $u_k \in \mathcal{U}$, and $A \in \mathcal{B}(\mathcal{X})$, this stochastic kernel is defined as

$$\mathbf{t}(A \mid x_k, u_k) = \mathbb{P}(w_k \in \mathcal{W} \mid f(x_k, u_k, w_k) \in A). \tag{5.2}$$

Specifications To define control tasks (called *specifications*), we equip the system $\mathbf{S}$ with a universally measurable *labeling function* $h_{\mathbf{S}} : \mathcal{X} \to 2^{\mathcal{Y}}$ over a finite set of labels $\mathcal{Y}$. In this chapter, we focus on (infinite-horizon) *reach-avoid specifications*, which require the system to eventually reach the goal states $\mathcal{X}_G \subset \mathcal{X}$ while avoiding unsafe states $\mathcal{X}_U \subset \mathcal{X}$ until then. Such a specification uses the labels $\mathcal{Y} = \{goal, safe\}$ and labeling function $h_{\mathbf{S}}$ defined for all $x \in \mathcal{X}$ as

$$\mathsf{goal} \in h_{\mathbf{S}}(x) \iff x \in X_G, \quad \mathsf{safe} \in h_{\mathbf{S}}(x) \iff x \in X \setminus X_U.$$

The reach-avoid specification is then identified by its set of satisfying output traces $\mathbf{Y}$, defined as

$$\mathbf{Y} := \Big\{ \{h_{\mathbf{S}}(x_k)\}_{k \in \mathbb{N}_0} \in (2^{\mathcal{Y}})^{\mathbb{N}_0} : \exists k \in \mathbb{N}.\, goal \in h_{\mathbf{S}}(x_k) \wedge \forall k' \leq k.\, safe \in h_{\mathbf{S}}(x_{k'}) \Big\}. \tag{5.3}$$

5

Policies The actions in S are selected by a memoryless policy $\pi : \mathcal{X} \to \mathcal{U}$, which is a universally measurable function. The set of all such policies is denoted by $\Pi_S = \{\mathcal{X} \to \mathcal{U}\}$. For the policy π, the sequence of states $(x_0, x_1, \ldots)$ is obtained as $x_0 \sim \mu_{x_0}$ and $x_{k+1} \sim \mathbf{t}(\cdot \mid x_k, \pi(x_k))$ for all $k \in \mathbb{N}_0$.

Remark 5.1. More general specifications are often expressed in linear temporal logic (LTL) [261] or probabilistic computation tree logic (PCTL) [127]. Our techniques are compatible with these specifications via the standard approach: express the specification as a deterministic finite automaton (DFA) with states $\mathcal{Q}$, construct the product of system S and the DFA, and consider a reach-avoid specification on the product state space $\mathcal{X} \times \mathcal{Q}$ [24, Sec.10.6]. In such a case, the policy $\pi : \mathcal{X} \times \mathcal{Q} \to \mathcal{U}$ maps from the product state space to actions. A special case is the reach-avoid specification over a finite horizon of $K \in \mathbb{N}$ steps, in which case the DFA states $\mathcal{Q} = \{0, 1, \ldots, K\}$ encode precisely the time steps up to the horizon, and the policy is *Markovian*, i.e., has the form $\pi : \mathcal{X} \times \{0, 1, \ldots, K\} \to \mathcal{U}$. For simplicity, we restrict ourselves to specifications that do not need this product construction.

Fixing a policy π creates a stochastic process over paths $\{x_k\}_{k \in \mathbb{N}_0}$ and thus also over output traces $\{y_k\}_{k \in \mathbb{N}_0} = \{h_S(x_k)\}_{k \in \mathbb{N}_0}$. This process, also called the *closed-loop system*, is defined on the sample space $\Omega := \prod_{k=1}^{\infty}(2^{\mathcal{Y}})$ endowed with its product topology $\mathcal{B}(\Omega)$ and a probability measure $\mathbb{P}_S^\pi$ over output traces $(y_0, y_1, \ldots)$ uniquely generated by the transition kernel $\mathbf{t}$ [40, Proposition 7.45]. Intuitively, $\mathbb{P}_S^\pi(\mathrm{Y})$ *measures* the probability that the closed-loop system generates an output trace $(y_0, y_1, \ldots)$ contained in $\mathrm{Y} \subseteq (2^{\mathcal{Y}})^{\mathbb{N}_0}$.

5

5.3.2 Problem Statement

Our goal is to compute a policy π for the system S that (1) minimizes a desired cost function, and (2) guarantees that a given specification is satisfied with at least a desired probability of $\lambda \in [0, 1]$. This cost function is defined as a real-valued *cost function* $J : (\mathcal{X} \times \mathcal{U})^{\mathbb{N}_0} \to \mathbb{R}$ over paths and can model various performance indicators, such as total control effort, and even supports non-Markovian objectives. We write $\mathbb{E}_S^\pi[J(x_0 u_0 x_1 u_1 \cdots)]$ for the expectation over this cost function in the system S for the fixed policy π, i.e. w.r.t. the probability $\mathbb{P}_S^\pi(\mathrm{Y})$. Then, the problem we consider is stated as follows.

Problem 5.1. Consider a system S as in (5.1), a reach-avoid specification Y as in (5.3), a threshold $\lambda \in [0, 1]$, and a cost function $J : (\mathcal{X} \times \mathcal{U})^{\mathbb{N}_0} \to \mathbb{R}$. Compute a policy π such that $\mathbb{P}_S^\pi(\mathrm{Y}) \geq \lambda$ and that minimizes $\mathbb{E}_S^\pi[J(x_0 u_0 x_1 u_1 \cdots)]$, i.e., a policy that solves the following optimization problem:

$$\begin{aligned} \min_{\pi \in \Pi_S} \ & \mathbb{E}_S^\pi[J(x_0 u_0 x_1 u_1 \cdots)] \\ \text{s.t.} \ & \mathbb{P}_S^\pi(\mathrm{Y}) \geq \lambda. \end{aligned} \tag{5.4}$$

Example 5.1. Consider again, the reach-avoid specification $\mathcal{Y}$ defined in (5.3). Suppose we want to synthesize a policy that satisfies this specification with probability at least $\lambda = 0.9$. In addition, we want to minimize the total control effort required to reach the goal states $\mathcal{X}_G$, modeled by the cost function $J = \sum_{k \in \mathbb{N}_0} \|u_k\|_2^2$. Standard abstraction techniques optimize solely for this satisfaction probability but not for the cost function J. Our goal is, instead, to additionally minimize the expected cost J. As a result, the satisfaction probability $\mathbb{P}_S^\pi(\mathrm{Y})$ may be slightly lower, but it cannot violate the threshold of λ.

Solving Prob. 5.1 exactly amounts to solving a stochastic and nonconvex optimization program that is generally intractable [40]. Thus, we will treat $\mathbb{P}^{\pi}_{\mathbf{S}}(\mathbf{Y}) \geq \lambda$ as a *hard constraint* and minimizing $\mathbb{E}^{\pi}_{\mathbf{S}}[J(x_0 u_0 x_1 u_1 \cdots)]$ as a *soft requirement*. In other words, we seek a policy that satisfies the specification $\mathbf{Y}$ with probability at least λ, and that yields a low (but possibly suboptimal) expected cost.

Overview of our approach Our approach to solving Prob. 5.1 is based on a finite-state abstraction in the form of an IMDP (cf. Def. 5.1). Existing abstractions are predominantly based on a partition of the state space and a gridding of the action space [20, 125, 181, 219]. We, by contrast, define in Sec. 5.6 a novel and more general abstraction: the discrete states are still based on a partition, but the abstract actions are no longer defined by a gridding of the action space. Instead, each abstract action corresponds to a *set of inputs*, leading to the novel notion of a *set-valued interface function* (defined in Sec. 5.5). Doing so, we can offload part of the policy synthesis to an online control algorithm (for which we use MPC; see Sec. 5.7) that further refines the policy by selecting precise inputs from these sets.

5.4 Interval MDPs

In this chapter, we use finite-state interval Markov decision processes (IMDPs) [246, 320] to represent abstractions of system $\mathbf{S}$.

Definition 5.1 (IMDP)**.** An *interval MDP* (IMDP) $\mathbf{M}$ is a tuple $(\mathcal{S}, \mu_{s_0}, \mathcal{A}, \check{P}, \hat{P}, \mathcal{Y}, h_{\mathbf{M}})$, where

- $\mathcal{S}$ is a finite set of states,
- $\mu_{s_0} \in \Delta(\mathcal{S})$ is the initial distribution,
- $\mathcal{A}$ is a finite set of actions, and we write $\mathcal{A}(s) \subseteq \mathcal{A}$ for the actions enabled in state $s \in \mathcal{S}$,
- $\check{P}, \hat{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightharpoonup [0,1]$ assign[1] a lower and upper bound to each transition probability, respectively, which define the uncertain transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \rightharpoonup 2^{\Delta(\mathcal{S})}$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$ as

$$\mathcal{P}(s,a) = \left\{ \mu \in \Delta(\mathcal{S}) : \forall s' \in \mathcal{S}, \mu(s') \in [\check{P}(s,a,s'), \hat{P}(s,a,s')] \right\},$$

- $\mathcal{Y}$ is a finite set of labels, and
- $h_{\mathbf{M}} : \mathcal{S} \to 2^{\mathcal{Y}}$ is a labeling function over $\mathcal{Y}$.

We call $[\check{P}(s,a,s'), \hat{P}(s,a,s')] \subseteq [0,1]$ the *probability interval* for the transition (s,a,s'). Analogous to the system $\mathbf{S}$, the actions in an IMDP are selected by a memoryless policy[2] $\sigma : \mathcal{S} \to \mathcal{A}$. The set of all policies for $\mathbf{M}$ is $\Pi_{\mathbf{M}}$.

[1] The functions $\check{P}$, $\hat{P}$, and $\mathcal{P}$ are partial maps, written as $\rightharpoonup$, due to the fact that not all actions may be enabled in every state.

[2] For clarity, we write policies for the system $\mathbf{S}$ as π and policies for the IMDP $\mathbf{M}$ as σ.

Adversary An IMDP defines a game between the policy that selects actions and an *adversary* that fixes distributions $P(s,a) \in \mathcal{P}(s,a)$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$. A different probability can be chosen every time the same pair (s,a) is encountered (called the *dynamic* uncertainty model [147, 320]). We overload notation and write $P \in \mathcal{P}$ for fixing a distribution $P(s,a) \in \mathcal{P}(s,a)$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$.

Optimal robust policies Fixing $\sigma \in \Pi_{\mathbf{M}}$ and $P \in \mathcal{P}$ for $\mathbf{M}$ yields a Markov chain with (standard) probability measure $\mathbb{P}_{\mathbf{M}}^{\sigma,P}$ [24]. Analogous to the system $\mathbf{S}$, a reach-avoid specification $\mathbf{Y}'$ for $\mathbf{M}$ is identified by its set of satisfying output traces, such that $\mathbf{Y}' \subseteq \prod_{k=1}^{\infty}(2^{\mathcal{Y}})$. The probability that $\mathbf{M}$ with policy σ and adversary P satisfies $\mathbf{Y}'$ is written as $\mathbb{P}_{\mathbf{M}}^{\sigma,P}(\mathbf{Y}')$. An *optimal (robust) policy* $\sigma^\star \in \Pi_{\mathbf{M}}$ maximizes this probability under the worst-case adversary:[3]

$$\sigma^\star \in \operatorname*{argmax}_{\sigma \in \Pi_{\mathbf{M}}} \min_{P \in \mathcal{P}} \mathbb{P}_{\mathbf{M}}^{\sigma,P}(\mathbf{Y}'). \tag{5.5}$$

As the IMDP has finitely many states and actions, optimal robust policies can be computed using *robust value iteration* [125, 344]. Robust value iteration has been implemented in common probabilistic model checking tools such as PRISM [178] and Storm [131].

5

5.5 Set-Valued Interface Functions

At the core of an abstraction of system $\mathbf{S}$ into an IMDP $\mathbf{M}$ is a measurable relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{S}$ between the states of both models. In this work, we only consider relations such that $|\mathcal{R}(x)| = 1$ for all $x \in \mathcal{X}$, i.e., $\mathcal{R}$ represents a *partition* of $\mathcal{X}$.[4] We overload notation and equivalently write such a relation as a function $\mathcal{R} : \mathcal{X} \to \mathcal{S}$, whose preimage is defined as $\mathcal{R}^{-1}(s) = \{x \in \mathcal{X} \mid \mathcal{R}(x) = s\}$.

Next, we define the notion of an interface function, which is commonly used in abstraction-based control [20, 110, 125, 331].

Definition 5.2 (Interface)**.** An *interface function* for the system $\mathbf{S}$, the IMDP $\mathbf{M}$, and the relation $\mathcal{R} : \mathcal{X} \to \mathcal{S}$ is a function $\mathcal{F} : \mathcal{X} \times \mathcal{A} \to \mathcal{U}$ that maps every $x \in \mathcal{X}$ and $a \in \mathcal{A}$ to an input $u \in \mathcal{U}$ for $\mathbf{S}$.

An interface function refines every state $x \in \mathcal{X}$ and abstract action $a \in \mathcal{A}$ into a *single* input $u \in \mathcal{U}$ for the system $\mathbf{S}$. As a result, the interface completely determinizes the stochastic system $\mathbf{S}$, without leaving any room to optimize for the cost function J in Prob. 5.1. Therefore, we propose a novel definition that generalizes the interface to be a map from states to *sets of inputs*.

Definition 5.3 (Set-valued interface)**.** A *set-valued interface function* for the system $\mathbf{S}$, the IMDP $\mathbf{M}$, and the relation $\mathcal{R} : \mathcal{X} \to \mathcal{S}$ is a function $\mathcal{F}_{\text{set}} : \mathcal{X} \times \mathcal{A} \to 2^{\mathcal{U}}$.

Note that a (standard) interface function $\mathcal{F}$ is a special case of a set-valued interface function $\mathcal{F}_{\text{set}}$ with singleton input sets.

[3] Conversely, we may define variants of robust policies for IMDPs where the max and/or min operators are inverted.

[4] For more general abstractions based on, for example, a cover of the state space, see, e.g., [125]. However, these abstractions lead to more involved refinement strategies.

The next notion we introduce is that of a lifting for a relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{S}$, as defined in [126]. Intuitively, such a lifting lifts a relation on two state spaces to distributions over these state spaces.

Definition 5.4 (Lifting [126])**.** Let $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{S}$ be a relation for $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ and $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$. The relation $\mathcal{R}^{\mathcal{P}} \subseteq \Delta(\mathcal{X}, \mathcal{B}(\mathcal{X})) \times \Delta(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ is called a *lifting of relation* $\mathcal{R}$ if $(\mu, \nu) \in \mathcal{R}^{\mathcal{P}}$ holds for all $\mu \in \Delta(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ and $\nu \in \Delta(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ for which there exists a probability space $(\mathcal{X} \times \mathcal{S}, \mathcal{B}(\mathcal{X} \times \mathcal{S}), \mathbb{W})$ satisfying:

1. for all $A \in \mathcal{B}(\mathcal{X})$ it holds that $\mathbb{W}(A, \mathcal{S}) = \mu(A)$,
2. for all $B \in \mathcal{B}(\mathcal{S})$ it holds that $\mathbb{W}(\mathcal{X}, B) = \nu(B)$,
3. $\mathbb{W}(\mathcal{R}) = 1$.

Based on the set-valued interface and lifted relation, we now define the *behavioral relation* between the system $\mathbf{S}$ and the IMDP $\mathbf{M}$ that is fundamental to solving Prob. 5.1.

Definition 5.5 (Probabilistic alternating simulation)**.** Consider a system $\mathbf{S}$ as in (5.1) and an IMDP $\mathbf{M} = (\mathcal{S}, \mu_{s_0}, \mathcal{A}, \check{P}, \hat{P}, \mathcal{Y}, h_{\mathbf{M}})$. If there exist

- A relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{A}$ such that $|\mathcal{R}(x)| = 1$ for all $x \in \mathcal{X}$,
- A lifting $\mathcal{R}^{\mathcal{P}}$ of the relation $\mathcal{R}$, and
- An set-valued interface function $\mathcal{F}_{\text{set}} : \mathcal{X} \times \mathcal{A} \to 2^{\mathcal{U}}$,

such that the following conditions are satisfied:

1. The initial distributions are related: $(\mu_{x_0}, \mu_{s_0}) \in \mathcal{R}^{\mathcal{P}}$,
2. For all $(x, s) \in \mathcal{R}$, the labels coincide: $h_{\mathbf{S}}(x) = h_{\mathbf{M}}(s)$,
3. For all $(x, s) \in \mathcal{R}$, $a \in \mathcal{A}$, and $u \in \mathcal{F}_{\text{set}}(x, a)$, there exists $\nu \in \mathcal{P}(s, a)$ such that $\big(\mathbf{t}(\cdot \mid x, u), \nu\big) \in \mathcal{R}^{\mathcal{P}}$,

then the relation $\mathcal{R}$ is a *probabilistic alternating simulation relation* (PASR) from $\mathbf{S}$ to $\mathbf{M}$, denoted by $\mathbf{S} \preceq_{\text{alt}} \mathbf{M}$.

The PASR in Def. 5.5 is a variant of [20, Def. 7], tailored to one system with a precise stochastic kernel (the system $\mathbf{S}$) and one with an uncertain stochastic kernel (the IMDP). Furthermore, we generalized the interface function to sets of inputs, requiring an additional quantification over all $u \in \mathcal{F}_{\text{set}}(x, a)$ in condition 3.

Theorem 5.1 (Policy refinement)**.** *Let* $\mathbf{S} \preceq_{\text{alt}} \mathbf{M}$*. Then for every IMDP policy* $\sigma : \mathcal{S} \to \mathcal{A}$ *and every specification* $\mathbf{Y} \subseteq \prod_{k=1}^{\infty}(2^{\mathcal{Y}})$*, it holds that*

$$\min_{P \in \mathcal{P}} \mathbb{P}_{\mathbf{M}}^{\sigma, P}(\mathbf{Y}) \leq \mathbb{P}_{\mathbf{S}}^{\pi}(\mathbf{Y}) \leq \max_{P \in \mathcal{P}} \mathbb{P}_{\mathbf{M}}^{\sigma, P}(\mathbf{Y}), \tag{5.6}$$

where the policy π *is defined for all* $k \in \mathbb{N}_0$ *and* $x \in \mathcal{X}$ *as*

$$\pi(x_k) \in \mathcal{F}_{\text{set}}(x_k, \sigma(\mathcal{R}(x_k))). \tag{5.7}$$

Proof. As in [125], the satisfaction probability $\mathbb{P}_{\mathrm{S}}^{\hat{\pi}}(\mathbf{Y})$ under a fixed policy $\hat{\pi} : \mathcal{X} \to \mathcal{U}$ can be computed via a dynamic programming recursion on the value function $V_k^{\hat{\pi}} : \mathcal{X} \to \mathbb{R}$, $k \in \mathbb{N}$, where

$$V_{k+1}^{\hat{\pi}}(x) = \int_{\mathcal{X}} \max\left\{1_{\mathcal{X}_F}(\xi), V_k^{\hat{\pi}}(\xi)\right\} \cdot \mathbf{t}(\mathrm{d}\xi \mid x, \hat{\pi}(x)), \tag{5.8}$$

where $\mathcal{X}_F \subseteq \mathcal{X}$ is the set of states from which $\mathcal{Y}$ is satisfied with probability one. Let $V_\star^{\hat{\pi}}$ be the least fixed point of this recursion. The satisfaction probability is given as $\mathbb{P}_{\mathrm{S}}^{\hat{\pi}}(\mathbf{Y}) = \int_{\mathcal{X}} V_\star^{\hat{\pi}}(\xi)\mu_{x_0}(\mathrm{d}\xi)$.

Due to the monotonicity of the value function, the value function for any policy π that satisfies (5.7) is bounded as $\check{V}_\star(x) \leq V_\star^{\hat{\pi}}(x) \leq \hat{V}_\star(x)$ for all $x \in \mathcal{X}$, where $\check{V}_\star(x)$ and $\hat{V}_\star(x)$ are the fixed points of the recursions

$$\check{V}_{k+1}(x) = \min_{u \in \mathcal{F}_{\mathrm{set}}(x,\sigma)} \int_{\mathcal{X}} \max\left\{1_{\mathcal{X}_F}(\xi), \check{V}_k(\xi)\right\} \cdot \mathbf{t}(\mathrm{d}\xi \mid x, \hat{\pi}(x))$$

$$\hat{V}_{k+1}(x) = \max_{u \in \mathcal{F}_{\mathrm{set}}(x,\sigma)} \int_{\mathcal{X}} \max\left\{1_{\mathcal{X}_F}(\xi), \hat{V}_k(\xi)\right\} \cdot \mathbf{t}(\mathrm{d}\xi \mid x, \hat{\pi}(x)),$$

with the shorthand notation $\mathcal{F}_{\mathrm{set}}(x, \sigma) \coloneqq \mathcal{F}_{\mathrm{set}}(x, \sigma(\mathcal{R}(x)))$.

To prove (5.6), we will show that $\check{V}_{k+1}$ and $\hat{V}_{k+1}$ can be lower bounded by dynamic programming recursions on the IMDP. For the lower bound, define the following recursion for the IMDP:

$$\check{W}_{k+1}^{\sigma}(s) = \min_{\nu \in \mathcal{P}(s,\sigma(s))} \sum_{s' \in \mathcal{S}} \max\{1_{\mathcal{S}_F}, W_k^{\sigma}(s')\} \cdot \nu(s'). \tag{5.9}$$

The minimum satisfaction probability $\min_{P \in \mathcal{P}} \mathbb{P}_{\mathrm{M}}^{\sigma,P}(\mathbf{Y})$ for the IMDP is computed as $\sum_{s \in \mathcal{S}} \mu_{s_0}(s) \cdot \check{W}_\star^{\sigma}(s)$, where $\check{W}_\star^{\sigma}$ is the least fixed point of the recursion in (5.9). Let us show that, if $\check{W}_k^{\sigma}(s) \leq \check{V}_k(x)$ for all $(x, s) \in \mathcal{R}$, then also $\check{W}_{k+1}^{\sigma}(s') \leq \check{V}_{k+1}(x')$ for all $(x, s) \in \mathcal{R}$. First, since $\mathcal{R}$ models a partition of $\mathcal{X}$, the integral over $\mathcal{X}$ can be decomposed as the sum over the IMDP states $s' \in \mathcal{S}$:

$$\check{V}_{k+1}(x) = \min_{u \in \mathcal{F}_{\mathrm{set}}(x,\sigma)} \sum_{s' \in \mathcal{S}} \int_{\mathcal{R}^{-1}s'} \max\left\{1_{\mathcal{X}_F}(\xi), \check{V}_k(\xi)\right\} \cdot \mathbf{t}(\mathrm{d}\xi \mid x, \hat{\pi}(x))$$

Next, observe that for all $(x, s) \in \mathcal{R}$, we have $\check{V}_k(x) \geq \check{W}_k^{\sigma}(s)$. Furthermore, due to condition (2) of the PASR, it holds that $1_{\mathcal{X}_F}(x) = 1_{\mathcal{S}_F}(s)$, and due to (3), there exists a distribution $\nu \in \mathcal{P}(s, \sigma(s))$ that *underapproximates* the integral $\mathbf{t}(\mathrm{d}\xi \mid x, \hat{\pi}(x))$. Hence, we obtain

$$\begin{aligned}\check{V}_{k+1}(x) &\geq \min_{u \in \mathcal{F}_{\mathrm{set}}(x,\sigma)} \sum_{s' \in \mathcal{S}} \max\left\{1_{\mathcal{S}_F}(s'), \check{W}_k^{\sigma}(s')\right\} \int_{\mathcal{R}^{-1}s'} \mathbf{t}(\mathrm{d}\xi \mid x, \hat{\pi}(x)),\\ &\geq \min_{\nu \in \mathcal{P}(s,\sigma(s))} \sum_{s' \in \mathcal{S}} \max\left\{1_{\mathcal{S}_F}(s'), \check{W}_k^{\sigma}(s')\right\} \cdot \nu(s') = \check{W}_{k+1}^{\sigma}(s),\end{aligned}$$

which proves that $\check{V}_{k+1}(x) \geq \check{W}_{k+1}^{\sigma}(s)$ for all $(x, s) \in \mathcal{R}$. The upper bound $\hat{V}_{k+1}^{\sigma}(x) \leq \hat{W}_{k+1}(s)$ follows analogously and is thus omitted. Altogether, it follows that the fixed points of these recursions satisfy

$$\check{W}_\star^{\sigma}(s) \leq \check{V}_\star(x), \qquad \hat{V}_\star(x) \leq \hat{W}_\star^{\sigma}(s), \qquad \forall (x, s) \in \mathcal{R}. \tag{5.10}$$

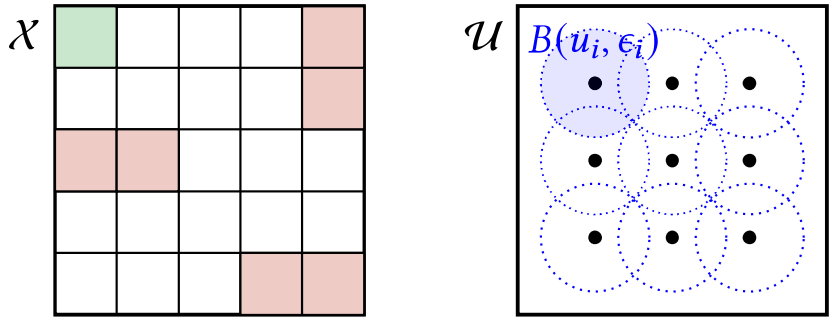


Figure 5.2: Our abstraction procedure is based on a partition of the state space $\mathcal{X}$ (left; showing a reach-avoid problem with goal states in green and avoid states in red), whereas each abstract action a_i corresponds to an L_p-ball $B(u_i, \epsilon_i)$ in the input space $\mathcal{U}$ (right; highlighting one such ball in blue).

Since the values in the initial states are linear combinations of the values in (5.10), the bounds in (5.6) follow. □

Intuitively, Thm. 5.1 asserts that the existence of a PASR implies that the satisfaction probability $\mathbb{P}^{\pi}_{\mathbf{S}}(\mathbf{Y})$ is bounded by the min/max satisfaction probability on the IMDP, as long as the policy π for $\mathbf{S}$ is selected based on the (set-valued) interface function $\mathcal{F}_{\text{set}}$. In other words, the interface function $\mathcal{F}_{\text{set}}$ takes care of the required lower bound $\mathbb{P}^{\pi}_{\mathbf{S}}(\mathbf{Y}) \geq \lambda$, whereas we can freely choose within $\mathcal{F}_{\text{set}}$ to optimize the expected cost $\mathbb{E}^{\pi}_{\mathbf{S}}[J(x_0 u_0 x_1 u_1 \cdots)]$.

5.6 IMDP Abstraction

In this section, we present a model-based IMDP abstraction procedure that induces the PASR defined in Def. 5.5. Our abstraction procedure is a variant of [20, 181], adapted to the set-valued interface function in Def. 5.3. In what follows, we define the IMDP's states $\mathcal{S}$, actions $\mathcal{A}$, transition probabilities $\check{P}, \hat{P}$, and labeling function $h_{\mathbf{M}}$.

States and labels The states are defined by a partition of the compact state space $\mathcal{X} \subset \mathbb{R}^{n_x}$, as shown by Fig. 5.2. Without loss of generality, we assume a partition into $L-1$ polytopes $\{\mathcal{V}_1, \ldots, \mathcal{V}_{L-1}\}$, plus one region $\mathcal{V}_L$ covering the remaining state space:

1. Each region $\mathcal{V}_1, \ldots, \mathcal{V}_{L-1}$ is a convex polytope, i.e., $\mathcal{V}_i = \{x \in \mathbb{R}^n : H_i x \leq b_i\}$ for $H_i \in \mathbb{R}^{\xi_i \times n}$, $b_i \in \mathbb{R}^{\xi_i}$, and $\xi_i \in \mathbb{N}$;
2. The union of the regions covers $\mathcal{X}$, i.e., $\mathcal{X} = \bigcup_{i=1}^{L} \mathcal{V}_i$;
3. The interiors of all regions are disjoint, i.e., $\text{int}(\mathcal{V}_i) \bigcap \text{int}(\mathcal{V}_j) = \emptyset$, $\forall i, j \in \{1, \ldots, L\}$, $i \neq j$.

Thus, each region $\mathcal{V}_1, \ldots, \mathcal{V}_{L-1}$ is a convex polytope, and the remaining region $\mathcal{V}_L$ is the complement of $\cup_{i=1}^{L-1} \mathcal{V}_i$. We define one IMDP state for each of the L regions, such that $\mathcal{S} \coloneqq \{s_1, \ldots, s_L\}$.[5] The partition induces a relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{S}$ such that $(x, s_i) \in \mathcal{R} \iff x \in \mathcal{V}_i$.

[5] Technically, states x on the boundary between regions are related to multiple IMDP states. However, Asm. 5.1 means that the probability of reaching such a state is zero, so this technicality does not affect the correctness of the abstraction.

The initial distribution μ_{s_0} is defined as $\mu_{s_0}(s) = \mu_{x_0}(\mathcal{R}^{-1}(s))$ for all $s \in \mathcal{S}$. For simplicity, we assume the partition is *label-preserving*, which allows us to satisfy condition (2) in Def. 5.5.

Assumption 5.2 (Label-preserving)**.** For all states $x, x' \in \mathcal{X}$ such that $\mathcal{R}(x) = \mathcal{R}(x')$, it holds that $h_{\mathrm{S}}(x) = h_{\mathrm{S}}(x')$.

For a label-preserving partition, the abstract labeling function h_{M} is trivially defined for all $s \in \mathcal{S}$ as $h_{\mathrm{M}}(s) \coloneqq h_{\mathrm{S}}(\mathcal{R}^{-1}(s))$.

Remark 5.2. Constructing a label-preserving polyhedral partition can be challenging (or even impossible, e.g., a circular goal region cannot be represented using finitely many convex polytopes). At the cost of weaker bounds on the satisfaction probability in Thm. 5.1, one can weaken Asm. 5.2 by suitably over- and underapproximating the labels, or by defining a metric on the output space as in [126].

Actions In a standard abstraction, each abstract action $a_i \in \mathcal{A}$ corresponds to a discrete input $u_i \in \mathcal{U}$. To accommodate the set-valued interface function, we instead associate each a_i to a *set of inputs*, defined as an L_p-ball, as also visualized in Fig. 5.2.

Definition 5.6. The L_p-ball of radius $\epsilon > 0$ centered at $c \in \mathbb{R}^n$ is defined as $B(c, \epsilon) = \{c' \in \mathbb{R}^n : \|c' - c\|_p \leq \epsilon\}$.

We define the set of IMDP actions as $\mathcal{A} \coloneqq \{a_1, \ldots, a_M\}$, where each $a_i \in \mathcal{A}$ is associated with an L_p-ball $B(u_i, \epsilon_i)$ of radius ϵ_i centered at $u_i \in \mathcal{U}$. We define the set-valued interface function $\mathcal{F}_{\mathrm{set}} : \mathcal{X} \times \mathcal{A} \to 2^{\mathcal{U}}$ for all $x \in \mathcal{X}$, $a_i \in \mathcal{A}$ as $\mathcal{F}_{\mathrm{set}}(x, a_i) = B(u_i, \epsilon_i)$. For simplicity, but without loss of generality, we do not make the set-valued interface $\mathcal{F}_{\mathrm{set}}$ dependent on the state. We discuss a possible criterion for selecting the size ϵ of each ball in Sec. 5.8.2.

Transition probabilities For every triple $(s, a, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$, we need to compute the probability interval $[\check{P}(s, a, s'), \hat{P}(s, a, s')]$ that defines the set of distributions $\mathcal{P}(s, a)$. For the standard IMDP abstraction procedure [20, 59, 181] (where each action a_i corresponds to a singleton input u_i), this interval would be defined as

$$\left[\min_{x \in \mathcal{R}^{-1}(s)} \mathbf{t}(\mathcal{R}^{-1}(s') \mid x, u_i), \max_{x \in \mathcal{R}^{-1}(s)} \mathbf{t}(\mathcal{R}^{-1}(s') \mid x, u_i) \right].$$

For our setting, we use the set-valued interface $\mathcal{F}_{\mathrm{set}}$ to additionally minimize and maximize over the ball $B(u_i, \epsilon_i)$. Thus, the probability bounds for the transition (s, a, s') are defined as

$$\check{P}(s, a, s') = \min_{\substack{x \in \mathcal{R}^{-1}(s) \\ u \in \mathcal{F}_{\mathrm{set}}(x, a)}} \mathbf{t}(\mathcal{R}^{-1}(s') \mid x, u_i), \tag{5.11}$$

$$\hat{P}(s, a, s') = \max_{\substack{x \in \mathcal{R}^{-1}(s) \\ u \in \mathcal{F}_{\mathrm{set}}(x, a)}} \mathbf{t}(\mathcal{R}^{-1}(s') \mid x, u_i). \tag{5.12}$$

For systems with additive (Gaussian) noise, these probabilities can be computed efficiently [59]. For more general dynamics, sampling-based approaches can be used to derive *probably approximately correct* (PAC) bounds on these probabilities [23].

The full uncertain transition function $\mathcal{P} : S \times \mathcal{A} \rightharpoonup 2^{\Delta(S)}$ is obtained by computing the lower and upper bounds in (5.11), (5.12) for every transition (s, a, s').

Policy Synthesis and Refinement Above, we defined an abstract IMDP $\mathbf{M}$ for system $\mathbf{S}$. This IMDP is a probabilistic simulation of the original system, i.e., $\mathbf{S} \preceq_{\text{alt}} \mathbf{M}$, as formalized by Thm. 5.2.

Theorem 5.2. *Let* $\mathbf{M} = (S, \mu_{s_0}, \mathcal{A}, \check{P}, \hat{P}, \mathcal{Y}, h_{\mathbf{M}})$ *be the IMDP abstraction for the system* $\mathbf{S}$ *equipped with the labeling function* $h_{\mathbf{S}} : \mathcal{X} \to 2^{\mathcal{Y}}$*, obtained using the abstraction procedure above. It holds that* $\mathbf{S} \preceq_{\text{alt}} \mathbf{M}$.

Proof. Let us show the three conditions in Def. 5.5 are satisfied:

1. The initial distribution is defined as $\mu_{s_0}(s) = \mu_{x_0}(\mathcal{R}^{-1}(s))$ for all $s \in S$, so it holds that $(\mu_{x_0}, \mu_{s_0}) \in \mathcal{R}^{\mathcal{P}}$.

2. The partition is label-preserving (Asm. 5.2), so it trivially holds that $h_{\mathbf{S}}(x) = h_{\mathbf{M}}(s)$ for all $(x, s) \in \mathcal{R}$.

3. For the third condition, pick any kernel $\mu := \mathbf{t}(\cdot \mid \hat{x}, \hat{u})$ for $\hat{x} \in \mathcal{X}$, $s \in \mathcal{R}^{-1}\hat{x}$, $a \in \mathcal{A}$, and $\hat{u} \in \mathcal{F}_{\text{set}}(\hat{x}, a)$. We need to show that there exists a distribution $\nu \in \Delta(S)$ such that $\nu \in \mathcal{P}(\mathcal{R}(\hat{x}), a)$ and $(\mu, \nu) \in \mathcal{R}^{\mathcal{P}}$. Define $\nu \in \Delta(S)$ for all $s' \in S$ as $\nu(s') = \mathbf{t}(\mathcal{R}^{-1}(s') \mid \hat{x}, \hat{u})$. Since $\hat{x} \in \mathcal{R}^{-1}(s)$ and $\hat{u} \in C$, $\check{P}(s, a, s') \leq \hat{P}(s, a, s')$ for all $s' \in S$, it indeed holds that $\nu \in \mathcal{P}(\mathcal{R}(\hat{x}), a)$. What remains to show is that $(\mu, \nu) \in \mathcal{R}^{\mathcal{P}}$, i.e., there exists a probability measure $\mathbb{W}$ such that the conditions in Def. 5.4 hold. Define $\mathbb{W}$ for all $Z \in \mathcal{B}(\mathcal{X} \times S)$ as

$$\mathbb{W}(Z) = \mathbf{t}(\{x \in \mathcal{X} : \exists s \in S, (x, s) \in Z \cap \mathcal{R}\} \mid \hat{x}, \hat{u}).$$

It is easily verified that $\mathbb{W}$ is a lifting for (μ, ν), i.e., $(\mu, \nu) \in \mathcal{R}^{\mathcal{P}}$, because: (1) for all $A \in \mathcal{B}(\mathcal{X})$, it holds that

$$\begin{aligned}\mathbb{W}(A, S) &= \mathbf{t}(\{x \in \mathcal{X} : \exists s \in S, (x, s) \in (A, S) \cap \mathcal{R}\} \mid \hat{x}, \hat{u}) \\ &= \mathbf{t}(A \mid \hat{x}, \hat{u}) = \mu(A);\end{aligned}$$

(2) for all $B \in \mathcal{B}(S)$, it holds that

$$\begin{aligned}\mathbb{W}(\mathcal{X}, B) &= \mathbf{t}(\{x \in \mathcal{X} : \exists s \in S, (x, s) \in (A, S) \cap \mathcal{R}\} \mid \hat{x}, \hat{u}) \\ &= \mathbf{t}(\cup_{s \in B} \mathcal{R}^{-1}(s) \mid \hat{x}, \hat{u}) = \nu(B);\end{aligned}$$

(3) $\mathbb{W}(\mathcal{R}) = \mathbf{t}(\{x \in \mathcal{X} : \exists s \in S, (x, s) \in \mathcal{R} \cap \mathcal{R}\} \mid \hat{x}, \hat{u}) = \mathbf{t}(\mathcal{X} \mid \hat{x}, \hat{u}) = 1$.

This concludes the proof. □

As a consequence of Thm. 5.2, for any specification $\mathbf{Y}$, we can take any policy $\sigma : S \to \mathcal{A}$ for the IMDP and use Thm. 5.1 to refine this policy σ into a policy π for the system $\mathbf{S}$ that satisfies the specification with probability at least $\min_{P \in \mathcal{P}} \mathbb{P}_{\mathbf{M}}^{\sigma, P}(\mathbf{Y})$ and at most $\max_{P \in \mathcal{P}} \mathbb{P}_{\mathbf{M}}^{\sigma, P}(\mathbf{Y})$. Thus, if $\min_{P \in \mathcal{P}} \mathbb{P}_{\mathbf{M}}^{\sigma, P}(\mathbf{Y}) \geq \lambda$, then the refined policy π is an admissible solution to Prob. 5.1.

Recall from Sec. 5.4 that abstract policies can be computed using robust value iteration. In practice, we use robust value iteration to compute an optimal robust IMDP policy σ that *maximizes* the probability of satisfying the specification on the IMDP under the *worst-case* adversary, i.e., a policy $\pi^\star$ satisfying (5.5). In our numerical experiments, we use an implementation of robust value iteration in the probabilistic model checker Storm [131].

5.7 Abstraction-Driven Model Predictive Control

In this section, we close the loop and feed the set of certified policies obtained based on the IMDP abstraction to an MPC controller that optimizes for the cost function J. This yields a solution to Prob. 5.1 that minimizes the expected cost $\mathbb{E}^\pi_{\mathbf{S}}[J(x_0 u_0 x_1 u_1 \cdots)]$ in (5.4) approximately, but that treats $\mathbb{P}^\pi_{\mathbf{S}}(\mathbf{Y}) \geq \lambda$ as a hard constraint.

5.7.1 MPC Architecture

At every time step $k \in \mathbb{N}$, we use the measured state x_k to determine the IMDP state $s_i = \mathcal{R}(x_k)$ and action $a_i = \sigma(s_i)$ taken by the abstract policy. The MPC controller then solves an optimization problem that minimizes the cost function J over a finite horizon, where the current input u_k must belong to the L_p-ball associated with action a_i, i.e., $u_k \in \mathcal{F}_{\text{set}}(x, a_i) = B(u_i, \epsilon_i)$. This input u_k is then implemented in the dynamical system, and the MPC is solved again for time step $k+1$. However, due to the nonlinear stochastic dynamics of system $\mathbf{S}$ together with the piecewise-constant form of the policies, the MPC controller necessitates two fundamental constructs to be deployed in this case:

1. a model of the system to be controlled accounting for the nonlinearity, which allows for accurate prediction of its future behavior; and
2. a logic-driven structure that enables the selection of the appropriate inputs from the associated balls $B(u_i, \epsilon_i)$.

To address point (1), we use a piecewise affine (PWA) approximations [314] of $\mathbf{S}$ to represent the nonlinear dynamics with an arbitrary degree of accuracy. These will be discussed in the experimental section for each individual case. For point (2), we resort to hybrid dynamical systems to represent the logic conditions linking measured and predicted states to respective cells in the partition of the abstraction and then to the associated L_p-balls given by the policy. Such an approach is possible thanks to mixed logical dynamical systems, as discussed in the remainder of this section, and represents the core of our abstraction-driven MPC policy reconstruction.

5.7.2 Logic-Driven L_p-Balls Selection Model

To allow the MPC formulation to meet the logical requirements described above, we use operational constraints that are activated through binary variables, thereby creating a hybrid system. These logical requirements can be written as:

$$x \in \mathcal{R}^{-1}(s_i) \implies u \in \mathcal{F}_{\text{set}}(x, \sigma(s_i)) = B(u_i, \epsilon). \tag{5.13}$$

To embed these constraints in an MPC problem, we define $N_s = |S|$ as the number of states in the IMDP abstraction, and N as the prediction horizon for the MPC. Then, we introduce

$N_s \cdot (N+1)$ binary variables δ that satisfy the following conditions:

$$\delta_k^i = 1 \iff x_k \in \mathcal{R}^{-1}(s_i), \qquad \sum_{i=1}^{N_s} \delta_k^i = 1 \quad \text{for } k = 1,\dots,N. \tag{5.14}$$

For each region $\mathcal{R}^{-1}(s_i)$, we introduce the n_x-vectors M_s^i and m_s^i that represent the element-wise maximum and minimum boundaries of the regions associated with each state s_i, i.e.,

$$(M_s^i)_j = \max((\mathcal{R}^{-1}(s_i))_j), \qquad (m_s^i)_j = \min((\mathcal{R}^{-1}(s_i))_j), \tag{5.15}$$

for $j = 1,\dots,n_x$ where $(\cdot)_j$ is the j-th component of the set. Following the procedure illustrated in [36], the logical conditions in (5.14) can be rewritten using the inequalities:

$$m_s^i \delta_k^i \le x_k \delta_k^i \le M_s^i \delta_k^i \quad \text{for } i = 1,\dots,N_s;\ k = 1,\dots,N. \tag{5.16}$$

These inequality constraints cannot be used directly into an optimization problem due to the nonlinearity introduced by the product $x \cdot \delta$. Therefore, we define $N_s \cdot (N+1)$ of auxiliary variables $z_k^i = x_k \delta_k^i$ and we rewrite (5.16) as:

$$\begin{aligned} &z_k^i \le M_s^i \delta_k^i \\ &z_k^i \ge m_s^i \delta_k^i \\ &z_k^i \le x_k - m_s^i (1 - \delta_k^i) \\ &z_k^i \ge x_k - M_s^i (1 - \delta_k^i) \quad \text{for } i = 1,\dots,N_s;\ k = 1,\dots,N. \end{aligned} \tag{5.17}$$

These inequalities force the variables $z_k^i = x_k^i$ when x_k is in the region $\mathcal{R}^{-1}(s_i)$, i.e. $\delta_k^i = 1$, and make $z_k^i = 0$ elsewhere. In addition, the (5.17) can only be satisfied if (5.14) holds, thus linking the state of the system to a unique state in the abstraction.

The next step is to link the control input to the policy. For this, we can use a linear combination of the L_p-balls. It is immediate to verify that the control input u_k at each time step k must belong to the union of $B(u_i, \epsilon)\ \forall u_i$, i.e. $u_k \in \bigcup_{i=1}^{N_s} B(u_i, \epsilon)\ \forall k \in \{1,\dots,N\}$. Using a definition similar to (5.15), we introduce a number N_s of n_u-vector boundaries M_a^s, m_a^i for the L_p-balls:

$$(M_a^i)_j = \max(B(u_i, \epsilon))_j), \qquad (m_a^i)_j = \min((B(u_i, \epsilon))_j) \tag{5.18}$$

Therefore, leveraging the binary variables δ, we can write:

$$\sum_{i=1}^{N_s} m_a^i \delta_k^i \le u_k \le \sum_{i=1}^{N_s} M_a^i \delta_k^i \quad \text{for } i = 1,\dots,N_s;\ k = 1,\dots,N. \tag{5.19}$$

For the input constraints, only one element of the sums can be nonzero at each time step, thanks to the relation of the binary variables with the abstraction. Therefore, at each time step, the control input can exclusively belong to the L_p-ball associated with the policy defined by the respective state in the abstraction.

5.7.3 MPC Formulation

To complete the definition of the MPC problem, we introduce a mathematical model of the system $f^{\text{MPC}}(x_k, u_k)$, based on PWA approximation for nonlinear systems, which provides the predicted state behavior across the optimization horizon N; and a stage cost $J(x_k, u_k)$ that captures the performance of the system, and that has to be minimized. We use the notation $j|k$ to represent the prediction step $j = 1,\dots,N$ given the measured state of the system at the time step k. Then, the optimization program associated to the MPC problem at time step k is:

$$
\begin{aligned}
\min_{\substack{x_{1|k},\dots,x_{N|k}\in\mathcal{X}\\ u_{0|k},\dots,u_{N-1|k}\in\mathcal{U}}} \quad & \sum_{j=1}^{N} J(x_{j|k}, u_{j-1|k}) \qquad (5.20)\\
\text{s.t.} \quad & x_{k+1} = f^{\text{MPC}}(x_k, u_k)\\
& x_{0|k} = x_k\\
& z^i_{j|k} \le M^i_s \delta^i_{j|k}\\
& z^i_{j|k} \ge m^i_s \delta^i_{j|k}\\
& z^i_{j|k} \le x_k - m^i_s(1-\delta^i_{j|k})\\
& z^i_{j|k} \ge x_k - M^i_s(1-\delta^i_{j|k})\\
& \qquad \text{for } i = 1,\dots,N_s;\ j = 0,\dots,N\\
& u_{j-1|k} \le \textstyle\sum_i^{N_s} M^i_a \delta^i_{j-1|k}\\
& u_{j-1|k} \ge \textstyle\sum_{i=0}^{N_s} m^i_a \delta^i_{j-1|k}\\
& \qquad \text{for } j = 1,\dots,N\\
& \textstyle\sum_{i=1}^{N_s} \delta^i_{j|k} = 1\\
& \qquad \text{for } j = 0,\dots,N
\end{aligned}
$$

In the experiments in Sec. 5.8, we use (5.20) with a standard quadratic stage cost $J(x_k, u_k) = \|x^{\mathrm{T}} - x_k\|_Q + \|u_k\|_R$, where $\|\cdot\|_Q$ and $\|\cdot\|_R$ are the quadratic forms for positive definite weighting matrices Q and R. This stage cost allows optimising control effort, but also, e.g., minimizing the distance to a target state[6] x^{T}. This type of cost function enables the optimizer to find a control solution that is the contribution of two opposing effects: on one side, the target error forces the control action to drive the system toward the destination, while, on the other, the control effort tries to minimize the energy required to execute the action.

[6] For reach-avoid specifications, it is common to consider a target area rather than a single specific point of the state space, as is instead common practice in MPC. In this setting, we consider the point in the target area closest to the current measured state. This is done by constructing a table in which each region of the partition used for the abstraction is associated with the closest target cell. We perform these computations offline and make the resulting table available to the MPC.

5.8 Numerical Experiments

We validate our framework on several benchmarks from the literature to answer two questions about our approach:

Q1 How does the size ϵ of the L_p-balls affect the lower bound λ on the satisfaction probability?

Q2 Can our abstraction-driven MPC controller reduce the value of the cost function J, while retaining the certified bound on the satisfaction probability from the abstraction?

We consider well-known benchmarks, which, in order of increasing complexity, are a double integrator, mountain car [328], and a Dubins car. To construct finite-state IMDP abstractions, we build upon the implementation from [20], which we extend to generate L_∞-balls around each action with a radius of ϵ. We use the model checker Storm [131] to compute optimal policies for IMDPs, and Gurobi [123] to solve the MPC optimization problems. All experiments are run on a laptop with an Intel i7-1185G7 CPU, and 16 GB of RAM. We will make the code available via the Repeatability Evaluation.

5.8.1 Benchmarks and IMDP Abstractions

Double integrator

We consider the linear discrete-time stochastic dynamics described by

$$x_{k+1} = \begin{bmatrix} 1 & \tau \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} \frac{\tau^2}{2} \\ \tau \end{bmatrix} u_k + w_k, \tag{5.21}$$

with the sampling time $\tau = 1$, stochastic noise $w_k \sim \mathcal{N}(0, 0.15 I_2)$, state space $\mathcal{X} = [-21, 21]^2$, and input constraints defined as $u_k \in \mathcal{U} = [-5, 5]$. The reach-avoid specification is to reach $\mathcal{X}_G = [-4, 4] \times [-2, 2]$ without leaving $\mathcal{X}$. We partition the state space into $21 \times 21 = 441$ regions and define $|\mathcal{A}| = 21$ abstract actions, each associated with an L_∞-ball with the centers $u_1, \ldots, u_{21}$ forming a uniform gridding of $\mathcal{U}$ and with a fixed radius ϵ (as specified in Sec. 5.8.2).

Mountain car

The discrete-time nonlinear dynamics considered for the mountain car is [328]:

$$v_{k+1} = v_k + \tau \left(P u_k - g \cos(p_k) \right) + w_k^{[2]}, \tag{5.22}$$

$$p_{k+1} = p_k + \tau (v_{k+1} - w_k^{[2]}) + w_k^{[1]}, \tag{5.23}$$

where $p_k \in [-1.2, 0.6]$ is the position and $v_k \in [-0.07, 0.07]$ the velocity, $\tau = 2$, $P = 0.0015$, and $g = 0.0025$. The stochastic noise is distributed as $w_k \sim \mathcal{N}(0, \mathrm{diag}(0.005, 0.0005))$, and the input is contrained as $u_k \in \mathcal{U} = [-1, 1]$. In addition, the position dynamics is affected by the nominal velocity update v_{k+1}, without the effect of the noise $w_k^{[2]}$. The reach-avoid specification is to reach $\mathcal{X}_G = [0.45, 0.6] \times [-0.07, 0.07]$ without leaving $\mathcal{X}$. We partition $\mathcal{X}$ into 360×140 cells and define $|\mathcal{A}| = 5$ actions, each associated with an L_∞-ball with uniform gridding of $\mathcal{U}$ and with a fixed radius ϵ.

Dubins car

We consider a 3D Dubins car, where the nonlinear discrete-time dynamics is described by

$$x_{k+1} = x_k + \tau u_k^{[2]} \cos\theta_k, \tag{5.24}$$

$$y_{k+1} = y_k + \tau u_k^{[2]} \sin\theta_k, \tag{5.25}$$

$$\theta_{k+1} = \text{wrap}\left[\theta_k + \tau\alpha, u_k^{[1]}\right] \tag{5.26}$$

where $x_k \in [-10, 10]$, $y_k \in [-10, 10]$, and $\theta_k \in [-\pi, \pi]$ are the X and Y positions and the steering angle, respectively, and the two inputs represent the change in steering angle $u_k^{[1]} \in [-\pi/2, \pi/2]$ and the driving speed $u_k^{[2]} \in [-3, 3]$. The parameters are set as $\tau = 1$ and $\alpha = 0.85$. The function 'wrap' brings the angle update θ_{k+1} within the range $[-\pi, \pi]$. The reach-avoid specification is to reach $\mathcal{X}_G = [-10, -5] \times [5, 10] \times [-\pi, \pi]$ while avoiding the black states shown in Fig. 5.8 and while not leaving $\mathcal{X}$. We partition the state space into $20 \times 20 \times 11$ cells and define 7×5 actions, again with L_∞-balls with uniformly gridded centers and with a fixed radius ϵ.

5

5.8.2 IMDP Abstraction and Selection of ϵ

To answer Q1, we generate the IMDP abstraction (as described in Sec. 5.5, 5.6) for the different values of ϵ shown in Tab. 5.1. For each IMDP, we compute an optimal robust policy $\sigma^\star$ as in (5.5) and the corresponding lower bound on the satisfaction probability $\lambda = \min_{P \in \mathcal{P}} \mathbb{P}_{\mathbf{M}}^{\sigma, P}(\mathrm{Y})$ for a fixed initial state (which is the same as the one used for the simulations in Sec. 5.8.3). In addition, Tab. 5.1 shows the *coverage* of $\mathcal{U}$, which is computed as the ratio between the total volume of all L_∞-balls and the volume of $\mathcal{U}$. From an optimization perspective, the $\mathcal{U}$ coverage can be an indicator for the chance of obtaining good performance in the online MPC control.

The results in Tab. 5.1 show that by increasing the value of ϵ, i.e. by enlarging the L_∞-ball $\mathcal{F}_{\text{set}}(x, \sigma(s_i)) = B(u_i, \epsilon)$, the value of λ decreases, but not in a linear fashion. In particular, we observe that increasing ϵ will slowly decrease λ until an *"elbow"* point where a sharp decrease occurs; we thus select this point as the best trade-off for the selection of ϵ.

The heat maps in Fig. 5.3 to 5.5 show the lower bound on the satisfaction probability λ for *any* initial state $x_0 \in \mathcal{X}$, for different benchmarks and values of ϵ. These figures confirm that a higher ϵ might decrease the lower bound on the satisfaction probability, but by how much depends strongly on the particular benchmark.

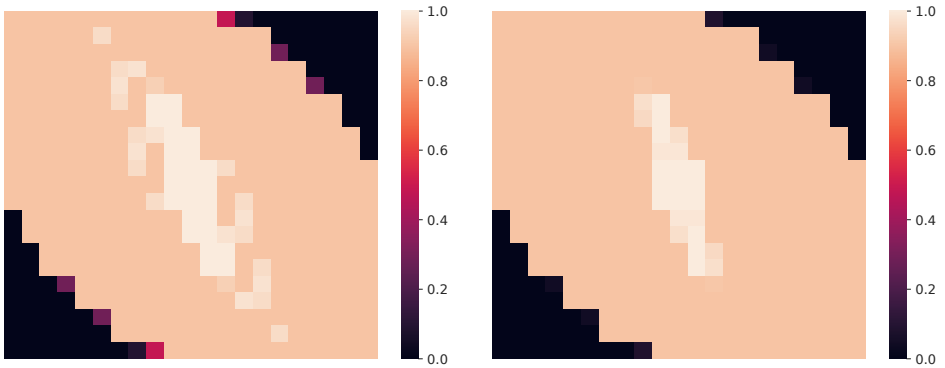
5.8.3 Simulation Results

Next, we answer Q2 by running simulations under the MPC controller as described in Sec. 5.7. For each benchmark and value of ϵ, we run 100 simulations of the closed-loop (stochastic) system under the MPC controller. The performance for $\epsilon = 0$ is equivalent to a vanilla IMDP abstraction (as in, e.g., [20, 181]) and is thus the baseline to which we compare our performnace. For fair comparison, we use the same sequence of noise values on a simulation horizon N_{sim}, i.e., $\{\{w_{i,k}\}_{k=0}^{N_{\text{sim}}-1}\}_{i=1}^{100}$, for different values of ϵ of the same benchmark.

The simulation results are presented in Tab. 5.2. For each benchmark and ϵ, Tab. 5.2 shows the average total cost $\mathbb{E}[J]$ (over the 100 simulations), as well as the decomposition

Table 5.1: Values of ϵ used for the abstraction.

Double Integrator		
ϵ	λ	$\mathcal{U}$ coverage
0	0.9987	0
0.1	0.9922	0.42
0.3	0.8918	1.26
0.5	0.8979	2.1
Mountain Car		
ϵ	λ	$\mathcal{U}$ coverage
0	0.9911	0
0.1	0.9865	0.5
0.15	0.9640	0.75
0.20	0.7440	1
Dubins Car		
ϵ	λ	$\mathcal{U}$ coverage
$[0;0]$	0.9969	0
$[0.1;0.2]$	0.9952	0.2769
$[0.15;0.3]$	0.9922	0.4152
$[0.2;0.4]$	0.1206	0.5536



$\epsilon = 0.3$ $\epsilon = 0.5$

Figure 5.3: Heat maps of the lower bound of the satisfaction probability for the double integrator.

into the reference error $\mathbb{E}[J_{\text{state}}]$ and the control effort $\mathbb{E}[J_{\text{input}}]$ (such that $\mathbb{E}[J] = \mathbb{E}[J_{\text{state}}] + \mathbb{E}[J_{\text{input}}]$). Furthermore, we also compare the average length $\mathrm{L}_{\text{trace}}^{\text{average}}$ of each trajectory until reaching the goal. We introduce this additional metric for fairness in evaluating the results since the minimization of the Euclidean distance from the current state to the closest target cell might not always be the best indicator of performance when a task-driven policy is involved. The metric $\mathrm{L}_{\text{trace}}^{\text{average}}$ quantifies the efficiency of the paths resulting from different control approaches across different initial samples and noise sequences. The final two columns show the time required for the abstraction (T^{abs}) and the average time required to solve the optimization problem for the MPC at each time step k ($\mathrm{T}_{\text{step}}^{\text{MPC}}$). Fig. 5.6 to 5.8 show simulated state trajectories under both the vanilla abstraction policy, versus the MPC controller.

Double integrator

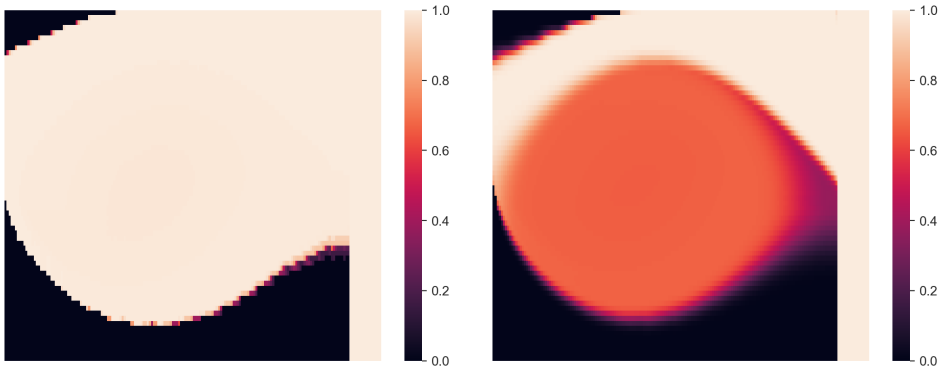
We consider two different prediction horizons $N = 3, 5$ and weighting matrices $Q = \text{diag}(1, 1)$, $R = 1$. The time required to construct the MPC model is approximately 28 seconds for all cases. For both $N = 3, 5$, the best performance is achieved for $\epsilon = 0.5$, which reduces the cost by 11.6% at a loss in λ around 10%, and trajectories that are 8% shorter. Furthermore, the cost reduction is slightly higher for $N = 5$ than $N = 3$, showing that a longer horizon indeed allows for better online optimization, at the cost of higher time ($\mathrm{T}_{\text{step}}^{\text{MPC}}$) to solve each MPC instance.

Mountain car

We consider an MPC horizon of $N = 3$ and weighting matrices $Q = \text{diag}(1, 0)$, $R = 1$. The time to construct the general MPC problem is around 6 minutes. In this case, the optimization shows significant performance improvement for $\epsilon = 0.15$, with a relatively low loss in λ of 2.7%. Of particular relevance is the improvement in the control effort, with a gain of 61.5%, which can have a huge impact on the energy requirements to execute the traces. This is achieved with a minor increase in the length of the trajectories executed, about 6.6%, as

Table 5.2: MPC simulations results for all benchmarks.

Double Integrator with $N = 3$						
ϵ	$\mathbb{E}[J]$	$\mathbb{E}[J_{\text{state}}]$	$\mathbb{E}[J_{\text{input}}]$	$L_{\text{trace}}^{\text{average}}$	T^{abs} $[s]$	$T_{\text{step}}^{\text{MPC}}$ $[s]$
0 (baseline)	141.21	74.32	66.90	21.46	6.53	–
0.1	137.42	73.48	63.93	21.33	6.99	0.23
0.3	137.61	74.99	62.62	21.52	9.23	0.18
0.5	126.58	71.05	55.53	19.75	11.92	0.18
Double Integrator with $N = 5$						
ϵ	$\mathbb{E}[J]$	$\mathbb{E}[J_{\text{state}}]$	$\mathbb{E}[J_{\text{input}}]$	$L_{\text{trace}}^{\text{average}}$	T^{abs} $[s]$	$T_{\text{step}}^{\text{MPC}}$ $[s]$
0 (baseline)	141.21	74.32	66.90	21.46	6.53	–
0.1	137.49	73.17	64.31	21.31	6.99	0.53
0.3	136.84	74.48	62.37	21.41	9.23	0.74
0.5	124.79	70.38	54.41	19.73	11.92	0.74
Mountain Car						
ϵ	$\mathbb{E}[J]$	$\mathbb{E}[J_{\text{state}}]$	$\mathbb{E}[J_{\text{input}}]$	$L_{\text{trace}}^{\text{average}}$	T^{abs} $[s]$	$T_{\text{step}}^{\text{MPC}}$ $[s]$
0 (baseline)	192.80	107.42	85.38	3.11	743.69	–
0.1	90.96	58.02	32.94	3.32	750.12	7.05
0.15	91.82	59.02	32.80	3.33	773.65	7.04
0.2	92.38	59.14	33.23	3.11	904.07	7.07
Dubins Car						
ϵ	$\mathbb{E}[J]$	$\mathbb{E}[J_{\text{state}}]$	$\mathbb{E}[J_{\text{input}}]$	$L_{\text{trace}}^{\text{average}}$	T^{abs} $[s]$	$T_{\text{step}}^{\text{MPC}}$ $[s]$
$[0;0]$ (baseline)	1360.18	1244.89	115.28	47.88	72.80	–
$[0.1;0.2]$	1415.68	1310.64	105.03	43.89	79.16	8.86
$[0.15;0.3]$	1539.89	1438.25	101.64	45.80	98.76	8.81
$[0.2;0.4]$	1653.04	1552.56	100.47	41.15	3202.02	10.16



$\epsilon = 0.1$ $\epsilon = 0.2$

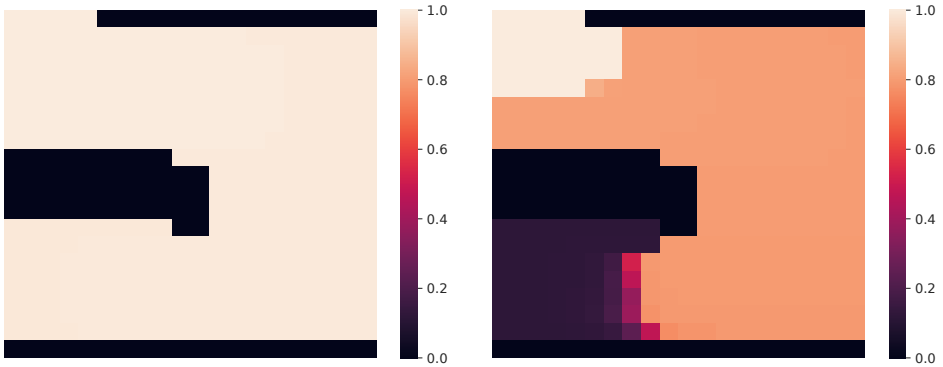
Figure 5.4: Heat maps of the lower bound of the satisfaction probability for the mountain car.

shown in the column $\mathrm{L}_{\mathrm{trace}}^{\mathrm{average}}$. A graphical representation of this simulation is given in Fig. 5.7.

Dubins car

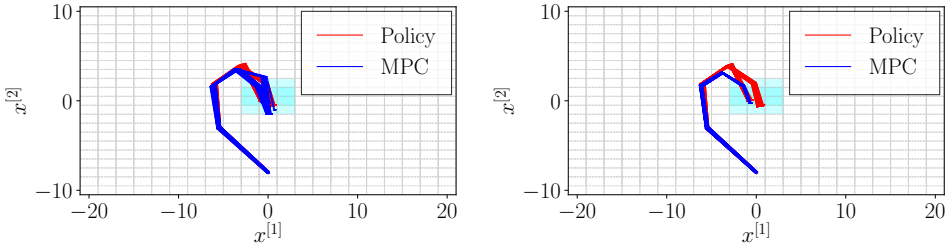
We consider an MPC horizon $N = 3$ and weighting matrices $Q = \mathrm{diag}(1,1,0)$, $R = \mathrm{diag}(1,1)$. The time required to construct the general MPC problem is approximately 3 minutes. Interestingly, for this benchmark, the MPC actually increases the cost function J. This increase in the cost J is caused by the *shortsightedness* of the MPC (a short horizon of $N = 3$ steps), which cannot optimize across the length of the logically-constrained trajectories provided by the policy. This fact is illustrated around the third time step in Fig. 5.8, where the MPC controller pushes the blue traces up toward the goal, without accounting for the effect of the coming constraints in the L_∞-balls: this leads to a cost increase for the Dubins car. At the same time, the MPC controller decreases the average trajectory length ($\mathrm{L}_{\mathrm{trace}}^{\mathrm{average}}$), showing a gain in path efficiency. For this benchmark, the best trade-off between increasing $\mathbb{E}[J_{\mathrm{state}}]$ and decreasing $\mathbb{E}[J_{\mathrm{input}}]$ and $\mathrm{L}_{\mathrm{trace}}^{\mathrm{average}}$ may be obtained for $\epsilon = [0.1; 0.2]$, which, for a 1% reduction of λ, yields control effort reduction of 12%, and 8% shorter trajectories.

Remark 5.3. The complexity of the MIQP for the MPC problem grows exponentially with the number of cells in the state space partition. Thus, constructing a single MIQP for the entire problem can be prohibitive if the abstraction is too fine-grained. Alternatively, we can consider a different 'general' problem for each cell of the partition, thus constructing a set of N_s smaller problems, which are then updated with constraints accounting for the measured state. Such an approach can significantly improve computational performance, although constructing these local problems requires computing the subset of reachable cells within the prediction horizon.



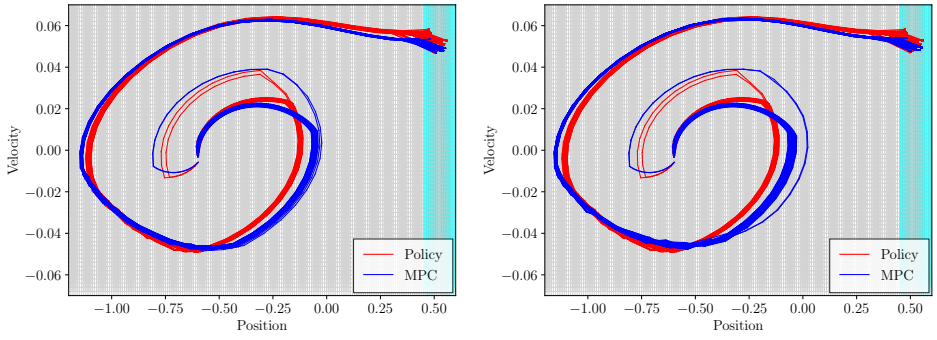
$\epsilon = [0.15; 0.3]$ $\epsilon = [0.2; 0.4]$

Figure 5.5: Heat maps of the lower bound of the satisfaction probability for the Dubins car.



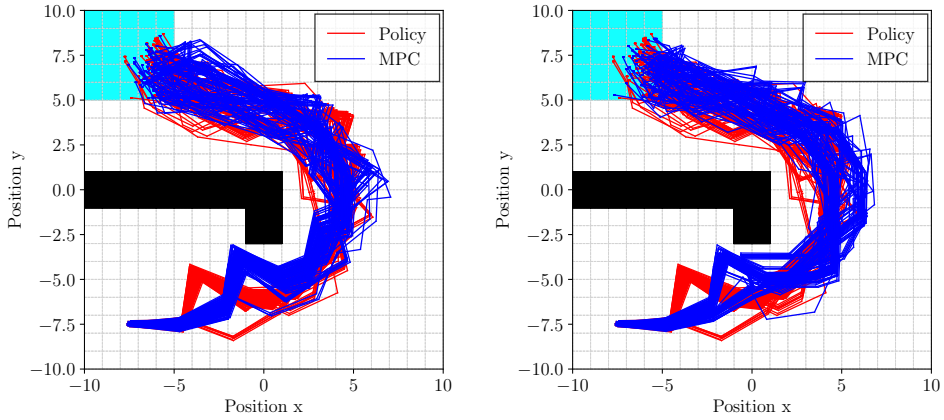
$N = 3$, $\epsilon = 0,3$ $N = 5$, $\epsilon = 0,5$

Figure 5.6: Simulated state trajectories for the double integrator, showing the baseline without MPC (policy) and the MPC.



$\epsilon = 0.1$ $\epsilon = 0.2$

Figure 5.7: Mountain car simulations with different ϵ for the baseline (policy; red) and our MPC controller (blue).



$\epsilon = [0.1; 0.2]$ $\epsilon = [0.15; 0.3]$

Figure 5.8: Dubins simulations with different ϵ for the baseline without MPC (policy; red) and our MPC controller (blue).

5.9 Conclusions

We have presented a novel policy synthesis framework for discrete-time stochastic systems that integrates offline abstraction with online model predictive control (MPC). In the abstract model, which we represented as an interval Markov decision process (IMDP), each action is associated with a set of control inputs for the dynamical system over which we reason robustly. By performing robust value iteration on the abstract IMDP, we obtain a set of policies for the dynamical system, each of which satisfies a given logical specification with a certified minimum probability. Online, we use MPC to further optimize a desired cost function (e.g., total control effort) by choosing inputs within this certified set of policies from the abstraction. The resulting MPC controller optimizes the cost function while retaining the certified lower bound on the satisfaction probability from the abstraction. Our experiments demonstrated that our approach yields better control performance than vanilla abstraction techniques, with only a small degradation of the certified lower bound on the probability of satisfying the specification.

To further improve performance, we intend to explore adaptive abstraction schemes that use a variable ϵ (the size of the L_p-balls to define actions) across the state space. Moreover, to improve the tightness of the abstract model, we will investigate using models other than IMDPs to represent the abstraction, e.g., as done in [115, 219]. Finally, we will explore other approximations for the MPC solution that will improve the online computation times.

6

Conclusions and Recommendations

"No, life is not fair. Not intrinsically. It's something we can try to make it, though, a goal we can aim for. You can choose to do so, or not."

Iain M. Banks, The Player of Games, 1988

In this thesis, we have approached the problems of partitioning for non-centralized model predictive control (MPC) of networks of systems, and of the integration of abstraction-based policy synthesis with online optimization for complex systems. The first research line falls within the field of control of large-scale multi-agent networks, while the second falls within the field of control of complex systems. In this chapter, we complete the discussions developed throughout the thesis by providing a summary of the main research contributions and presenting potential directions for future research. Additionally, we highlight the potential synergies among the two research topics that can be explored in further work for the integration of the methods presented, which could lead to the development of strategies for the predictive control of networks of systems under dynamic topologies while satisfying safety-critical specifications.

6.1 Summary of Research Contributions

The main contributions of this thesis have been discussed in the Chapters 2–5, and we summarize them briefly in the following:

1. **Development of a benchmark for non-centralized control of multi-agent systems**: We have provided the implementation of a benchmark system representing an abstract formulation of the electricity network of the European economic area. The benchmark is constructed using real-world data about renewable electricity production and load demand. It can be used for the development of non-centralized control strategies, and reference studies using centralized MPC, and distributed MPC are available for direct performance comparison. The benchmark has been used already to test partitioning strategies [278] and to develop an integrated multi-agent reinforcement learning and MPC control approach [317]. Versions of the benchmarks are available both in MATLAB and Python, with linear or hybrid definition of the energy storage systems dynamics [279, 280].

2. **Definition of a generalized partitioning strategy for non-centralized MPC control**: We have approached the problem of partitioning a network for the application of non-centralized MPC by deriving a unified approach for bottom-up and top-down approaches. The resulting strategy is a two-step procedure in which first the fundamental units of the network are precisely defined, and then they are aggregated to optimize a global performance metric. We provide several aggregation procedures, based on specialized and evolutionary algorithms, or on the solution of a mixed-integer optimization problem, where each approach has different strengths and limitations. We empirically demonstrated the efficacy of our partitioning strategy through the application of distributed MPC to networks with different topologies, obtaining substantial improvements in computational cost for linear systems, and control performance for hybrid systems.

3. **Systematization of the work in the field of partitioning for non-centralized MPC control**: We have provided a systematic review of the work in partitioning for non-centralized MPC control. We have approached the topic using both a synthetic and an analytic approach. First, we have provided a classification of the strategies based on their partitioning methodology, objective, and other criteria relevant for control. Then we have provided a brief technical discussion of each strategy. The survey has been completed with a a classification of the application domains in which each strategy has been applied, where the relevant benchmarks available are also reported.

4. **New theoretical insights for partitioning in non-centralized MPC control**: We have provided a definition of optimal partitioning for performance maximization, from which we have concluded that partitioning for non-centralized MPC is essentially a multi-level optimization problem. Therefore, multi-step procedures are necessary for its solution. This formalization has also allowed us to introduce the concept of predictive partitioning, in which the allocation of fundamental systems into composite units is given by the predicted behavior of the network state and topology.

Such an approach is supported by the additional concept of multi-topological graph representations that using the hybrid systems formalism allow the introduction of logic and event-driven behaviors in the topology dynamics.

5. **Development of an abstraction-driven MPC controller for nonlinear stochastic systems**: We have achieved the integration of policy synthesis for the satisfaction of safety critical specification with online performance optimization by deriving an abstraction-driven MPC controller. We consider a nonlinear stochastic system, for which we produce an abstraction using a modified definition of interval Markov decision processes (IMDPs) in which the actions correspond to portions of the input space. Then, using a hybrid system formulation and mixed integer programming, we define an optimization problem that selects the input regions and the inputs within such regions over a prediction horizon on the basis of a piecewise linear approximation of the nonlinear system. The resulting architecture allows for the satisfaction of a given specification within a predefined probability bound, while improving control performance w.r.t. conventional policies obtained using robust dynamic programming. The architecture is developed over two distinct levels: an abstract level where the policy is derived, and a control level in which online optimization is executed using MPC as an optimizer to reconstruct the policy.

6.2 Suggestions for Future Research

During this research, we had the possibility to address part of some of the most relevant problems associated with non-centralized MPC control, and complex large-scale multi-agent networks. Together with our answers, we have found also new directions that we consider of interest for future work in this field. Specifically, reported in conceptual order w.r.t. the contributions in this thesis, we have:

- **Extensions of the EEA-ENB**: The benchmark that we have developed for non-centralized control of multi-agent systems and that represents an abstract version of the electricity network of the European economic area can be further enhanced, mainly in two directions. It can be extended with more detailed data about local generation, also adding further areas on regional or sub-regional levels, and defining accordingly the associated dispatchable generation sources, potentially with technical details about the production technology. Alternatively, the dynamics of local systems can be made more complex to capture nonlinearities or stochastic phenomena, both in local systems and on the transmission lines. Such extensions can help in validating advanced non-centralized control strategies.

- **Predictive partitioning:** We have introduced the definition of optimization problems that can partition the network according to the predicted behavior of the state of its fundamental systems. The natural step following this definition is the development of systematic studies addressing the potential research lines in this field. For example, we can consider the theoretical properties related to predictive partitioning for non-centralized MPC, which include the stability, the robustness, and the optimality of the overall architecture. In addition, uncertain topological descriptions can be considered, subject to disturbances or unknown phenomena; thus introducing

stochasticity in the network setting and the related partitioning problem. Moreover, additional research lines that can benefit from predictive partitioning include the topics of network resilience, privacy of operation and security, as well as topologies with time-varying sizes.

- **Network topology dynamics and multi-topological representations:** One aspect not studied in literature is the behavior of the topology of a network as a function of the state of its fundamental systems, external or logic-based events, or stochastic phenomena. Introducing the concept of multi-topological network representation through hybrid systems, we have defined a mathematical model for topology dynamics that can be used to predict network evolution, and potentially be used in a mixed-integer optimization program for performance optimization.
- **Abstraction-driven MPC control:** The novel multi-level MPC strategy that we have defined using the abstraction framework represented only the first step for the development of network control approaches that are able to comply with complex logical specifications. The main limitation in this regard is the computational complexity associated with both online mixed-integer optimization and the abstraction procedure. For the former aspect, non-centralized approaches can be potentially leveraged to improve the computational performance; for the latter, instead, we refer to the next point.

- **Compositional abstraction of complex large-scale multi-agent networks**: One approach to overcome the computational complexity associated with the abstraction procedure is to use a compositional abstraction approach, where elements in a network are abstracted individually and then composed to obtain an abstraction that, under a specified assumption, is approximately similar to the abstraction of the entire network. The compositional approach is naturally close to the partitioning problem; therefore, similar techniques for the definition of the best aggregation of fundamental units to abstract together can be deployed. Moreover, compositional abstraction for networks with stochastic topologies has not been performed yet, and in this regard, multi-topological network representation can also play a role. In this context, abstraction-driven MPC control for a network could be potentially applicable once a compositional abstraction of a network has been obtained. Merging all the elements discussed so far can lead to the definition of composition-abstraction-driven MPC control for the optimal satisfaction of safety-critical specifications in networks of nonlinear stochastic systems subject to dynamical topologies. This setting can capture most of the complexity of current networks of systems.

A

A Generalized Partitioning Strategy for Distributed Control

"When the past is always with you, it may as well be present; and if it is present, it will be future as well."

William Gibson, Neuromancer, 1984

The partitioning problem is a key problem for distributed control techniques. The problem consists in the definition of the subnetworks of a dynamical system that can be considered as individual control agents in the distributed control approach. Despite its relevance and the different approaches proposed in the literature, no generalized technique to perform the partitioning of a network of dynamical systems is present yet. In this article, we introduce a general approach to partitioning for distributed control. This approach is composed by an algorithmic part selecting elementary subnetworks, and by an integer program, which aggregates the elementary components according to a global index. We empirically evaluated our approach on a distributed predictive control problem in the context of power systems, obtaining promising performances in terms of reduction of computation speed and resource cost, while retaining a good level of performance.

This chapter is based on [278].

A

A.1 Introduction

The development of a distributed control architecture for a network of dynamical systems requires the selection of smaller subnetworks to be controlled independently by local controllers [303]. Coordination among local controllers is then achieved through a communication protocol [295]. The selection of the subnetworks is referred to as the *partitioning problem* [64]. In the literature concerning the partitioning problem, one can distinguish between two main approaches: i) the *top-down* approach, where a monolithic system is decomposed into smaller components defining the subnetworks; ii) the *bottom-up* approach, where individual systems, already predefined, aggregate to form bigger control entities. Despite being a central problem of distributed control a generalized partitioning technique is not present in the literature yet. With generalized approach we are referring to a partitioning technique that can be applied to every type of control system according to well-defined rules. The absence of a general theory unifying all the approaches leads to a superposition of terminology and concepts. Moreover, the technical problem of the definition of the number of subnetworks constituting the partitioning is not addressed in general, and it is often solved by heuristics or assumptions.

A.1.1 Contribution

In this work, we propose a novel approach to partitioning based on integer optimization and a novel global network metric. To approach the gaps in the literature, we propose a systematization of the definitions used in partitioning, supported by an initial mathematical formalization of the framework for a systematic approach to partitioning. Then, we define an algorithm for the selection of the elementary and indivisible control units in a network, to which we will refer to as *atomic control agents*. These elementary units will aggregate to constitute the subnetworks of the partitioning. Moreover, we define a global network metric accounting for both information about the dynamics and topology of the network. According to this novel metric, we define a partitioning strategy addressing simultaneously the two problems of: i) defining the number of subnetworks and ii) of which elementary system must belong to each set. The novel metric allows us to define the partitioning problem as an integer program. To illustrate the partitioning strategy, we use it to define the subnetworks for the application of a Distributed Model Predictive Control (DMPC) technique [288]. We test the resulting control architecture on a power system called the European Economic Area Electricity Network benchmark EEA-ENB [280].

A.2 Literature Survey

In the following we report some of the most common partitioning techniques used in the literature, classifying them according to two main approaches.

A.2.1 Top-Down Approaches

Among the most relevant approaches there are the partitioning techniques based on the use of the modularity metric [101], also referred to as community detection strategies [210, 298, 324]. Modularity is a metric involving both information about the state transition matrix of a linear dynamical system, and topological information about the network. The scope of modularity-based techniques is to maximize the modularity. Maximization of

modularity is known to be an NP-hard problem [45], and efficient algorithms have been developed to obtain approximate solutions to the problem of finding a partitioning of a network maximizing the modularity metric. The structure of these algorithms is based on the following pattern: given a certain network, modularity is maximized through an exchange procedure of state nodes across these communities until no further improvement in modularity is obtained.

Often partitioning techniques are derived in specialized contexts for ad-hoc applications. This is the case of the technique proposed in [306], where the partitioning of a wind farm is performed based on the wake-effect affecting down-stream turbines in the farm. In [29], flow-based distribution systems are considered, and more specifically a water distribution network. [29] proposes a multi-criteria optimization approach to obtain the partitioning. However, the number of sets of the partitioning have to be specified a priori.

A.2.2 Bottom-Up Approaches

A different approach to the partitioning problem is sought in coalitional control [94, 96] where control theory and game theory are used in combination. Coalitional control can be classified as a bottom-up approach, where individual control systems are aggregated into coalitions. These elementary control systems cannot be further divided, and constitute the fundamental blocks of a coalition, and therefore they are called agents. Each coalition, which is a collection of agents, is associated with a performance index, which is a function of the aggregated state and input values of the coalition. The performance cost is interpreted as an economic index, i.e. a transferable utility that can be reallocated among the agents. According to an iterative procedure, agents are exchanged across the coalitions until no further optimization of the coalitional cost is possible. Coalitional control inherits the complexity of the general clustering problem, thus requiring the solution of an NP-hard problem. A heuristic solution has been developed using binary quadratic programming in [67], thus providing better scalability for large systems when applying coalitional control.

Some partitioning techniques produce a non-stationary definition of the subnetworks. This is the case in the approaches defined in [11, 12], where linear switching systems and event-driven systems are considered. In these cases, it is necessary to establish the conditions that trigger a re-partition of the network, and the associated procedures to perform it.

Other works explore the effect of different choices of partitioning on the control properties of the systems, as done for distributed tube-based MPC in [27].

A.3 Preliminary Concepts and Notation

A preliminary concept required to approach partitioning techniques is the one of equation graph, or hypergraph, or network graph [303]. Consider a linear discrete-time dynamical system of the form:

$$x(k+1) = Ax(k) + Bu(k) \tag{A.1}$$

with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^p$, and A, B are matrices of appropriate dimensions. The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with the system is defined as a set of nodes $\mathcal{V} = \{u_1, \dots, u_p, x_1, \dots, x_n\}$, and a set of edges $\mathcal{E} = \{(i,j), |, i,j \in \mathcal{V}\}$, where an edge (i,j) exists if and only if a relation between nodes i and j is defined by a nonzero entry of matrices A or B. Moreover, we define a set of control

A

nodes $\mathcal{U} = \{u_1, \dots u_p\}$, and a set of state nodes $\mathcal{X} = \{x_1, \dots x_n\}$. Accordingly, the entries of matrix B can be seen as the weights of directed edges in $\mathcal{E}_B$ connecting nodes from set $\mathcal{U}$ to set $\mathcal{X}$, and the entries of matrix A as edges in $\mathcal{E}_A$ connecting nodes in $\mathcal{X}$. Thus, we can see a dynamical system also as composed of two graphs, denoted by $(\mathcal{U} \bigcup \mathcal{X}, \mathcal{E}_B)$ and $(\mathcal{X}, \mathcal{E}_A)$. We can indicate a dynamical system with the notation $\mathcal{S}_i = \{\mathcal{U}_i \bigcup \mathcal{X}_i, \mathcal{E}_{A_i} \bigcup \mathcal{E}_{B_i}\} = \{\mathcal{V}_i, \mathcal{E}_i\}$. For a given node i, the set of all nodes connected to it is called the neighborhood of i, and denoted by $\mathcal{N}_i = \{j \in \mathcal{V} \mid (i,j) \in \mathcal{E}_i\}$. If we define a subgraph $\mathcal{S}_i = (\mathcal{V}_i, \mathcal{E}_i)$, then the frontier of the subgraph is the set of nodes inside the subgraph that are connected to nodes outside the subgraph, and we denote it as $\mathcal{F}_i = \{i \in \mathcal{V}_i \mid (i,j) \in \mathcal{E}, j \in \mathcal{V} \setminus \mathcal{V}_i\}$. Once the graph associated with the system is defined, the partitioning problem is then converted into the problem of finding suitable subsets of nodes and edges defining sub-graphs constituting the subnetworks.

A.4 The Concepts of Control Agent and Atomic Control Agent

For a given dynamical system, the scope of a partitioning strategy is to select subsystems that can be considered as individual control units. This partitioning is generally obtained through specified metrics, topological or non-topological, and/or objectives, which might be related to the optimality of the control actions of individual control units, or other criteria.

In order to formally define a partitioning strategy that can have a general applicability for control systems, we introduce the concept of *control agent*:

Definition A.1 (Control Agent)**.** A control agent is a dynamical system whose inputs affect only the state dynamics of the agent itself. All the dynamic relations that the control agent has with other control agents occur through dynamical coupling among the states, or directed output-input connections.

In this chapter, we consider linear discrete-time time-invariant systems of the form (A.1). From Def. A.1, a control agent[1], indexed by i, will take the form:

$$\begin{aligned} x^{[i]}(k+1) &= A^{[i]}x^{[i]}(k) + B^{[i]}u^{[i]}(k) + w^{[i]}(k) \\ w^{[i]}(k) &= \sum_{j \in \mathcal{N}_i} A^{[ij]}x^{[j]}(k) \end{aligned} \tag{A.2}$$

where $x^{[i]} \in \mathbb{R}^{n_i}$, $u^{[i]} \in \mathbb{R}^{p_i}$, and $A^{[i]}$, $B^{[i]}$ are matrices of appropriate dimensions; $\sum_i n_i = n$, $\sum_i p_i = p$; $\mathcal{N}_i$ represents the neighborhood of agent i, $i \notin \mathcal{N}_i$; and $A^{[ij]}$ are matrices representing how the evolution of the states $x_j \in \mathcal{N}_i$ affect the dynamics of state x_i. The signal $w_k^{[i]}$ is the coupling effect that agent i experiences with its neighboring states. How

[1]Equation (A.2) defining our control agent is a slightly different version of this form is used to specify a general subsystem [64], where the term $w_k^{[i]}$ also contains the coupling through inputs $B^{[ij]}u_k^{[j]}$. This approach is more general, but not of practical use for distributed control. Instead, it is advisable to achieve a decomposition of the network such that terms $B^{[ij]}$ are equal to zero to avoid coupling through input signals among subsystems. This selection of subsystems is always possible, and in the worst case it will provide a single agent corresponding to the entire network.

this signal is interpreted is often at the basis of the definition of a non-centralized control strategy: in decentralized control, w_i is regarded as a disturbance; in DMPC, w_i may be considered as a bounded disturbance in non-iterative strategies, or as a known predicted signal over which agreement is sought through communication in iterative strategies [295].

In the definition of a general partitioning strategy, a fundamental task is to find a collection of smallest control agents in size that together cover the whole system, where the size of the control agent is the number of nodes constituting it. We specify this concept through the definition of atomic control agent.

Definition A.2 (Atomic Control Agent)**.** Given a network of dynamical systems, an atomic control agent is the smallest control agent definable through any decomposition of the network.

Atomic control agents represent the smallest individual components in a network of control agents. If a bottom-up partitioning approach is used, the definition of control agents is usually provided, but these might not coincide with the atomic control agents. We can set up a verification procedure to ensure these fundamental units are atomic control agents, and then proceed with the partitioning. Instead, in top-down partitioning, the definition of the atomic control agents is not present. These considerations are at the basis of the definition of an algorithm for the selection of atomic control agents. We conclude this section with a proposition that allows the construction of control agents as the result of the aggregation of other control agents. This result guarantees that the optimization-based partitioning will produce control agents by aggregation of the atomic control agents.

Proposition A.1. *The aggregation of multiple control agents is a control agent itself.*

Proof: The statement can be verified by construction. Consider two control agents, $\mathcal{S}_1 = (\mathcal{V}_1, \mathcal{E}_1) = (\mathcal{X}_1 \bigcup \mathcal{U}_1, \mathcal{E}_1)$, $\mathcal{S}_2 = (\mathcal{V}_2, \mathcal{E}_2) = (\mathcal{X}_2 \bigcup \mathcal{U}_2, \mathcal{E}_2)$. By definition of control agent i, for both $\mathcal{S}_1$ and $\mathcal{S}_2$ it holds that no edges are present from the set of nodes $\mathcal{U}$ to the set of states $\mathcal{X}_1$ or $\mathcal{X}_2$. Now we define the agent resulting from their aggregation as $(\mathcal{V}_1 \bigcup \mathcal{V}_2, \mathcal{E}_1 \bigcup \mathcal{E}_2 \bigcup \{(i,j),(j,i), i \in \mathcal{X}_1, j \in \mathcal{X}_2\})$. The aggregation merges the sets of edges, with the addition of the edges in $\mathcal{E}$ linking the two sets of nodes, but not present in the individual sets of edges. The operation does not add any edge connecting $\mathcal{U}_i$ to $\mathcal{X}_j$. Thus, the agent resulting from the aggregation is a control agent according to Def. A.1 since the operation preserves the characteristics required in the definition.

A.5 Algorithm for the Selection of Atomic Control Agents

The selection of atomic control agents produces fundamental entities that should be considered in a control strategy without further subdivisions. In our generalized approach, we first select the atomic control agents, and then determine the partitioning of the network by solving an integer program based on a global metric. This procedure is only a preliminary part of the overall network partitioning strategy, which will be completed by the optimization-based partitioning problem of **Section A.8**. We remark that, while the selection of atomic control agents may not be required for networks in which the smallest possible agents are given, it is required for top-down approaches to select the control

agents for the partitioning. Moreover, the selection algorithm can be useful in bottom-up approaches to verify that the elementary agents provided are atomic control agents. To define an algorithm to select the atomic control agents according to the Def. A.1 and **A.2**, we will use the graph representation of the network. We will indicate the atomic agents as sets $\mathcal{A}_i$, for $i = 1, \dots, N_{\text{Atomic}}$, where $N_{\text{Atomic}} \leq |\mathcal{U}|$, and the collection of all the atomic control agents as $\mathcal{A} = \{\mathcal{A}_1, \dots, \mathcal{A}_{N_{\text{Atomic}}}\}$. Also, we will make use of the additional set $\mathcal{L} \subset \mathcal{X}$, indicating the state nodes that remain to be assigned. We will use the superscript $[k]$ to indicate a certain set during k-th iteration. The algorithm consists of three main steps that we present in the following, and that will be performed in a specific order illustrated in **Algorithm A.1**:

1. *Identify the roots of the atomic control agents.* These roots consist of at least one input node and one state node directly connected by a link. If a state node is connected to two or more input nodes, then the entire group constitutes a root, and all are merged into the same atomic control agent. This procedure is performed on the graph $(\mathcal{U} \bigcup \mathcal{X}, \mathcal{E}_B)$. We can have a maximum number of roots equal to $N_{\text{Atomic}} \leq |\mathcal{U}|$, where $N_{\text{Atomic}} \leq |\mathcal{U}|$ is the maximum number of atomic control agents. The roots are the first components of the atomic control agents.

2. Perform the assignment of the state nodes not belonging to the roots for which a directed edge from the state nodes in the roots exists. We call this procedure *forward assignment*. Here the graph considered is $(\mathcal{X}, \mathcal{E}_A)$. For each state node in the roots, e.g. x_i, we scan the set of unassigned state nodes. We assign one free state, e.g. x_j, to the root of x_i if the element $|A_{ji}|$ is the maximum over the row $A_{j\cdot}$. Note that, in this case, we are considering the directed edge ϵ_{ji}, from x_i to x_j with weight A_{ji}, which is different from the edge ϵ_{ij}. We call this phase *forward assignment*, since we look at the states from the roots to the periphery of the graph.

3. Perform the *backward assignment*, from the nodes belonging to the periphery of the graph toward the direction of the roots. Following a similar procedure as above, given a state x_j for which a forward assignment is not possible, and a state x_i belonging to a certain atomic control agent, we assign x_j to the agent of x_i if $|A_{ij}|$ is the maximum over the column $A_{\cdot j}$. In this case we are considering the directed edge ϵ_{ji}.

Once atomic control agents have been selected, partitioning techniques for control strategies can be designed considering as the smallest control agents the sets $\mathcal{A}_i$. We have implemented the algorithm for the selection of atomic control agents described above, and made it available in the open source toolbox [277]. The toolbox is designed to perform the selection starting from any linear system description of the form (A.1) representing a complex network. An example of the application of the algorithm is presented in Sec. A.9. We conclude this section with a remark: if outputs are also specified in the dynamics of the system, the procedure defined above still holds, but it is necessary to assign the output nodes to each agent in the same way as done for the state nodes.

Algorithm A.1 Selection of the Atomic Control Agents

Part 1 - Selection of the roots
Given $(\mathcal{V} \bigcup \mathcal{X}, \mathcal{E}_B)$ perform step 1)
$(\mathcal{V}, \mathcal{X}) \rightarrow (\mathcal{A}^{[0]}, \mathcal{L}^{[0]})$
Part 2 - Selection of the atomic control agents
Given $(\mathcal{X}, \mathcal{E}_A)$
$k \leftarrow 0$
while $\mathcal{L}^{[k]}$ not empty $\vee$ $\mathcal{L}^{[k]}$ connected: **do**
    perform *forward assignment*: step 2)
    $(\mathcal{A}^{[k]}, \mathcal{L}^{[k]}) \rightarrow (\mathcal{A}^{[k+1]}, \mathcal{L}^{[k+1]})$
    perform *backward assignment*: step 3)
    $(\mathcal{A}^{[k+1]}, \mathcal{L}^{[k+1]}) \rightarrow (\mathcal{A}^{[k+2]}, \mathcal{L}^{[k+2]})$
    $k \leftarrow k+2$
end while

A.6 The Optimal Partitioning

Before proceeding with the definition of a partitioning algorithm, we want to specify some concepts that are of general relevance for distributed control strategies and partitioning techniques.

Definition A.3 (Control Partition of a Network)**.** The control partition $\mathcal{P}$ of a network is defined as the collection of m control agents:

$$\mathcal{P} = \{\mathcal{S}_1, \dots, \mathcal{S}_m\} \tag{A.3}$$

where control agents $\mathcal{S}_i$ are sets of atomic control agents $\mathcal{A}_j$.

For each set $\mathcal{S}_i$ of the partition, we define the set of its neighboring nodes as $\mathcal{N}_i = \{j \in \mathcal{X} \setminus \mathcal{V}_i \mid (i,j) \in \mathcal{E}_{A_i}\}$.

The definition of optimal partitioning strategy in the context of optimization-based control is related to the cost function used to obtain the control action, which we denote by J. When comparing a centralized control action u_c with a distributed u_d alternative applied to the same network, the optimal control partition $\mathcal{P}^*$ would be the one minimizing the difference $J(x, u_\mathrm{d}) - J^*(x, u_\mathrm{c})$. However, this index can not be computed for the cases in which the centralized control action u_c requires an intractable computation time. Thus, this is a theoretical index that can only be used to benchmark small-scale test cases and not as a cost function to obtain the optimal partitioning directly.

A.7 The Partitioning Index

A.7.1 The Two Main Problems

According to the discussion above, what we would like to obtain is a partitioning strategy that using information about the dynamics, and possibly the state, of the system will result in a distributed control cost as close as possible to the centralized strategy cost. As an example, for linear systems with linear constraints, in a completely decentralized control

setting, local optimizers will not account for the interaction with neighbors to retrieve the local control action. In the ideal case in which no interaction is present among agents, the sum of individual contributions will match the overall cost. The difference between the global cost and the sum of local costs will increase as the interaction among control agents strengthens. Thus, for decentralized MPC, it is intuitive to require a partitioning strategy that will lead to the definition of control agents that have the least possible interaction among them. A similar principle applies to distributed MPC when non-iterative algorithms are considered. A further extension of this principle will be the one of strengthening as much as possible the interaction among the atomic agents participating in the same set.

Another aspect to consider in designing a partitioning strategy is the definition of the number of sets constituting the partition. Often, this number is assumed to be fixed, obtained through heuristics, or with iterative evaluation of different numbers of sets. What we can do instead is to define an optimization problem that will try to minimize a global metric of the network. We select this metric as the ratio between the inter-agent interaction and the intra-agent interaction, simultaneously defining to which set of the partition an atomic agent belongs using binary variables.

A.7.2 The Metric

To merge the two previous objectives simultaneously we define a new metric called *partitioning index*. The scope is to obtain a simple and static metric for partitioning. A similar approach can be used to exploit other types of information about the system, or the cost function of the optimization-based controller. Assume that, on the basis of the atomic control agents, we have a certain partition, constituted by control agents according to Def. A.1 and A.2. We denote this partition as $\mathcal{P} = \{\mathcal{S}_1, \dots, \mathcal{S}_l\}$, with $l \leq m$. Each set $\mathcal{S}_i$ is obtained as the grouping of atomic agents $\mathcal{A}_j$. As a consequence, each set $\mathcal{S}_i$ is associated with a graph $(\mathcal{S}_i, \mathcal{E}_{\mathcal{S}_i})$ where the set of nodes groups all the nodes of the atomic control agents, and the set of the edges accounts for both the edges inside and between atomic control agents in the same set.

For each control agent $\mathcal{S}_i$ we define two indices, the intra-agent interaction $W^{\text{intra}}_{\mathcal{S}_i}$, and the inter-agent interaction $W^{\text{inter}}_{\mathcal{S}_i}$, which are functions of the sets $\mathcal{S}_i$, where the sets are not specified yet at this stage.

Each $\mathcal{S}_i$ is a network control agent by **Proposition A.1**, obtained as the union of atomic control agents. Thus, we denote the state and input matrices associated with $\mathcal{S}_i$ as $A_{\mathcal{S}_i}$ and $B_{\mathcal{S}_i}$. We define the intra-agent interaction as:

$$W^{\text{inter}}_{\mathcal{S}_i} = \frac{\left(\sum_{z \in \mathcal{F}_{\mathcal{S}_i}} \sum_{j \in \mathcal{N}_{\mathcal{S}_i}} A^2_{z,j}\right)^{\frac{1}{2}}}{\sum_{k=1}^{l} \left(\sum_{z \in \mathcal{F}_{\mathcal{S}_k}} \sum_{j \in \mathcal{N}_{\mathcal{S}_k}} A^2_{z,j}\right)^{\frac{1}{2}}} \tag{A.4}$$

where $\mathcal{F}_{\mathcal{S}_i}$ is the frontier of the control agent $\mathcal{S}_i$, i.e. the set of nodes connecting to nodes outside the control agent, and $A_{z,j}$ are the entries of matrix A corresponding to the weights of the edges connecting the frontier nodes of the control agent $\mathcal{S}_i$ to its neighbors belonging to other control agents. The second index is:

$$W^{\text{intra}}_{\mathcal{S}_i} = \frac{\|A_{\mathcal{S}_i}\|_{\text{F}}}{\sum_{k=1}^{l} \|A_{\mathcal{S}_k}\|_{\text{F}}} \tag{A.5}$$

where $\|\cdot\|_F$ indicates the Frobenius norm. Both indices are normalized, and are in the interval $[0,1]$. The normalization provides numerical results that will be suitable for comparing the ratios between different partitions, with different numbers of control agents.

With W_i^{intra} and W_i^{inter} we can associate to the partition $\mathcal{P} = \{\mathcal{S}_1, \dots, \mathcal{S}_l\}$ the following *specialized partition index*:

$$p^{\text{idx}}(\mathcal{P}) = \sum_{i=1}^{l} \left(\frac{W_{\mathcal{S}_i}^{\text{inter}}}{W_{\mathcal{S}_i}^{\text{intra}}} \right)^2 \tag{A.6}$$

The term p^{idx} constitutes our metric to define the partition minimizing the ratio between inter- and intra-agents coupling. The key feature of p^{idx} is that it accounts for *global* information about the network, which makes it a global metric. This fact will allow us to solve the two problems in Sec. A.7.1 simultaneously, as specified later in Sec. A.8 with the definition of the partitioning strategy.

A.8 The Partitioning Strategy

The strategy that we use for partitioning is based on the solution of a mixed integer optimization problem using the global metric (A.6). We introduce binary variables $\gamma_{ij} \in \{0,1\}$ to specify if an atomic control agent $\mathcal{A}_i$ belongs to a set $\mathcal{S}_j$:

$$\gamma_{ij} = 1 \iff \mathcal{A}_i \subset \mathcal{S}_j \tag{A.7}$$

We have in total a number of m^2 binary decision variables, since $i, j = 1, \dots, m$, where i is related to the number of atomic control agents, and j to the number of sets. The number of sets $\mathcal{S}_i$ that will be generated by the procedure is not known a priori but it can not exceed the number m of atomic control agents by definition. We solve this practical problem in the implementation allowing for empty sets in the partitioning optimization problem. To be precise, we define the general partition of the network as $\mathcal{P} = \{\mathcal{S}_1, \dots, \mathcal{S}_m\}$, where m is the number of atomic control agents. However, of the m sets in $\mathcal{P}$, a certain number $m-l$ may be empty sets, where l is the number of non-empty sets. All decision variables γ_{ij} are collected into the vector γ, and the partition of the network is now a function of this vector, i.e. $\mathcal{P}(\gamma)$. Then, we define the general mixed integer partitioning problem as:

$$\begin{aligned} &\min_{\gamma} \; p^{\text{idx}}(\mathcal{P}(\gamma)) \\ &\text{s.t.} \sum_{j=1}^{m} \gamma_{ij} = 1 \quad \text{for } i = 1, \dots, m \\ &\gamma_{ij} \in \{0,1\} \end{aligned} \tag{A.8}$$

where the equality constraint expresses the condition that each atomic control agent must belong exactly to one set, avoiding multiple assignments. This condition also ensures that all m atomic control agents are assigned. It can happen that the assignment will result in some sets being empty as specified before. In this case, we can just retain the l non-empty resulting sets. To use the metric (A.6) in the formulation (A.8) it is sufficient to notice that both $W_{\mathcal{S}_i}^{\text{inter}}$ and $W_{\mathcal{S}_i}^{\text{intra}}$ are functions of γ.

A

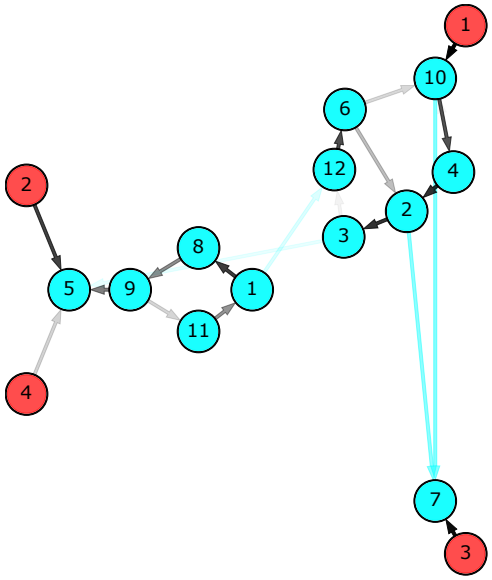


Figure A.1: Graph representation of the atomic control agents. Input nodes are in red, state nodes in blue. The nodes belonging to the same control agent are connected through black arrows. The interconnections among the control agents are represented by blue arrows. The transparency of the arrows represents the strength of interaction among nodes.

A.9 Example of Selection of Atomic Control Agents

In this example, we consider a network in the form (A.1) with 4 inputs and 12 states. We have sparse matrices A and B, which we report in the following indicating the element in the i-th row and j-th column in the subscript:

$$\begin{array}{llll} a_{1,11}=0.5 & a_{2,4}=0.98 & a_{2,6}=0.31 & a_{3,2}=0.97 \\ a_{4,10}=0.8 & a_{5,3}=0.11 & a_{5,9}=0.6 & a_{6,12}=0.84 \\ a_{7,2}=0.54 & a_{7,10}=0.58 & a_{8,1}=0.92 & a_{9,8}=0.53 \\ a_{10,4}=0.36 & a_{10,6}=0.18 & a_{10,10}=0.02 & a_{11,9}=0.2 \\ a_{12,1}=0.17 & a_{12,3}=0.06 & & \\ b_{5,2}=0.65 & b_{5,4}=0.15 & b_{7,3}=0.84 & b_{10,1}=0.87 \end{array} \tag{A.9}$$

We represent the network through a graph as indicated in Sec. A.3, thus having a set of input nodes $\mathcal{U}=\{u_1,\dots,u_4\}$, and of state nodes $\mathcal{X}=\{x_1,\dots,x_{12}\}$. No clear decomposition of this network can be established just by looking at the matrices. We can decompose this network by looking at its atomic control agents. Applying the algorithm presented in Sec. A.4 and implemented in the toolbox [277] we obtain the network in Fig. A.1, where input nodes are represented in red, and state nodes in blue. The selection procedure produces 3 atomic control agents from the original network, which has 4 inputs. Two inputs contribute to the same atomic control agent. Also, it is possible to see that two atomic control agents are not directly connected. The representation uses blue arrows to

highlight the connection between atomic control agents. Once this selection is performed, distributed control strategies as well as further partitioning strategies can be applied to the network.

A.10 Case Study: Partitioning for Distributed Predictive Control

A.10.1 System Description

In this section, we apply the partitioning strategy derived in this chapter to define the subnetworks of an electrical system for the application of a distributed control technique. The system considered is a benchmark for power networks called the European Economic Area Electricity Network Benchmark (EEA-ENB) [280]. It consists of 26 interconnected electrical areas, each representing a portion of the European power network. The topology of the network is illustrated in Fig. A.2.

Each electrical area is modeled as an equivalent electrical machine subject to a load request that has to be satisfied to maintain the network operating frequency of 50 [Hz] in the boundaries of ±0.04 [Hz]. The problem of regulating the frequency deviation to zero under load variations is referred to as the Load Frequency Control (LFC) problem.

The benchmark includes data regarding the load demand, and the renewable energy generation for each area, which represent known external signals. The presence of renewable generation, with its intermittent and inertia-less characteristics, complicates the solution of the LFC problem. To help compensate for the presence of renewable generation, each electrical area is equipped with an energy storage system (ESS).

The dynamics of each electrical area is represented by the linear discrete-time system

$$
\begin{aligned}
\Delta\delta_i(k+1) &= \Delta\delta_i(k) + \tau 2\pi \Delta f_i(k) \\
\Delta f_i(k+1) &= \left(1 - \frac{\tau}{T_{\mathrm{p},i}}\right)\Delta f_i(k) + \tau \frac{K_{\mathrm{p},i}}{T_{\mathrm{p},i}} g_i(k) \\
e_i(k+1) &= e_i(k) + \tau\left(\eta_i^{\mathrm{c}} P_i^{\mathrm{ESS,\,c}}(k) - \frac{1}{\eta_i^{\mathrm{d}}} P_i^{\mathrm{ESS,\,d}}(k)\right) \\
g_i(k) &= \Delta P_i^{\mathrm{disp}}(k) - \Delta P_i^{\mathrm{load}}(k) + \Delta P_i^{\mathrm{ren}}(k) - \Delta P_i^{\mathrm{tie}}(k) - P_i^{\mathrm{ESS,\,c}}(k) + P_i^{\mathrm{ESS,\,d}}(k) \\
\Delta P_i^{\mathrm{tie}}(k) &= \sum_{j \in \mathcal{N}_i} T_{ij}(\Delta\delta_i(k) - \Delta\delta_j(k)),
\end{aligned}
\tag{A.10}
$$

where $\Delta\delta_i$, Δf_i, and e_i are respectively the power angle and frequency deviations from nominal conditions, and the energy stored in the ESS of the i-th electrical area. These variables represent the state of each area. The inputs of the system are $\Delta P_i^{\mathrm{disp}}$, $P_i^{\mathrm{ESS,\,c}}$, and $P_i^{\mathrm{ESS,\,d}}$, respectively the dispatchable power allocation of the i-th area, and the charging and discharging powers of the ESS. The signal $\Delta P_i^{\mathrm{load}}$ and $\Delta P_i^{\mathrm{ren}}$ are the load demand and the renewable energy production. The electrical areas exchange power according to $\Delta P_i^{\mathrm{tie}}$ representing the power transmission between the i-th and its neighbors $j \in \mathcal{N}_i$. The other symbols define parameters of the system used to model the network. A detailed explanation of their meaning and selection is present in [280], and the source code and documentation

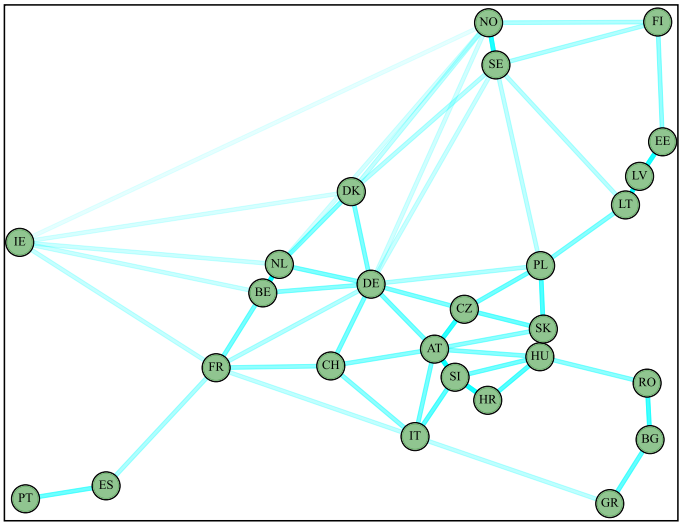


Figure A.2: Electrical topology of the EEA-ENB. Each node represents an electrical area, coinciding with a single country, whereas each edge is a transmission line.

for the benchmark are present at [276]. Data for the load and renewable generation of the European network have been acquired from [92].

A.10.2 Distributed Control Strategy

To address the LFC control problem using distributed control, we implemented a distributed model predictive control scheme based on the alternating direction method of multipliers (DMPC-ADMM) [49, 288, 321]. This approach has proven to be suitable for controlling large-scale systems where the application of centralized model predictive control (MPC) results unpractical, or not suited for real-time operation. In this case study we will compare the same DMPC-ADMM scheme implemented using different partitionings obtained with the strategy in Sec. A.8. The performance of the conventional DMPC-ADMM implementation differs only marginally from the one of a centralized MPC, which is in line with the literature [321]. In the DMPC-ADMM scheme, we select the parameter $\rho = 0.1$, as in [288].

A.10.3 Partitioning Strategy

In DMPC-ADMM it is usually assumed to have a network where control agents are already defined. In the following, we refer to this partitioning as $\mathcal{P}_{\text{DMPC}}$. In this context, $\mathcal{P}_{\text{DMPC}}$ will be our reference case for the other choices to compare performance, computation time, and computation resources, i.e. the number of [cores×seconds] necessary to run the implementation in parallel.

The first approach we consider involves the specialized partitioning index defined in (A.6). We use this index as cost for the integer program (A.8). The integer program is solved using the genetic algorithm implementation of Matlab. The resulting partitioning $\mathcal{P}_{\text{Opt,1}}$ consists of 10 control agents.

For the second approach to partitioning, we use a modified version of the index (A.6).

Table A.1: Comparison metrics of different partitionings.

Metric → Partitioning ↓	Optimality gap [%]	Parallel execution time [s]	Number of agents	Core seconds[s]	Cost gap [%]
$\mathcal{P}_{\text{DMPC}}$	0	113	26	2931	748
$\mathcal{P}_{\text{Opt,1}}$	3.56	88	10	884	156
$\mathcal{P}_{\text{Opt,2}}$	4.13	74	8	594	72
$\mathcal{P}_{\text{Rnd}}$	13.34	103	8	826	139
$\mathcal{P}_{\text{Geo}}$	15.44	43	8	346	0

We add the term $1/(1+|\bar{\mathcal{S}}_i|)$ to the cost of each control agent. Where $\bar{\mathcal{S}}_i$ is the system considered for the local DMPC optimization in ADMM. This system consists of the state and input variables of agent i and of all its direct neighbors. This choice aims to obtain the partitioning that simultaneously minimizes (A.6) and maximizes the size of the overlapping systems used in the local minimization of the stage cost for $\bar{\mathcal{S}}_i$. This choice will increase the complexity of local problems, because they will be larger, but will reduce the communication overhead, and possibly improve the agreement on shared variables among the control agents. We denote this partitioning as $\mathcal{P}_{\text{Opt,2}}$, and it has 8 control agents.

For an additional benchmark, we also consider a heuristic approach based on the geographical proximity of the atomic agents. We start from the consideration that the smallest previous partitioning is $\mathcal{P}_{\text{Opt,2}}$ with 8 control agents, thus we define 8 geographical areas by manual inspection on the network topology in Fig. A.2. We call this partitioning $\mathcal{P}_{\text{Geo}}$.

As a final comparison strategy, we select a random network partitioning, still with 8 control agents. We denote this as $\mathcal{P}_{\text{Rnd}}$. For each case, we use a static partitioning approach. This choice is motivated by the fact that the system at study is linear, and therefore the topology of its associated graph does not vary with time.

A.10.4 Partitioning Results and Discussion

For each of the five partitionings, we have executed the DMPC-ADMM strategy to control the EEA-ENB for the equivalent of one hour of network data using the same tuning parameters. We report the results of these executions in Tab. A.1, which we discuss in the following.

The first column is the optimality gap. In this aspect, partitioning $\mathcal{P}_{\text{DMPC}}$ is the best performer, and it is used as a reference. All other partitionings perform worse. However, the results for $\mathcal{P}_{\text{Opt,1}}$, and $\mathcal{P}_{\text{Opt,2}}$, which are obtained through optimization, are relatively close with a gap of less than 4.13%. For $\mathcal{P}_{\text{Geo}}$ and $\mathcal{P}_{\text{Rnd}}$ we start seeing a degradation of the performance with a gap of more than a 13.34%, which is much worse than the others. The second column is the execution time required to run the algorithm in parallel using the number of cores reported in the third column. In this category $\mathcal{P}_{\text{Opt,1}}$ and $\mathcal{P}_{\text{Opt,2}}$ have a significantly lower execution time than $\mathcal{P}_{\text{DMPC}}$ with a comparably small loss in performance. The fourth column represents the number of core seconds necessary to implement the strategy, and it is obtained as the product of the number of cores times the

parallel execution time. This measure is commonly used in parallel computing to express computation resource cost. We further clarify that core seconds are computed taking at each iteration the slowest performing agent. We sum the computation times of the slowest performer for each step to obtain the parallel execution time, which is the time that a number of cores equal to the number of agents have to operate in parallel to execute the algorithm in the least amount of time. When the algorithm is running, it occupies a number of computation resources (e.g. CPU cores) equal to the number of agents for an amount of time equal to the parallel computation time. With the product of these two factors, we obtain the amount of core seconds necessary to run the algorithm.

The value of the cost gap reported in the last column represents how expensive it is to run the algorithm in parallel in terms of computation resources cost w.r.t. $\mathcal{P}_{\text{Geo}}$, which has the lowest computation resources cost, and we take it as a reference in this category. The strategy $\mathcal{P}_{\text{Geo}}$ is the cheapest in terms of computation resources cost, but it also is the worst in terms of control performance. With strategy $\mathcal{P}_{\text{Opt,2}}$ we have an increase of 11.31% in performance from $\mathcal{P}_{\text{Geo}}$, but a computation cost raise of 72%. To further improve the performance of a 0.57% from $\mathcal{P}_{\text{Opt,2}}$ to $\mathcal{P}_{\text{Opt,1}}$, we need an additional computation cost of 84%. Strategy $\mathcal{P}_{\text{DMPC}}$ is the most expensive, but it also provides the best performance. Lastly, the random partitioning $\mathcal{P}_{\text{Rnd}}$ has both a high computation cost and bad performance.

According to these results, we can conclude that the most balanced partitionings are $\mathcal{P}_{\text{Opt,1}}$ and $\mathcal{P}_{\text{Opt,2}}$. These two strategies provide a comparably small loss in performance, but considerable improvements in computation cost. The selection between $\mathcal{P}_{\text{Opt,1}}$ and $\mathcal{P}_{\text{Opt,2}}$ can be performed according to the specific needs of the application. Moreover, the cost index (A.6) can be further refined to suit these needs.

We also present some simulation results for the EEA-ENB controlled with DMPC-ADMM for the different partitionings. The EEA-ENB consists of 26 areas, thus visualizing the state of each area on a single graph for five different partitionings is prohibitive. Therefore, to improve the interpretation of the results, we decided to show the evolution of the norm of each state for each partitioning. In Fig. A.3 we report the evolution of the power angle deviation from the nominal value. The results reflect the situation reported in Tab. A.1, with $\mathcal{P}_{\text{DMPC}}$ providing the smallest angle deviation, and $\mathcal{P}_{\text{Opt,1}}$, $\mathcal{P}_{\text{Opt,2}}$ closely following. Also, the gap with $\mathcal{P}_{\text{Rnd}}$ and $\mathcal{P}_{\text{Geo}}$ is evident in this regard. This situation is evident also in Fig. A.4, where we report the variation in dispatchable power allocation required to compensate for the load demand.

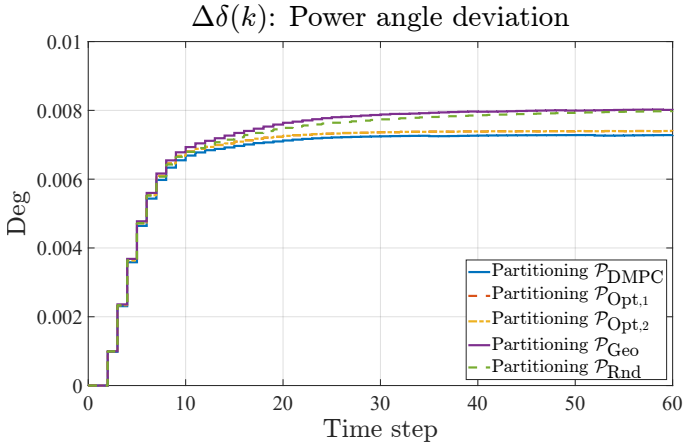


Figure A.3: Evolution of the power angle deviation.

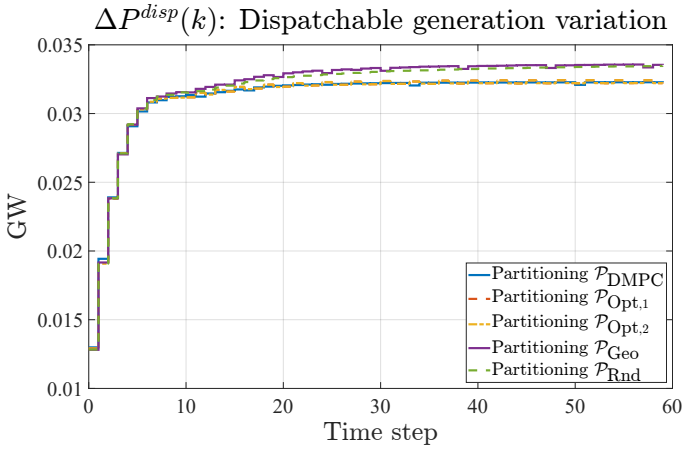


Figure A.4: Evolution of dispatchable generation.

A

A.11 Conclusions and Future Work

In this article, we have introduced a general method for partitioning and we have shown how an accurate selection of the partitioning in the application of distributed control can significantly reduce the computation speed and resource cost, with marginal performance degradation. To do this, first, we introduced the basic definition of *atomic control agent* and *control agent* to construct a generalized partitioning strategy, and we defined an algorithm to select the atomic control agents, and a global specialized partitioning metric. Then, this metric is used to partition the system through an integer program that returns both the number of sets and the elements in the sets of the partitioning. We provided a case study for the distributed predictive control of power systems in which we tested different partitionings, obtaining with our novel strategy, considerable improvements in the computation resources required to solve the distributed problem. Future work could extend the mathematical framework for the generalized partitioning technique with formal definitions of *atomic control agent* and *control agent*, extend it beyond linear systems, and perform an extensive assessment of the proposed method for a wide range of distributed control benchmarks. Providing a closed form for the optimization metric is another direction of interest. Also, ensuring control properties, such as controllability of the agents resulting from partitioning remains an open issue. Moreover, in a more general setting where online re-partitioning of large-scale networks is required, it is important to develop efficient online distributed optimization strategies for the partitioning problem.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101018826) – Project CLariNet.

B

Distributed Residual Deep Reinforcement Learning for Load Frequency Control

"Everything in life is just for a while."

Philip K. Dick, A Scanner Darkly, 1977

Philip K. Dick, A Scanner Darkly

This chapter introduces two residual reinforcement learning frameworks for multi-agent load frequency control. The large-scale nature of the power system complicates real-time control. Furthermore, the system used in this chapter is unstable due to inter-area coupling, which makes reinforcement learning difficult. In both approaches, a decentralized model predictive control structure is deployed as a stabilizing baseline layer. To improve the coordination, the first approach deploys a centralized residual reinforcement learning layer, and the second approach deploys a distributed residual reinforcement learning layer. By providing a good enough approximation of the optimal policy, the baseline layer increases sample efficiency, reduces exploration difficulties, and improves the accuracy of the residual reinforcement learning controller. To further reduce online computational cost, the residual reinforcement learning algorithm is trained offline. The simulation results demonstrate that the addition of a coordinating layer significantly improves the performance, while adding a negligible amount of computation time. .

This chapter is based on [317]

B.1 Introduction

One of the primary control objectives in modern power systems is Load Frequency Control (LFC), which aims to maintain a constant operating frequency by balancing power generation and demand at all times [42]. In recent decades, however, structural changes have increased the complexity of the grid, necessitating more advanced control strategies.

For example, the widespread integration of Renewable Energy Sources (RESs), such as photovoltaic panels and wind farms, poses a challenge due to their weather-dependent and therefore less predictable output [143]. Moreover, unlike conventional plants, RESs lack rotational inertia and cannot directly support grid stability [42].

Furthermore, the electricity grid is now highly interconnected between countries, forming a large-scale, integrated power system [92]. This expands the scope of the LFC task, which must now regulate not only the frequency deviations within individual countries but also the power exchanges between them, necessitating control structures capable of effective multi-area coordination [42, 248]. Finally, the power system demands real-time control, meaning that control actions must be computed and executed within the timing requirements of the system.

B.1.1 Related Work

Centralized Model Predictive Control (MPC) is a state-of-the-art technique that can compute optimal control actions while satisfying system constraints. However, centralized MPC is computationally intensive and generally unsuitable for real-time applications in large-scale systems [280]. To reduce the computational burden, more scalable, non-centralized approaches are needed [35].

One such approach is decentralized MPC (dMPC), where each controller operates based on local information and controls only its own area [4]. While this reduces computation significantly, it lacks coordination among areas, often leading to suboptimal global performance. Distributed MPC methods, such as those based on the Alternating Direction Method of Multipliers (ADMM), enable limited communication between neighboring controllers and can coordinate actions across areas [321]. However, the iterative nature of the ADMM-based optimization makes these methods still computationally demanding and potentially unsuitable for real-time control.

Recently, data-driven approaches, such as Reinforcement Learning (RL) [322], have gained increasing attention from researchers for the control of complex systems, such as LFC [114, 238]. In RL, the agent learns a control law (policy) through interaction with the environment, using feedback in the form of rewards. This approach has also been extended to multi-area LFC systems, where multi-agent RL has shown promising results by cooperatively minimizing frequency deviations under RES variability [238, 289, 347].

For real-time control of large-scale LFC, it is possible to train the policy offline, so that online execution requires only simple forward computations to determine the control action. Offline reinforcement learning achieves this by learning from historical data or by interacting with a model of the environment. Since the policy is developed without interacting with the real system, offline training also avoids the risks associated with unstable or unsafe behavior during learning [302].

Nonetheless, purely RL approaches face challenges in unstable systems, as exploration noise can quickly cause the system state to diverge [130]. In such cases, relying solely

on an RL-based controller is often impractical. The author of [238] concludes that RL is not intended to fully replace conventional model-based LFC techniques, but rather serves as an effective alternative for specific tasks. This motivated the implementation of a hybrid approach that combines well-known advanced control principles with the learning capability of RL, leveraging the strengths of both frameworks. Specifically, using the concept of Residual Reinforcement Learning (RRL), a decentralized MPC is deployed for system stabilization, and an RL controller computes a residual action to improve the coordination across electrical areas in a computationally efficient manner.

B.1.2 Contributions

In this work, two novel Residual Reinforcement Learning (RRL) control structures are proposed. The first one combines decentralized MPC with a centralized Deep Deterministic Policy Gradient (dMPC+CDDPG) algorithm. The second structure combines decentralized MPC with distributed DDPG (dMPC+DDDPG). To the best of the author's knowledge, this work contributes to the current literature in the following aspects:

- Implementation of RRL with a baseline layer that primarily aims to stabilize a networked system, such that offline, off-policy learning is more efficient. In this work, the dMPC layer stabilizes the network, and a CDDPG or DDDPG layer improves the performance.
- Implementation of both centralized and distributed RRL for the coordination of decentralized agents in a networked LFC problem.

B.2 Background

This section introduces the background on networked LFC, the associated control objective, and residual reinforcement learning. For general background on RL, we refer to [322].

B.2.1 Multi-Area Load Frequency Control

A network of connected LFC subsystems can be represented by the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Here, the set of M nodes $\mathcal{V} = \mathcal{A}_i, ..., \mathcal{A}_M$ corresponds to the electrical areas in the topology, with i denoting the area index. If nodes $\mathcal{A}_i$ and $\mathcal{A}_j$ are adjacent, they are connected by an undirected edge $\epsilon_{ij} = \epsilon_{ji} = (\mathcal{A}_i, \mathcal{A}_j) \in \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, allowing for bidirectional power flow. Moreover, the neighborhood, denoted by $\mathcal{N}_i = \mathcal{A}_j \in \mathcal{V} \mid (\mathcal{A}_i, \mathcal{A}_j) \in \mathcal{E}$, is the set of all nodes connected to area $\mathcal{A}_i$.

The discrete-time dynamics of a linearized interconnected system can be represented by the following equation:

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{K}\mathbf{w}(k), \tag{B.1}$$

where $\mathbf{x} = \begin{bmatrix} x_1^\top & \cdots & x_M^\top \end{bmatrix}^\top$, $\mathbf{u} = \begin{bmatrix} u_1^\top & \cdots & u_M^\top \end{bmatrix}^\top$, and $\mathbf{w} = \begin{bmatrix} w_1^\top & \cdots & w_M^\top \end{bmatrix}^\top$ are the augmented states, control inputs, and external signals of the M areas, respectively. In LFC, the external signals generally consist of the load and renewable energy generation, which are exogenous inputs that directly influence the frequency dynamics. The matrix $\mathbf{A}$ is composed of blocks on the diagonal representing each area's own dynamics, and off-diagonal terms for the tie-line couplings between areas. The input matrix $\mathbf{B}$ is block-diagonal with

each block mapping local input u_i directly into subsystem i. Similarly, the external signal matrix $\mathbf{K}$ is block-diagonal, mapping the local signal w_i into subsystem i.

B.2.2 Control Objective

B

In discrete-time control systems, the control objective generally consists of minimizing a cost function over a finite or infinite horizon with respect to a reference trajectory $\mathbf{x}^{\text{ref}}$. This cost function is given in (B.2).

$$J(x,u) = \sum_{k=1}^{\infty} \left[\|\mathbf{x}(k) - \mathbf{x}^{\text{ref}}(k)\|_{\mathbf{Q}}^2 + \|\mathbf{u}(k)\|_{\mathbf{R}}^2 \right], \tag{B.2}$$

where $\mathbf{Q} = \text{blkdiag}(Q_1, \ldots, Q_M)$ and $\mathbf{R} = \text{blkdiag}(R_1, \ldots, R_M)$.

Furthermore, in large interconnected systems, computational efficiency is a critical factor in the deployment of control strategies. Therefore, in addition to minimizing the control cost, we also evaluate the structures on two computational efficiency indicators, both of which relate to the effort required to compute the control inputs. The first indicator assesses whether real-time control is possible, as real-time control is only feasible if the control input is computed within the system's sampling time. It is worth noting that in decentralized or distributed control structures, parallel computation of the controllers is possible. In such structures, the longest computational time of the parallel controllers is measured.

The second indicator quantifies the overall computational burden of the control architecture. First, the cumulative computational time over the simulation horizon is calculated for all controllers in the architecture. Then, these values are summed to obtain the total core computational time indicator.

B.2.3 Residual Reinforcement Learning

In this section, the concept of RRL is introduced [157, 200, 305]. The main idea is to learn a *residual* policy on top of a base policy provided by an arbitrary controller. The resulting control input is the sum of the base policy and the residual policy:

$$\pi_{\text{base}}(x(k)) + \pi_{\text{res}}(x(k)) = u_{\text{base}}(k) + u_{\text{res}}(k) = u(k), \tag{B.3}$$

where the base policy π_{base} can be any existing controller. This allows the RRL method to leverage the properties of the base controller, such as robustness or stability [157]. The goal of the RL algorithm is then to find the optimal residual policy $\pi_{\text{res}}^{\star} = \pi^{\star} - \pi_{\text{base}}$. This method significantly improves data efficiency by avoiding the need for RL to learn the full policy from scratch [305]. Furthermore, assuming that the base policy is already close to the optimal policy, the action space for the RRL algorithm can be reduced. This offers two additional advantages: 1) it reduces exploration challenges, and 2) it improves actor accuracy due to smaller approximation errors in the critic [200].

Since the control inputs are computed sequentially, the residual policy can take the baseline action as an input. This information is essential for the RL agent, as it defines the baseline behavior upon which the residual policy is intended to improve. If the base action is not included, the policy would have to implicitly infer it from the system response, which significantly complicates the learning task.

In RRL algorithms that use a replay buffer, only the residual action $u_{\text{res}}(k)$ is stored as the action. The total action $u(k)$ is not stored because the base policy π_{base} is fixed and known, and the learning algorithm must only optimize the residual policy π_{res}.

B.3 Approach

This chapter proposes two control structures to perform offline learning in unstable systems with changing external signals using RRL. Both structures consist of two layers, where the first baseline layer is designed to stabilize the system, and the second layer learns a coordinating residual control input on top of this baseline. In this work, a decentralized MPC (dMPC) is used as a stabilizing baseline layer. For the RRL, the Deep Deterministic Policy Gradient (DDPG) algorithm has been selected, which is an off-policy Actor-Critic algorithm that allows offline learning [192]. In the first structure, the RRL layer consists of a single centralized residual DDPG agent, referred to as dMPC+CDDPG. In the second structure, the RRL layer comprises multiple distributed residual DDPG agents, referred to as dMPC+DDDPG.

B.3.1 Decentralized MPC + Centralized DDPG

In Fig. B.1, the closed-loop system during the offline learning phase is depicted. For the first layer, a stabilizing dMPC[1] is deployed, where each dMPC controller i controls subsystem $\mathcal{A}_i$. The key limitation of dMPC is the lack of coordination between individual control inputs, which leads to suboptimal performance [204]. To address this, in the first structure, a centralized residual DDPG algorithm is introduced to learn a corrective control signal that coordinates the decentralized inputs. Since only a single DDPG agent is trained, the training procedures of [192] require only minor modifications.

The training loop proceeds as follows. At time step k, each dMPC solves its respective optimization problem using only the local state $x_i(k)$ and the external signals $w_i(k), ..., w_i(k+N-1)$, where N denotes the prediction horizon. The resulting local control inputs $u_i^{\text{dMPC}}(k)$ are aggregated to form the global dMPC input $\mathbf{u}^{\mathbf{dMPC}}(k) = \begin{bmatrix} u_1^{\text{dMPC}}(k) & \cdots & u_M^{\text{dMPC}}(k) \end{bmatrix}$, as illustrated by the black bar in Fig. B.1. This input, along with the global system state $\mathbf{x}(k)$, current and future external signals, is provided as input to the CDDPG actor network.

[1] This result can be found by empirical testing. Formal network stabilization using dMPC can be verified using [4], but this is out of the scope of this chapter.

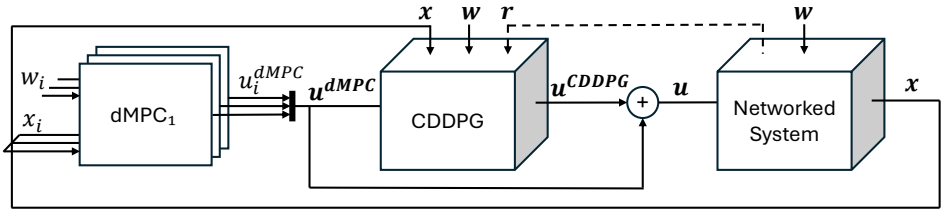


Figure B.1: The dMPC with CDDPG control structure. The dashed line for the reward indicates that the information is used only during offline training, and not during online control.

These components are concatenated to form the augmented observation state:

$$\mathbf{x}_{\mathbf{aug}}^{\mathbf{CDDPG}}(k) = \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{w}(k) \\ \vdots \\ \mathbf{w}(L-1) \\ \mathbf{u}^{\mathbf{dMPC}}(k) \end{bmatrix} \tag{B.4}$$

The external signal horizon length L can be chosen based on the dynamics of the system. However, selecting a value that is too large may cause the external signals to dominate the augmented representation, potentially overshadowing the influence of the state and baseline action. Based on this information, the CDDPG computes a residual control input $\mathbf{u}^{\mathbf{CDDPG}}$. The final control input applied to the system is the sum of both components:

$$\mathbf{u}(k) = \mathbf{u}^{\mathbf{dMPC}}(k) + \mathbf{u}^{\mathbf{CDDPG}}(k) \tag{B.5}$$

The system then transitions to the next state and produces a scalar reward $\mathbf{r}(k)$. This reward is the negative cost from B.2 at the current time step k times a scalar value to scale the reward to a range in which the CDDPG agent learns efficiently [130]. The following transition is saved in the replay buffer:

$$\left(\mathbf{x}_{\text{aug}}^{\mathbf{CDDPG}}(k),\ \mathbf{u}^{\text{CDDPG}}(k),\ \mathbf{r}(k),\ \mathbf{x}_{\text{aug}}^{\mathbf{CDDPG}}(k+1),\ d\right), \tag{B.6}$$

which is used by the CDDPG algorithm to update its parameters. It should be noted that constructing the augmented next state $\mathbf{x}_{\mathbf{aug}}^{\mathbf{CDDPG}}(k+1)$ requires computing the dMPC input $\mathbf{u}^{\mathbf{dMPC}}(k+1)$ at the subsequent time step. Since the future external signals and system state are known during simulation and the dMPC is deterministic, this computation is feasible. Consequently, it is convenient to store the computed dMPC input for the next time step during training.

During the online control phase, only the pre-trained actor network of the CDDPG algorithm is used. This policy can be executed using simple, computationally efficient forward mathematical operations.

B.3.2 Decentralized MPC + Distributed DDPG

The second control structure replaces the centralized residual DDPG layer with a residual distributed DDPG layer, while keeping the dMPC as the baseline controller. The core principles of the residual architecture remain unchanged: the dMPC layer stabilizes the system, enabling offline and off-policy learning. The additional learning advantages are still applicable, which include improved sample efficiency and actor accuracy. The main motivation for this structure is improved scalability compared to the CDDPG variant. The single CDDPG agent would be required to learn a complex actor neural network, as the dimension of $\mathbf{x}_{\mathbf{aug}}^{\mathbf{CDDPG}}$ scales linearly with the number of electrical areas. By limiting the observation space of the agents to only the direct neighbors, their optimization problem remains small, which would require a less complex actor network.

In Fig. B.2, the closed-loop system of the training and control (without dashed lines) phase for dMPC+DDDPG is depicted. Each area $\mathcal{A}_i$ is now subjected to a dMPC controller

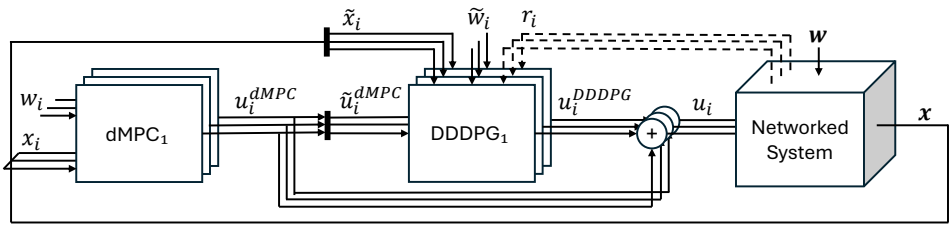


Figure B.2: The dMPC with DDDPG control structure. The dashed lines for the rewards indicate that the information is used only during offline training, and not during online control.

and a DDDPG controller. Similar to the CDDPG structure, a base input u_i^{dMPC} is first computed by a dMPC layer. The DDDPG agents, which can only observe the system information from themselves and their direct neighbors, compute a local corrective residual control input that should coordinate the local system state with that of the neighboring states. The goal is for the agents to operate cooperatively to minimize the global cost function B.2. Each DDDPG agent requires the states, external signals, and dMPC inputs from itself and its neighborhood, which is indicated by the tilde notation above the state $\tilde{x}_i$, input $\tilde{u}_i^{\text{dMPC}}$, and external signals $\tilde{w}_i$:

$$\tilde{x}_i = \left[x_i,\ x_j \,\middle|\, \forall j \in \mathcal{N}_i \right] \tag{B.7}$$

$$\tilde{u}_i^{\text{dMPC}} = \left[u_i^{\text{dMPC}},\ u_j^{\text{dMPC}} \,\middle|\, \forall j \in \mathcal{N}_i \right] \tag{B.8}$$

$$\tilde{w}_i = \left[w_i,\ w_j \,\middle|\, \forall j \in \mathcal{N}_i \right] \tag{B.9}$$

The base input and the residual input for each area i can be summed and inserted into the system.

$$u_i = u_i^{\text{dMPC}} + u_i^{\text{DDDPG}} \tag{B.10}$$

While transitioning to the next state, the system provides each DDDPG agent with a local reward dependent on the local performance:

$$r_i(k) = -\hat{\sigma}_i * \left(\|x_i(k) - x_i^{\text{ref}}(k)\|_{Q_i}^2 + \|u_i(k)\|_{R_i}^2 \right) \tag{B.11}$$

where $\hat{\sigma}_i$ is the local reward scale factor. These rewards allow each agent to independently learn its control policy based on local system information. This policy is denoted by π_i^{DDDPG}, and the distributed agents collectively define the joint residual policy $\boldsymbol{\pi}^{\mathbf{DDDPG}}$. Ideally, the agents learn their local policies such that the resulting joint policy approximates the optimal residual policy: $\boldsymbol{\pi}^{\mathbf{DDDPG}^\star} = \boldsymbol{\pi}^\star - \boldsymbol{\pi}^{\mathbf{dMPC}}$. To achieve this, at each time step k, all DDDPG agents sequentially update their parameters based on batches of transitions sampled from the agent's own replay buffer. These transitions have the form:

$$\left(\tilde{x}_{\text{aug},i}^{\text{DDDPG}}(k),\ u_i^{\text{DDDPG}}(k),\ r_i(k),\ \tilde{x}_{\text{aug},i}^{\text{DDDPG}}(k+1),\ d \right), \tag{B.12}$$

where $\tilde{x}_{\text{aug},i}^{\text{DDDPG}}(k)$ follows the structure of B.4, but includes only local and neighborhood information. Similarly to the CDDPG execution, only the trained actor networks are used for the online control.

B.4 Case study

In this section, the LFC benchmark is explained, where the novel control structures are tested. The simulation results are presented, showing the advantages of the proposed architecture with respect to conventional MPC and distributed MPC based on the ADMM consensus method.

B.4.1 Benchmark Description

To assess the performance and computational complexity of the novel control structures, we use the European Economic Area Electricity Benchmark (EEA-ENB) [280]. The countries of the EEA are represented as interconnected electrical areas, for which the benchmark provides topological information, real-world electrical data, and a mathematical model. The electrical power sources, loads, and renewable sources are aggregated per area and can therefore be modelled as an equivalent electrical machine. Neighboring areas are connected by tie-lines, which allow bidirectional power flow.

The linearized LFC system of the i-th area consists of the 5 states x_i, the 3 control inputs u_i, and the 2 external signals w_i:

$$x_i = \begin{bmatrix} \Delta\delta_i & \Delta f_i & e_i & P_i^{\text{tie}} & P_i^{\text{disp}} \end{bmatrix}^\top \tag{B.13}$$

$$u_i = \begin{bmatrix} \Delta P_i^{\text{disp}} & P_i^{\text{ESS,c}} & P_i^{\text{ESS,d}} \end{bmatrix}^\top \tag{B.14}$$

$$w_i = \begin{bmatrix} \Delta P_i^{\text{load}} & \Delta P_i^{\text{ren}} \end{bmatrix}^\top \tag{B.15}$$

Here, $\Delta\delta_i$ [deg] is the variation of the machine angle with respect to the nominal angle $\delta_{0,i} = 30\,[\text{deg}]$, and Δf_i [Hz] is the variation of the frequency with respect to the nominal frequency $f_0 = 50\,[\text{Hz}]$, e_i [GWs] is the energy stored in the ESS, P_i^{tie} [GW] is the total energy exchange over the tie-lines to the i-th area, and P_i^{disp} [GW] is the dispatchable power allocation. The control inputs are the variation in dispatchable power ΔP_i^{disp} [GW/s], and the ESS charging power $P_i^{\text{ESS,c}}$ [GW] or discharging power $P_i^{\text{ESS,d}}$ [GW]. The two external signals are the variation in load request ΔP_i^{load} [GW/s] and the variation in renewable energy generation ΔP_i^{ren} [GW/s].

A sampling time τ of 2.5 [s] is used for the simulations, where the discrete-time index is denoted by k. The dynamics associated with the i-th area $\mathcal{A}_i$ of the equivalent electrical machine are provided by the system of linear discrete-time difference equations reported in the following:

$$\Delta\delta_i(k+1) = \Delta\delta_i(k) + \tau 2\pi \Delta f_i(k) \tag{B.16}$$

$$\Delta f_i(k+1) = \left(1 - \frac{\tau}{\tau_{R,i}}\right)\Delta f_i(k) + K_{p,i}\frac{\tau}{\tau_{R,i}} g_i(k) \tag{B.17}$$

$$e_i(k+1) = e_i(k) + \tau\left(\eta_i^c P_i^{\text{ESS,c}}(k) - \frac{1}{\eta_i^d} P_i^{\text{ESS,d}}(k)\right) \tag{B.18}$$

$$P_i^{\text{tie}}(k+1) = P_i^{\text{tie}}(k) + \tau \Delta P_i^{\text{tie}}(k) \tag{B.19}$$

$$P_i^{\text{disp}}(k+1) = P_i^{\text{disp}}(k) + \tau \Delta P_i^{\text{disp}}(k), \tag{B.20}$$

with:

$$g_i(k) = \Delta P_i^{\text{disp}}(k) - \Delta P_i^{\text{load}}(k) + \Delta P_i^{\text{ren}}(k) - \Delta P_i^{\text{tie}}(k) - P_i^{\text{ESS,c}}(k) + P_i^{\text{ESS,d}}(k) \tag{B.21}$$

$$\Delta P_i^{\text{tie}}(k) = \sum_{j \in \mathcal{N}_i} T_{ij}(\Delta\delta_i(k) - \Delta\delta_j(k)), \tag{B.22}$$

where T_{ij} in $\frac{\text{GW}}{\text{deg}}$ is the gain associated with the tie-line connecting area i and j; $K_{p,i}$ in $[\frac{\text{Hz}\times s}{\text{GW}}]$ is the gain of the rotating mass dynamics; and $\tau_{R,i}$ in [s] is the time constant of the rotating mass dynamics. Furthermore, η_i^c and η_i^d are the charging and discharging efficiencies of the ESS. A challenge to the control of the EEA-ENB arises from the instability of the system due to the inter-area coupling.

All areas must satisfy the following constraints. The angle deviation and frequency deviation states are bounded by $-3.5 \leq \Delta\delta_i \leq 3.5$ and $-0.04 \leq \Delta f_i \leq 0.04$, respectively. Furthermore, the total ESS capacity of area i is assumed to equal the total dispatchable capacity $P_i^{\text{disp,max}}$, which varies by area. The dispatchable power control input is bounded by $-\Delta P_i^{\text{disp,max}} \leq \Delta P_i^{\text{disp}} \leq \Delta P_i^{\text{disp,max}}$, and the charging and discharging inputs by $0 \leq P_i^{\text{ESS,c}}, P_i^{\text{ESS,d}} \leq \Delta P_i^{\text{disp,max}}$, where $\Delta P_i^{\text{disp,max}}$ is selected such that the total dispatchable power or ESS capacity can be allocated over one hour. To enforce these constraints, saturation functions on the combined control input have been implemented.

B.4.2 Experimental Setups

This chapter focuses on a four-area system and a six-area system. The four-area case study considers a circular topology with the following countries: Denmark (DK), Germany (DE), the Netherlands (NL), and Sweden (SE). In the actual EEA-ENB topology, DK and DE are connected. However, this would result in some distributed controllers having the same observability as centralized controllers. For the six-area case study, Norway (NO) and Poland (PL) are added to the four-area system. For the same reason as the four-area case, connections DE-NO and DE-SE are removed from the network, and connection DK-DE is restored.

This chapter uses the measured data for ΔP_i^{load} and ΔP_i^{ren}, and assumes to have a perfect forecast. To introduce variability between the training and testing external signals, Gaussian noise with a standard deviation of 1% of the maximum value of the external signal is added to the data.

All simulations presented in this work were executed on the *DelftBlue* high-performance computing cluster. Using this system ensured isolated and reproducible runs, free from interference by external processes. This enabled valid comparisons of computational time between control structures, without the necessity of re-running simulations several times for average results in computation time.

B.4.3 Simulation Results

The novel control structures are tested against a dMPC, a Centralized MPC (CMPC), and an ADMM-based Distributed MPC (DMPC-ADMM) with respect to the cost function and the computational efficiency indicators. These evaluations will be done using Table B.1 and Table B.2. The tables report the best-performing runs over 5 seeds for dMPC+CDDPG

Method four-area	Cost	% from optimal	Comp. time online	Mean comp. time [s]	Core time online	Comp. time offline
CMPC	1.4787×10^{-2}	0.00	05:18:55	0.554	05:18:55	-
DMPC-ADMM	1.4829×10^{-2}	0.28	04:57:16	0.516	19:49:06	-
dMPC	2.0042×10^{-2}	35.55	00:16:58	0.029	01:07:52	-
dMPC+CDDPG	1.6415×10^{-2}	11.01	00:16:59	0.029	01:07:06	01:30:06
dMPC+DDDPG	1.7445×10^{-2}	17.97	00:17:59	0.031	01:11:58	01:49:15

Table B.1: Cumulative cost and time comparison for the different control structures in the four-area case study. Time is given in $hh:mm:ss$.

Method six-area	Cost	% from optimal	Comp. time online	Mean comp. time [s]	Core time online	Comp. time offline
CMPC	1.4008×10^{-2}	0.00	08:56:56	0.932	08:56:56	-
DMPC-ADMM	1.4020×10^{-2}	0.09	08:04:47	0.842	48:28:39	-
dMPC	5.8646×10^{-2}	318.66	00:20:10	0.035	02:00:59	-
dMPC+CDDPG	2.4601×10^{-2}	75.62	00:19:09	0.033	01:53:01	01:32:39
dMPC+DDDPG	1.9659×10^{-2}	40.34	00:20:45	0.036	02:04:35	03:35:39

Table B.2: Cumulative cost and time comparison for the different control structures in the six-area case study. Time is given in $hh:mm:ss$.

and dMPC+DDDPG, which is justified since the controllers are trained offline, allowing for the selection of the most effective parameter configurations for deployment.

CMPC yields the optimal control solution by leveraging full system information over the prediction horizon. The performance of alternative strategies is expressed as their percentage deviation from this optimum, shown in the *% from optimal* columns. DMPC-ADMM achieves nearly optimal performance with slightly lower computation time. However, as the number of areas increases, this time advantage grows due to parallelization. The need for coordination of the dMPC becomes especially evident in the six-area case, where its performance drops sharply while CMPC remains consistent. Nevertheless, dMPC offers much lower and scalable computation times, remaining nearly constant with increasing system size.

In the four-area case, the addition of a coordinating DDPG layer reduces the performance gap between dMPC and the optimal CMPC by 69.03% for the dMPC+CDDPG structure and by 49.45% for the dMPC+DDDPG configuration. Since the computational burden of the DDPG layers is shifted to the offline phase, both structures introduce only a minimal increase in computation time during the online control phase. The core computation time of the dMPC+CDDPG is even slightly lower than that of dMPC; however, this is not expected to hold on average. It is also worth noting that both novel control structures remain faster than CMPC and DMPC-ADMM when the offline computation time is included. This suggests that an online RL coordination layer is also possible for real-time control.

For the six-area case study, the reductions in performance gap are even greater: 76.27% for the dMPC+CDDPG structure and 87.34% for the dMPC+DDDPG structure. The distributed variant performs better than the centralized one, despite additional challenges of multi-agent reinforcement learning. Given that the CDDPG has full observability of the system, it would be expected to achieve superior performance, as observed in the four-area case. This could imply that the hyperparameters for the CDDPG controller might not yet

be fully optimized. The computation time per time step and core computation time remain similar to those of the baseline dMPC, much lower than those of CMPC and DMPC-ADMM.

B.5 Conclusion and Future Work

In this work, two novel RRL control structures have been developed, combining stabilizing dMPC with a coordinating centralized or distributed DDPG layer. These structures are tested on a four- and six-area case study in the EEA-ENB, which has inherent system instability and time-varying external signals. Although the cost performance of dMPC+CDDPG and dMPC+DDDPG does not match that of CMPC or DMPC-ADMM, both represent a clear improvement over the dMPC baseline, while retaining the same convenient computational speed for real-time control. These computation times are at least 15 times faster than those observed for the CMPC and DMPC-ADMM approaches. Future work should investigate the satisfaction of constraints without an input saturation function, which is a pragmatic but suboptimal solution.

Acknowledgments

First and foremost, this work would not have been possible without the support and guidance of my promotor **Bart De Schutter** and copromotor **Luca Laurenti**. I owe them most of my professional growth as a researcher. Thank you for your time and dedication. I would also like to sincerely thank **Thom Badings** and **Alessandro Abate**, as well as all the members of the Oxford Control & Verification (OXCAV) group, for their hospitality and the wonderful collaboration that resulted from our discussions. A special thanks goes to the MSc students that I had the pleasure to follow in the development of their thesis: **Loek Steenhoff**, **Joost van der Weerd**, and **Demi Spée**. Your questions challenged me, and your ideas have been inspiring: I wish you all the best for your future work. My profound gratitude goes to **Francesco Cordiano** for the several collaborations and the days he brightened for me. I also want to thank, for the technical discussions and the everyday moments of the past four years, all the colleagues of the Clarinet project and related: **Caio**, **Filippo**, **Kanghui**, **Samuel**, **Gianpietro**, **Changrui**, **Giray**, **Anil**, **Leila**, **Athina**, **Reza**, and **Azita**. A special thanks goes to the first people I met at the beginning of my PhD, **Manu** and **Suad**, who made me feel very welcome. Also, I will remember all the moments I had with **Emilio**, **Léonore**, and **Claudia**, who have always been so kind to listen and support me, thank you. Finally, I want to thank all the other colleagues of DCSC, as well as the academic, technical, and administrative staff of the department: I think TU Delft is an exceptional reality especially for the work of people like you.

Bibliography

[1] A. Abate, M. Giacobbe, and D. Roy. Quantitative supermartingale certificates. In *Computer Aided Verification*, pages 3–28. Springer, 2025.

[2] A. Abate, M. Prandini, J. Lygeros, and S. Sastry. Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. *Automatica*, 44:2724–2734, 2008.

[3] A. Afram, F. Janabi-Sharifi, A. S. Fung, and K. Raahemifar. Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: A state of the art review and case study of a residential HVAC system. *Energy and Buildings*, 141:96–113, 2017.

[4] A. Alessio, D. Barcelli, and A. Bemporad. Decentralized model predictive control of dynamically coupled linear systems. *Journal of Process Control*, 21:705–714, 2011.

[5] A. Alessio and A. Bemporad. Decentralized model predictive control of constrained linear systems. In *2007 European Control Conference (ECC)*, pages 2813–2818, 2007.

[6] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[7] I. Alvarado, D. Limon, D. Muñoz De La Peña, J. M. Maestre, M. A. Ridao, H. Scheu, W. Marquardt, R. R. Negenborn, B. De Schutter, F. Valencia, and J. Espinosa. A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark. *Journal of Process Control*, 21:800–815, 2011.

[8] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62:3861–3876, 2017.

[9] K. An, Y. Chiu, X. Hu, and X. Chen. A network partitioning algorithmic approach for macroscopic fundamental diagram-based hierarchical traffic network management. *IEEE Transactions on Intelligent Transportation Systems*, 19:1130–1139, 2018.

[10] S. An, B. Lee, and D. Shin. A survey of intelligent transportation systems. In *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*, pages 332–337, 2011.

[11] W. Ananduta and C. Ocampo-Martinez. Event-triggered partitioning for non-centralized predictive-control-based economic dispatch of interconnected microgrids. *Automatica*, 132:1–8, 2021.

[12] W. Ananduta, T. Pippia, C. Ocampo-Martinez, J. Sijs, and B. De Schutter. Online partitioning method for decentralized control of linear switching large-scale systems. *Journal of the Franklin Institute*, 356:3290–3313, 2019.

[13] J. Annoni, C. Bay, T. Taylor, L. Pao, P. Fleming, and K. Johnson. Efficient optimization of large wind farms for real-time control. In *2018 Annual American Control Conference (ACC)*, pages 6200–6205, 2018.

[14] K. R. Apt and A. Witzel. A generic approach to coalition formation. *International Game Theory Review*, 11:347–367, 2009.

[15] A. Arastou, Y. Wang, and E. Weyer. Optimization-based network partitioning for distributed and decentralized control. *Journal of Process Control*, 146:1–10, 2025.

[16] K. J. Åström. *Introduction to Stochastic Control Theory*. Courier Corporation, 2012.

[17] E. Atam and E. C. Kerrigan. Optimal partitioning of multithermal zone buildings for decentralized control. *IEEE Transactions on Control of Network Systems*, 8:1540–1551, 2021.

[18] R. Attri, N. Dev, and V. Sharma. Interpretive structural modelling (ISM) approach: An overview. *Research Journal of Management Sciences*, 2:3–8, 2013.

[19] G. Auer, V. Giannini, I. Godor, P. Skillermark, M. Olsson, M. A. Imran, D. Sabella, M. J. Gonzalez, C. Desset, O. Blume, and A. Fehske. How much energy is needed to run a wireless network? *IEEE Wireless Communications*, 18:1–16, 2012.

[20] T. Badings and A. Abate. Probabilistic alternating simulations for policy synthesis in uncertain stochastic dynamical systems. DOI: 10.48550/arXiv.2508.05062, 2025.

[21] T. Badings, W. Koops, S. Junges, and N. Jansen. Policy verification in stochastic dynamical systems using logarithmic neural certificates. In *Computer Aided Verification*, pages 349–375. Springer, 2025.

[22] T. Badings, L. Romao, A. Abate, and N. Jansen. Probabilities are not enough: Formal controller synthesis for stochastic dynamical models with epistemic uncertainty. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 14701–14710, 2023.

[23] T. Badings, L. Romao, A. Abate, D. Parker, H. A. Poonawala, M. Stoelinga, and N. Jansen. Robust control for dynamical systems with non-Gaussian noise via formal abstractions. *Journal of Artificial Intelligence Research*, 76:341–391, 2023.

[24] C. Baier and J. Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[25] L. Bakule. Decentralized control: An overview. *Annual Reviews in Control*, 32:87–98, 2008.

[26] P. R. Baldivieso Monasterios and P. A. Trodden. Coalitional predictive control: Consensus-based coalition forming with robust regulation. *Automatica*, 125:14, 2021.

[27] P. R. Baldivieso Monasterios, P. A. Trodden, and M. Cannon. On feasible sets for coalitional MPC. In *IEEE 58th Conference on Decision and Control (CDC)*, pages 4668–4673, 2019.

[28] M. J. Barber. Modularity and community detection in bipartite networks. *Physical Review E*, 76:1–9, 2007.

[29] J. Barreiro-Gomez. *Partitioning for large-scale systems: Sequential DMPC design*, pages 163–178. Springer, 2019.

[30] J. Barreiro-Gomez, F. Dörfler, and H. Tembine. Distributed robust population games with applications to optimal frequency control in power systems. In *2018 Annual American Control Conference (ACC)*, pages 5762–5767, 2018.

[31] J. Barreiro-Gomez, C. Ocampo-Martinez, and N. Quijano. Time-varying partitioning for predictive control design: Density-games approach. *Journal of Process Control*, 75:1–14, 2019.

[32] J. Barreiro-Gomez and H. Tembine. Constrained evolutionary games by using a mixture of imitation dynamics. *Automatica*, 97:254–262, 2018.

[33] O. Bello, A. M. Abu-Mahfouz, Y. Hamam, P. R. Page, K. B. Adedeji, and O. Piller. Solving management problems in water distribution networks: A survey of approaches and mathematical models. *Water*, 11:1–29, 2019.

[34] C. Belta, B. Yordanov, and E. A. Gol. *Formal Methods for Discrete-Time Dynamical Systems*, volume 89. Springer, 2017.

[35] A. Bemporad, M. Heemels, and Johansson. *Networked control systems*. Springer, 2010.

[36] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–427, 1999.

[37] A. Bemporad and M. Morari. Robust model predictive control: A survey. In *Robustness in Identification and Control*, volume 245, pages 207–226. Springer, 1999.

[38] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.

[39] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[40] D. P. Bertsekas and S. E. Shreve. *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific, 1996.

[41] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 2015.

[42] H. Bevrani. *Robust Power System Frequency Control*. Springer, 2014.

[43] F. D. Bianchi, H. De Battista, and R. J. Mantz. *Wind Turbine Control Systems: Principles, Modelling and Gain Scheduling Design*. Springer, 2007.

[44] A. Blizard and S. Stockar. Optimality loss minimization in distributed control with application to district heating, 2025.

[45] V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008:1–12, 2008.

[46] B. Bollobás. *Modern Graph Theory*. Springer, 2001.

[47] J. A. Bondy and U. S. R. Murty. *Graph Theory*. Springer, 2008.

[48] M. Boulle. Compact mathematical formulation for graph partitioning. *Optimization and Engineering*, 5:315–333, 2004.

[49] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3:1–122, 2010.

[50] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[51] U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner. Maximizing modularity is hard. DOI: 10.48550/arXiv.physics/0608255, 2006.

[52] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20:172–188, 2008.

[53] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30:107–117, 1998.

[54] E. Bristol. On a new measure of interaction for multivariable process control. *IEEE Transactions on Automatic Control*, 11:133–134, 1966.

[55] Y. Cao, W. Yu, W. Ren, and G. Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9:427–438, 2013.

[56] S. Carr, N. Jansen, S. Junges, and U. Topcu. Safe reinforcement learning via shielding under partial observability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 14748–14756, 2023.

[57] J. M. Carrasco, L. G. Franquelo, J. T. Bialasiewicz, E. Galvan, R. C. PortilloGuisado, M. A. M. Prats, J. I. Leon, and N. Moreno-Alfonso. Power-electronic systems for the grid integration of renewable energy sources: A survey. *IEEE Transactions on Industrial Electronics*, 53:1002–1016, 2006.

[58] J. Castro, D. Gómez, and J. Tejada. Polynomial calculation of the shapley value based on sampling. *Computers & Operations Research*, 36:1726–1730, 2009.

[59] N. Cauchi, L. Laurenti, M. Lahijanian, A. Abate, M. Kwiatkowska, and L. Cardelli. Efficiency through uncertainty: scalable formal synthesis for stochastic hybrid systems. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, pages 240–251, 2019.

[60] J. C. Cesco. A convergent transfer scheme to the core of a tu-game. *Revista de Matemáticas Aplicadas*, 19:23–35, 1998.

[61] A. Chakrabortty and P. P. Khargonekar. Introduction to wide-area control of power systems. In *2013 American Control Conference*, pages 6758–6770, 2013.

[62] V. Chandan and A. Alleyne. Optimal partitioning for the decentralized thermal control of buildings. *IEEE Transactions on Control Systems Technology*, 21:1756–1770, 2013.

[63] P. Chanfreut, J. M. Maestre, and E. F. Camacho. Coalitional model predictive control on freeways traffic networks. *IEEE Transactions on Intelligent Transportation Systems*, 22:6772–6783, 2021.

[64] P. Chanfreut, J. M. Maestre, and E. F. Camacho. A survey on clustering methods for distributed and networked control systems. *Annual Reviews in Control*, 52:75–90, 2021.

[65] P. Chanfreut, J. M. Maestre, A. Ferramosca, F. J. Muros, and E. F. Camacho. Distributed model predictive control for tracking: A coalitional clustering approach. *IEEE Transactions on Automatic Control*, 67:6873–6880, 2022.

[66] P. Chanfreut, J. M. Maestre, A. J. Gallego, A. M. Annaswamy, and E. F. Camacho. Clustering-based model predictive control of solar parabolic trough plants. *Renewable Energy*, 216:10, 2023.

[67] P. Chanfreut, J. M. Maestre, T. Hatanaka, and E. F. Camacho. Fast clustering for multi-agent model predictive control. *IEEE Transactions on Control of Network Systems*, 9:1544–1555, 2022.

[68] C. Changqing, L. Xinran, and L. Hua. Frequency regulation control strategy of wind farms based on temporal and spatial uncertainty. *Sustainable Energy Technologies and Assessments*, 53:1–11, 2022.

[69] B. Chen and H. H. Cheng. A review of the applications of agent technology in traffic and transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 11:485–497, 2010.

[70] M. Chen, Jun Zhao, Z. Xu, Y. Liu, Y. Zhu, and Z. Shao. Cooperative distributed model predictive control based on topological hierarchy decomposition. *Control Engineering Practice*, 103:15, 2020.

[71] Z. Chen, J. Liu, Z. Lin, and Z. Duan. Closed-loop active power control of wind farm based on frequency domain analysis. *Electric Power Systems Research*, 170:13–24, 2019.

[72] P. D. Christofides, R. Scattolini, D. Muñoz de la Peña, and J. Liu. Distributed model predictive control: A tutorial review and future research directions. *Computers & Chemical Engineering*, 51:21–41, 2013.

[73] A. Clark. Control barrier functions for stochastic systems. *Automatica*, 130:1–9, 2021.

[74] H. H. Coban, A. Rehman, and M. Mousa. Load frequency control of microgrid system by battery and pumped-hydro energy storage. *Water*, 14:1–22, 2022.

[75] A. J. Conejo, editor. *Decomposition Techniques in Mathematical Programming: Engineering and Science Applications*. Springer, 2006.

[76] M. Conforti, G. Cornuéjols, and G. Zambelli. *Integer Programming*, volume 271. Springer, 2014.

[77] J. H. Conway and N. J. A. Sloane. *Sphere Packings, Lattices and Groups*. Springer, 1988.

[78] J. Cortés and M. Egerstedt. Coordinated control of multi-robot systems: A survey. *SICE Journal of Control, Measurement, and System Integration*, 10:495–503, 2017.

[79] G. Curzi, D. Modenini, and P. Tortora. Large constellations of small satellites: A survey of near future challenges and missions. *Aerospace*, 7:1–18, 2020.

[80] P. Daoutidis and C. Kravaris. Structural evaluation of control configurations for multivariable nonlinear processes. *Chemical Engineering Science*, 47:1091–1107, 1992.

[81] C. Dawson, S. Gao, and C. Fan. Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics*, 39:1749–1767, 2023.

[82] L. B. De Oliveira and E. Camponogara. Multi-agent model predictive control of signaling split in urban traffic networks. *Transportation Research Part C: Emerging Technologies*, 18:120–139, 2010.

[83] R. Diestel. *Graph Theory*, volume 173. Springer, fifth edition, 2017.

[84] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[85] D. Ding, Q. Han, X. Ge, and J. Wang. Secure state estimation and control of cyber-physical systems: A survey. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51:176–190, 2021.

[86] S. N. Dorogovtsev and J. F. F. Mendes. Evolution of networks. *Advances in Physics*, 51:1079–1187, 2002.

[87] A. Dorri, S. S. Kanhere, and R. Jurdak. Multi-agent systems: A survey. *IEEE Access*, 6:28573–28593, 2018.

[88] J. J. Downs and E. F. Vogel. A plant-wide industrial process control problem. *Computers & Chemical Engineering*, 17:245–255, 1993.

[89] W. B. Dunbar and D. S. Caveney. Distributed receding horizon control of vehicle platoons: stability and string stability. *IEEE Transactions on Automatic Control*, 57:620–633, 2012.

[90] T. e. Boukelia and M.-S. Mecibah. Parabolic trough solar thermal power plant: Potential, and projects development in algeria. *Renewable and Sustainable Energy Reviews*, 21:288–297, 2013.

[91] M. Elsisi, M. Soliman, M. A. S. Aboelela, and W. Mansour. Bat inspired algorithm based optimal design of model predictive load frequency control. *International Journal of Electrical Power & Energy Systems*, 83:426–433, 2016.

[92] ENTSO-E. Transparency platform. https://transparency.entsoe.eu/.

[93] A. M. Ersdal, L. Imsland, and K. Uhlen. Model predictive load-frequency control. *IEEE Transactions on Power Systems*, 31:777–785, 2016.

[94] F. Fele, E. Debada, J. M. Maestre, and E. F. Camacho. Coalitional control for self-organizing agents. *IEEE Transactions on Automatic Control*, 63:2883–2897, 2018.

[95] F. Fele, J. M. Maestre, and E. F. Camacho. Coalitional control: Cooperative game theory and control. *IEEE Control Systems Magazine*, 37:53–69, 2017.

[96] F. Fele, J. M. Maestre, and E. F. Camacho. Coalitional control: Cooperative game theory and control. *IEEE Control Systems Magazine*, 37:53–69, 2017.

[97] F. Fele, J. M. Maestre, S. M. Hashemy, D. Muñoz de la Peña, and E. F. Camacho. Coalitional model predictive control of an irrigation canal. *Journal of Process Control*, 24:314–325, 2014.

[98] T. L. Fine. *Feedforward Neural Network Methodology*. Springer, 2006.

[99] W. H. Fleming and R. Rishel. *Deterministic and Stochastic Optimal Control*. Springer, 2012.

[100] L. A. Fletscher, L. A. Suárez, D. Grace, C. V. Peroni, and J. M. Maestre. Energy-aware resource management in heterogeneous cellular networks with hybrid energy sources. *IEEE Transactions on Network and Service Management*, 16:279–293, 2019.

[101] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[102] S. Fortunato and D. Hric. Community detection in networks: A user guide. *Physics Reports*, 659:1–44, 2016.

[103] 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*, 89:563–577, 1970.

[104] S. Frank and S. Rebennack. An introduction to optimal power flow: Theory, formulation, and examples. *IIE Transactions*, 48:1172–1197, 2016.

[105] A. J. Gallego and E. F. Camacho. Adaptative state-space model predictive control of a parabolic-trough field. *Control Engineering Practice*, 20:904–911, 2012.

[106] B. Galloway and G. P. Hancke. Introduction to industrial control networks. *IEEE Communications Surveys & Tutorials*, 15:860–880, 2013.

[107] M. Gálvez-Carrillo, R. De Keyser, and C. Ionescu. Nonlinear predictive control with dead-time compensator: Application to a solar power plant. *Solar Energy*, 83:743–752, 2009.

[108] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß. Self-organized aggregation without computation. *The International Journal of Robotics Research*, 33:1145–1161, 2014.

[109] N. Geroliminis and C. F. Daganzo. Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings. *Transportation Research Part B: Methodological*, 42:759–770, 2008.

[110] A. Girard and G. J. Pappas. Hierarchical control system design using approximate simulation. *Automatica*, 45:566–571, 2009.

[111] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99:7821–7826, 2002.

[112] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[113] H. Golpîra and H. Bevrani. A framework for economic load frequency control design using modified multi-objective genetic algorithm. *Electric Power Components and Systems*, 42:788–797, 2014.

[114] X. Gong, X. Wang, and B. Cao. On data-driven modeling and control in modern power grids stability: Survey and perspective. *Applied Energy*, 350:1–18, 2023.

[115] I. Gracia and M. Lahijanian. Beyond interval MDPs: Tight and efficient abstractions of stochastic systems. DOI: 10.48550/arXiv.2507.02213, 2025.

[116] I. Gracia, L. Laurenti, M. J. Mazo, A. Abate, and M. Lahijanian. Temporal logic control for nonlinear stochastic systems under unknown disturbances. In *7th Annual Conference on Learning for Dynamics and Control (L4DC)*, volume 238, pages 1–13, 2025.

[117] C. Grigg, P. Wong, P. Albrecht, R. Allan, M. Bhavaraju, R. Billinton, Q. Chen, C. Fong, S. Haddad, S. Kuruganty, W. Li, R. Mukerji, D. Patton, N. Rau, D. Reppen, A. Schneider, M. Shahidehpour, and C. Singh. The IEEE reliability test system-1996. A report prepared by the reliability test system task force of the application of probability methods subcommittee. *IEEE Transactions on Power Systems*, 14:1010–1020, 1999.

[118] D. Groß. *Distributed Model Predictive Control with Event-Based Communication*. Kassel Univiversity Press, 2015.

[119] J. D. Grunnet, M. Soltani, T. Knudsen, M. Kragelund, and T. Bak. Aeolus toolbox for dynamics wind farm model, simulation and control. In *European Wind Energy Conference and Exhibition, EWEC 2010*, pages 1–10, 2010.

[120] Y. Guo, L. Yang, S. Hao, and J. Gao. Dynamic identification of urban traffic congestion warning communities in heterogeneous networks. *Physica A: Statistical Mechanics and its Applications*, 522:98–111, 2019.

[121] A. Gupta. Fast and effective algorithms for graph partitioning and sparse-matrix ordering. *IBM Journal of Research and Development*, 41:1–23, 1996.

[122] D. K. Gupta, A. V. Jha, B. Appasani, A. Srinivasulu, N. Bizon, and P. Thounthong. Load frequency control using hybrid intelligent optimization technique for multi-source power systems. *Energies*, 14:1–16, 2021.

[123] Gurobi Optimization, LLC. Reference Manual. https://www.gurobi.com.

[124] H. Haes Alhelou, M. E. Hamedani-Golshan, T. C. Njenda, and P. Siano. A survey on power system blackout and cascading events: Research motivations and challenges. *Energies*, 12:1–28, 2019.

[125] S. Haesaert and S. Soudjani. Robust dynamic programming for temporal logic control of stochastic systems. *IEEE Transactions on Automatic Control*, 66:2496–2511, 2021.

[126] S. Haesaert, S. E. Z. Soudjani, and A. Abate. Verification of general Markov decision processes by approximate similarity relations and policy refinement. *SIAM Journal on Control and Optimization*, 55:2333–2367, 2017.

[127] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6:512–535, 1994.

[128] W. He and S. Li. Enhancing topological information of the lyapunov-based distributed model predictive control design for large-scale nonlinear systems. *Asian Journal of Control*, 25:1476–1487, 2023.

[129] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37:1085–1091, 2001.

[130] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep reinforcement learning that matters. *The Association for the Advancement of Artificial Intelligence Press*, 392:3207–3214, 2019.

[131] C. Hensel, S. Junges, J. P. Katoen, T. Quatmann, and M. Volk. The probabilistic model checker Storm. *International Journal on Software Tools for Technology Transfer*, 24:589–610, 2022.

[132] T. A. Henzinger, K. Mallik, P. Sadeghi, and Đ. Žikelić. Supermartingale certificates for quantitative omega-regular verification and control. In *Computer Aided Verification*, pages 29–55. Springer, 2025.

[133] G. Heo and P. Gader. An extension of global fuzzy c-means using kernel methods. In *International Conference on Fuzzy Systems*, pages 1–6, 2010.

[134] J. M. Hernandez and P. van Mieghem. Classification of graph metrics. Technical report, Delft University of Technology, Faculty of Electrical Engineering, Mathematics, and Computer Science, 2011.

[135] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer. Learning an approximate model predictive controller with guarantees. *IEEE Control Systems Letters*, 2:543–548, 2018.

[136] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger. Learning-Based model predictive control: Toward safe lin control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:269–296, 2020.

[137] A. M. Howlader, T. Senjyu, and A. Y. Saber. An integrated power smoothing control for a grid-interactive wind farm considering wake effects. *IEEE Systems Journal*, 9:954–965, 2015.

[138] K. C. Hsu, H. Hu, and J. F. Fisac. The safety filter: A unified view of safety-critical control in autonomous systems. *Annual Review of Control, Robotics, and Autonomous Systems*, 7:47–72, 2024.

[139] C. P. E. Y. Huanca, G. P. Incremona, and P. Colaneri. Design of a distributed switching model predictive control for quadrotor UAVs aggregation. *IEEE Control Systems Letters*, 7:2964–2969, 2023.

[140] G. Hug-Glanzmann and G. Andersson. Decentralized optimal power flow control for overlapping areas in power systems. *IEEE Transactions on Power Systems*, 24:327–336, 2009.

[141] G. Hug-Glanzmann, R. R. Negenborn, G. Andersson, B. De Schutter, and H. Hellendoorn. Multi-area control of overlapping areas in power systems for FACTS control. In *2007 IEEE Lausanne Power Tech*, pages 1238–1243, 2007.

[142] IEEE DataPort. Power and energy. https://ieee-dataport.org/topic-tags/power-and-energy.

[143] International Energy Agency (IEA). World energy outlook 2023. https://www.iea.org/reports/world-energy-outlook-2023, 2023.

[144] S. Isapoor, A. Montazar, P. J. Van Overloop, and N. Van De Giesen. Designing and evaluating control systems of the dez main canal. *Irrigation and Drainage*, 60:70–79, 2011.

[145] H. Ishii and P. Tempo. The PageRank problem, multiagent consensus, and web aggregation: A systems and control viewpoint. *IEEE Control Systems Magazine*, 34:34–53, 2014.

[146] H. Ishii and R. Tempo. Distributed randomized algorithms for the pagerank computation. *IEEE Transactions on Automatic Control*, 55:1987–2002, 2010.

[147] G. N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30:257–280, 2005.

[148] J. Jackson, L. Laurenti, E. Frew, and M. Lahijanian. Strategy synthesis for partially-known switched stochastic systems. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, pages 1–11, 2021.

[149] P. Jagtap, S. Soudjani, and M. Zamani. Formal synthesis of stochastic systems via control barrier certificates. *IEEE Transactions on Automatic Control*, 66:3097–3110, 2021.

[150] A. Jain, A. Chakrabortty, and E. Biyik. Distributed wide-area control of power system oscillations under communication and actuation constraints. *Control Engineering Practice*, 74:132–143, 2018.

[151] M. B. Jamoom, E. Feron, and M. W. McConley. Optimal distributed actuator control grouping schemes. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 2, pages 1900–1905 vol.2, 1998.

[152] Z. Javid, I. Kocar, W. Holderbaum, and U. Karaagac. Future distribution networks: A review. *Energies*, 17:1–45, 2024.

[153] A. R. Jha. *Wind Turbine Technology*. CRC Press, 2010.

[154] Y. Jia, Z. Y. Dong, C. Sun, and K. Meng. Cooperation-Based distributed economic MPC for economic load dispatch and load frequency control of interconnected power systems. *IEEE Transactions on Power Systems*, 34:3964–3966, 2019.

[155] S. S. Jogwar. Distributed control architecture synthesis for integrated process networks through maximization of strength of input-output impact. *Journal of Process Control*, 83:77–87, 2019.

[156] S. S. Jogwar and P. Daoutidis. Community-based synthesis of distributed control architectures for integrated process networks. *Chemical Engineering Science*, 172:434–443, 2017.

[157] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. Ojea, E. Solowjow, and S. Levine. Residual reinforcement learning for robot control. In *2019 International Conference on Robotics and Automation*, pages 6023–6029, 2019.

[158] J. Jonkman, S. Butterfield, W. Musial, and G. Scott. Definition of a 5-MW reference wind turbine for offshore system development. Technical report, February 2009.

[159] O. Kallenberg. *Foundations of Modern Probability*. Springer, third edition, 2021.

[160] M. Kamel, W. Hamouda, and A. Youssef. Ultra-Dense networks: A survey. *IEEE Communications Surveys & Tutorials*, 18:2522–2545, 2016.

[161] S. Kamelian and K. Salahshoor. A novel graph-based partitioning algorithm for large-scale dynamical systems. *International Journal of Systems Science*, 46:227–245, 2015.

[162] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[163] G. Karypis and V. Kumar. METIS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. Technical report, University of Minnesota, Department of Computer Science, 1998.

[164] I. Katic, J. Højstrup, and N. O. Jensen. A simple model for cluster efficiency. In *European Wind Energy Association Conference and Exhibition*, pages 407–410, 1986.

[165] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49:291–307, 1970.

[166] B. Kersbergen, J. Rudan, T. van den Boom, and D. De Schutter. Towards railway traffic management using switching Max-plus-linear systems: Structure analysis and rescheduling. *Discrete Event Dynamic Systems*, 26:183–223, 2016.

[167] B. Kersbergen, T. van den Boom, and B. De Schutter. Distributed model predictive control for railway traffic management. *Transportation Research Part C: Emerging Technologies*, 68:462–489, 2016.

[168] H.K. Khalil. *Nonlinear Systems*. Pearson, third edition edition, 2002.

[169] S. Kim, S. Tak, D. Lee, and H. Yeo. Distributed model predictive approach for large-scale road network perimeter control. *Transportation Research Record: Journal of the Transportation Research Board*, 2673:515–527, 2019.

[170] J. P. Koeln and A. G. Alleyne. Stability of decentralized model predictive control of graph-based power flow systems via passivity. *Automatica*, 82:29–34, 2017.

[171] I K Kookos and A I Lygeros. An algorithmic method for control structure selection based on the RGA and RIA interaction measures. *Chemical Engineering Research and Design*, 76:458–464, 1998.

[172] M. Kordestani, A. A. Safavi, and M. Saif. Recent survey of large-scale systems: Architectures, controller strategies, and industrial applications. *IEEE Systems Journal*, 15:5440–5453, 2021.

[173] C. Kravaris and J. C. Kantor. Geometric methods for nonlinear process control. 1. Background. *Industrial & Engineering Chemistry Research*, 29:2295–2402, 1990.

[174] M. Krstic. Inverse optimal safety filters. *IEEE Transactions on Automatic Control*, 69:16–31, 2024.

[175] P. R. Kumar and P. Varaiya. *Stochastic Systems: Estimation, Identification, and Adaptive Control*. SIAM, 2015.

[176] R. Kumar and V. K. Sharma. Whale optimization controller for load frequency control of a two-area multi-source deregulated power system. *International Journal of Fuzzy Systems*, 22:122–137, 2020.

[177] P. S. Kundur and O. P. Malik. *Power System Stability and Control*. McGraw Hill, second edition, 2022.

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

[179] A. La Bella, F. Bonassi, M. Farina, and R. Scattolini. Two-layer model predictive control of systems with independent dynamics and shared control resources. *IFAC-PapersOnLine*, 52:96–101, 2019.

[180] A. La Bella, P. Klaus, G. Ferrari-Trecate, and R. Scattolini. Supervised model predictive control of large-scale electricity networks via clustering methods. *Optimal Control Applications and Methods*, 43:44–64, 2022.

[181] M. Lahijanian, S. B. Andersson, and C. Belta. Formal verification and synthesis for discrete-time stochastic systems. *IEEE Transactions on Automatic Control*, 60:2031–2045, 2015.

[182] K. Lakshmi, N. K. Visalakshi, and S. Shanthi. Data clustering using K-Means based on Crow Search Algorithm. *Sādhanā*, 43:1–12, 2018.

[183] G.K. H. Larsen, N. D. Van Foreest, and J. M. A. Scherpen. Power supply–demand balance in a smart grid: An information sharing model for a market mechanism. *Applied Mathematical Modelling*, 38:3350–3360, 2014.

[184] A. Lavaei. Abstraction-based synthesis of stochastic hybrid systems. In *Proceedings of the 27th ACM International Conference on Hybrid Systems: Computation and Control (HSCC)*, pages 1–11. ACM, 2024.

[185] A. Lavaei, M. Khaled, S. Soudjani, and M. Zamani. AMYTISS: Parallelized automated controller synthesis for large-scale stochastic systems. In *Computer Aided Verification*, pages 461–474. Springer, 2020.

[186] A. Lavaei, S. Soudjani, A. Abate, and M. Zamani. Automated verification and synthesis of stochastic hybrid systems: A survey. *Automatica*, 146:1–40, 2022.

[187] A. Lavaei, S. Soudjani, E. Frazzoli, and M. Zamani. Constructing MDP abstractions using data with formal guarantees. *IEEE Control Systems Letters*, 7:460–465, 2023.

[188] E. Lee. The past, present and future of cyber-physical systems: A focus on models. *Sensors*, 15:4837–4869, 2015.

[189] J. H. Lee. Model predictive control: Review of the three decades of development. *International Journal of Control, Automation and Systems*, 9:415–424, 2011.

[190] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Physical Review Letters*, 100:5, 2008.

[191] S. Li, Y. Zheng, and Z. Lin. Impacted-region optimization for distributed model predictive control systems with constraints. *IEEE Transactions on Automation Science and Engineering*, 12:1447–1460, 2015.

[192] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. DOI: 10.48550/arXiv.1509.02971, 2019.

[193] D. Limon, I. Alvarado, T. Alamo, and E. F. Camacho. Robust tube-based mpc for tracking of constrained linear systems with additive disturbances. *Journal of Process Control*, 20:248–260, 2010.

[194] C. Lin. Structural controllability. *IEEE Transactions on Automatic Control*, 19:201–208, 1974.

[195] Z. Lin, Z. Chen, C. Qu, Y. Guo, J. Liu, and Q. Wu. A hierarchical clustering-based optimization strategy for active power dispatch of large-scale wind farm. *International Journal of Electrical Power & Energy Systems*, 121:1–15, 2020.

[196] J. Liu, X. Chen, D. Muñoz de la Peña, and P. D. Christofides. Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. *AIChE Journal*, 56:2137–2149, 2010.

[197] J. Liu, D. Muñoz De La Peña, and P. D. Christofides. Distributed model predictive control of nonlinear process systems. *AIChE Journal*, 55:1171–1184, 2009.

[198] J. Liu, D. Muñoz De La Peña, B. J. Ohran, P. D. Christofides, and J. F. Davis. A two-tier control architecture for nonlinear process systems with continuous/asynchronous feedback. *International Journal of Control*, 83:257–272, 2010.

[199] P. Liu, A. Kurt, and U. Ozguner. Distributed model predictive control for cooperative and flexible vehicle platooning. *IEEE Transactions on Control Systems Technology*, 27:1115–1128, 2019.

[200] Q. Liu, Y. Guo, L. Deng, H. Liu, D. Li, and H. Sun. Residual deep reinforcement learning for inverter-based volt-var control. *IEEE Transactions on Sustainable Energy*, 16:269–283, 2024.

[201] R. Louf, C. Roth, and M. Barthelemy. Scaling in transportation networks. *PLoS ONE*, 9:1–8, 2014.

[202] X. Luan, B. De Schutter, L. Meng, and F. Corman. Decomposition and distributed optimization of real-time traffic management for large-scale railway networks. *Transportation Research Part B: Methodological*, 141:72–97, 2020.

[203] P. R. Lyman and C. Georgakis. Plant-wide control of the Tennessee Eastman problem. *Computers & Chemical Engineering*, 19:321–331, 1995.

[204] J. Machowski. *Predictive Control with Constraints*. Prentice Hall, 2002.

[205] J. M. Maestre and H. Ishii. A PageRank based coalitional control scheme. *International Journal of Control, Automation and Systems*, 15:1983–1990, 2017.

[206] J. M. Maestre, D. Muñoz de la Peña, A. Jiménez Losada, E. Algaba, and E. F. Camacho. A coalitional control scheme with applications to cooperative game theory. *Optimal Control Applications and Methods*, 35:592–608, 2014.

[207] J. M. Maestre, F. J. Muros, F. Fele, and E. F. Camacho. An assessment of coalitional control in water systems. In *2015 European Control Conference (ECC)*, pages 3286–3291, 2015.

[208] J. M. Maestre and R. R. Negenborn, editors. *Distributed Model Predictive Control Made Easy*. Springer, 2014.

[209] J. M. Maestre and C. Ocampo-Martinez, editors. *Control Systems Benchmarks*. Springer, 2025.

[210] F. D. Malliaros and M. Vazirgiannis. Clustering and community detection in directed networks: A survey. *Physics Reports*, 533:95–142, 2013.

[211] J. Martinez-Piazuelo, N. Quijano, and C. Ocampo-Martinez. Nash equilibrium seeking in full-potential population games under capacity and migration constraints. *Automatica*, 141:1–8, 2022.

[212] E. Masero, P. R. Baldivieso-Monasterios, J. M. Maestre, and P. A. Trodden. Robust coalitional model predictive control with plug-and-play capabilities. *Automatica*, 153:14, 2023.

[213] E. Masero, L. A. Fletscher, and J. M. Maestre. A coalitional model predictive control approach for heterogeneous cellular networks. In *European Control Conference (ECC)*, pages 448–453, 2020.

[214] E. Masero, L. A. Fletscher, and J. M. Maestre. A coalitional model predictive control for the energy efficiency of next-generation cellular networks. *Energies*, 13:6546, 2020.

[215] E. Masero, J. R. D. Frejo, J. M. Maestre, and E. F. Camacho. A light clustering model predictive control approach to maximize thermal power in solar parabolic-trough plants. *Solar Energy*, 214:531–541, 2021.

[216] E. Masero, J. M. Maestre, and E. F. Camacho. Market-based clustering of model predictive controllers for maximizing collected energy by parabolic-trough solar collector fields. *Applied Energy*, 306:1–12, 2022.

[217] E. Masero, J. M. Maestre, A. Ferramosca, M. Francisco, and E. F. Camacho. Robust coalitional model predictive control with predicted topology transitions. *IEEE Transactions on Control of Network Systems*, 8:1869–1880, 2021.

[218] E. Masero, S. Ruiz-Moreno, J. R. D. Frejo, J. M. Maestre, and E. F. Camacho. A fast implementation of coalitional model predictive controllers based on machine learning: Application to solar power plants. *Engineering Applications of Artificial Intelligence*, 118:1–10, 2023.

[219] F. B. Mathiesen, S. Haesaert, and L. Laurenti. Scalable control synthesis for stochastic systems via structural IMDP abstractions. In *Proceedings of the 28th ACM International Conference on Hybrid Systems: Computation and Control*, pages 1–12, 2025.

[220] MATLAB & Simulink. Optimization toolbox. https://www.mathworks.com/products/optimization.

[221] A. Maxim and C. Caruntu. A coalitional distributed model predictive control perspective for a cyber-physical multi-agent application. *Sensors*, 21:1–16, 2021.

[222] A. Maxim and C. Caruntu. Coalitional distributed model predictive control strategy for vehicle platooning applications. *Sensors*, 22:1–20, 2022.

[223] A. Maxim, J. M. Maestre, C. F. Caruntu, and C. Lazar. Robust coalitional distributed model predictive control algorithm with stability via terminal constraint. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 964–969, 2018.

[224] A. Maxim, O. Pauca, and C. Caruntu. Distributed model predictive control and coalitional control strategies - comparative performance analysis using an eight-tank process case study. *Actuators*, 12:1–23, 2023.

[225] A. Maxim, O. Pauca, and C. Caruntu. Coalitional distributed model predictive control strategy with switching topologies for multi-agent systems. *Electronics*, 13:18, 2024.

[226] A. Maxim, O. Pauca, J. M. Maestre, and C. F. Caruntu. Assessment of computation methods for coalitional feedback controllers. In *2022 European Control Conference (ECC)*, pages 1472–1477, 2022.

[227] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[228] D. Q. Mayne, M. M. Seron, and S. V. Raković. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41:219–224, 2005.

[229] T. J. McAvov and N. Ye. Base control for the Tennessee Eastman problem. *Computers & Chemical Engineering*, 18:383–413, 1994.

[230] A. Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36:30–44, 2016.

[231] A. Messmer and M. Papageorgiou. METANET: A macroscopic simulation program for motorway networks. *Traffic Engineering & Control*, 31:466–470, 1990.

[232] M. Metzger and G. Polakow. A survey on applications of agent technology in industrial process control. *IEEE Transactions on Industrial Informatics*, 7:570–581, 2011.

[233] M. Moharir, D. B. Pourkargar, A. Almansoori, and P. Daoutidis. Distributed model predictive control of an amine gas sweetening plant. *Industrial & Engineering Chemistry Research*, 57:13103–13115, 2018.

[234] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei. A survey of distributed optimization and control algorithms for electric power systems. *IEEE Transactions on Smart Grid*, 8:2941–2962, 2017.

[235] D. Monderer and L. S. Shapley. Potential games. *Games and Economic Behavior*, 14:124–143, 1996.

[236] A. W. Moore. Efficient memory-based learning for robot control. Technical report, University of Cambridge, Computer Laboratory, 1990.

[237] M. M. Morato and M. S. Felix. Data science and model predictive control: A survey of recent advances on data-driven MPC algorithms. *Journal of Process Control*, 144:1–17, 2024.

[238] R. Muduli, D. Jena, and T. Moger. A survey on load frequency control using reinforcement learning-based data-driven controller. *Applied Soft Computing*, 166:1–26, 2024.

[239] F. J. Muros, E. Algaba, J. M. Maestre, and E. F. Camacho. The banzhaf value as a design tool in coalitional control. *Systems & Control Letters*, 104:21–30, 2017.

[240] F. J. Muros, J. M. Maestre, C. Ocampo-Martinez, E. Algaba, and E. F. Camacho. A game theoretical randomized method for large-scale systems partitioning. *IEEE Access*, 6:42245–42263, 2018.

[241] M. Nazeri, T. Badings, S. Soudjani, and A. Abate. Data-Driven yet formal policy synthesis for stochastic nonlinear dynamical systems. In *7th Annual Conference on Learning for Dynamics and Control (L4DC)*, pages 1–15, 2025.

[242] R. R. Negenborn, G. Hug-Glanzmann, B. De Schutter, and G. Andersson. A novel coordination strategy for multi-agent control using overlapping subnetworks with application to power systems. In *Efficient Modeling and Control of Large-Scale Systems*, pages 251–278. Springer, 2010.

[243] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103:8577–8582, 2006.

[244] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:1–15, 2004.

[245] T. Ni and M. Kamgarpour. A learning-based approach to stochastic optimal control under reach-avoid constraint. In *Proceedings of the 28th ACM International Conference on Hybrid Systems: Computation and Control*, pages 1–8, 2025.

[246] A. Nilim and L. El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53:780–798, 2005.

[247] A. Núñez, C. Ocampo-Martinez, J. M. Maestre, and B. De Schutter. Time-varying scheme for noncentralized model predictive control of large-scale systems. *Mathematical Problems in Engineering*, 2015:1–17, 2015.

[248] A. Nurudeen, S. Lawal, and A. Beli. Load frequency control strategies in multi-area power systems: A comprehensive review. *Energies*, 57:1–19, 2022.

[249] C. Ocampo-Martinez, D. Barcelli, V. Puig, and A. Bemporad. Hierarchical and decentralised model predictive control of drinking water networks: Application to Barcelona case study. *IET Control Theory And Applications*, 6:62–71, 2012.

[250] C. Ocampo-Martinez, S. Bovo, and V. Puig. Partitioning approach oriented to the decentralised predictive control of large-scale systems. *Journal of Process Control*, 21:775–786, 2011.

[251] C Ocampo-Martinez, V Fambrini, D Barcelli, and V Puig. Model predictive control of drinking water networks: A hierarchical and decentralized approach. In *Proceedings of the 2010 American Control Conference*, pages 3951–3956. IEEE, 2010.

[252] C. Ocampo-Martinez, V. Puig, G. Cembrano, R. Creus, and M. Minoves. Improving water management efficiency by using optimization-based control strategies: the Barcelona case study. *Water Supply*, 9:565–575, 2009.

[253] K. Ogata. *Modern Control Engineering*. Prentice Hall, fifth edition, 2022.

[254] J. Paauw, B. Roossien, M. B. C. Aries, and O. Guerra Santin. Energy pattern generator - Understanding the effect of user behaviour on energy systems. In *First European Conference Energy Efficiency and Behaviour*, 2009.

[255] S. K. Pandey, S. R. Mohanty, and N. Kishor. A literature survey on load–frequency control for conventional and distribution generation power systems. *Renewable and Sustainable Energy Reviews*, 25:318–334, 2013.

[256] M. Papageorgiou, E. Kosmatopoulos, and I. Papamichail. Effects of variable speed limits on motorway traffic flow. *Transportation Research Record: Journal of the Transportation Research Board*, 2047:37–48, 2008.

[257] M. Papageorgiou and A. Kotsialos. Freeway ramp metering: An overview. *IEEE Transactions on Intelligent Transportation Systems*, 3:271–281, 2002.

[258] J. Paparrizos and L. Gravano. k-Shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1855–1870, 2015.

[259] A. Parisio, E. Rikos, and L. Glielmo. A model predictive control approach to microgrid operation optimization. *IEEE Transactions on Control Systems Technology*, 22:1813–1827, 2014.

[260] L. Pedroso, P. Batista, and W. P. M. H. Heemels. Distributed design of ultra large-scale control systems: Progress, challenges, and prospects. *Annual Reviews in Control*, 59:1–22, 2025.

[261] A. Pnueli. The temporal logic of programs. In *18th annual symposium on foundations of computer science (sfcs 1977)*, pages 46–57, 1977.

[262] A. Poullikkas. A comparative overview of large-scale battery systems for electricity storage. *Renewable and Sustainable Energy Reviews*, 27:778–788, 2013.

[263] D. B. Pourkargar, A. Almansoori, and P. Daoutidis. Impact of decomposition on distributed model predictive control: A process network case study. *Industrial & Engineering Chemistry Research*, 56:9606–9616, 2017.

[264] D. B. Pourkargar, M. Moharir, A. Almansoori, and P. Daoutidis. Distributed estimation and nonlinear model predictive control using community detection. *Industrial & Engineering Chemistry Research*, 58:13495–13507, 2019.

[265] S. Prajna, A. Jadbabaie, and G. J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Transactions on Automatic Control*, 52:1415–1428, 2007.

[266] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11:733–764, 2003.

[267] Y. Qin, M. Cao, and B. D. O. Anderson. Lyapunov criterion for stochastic systems and its applications in distributed computation. *IEEE Transactions on Automatic Control*, 65:546–560, 2020.

[268] T. D. Raj, C. Kumar, P. Kotsampopoulos, and H. H. Fayek. Load frequency control in two-area multi-source power system using bald eagle-sparrow search optimization tuned PID controller. *Energies*, 16:1–25, 2023.

[269] E. Rakhshani, K. Rouzbehi, A. J. Sánchez, A. C. Tobar, and E. Pouresmaeil. Integration of large scale PV-based generation into power systems: A survey. *Energies*, 12:1–19, 2019.

[270] M. Ranjan and R. Shankar. A literature survey on load frequency control considering renewable energy integration in power system: Recent trends and future prospects. *Journal of Energy Storage*, 45:1–33, 2022.

[271] J. B. Rawlings, D. Q. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, second edition, 2017.

[272] D. Ray. *A Game-Theoretic Perspective on Coalition Formation*. Oxford University Press, 2007.

[273] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74, 2006.

[274] R. Reiter, J. Hoffmann, D. Reinhardt, F. Messerer, K. Baumgärtner, S. Sawant, J. Boedecker, M. Diehl, and S. Gros. Synthesis of model predictive control and reinforcement learning: Survey and classification, 2025.

[275] A. Riccardi, T. Badings, L. Laurenti, A. Abate, and B. De Schutter. Temporal logic control of nonlinear stochastic systems with online performance optimization, 2026.

[276] A. Riccardi, L. Laurenti, and B. De Schutter. Code underlying the publication: A benchmark for the application of distributed control techniques to the electricity network of the European economic area. DOI: 10.4121/d2c0d075-1c49-41af-8113-5e50c27ca97e, 2024.

[277] A. Riccardi, L. Laurenti, and B. De Schutter. Code underlying the publication: A generalized partitioning strategy for control. DOI: 10.4121/90ada13d-a6c9-4e4c-a046-2b984595bcdd, 2024.

[278] A. Riccardi, L. Laurenti, and B. De Schutter. A generalized partitioning strategy for distributed control. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, pages 6134–6141, 2024.

[279] A. Riccardi, L. Laurenti, and B. De Schutter. A benchmark for multi-agent control of energy systems: The European economic area hybrid electricity network benchmark. In *2025 European Control Conference (ECC)*, pages 2575–2582, 2025.

[280] A. Riccardi, L. Laurenti, and B. De Schutter. A benchmark for the application of distributed control techniques to the electricity network of the European economic area. In *Control Systems Benchmarks*. Spinger, 2025.

[281] A Riccardi, L. Laurenti, and B. De Schutter. Code underlying the publication: A general partitioning strategy for non-centralized control. DOI: 10.4121/0e7dd651-66d7-451e-889b-d558e7d5b986, 2025.

[282] A. Riccardi, L. Laurenti, and B. De Schutter. A general partitioning strategy for non-centralized control. DOI: 10.48550/arXiv.2502.21126, 2025.

[283] A. Riccardi, L. Laurenti, and B. De Schutter. Partitioning techniques for non-centralized predictive control: A systematic review and novel theoretical insights. DOI: 10.48550/arXiv.2509.11470, 2025.

[284] A. Riccardi, L. Laurenti, and B. De Schutter. Partitioning techniques for non-centralized predictive control: A systematic review and novel theoretical insights. *Annual Reviews in Control*, 61:1–41, 2026.

[285] S. Riverso and G. Ferrari-Trecate. Tube-based distributed control of linear constrained systems. *Automatica*, 48:2860–2865, 2012.

[286] R. R. Rocha, L. C. Oliveira-Lopes, and P. D. Christofides. Partitioning for distributed model predictive control of nonlinear processes. *Chemical Engineering Research and Design*, 139:116–135, 2018.

[287] G. Rogers. *Power System Oscillations*. Springer, 2000.

[288] R. Rostami, G. Costantini, and D. Görges. ADMM-based distributed model predictive control: Primal and dual approaches. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 6598–6603, 2017.

[289] S. Rozada, D. Apostolopoulou, and E. Alonso. Deep multi-agent reinforcement learning for cost efficient distributed load frequency control. *IET Energy Systems Integration*, 3:327–343, 2020.

[290] A. Sánchez-Amores, P. Chanfreut, J. M. Maestre, and E. F. Camacho. Coalitional model predictive control with different inter-agent interaction modes. *European Journal of Control*, 68:1–7, 2022.

[291] A. Sánchez-Amores, P. Chanfreut, J. M. Maestre, and E. F. Camacho. Robust coalitional model predictive control with negotiation of mutual interactions. *Journal of Process Control*, 123:64–75, 2023.

[292] A. Sánchez-Amores, J. Martinez-Piazuelo, J. M. Maestre, C. Ocampo-Martinez, E. F. Camacho, and N. Quijano. Coalitional model predictive control of parabolic-trough solar collector fields with population-dynamics assistance. *Applied Energy*, 334:11, 2023.

[293] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohmé. Coalition structure generation with worst case guarantees i. *Artificial Intelligence*, 111:209–238, 1999.

[294] W. H. Sandholm. *Population Games and Evolutionary Dynamics*. The MIT Press, 2010.

[295] R. Scattolini. Architectures for distributed and hierarchical model predictive control – A review. *Journal of Process Control*, 19:723–731, 2009.

[296] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1:27–64, 2007.

[297] R. Segala and N. A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2:250–237, 1995.

[298] P. Segovia, V. Puig, E. Duviella, and L. Etienne. Distributed model predictive control using optimality condition decomposition and community detection. *Journal of Process Control*, 99:54–68, 2021.

[299] M. E. Sezer and D. D. Siljak. Nested e-decompositions and clustering of complex systems. *Automatica*, 22:321–331, 1986.

[300] V. Shaferman, M. Schwegel, T. Glück, and A. Kugi. Continuous-time least-squares forgetting algorithms for indirect adaptive control. *European Journal of Control*, 62:105–112, 2021.

[301] L. Shapley. A value for n-person games. In *Contributions to the Theory of Games II*, pages 307–317. Princeton University Press, 1953.

[302] L. Shi, R. Dadashi, Y. Chi, P. Castro, and M. Geist. Offline reinforcement learning with on-policy q-function regularization. In *Machine Learning and Knowledge Discovery in Databases: Research Track*, pages 455–471. Springer Nature Switzerland, 2023.

[303] D. D. Šiljak. *Decentralized Control of Complex Systems.* Academic Press, Inc., 1991.

[304] D. D. Šiljak. *Large-Scale Dynamic Systems: Stability and Structure.* Dover Publications Inc., 2008.

[305] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling. Residual policy learning. DOI: 10.48550/arXiv.1812.06298, 2019.

[306] S. Siniscalchi-Minna, F. D. Bianchi, C. Ocampo-Martinez, J. L. Domínguez-García, and B. De Schutter. A non-centralized predictive control strategy for wind farm active power control: A wake-based partitioning approach. *Renewable Energy*, 150:656–669, 2020.

[307] R. Sioshansi, P. Denholm, J. Arteaga, S. Awara, S. Bhattacharjee, A. Botterud, W. Cole, A. Cortés, A. de Queiroz, J. DeCarolis, Z. Ding, N. DiOrio, Y. Dvorkin, U. Helman, J. X. Johnson, I. Konstantelos, T. Mai, H. Pandžić, D. Sodano, G. Stephen, A. Svoboda, H. Zareipour, and Z. Zhang. Energy-storage modeling: State-of-the-art and future research directions. *IEEE Transactions on Power Systems*, 37:860–875, 2022.

[308] S. Siri, C. Pasquale, S. Sacone, and A. Ferrara. Freeway traffic control: A survey. *Automatica*, 130:1–25, 2021.

[309] R. Siringoringo and J. Jamaluddin. Initializing the fuzzy c-means cluster center with particle swarm optimization for sentiment clustering. *Journal of Physics: Conference Series*, 1361:1–6, 2019.

[310] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control: Analysis and Design.* John Wiley & Sons, second edition, 2001.

[311] SOBEK. User manual. Technical report, WL Delft Hydraulics, Delft, The Netherlands, 2000.

[312] T. Söderström. *Discrete-time Stochastic Systems: Estimation and Control.* Springer, 2002.

[313] Y. S. Son, R. Baldick, K. Lee, and S. Siddiqi. Short-term electricity market auction game analysis: Uniform and pay-as-bid pricing. *IEEE Transactions on Power Systems*, 19:1990–1998, 2004.

[314] E. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control*, 26:346–358, 1981.

[315] M. Srinivas and L. M. Patnaik. Genetic algorithms: A survey. *Computer*, 27:17–26, 1994.

[316] R. E. Stearns. Convergent transfer schemes for n-person games. *Transactions of the American Mathematical Society*, 134:449–459, 1968.

[317] L. Steenhoff, A. Riccardi, L. Laurenti, and B. De Schutter. Distributed residual deep reinforcement learning for load frequency control, 2025.

[318] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia. Cooperative distributed model predictive control. *Systems & Control Letters*, 59:460–469, 2010.

[319] K. Stoffel and A. Belkoniene. Parallel k/h-means clustering for large data sets. In *Euro-Par'99 Parallel Processing*, pages 1451–1454, 1999.

[320] M. Suilen, T. Badings, E. M. Bovy, D. Parker, and N. Jansen. Robust Markov decision processes: A place where AI and formal methods meet. In *Principles of Verification: Cycling the Probabilistic Landscape*, pages 126–154. Springer, 2025.

[321] T. H. Summers and J. Lygeros. Distributed model predictive consensus via the alternating direction method of multipliers. In *50th Annual Allerton Conference on Communication, Control, and Computing*, pages 79–84, 2012.

[322] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. The MIT Press, second edition, 2018.

[323] P. Tabuada. *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer, 2009.

[324] W. Tang, A. Allman, D. B. Pourkargar, and P. Daoutidis. Optimal decomposition for distributed optimization in nonlinear model predictive control through community detection. *Computers & Chemical Engineering*, 111:43–54, 2018.

[325] W. Tang, P. Carrette, Y. Cai, J. M. Williamson, and P. Daoutidis. Automatic decomposition of large-scale industrial processes for distributed MPC on the Shell-Yokogawa Platform for Advanced Control and Estimation (PACE). *Computers & Chemical Engineering*, 178:1–8, 2023.

[326] W. Tang and P. Daoutidis. Network decomposition for distributed control through community detection in input-output bipartite graphs. *Journal of Process Control*, 64:7–14, 2018.

[327] W. Tang, D. B. Pourkargar, and P. Daoutidis. Relative time-averaged gain array (RTAGA) for distributed control-oriented network decomposition. *AIChE Journal*, 64:1682–1690, 2018.

[328] M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. De Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis. Gymnasium: A standard interface for reinforcement learning environments, 2025.

[329] P. A. Trodden and J. M. Maestre. Distributed predictive control with minimization of mutual disturbances. *Automatica*, 77:31–43, 2017.

[330] F. Vallee, G. Brunieau, M. Pirlot, O. Deblecker, and J. Lobry. Optimal wind clustering methodology for adequacy evaluation in system generation studies using nonsequential Monte Carlo simulation. *IEEE Transactions on Power Systems*, 26:2173–2184, 2011.

[331] B. Van Huijgevoort, O. Schön, S. Soudjani, and S. Haesaert. SySCoRe: Synthesis via stochastic coupling relations. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, pages 1–11, 2023.

[332] J. E. van Zyl, D. A. Savic, and G. A. Walters. Operational optimization of water distribution systems using a hybrid genetic algorithm. *Journal of Water Resources Planning and Management*, 130:160–170, 2004.

[333] A. N. Venkat. *Distributed model predictive control: Theory and applications*. PhD thesis, University of Wisconsin-Madison, 2006.

[334] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, and S. J. Wright. Distributed MPC strategies with application to power system automatic generation control. *IEEE Transactions on Control Systems Technology*, 16:1192–1206, 2008.

[335] A. N. Venkat, J. B. Rawlings, and S. J. Wright. Stability and optimality of distributed model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 6680–6685, 2005.

[336] E. Vlahakis, L. Dritsas, and G. Halikias. Distributed LQR design for a class of large-scale multi-area power systems. *Energies*, 12:1–28, 2019.

[337] K. P. Wabersich and M. N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *Automatica*, 129:1–13, 2021.

[338] Y. Wan, K. Vamvoudakis, Y. Chen, and F. Lewis, editors. *Smarter Cyber Physical Systems: Enabling Methodologies and Applications*. CRC Press, 2025.

[339] J. Wang, C. Song, J. Zhao, Z. Mo, and Z. Xu. Distributed model predictive control-oriented network decomposition based on full dynamic response. *AIChE Journal*, 69:1–15, 2023.

[340] L. Wang, A. Dubey, A. H. Gebremedhin, A. K. Srivastava, and N. Schulz. MPC-based decentralized voltage control in power distribution systems with EV and PV coordination. *IEEE Transactions on Smart Grid*, 13:2908–2919, 2022.

[341] W. Wang and J. P. Koeln. Hierarchical clustering of constrained dynamic systems using robust positively invariant sets. *Automatica*, 147:1–8, 2023.

[342] Y. Wang, W. Yin, and J. Zeng. Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, 78:29–63, 2019.

[343] Y. Wei, S. Li, and J. Wu. Event-triggered distributed model predictive control with optimal network topology. *International Journal of Robust and Nonlinear Control*, 30:2186–2203, 2020.

[344] E. M. Wolff, U. Topcu, and R. M. Murray. Robust control of uncertain Markov decision processes with temporal logic specifications. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 3372–3379, 2012.

[345] L. Xie, X. Cai, J. Chen, and H. Su. GA based decomposition of large scale distributed model predictive control systems. *Control Engineering Practice*, 57:111–125, 2016.

[346] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16:645–678, 2005.

[347] Z. Yan and Y. Xu. A multi-agent deep reinforcement learning method for cooperative load frequency control of a multi-area power system. *IEEE Transactions on Power Systems*, 35:4599–4608, 2020.

[348] L. Ye, C. Zhang, Y. Tang, W. Zhong, Y. Zhao, P. Lu, B. Zhai, H. Lan, and Z. Li. Hierarchical model predictive control strategy based on dynamic active power dispatch for wind power cluster integration. *IEEE Transactions on Power Systems*, 34:4617–4629, 2019.

[349] X. Yin, Y. Liu, L. Yang, and W. Gao. Abnormal data cleaning method for wind turbines based on constrained curve fitting. *Energies*, 15:6373, 2022.

[350] G. Zames and A. El-Sakkary. Uncertainty in unstable systems: The gap metric. *IFAC Proceedings Volumes*, 14:149–152, 1981.

[351] A. Zecevic and D. D. Siljak. *Control of Complex Systems: Structural Constraints and Uncertainty*. Springer, 2010.

[352] X. Zhang, Q. Han, X. Ge, D. Ding, L. Ding, D. Yue, and C. Peng. Networked control systems: A survey of trends and techniques. *IEEE/CAA Journal of Automatica Sinica*, 7:1–17, 2020.

[353] X. Zhang and M. E. J. Newman. Multiway spectral community detection in networks. *Physical Review E*, 92:1–8, 2015.

[354] Y. Zhang, Y. Zheng, and S. Li. Enhancing cooperative distributed model predictive control for the water distribution networks pressure optimization. *Journal of Process Control*, 84:70–88, 2019.

[355] J. Zhao, T. Zhang, S. Tang, J. Zhang, Y. Zhu, and J. Yan. Optimal scheduling strategy of wind farm active power based on distributed model predictive control. *Processes*, 11:1–22, 2023.

[356] Y. Zheng, Y. Wei, and S. Li. Coupling degree clustering-based distributed model predictive control network design. *IEEE Transactions on Automation Science and Engineering*, 15:1749–1758, 2018.

[357] Y. Zhou, B. Rao, and W. Wang. UAV swarm intelligence: Recent advances and future trends. *IEEE Access*, 8:183856–183878, 2020.

[358] W. Zhu and S. Li. Study on discrete boundary-feedback-control strategy for traffic flow based on macroscopic fundamental diagram. *Physica A: Statistical Mechanics and its Applications*, 523:1237–1247, 2019.

[359] Y. Zhu, H. He, and D. Zhao. LMI-based synthesis of string-stable controller for cooperative adaptive cruise control. *IEEE Transactions on Intelligent Transportation Systems*, 21:4516–4525, 2020.

[360] Đ. Žikelić, M. Lechner, T. A. Henzinger, and K. Chatterjee. Learning control policies for stochastic systems with reach-avoid guarantees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11926–11935, 2023.

List of Publications

Publications

8. **A. Riccardi**, T. Badings, L. Laurenti, A. Abate, and B. De Schutter: Temporal logic control of nonlinear stochastic systems with online performance optimization, 2025, pp. 1–12, *submitted to Automatica*.

7. **A. Riccardi**, L. Laurenti, and B. De Schutter: Partitioning techniques for non-centralized predictive control: A systematic review and novel theoretical insights, in *Annual Reviews in Control* 61, 2026, pp. 1–41.

6. L. Steenhoff, **A. Riccardi**, L. Laurenti, and B. De Schutter: Distributed residual deep reinforcement learning for load frequency control, 2025, pp. 1–6, *to be submitted to: IEEE Control Systems Letters*.

5. **A. Riccardi**, L. Laurenti, and B. De Schutter: A general partitioning strategy for non-centralized control, 2025, pp. 1–14, DOI: 10.48550/arXiv.2502.21126, *under review at IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

4. **A. Riccardi**, L. Laurenti, and B. De Schutter: A benchmark for multi-agent control of energy systems: The European economic area hybrid electricity network benchmark, in *2025 European Control Conference (ECC)*, 2025, pp. 2575–2582.

3. **A. Riccardi**, L. Laurenti, and B. De Schutter: A benchmark for the application of distributed control techniques to the electricity network of the European economic area, in *Control Systems Benchmarks*, Springer, 2025, pp. 9–28.

2. **A. Riccardi**, L. Laurenti, and B. De Schutter: A generalized partitioning strategy for distributed control, in *2024 IEEE 63rd Conference on Decision and Control (CDC)*, 2024, pp. 6134–6141.

1. **A. Riccardi**, G. P. Furtado, J. Sikorski, M. Vendittelli, and S. Misra: Field model identification and control of a mobile electromagnet for remote actuation of soft robots, in *IEEE Robotics and Automation Letters* 8, 2023, pp. 4092–4098.

Included in this thesis.

Software

4. **A. Riccardi**, L. Laurenti, T. Badings, A. Abate, and B. De Schutter: Code for publication: Temporal logic control of nonlinear stochastic systems with online performance optimization, 2026. DOI: 10.4121/631b574d-40e8-4951-b3ac-e304e3f34b13

3. **A. Riccardi**, L. Laurenti, and B. De Schutter: Code for publication: A general partitioning strategy for non-centralized control, 2025. DOI: 10.4121/0e7dd651-66d7-451e-889b-d558e7d5b986

2. **A. Riccardi**, L. Laurenti, and B. De Schutter: Code underlying the publication: A generalized partitioning strategy for distributed control, 2024. DOI: 10.4121/90ada13d-a6c9-4e4c-a046-2b984595bcdd

1. **A. Riccardi**, L. Laurenti, and B. De Schutter: Code underlying the publication: A Benchmark for the application of distributed control techniques to the electricity network of the European economic area, 2024. DOI: 10.4121/d2c0d075-1c49-41af-8113-5e50c27ca97e

Modern networks of systems have grown to a scale and complexity where centralized or human-driven control strategies are no longer sufficient to guarantee safe and efficient operation. This thesis addresses fundamental challenges in the context of Model Predictive Control (MPC) for network systems, and is divided into two parts: network partitioning for non-centralized MPC of large-scale systems, and multi-level control integrating abstraction-based methods with online optimization for nonlinear stochastic systems subject to temporal specifications.

In the first part, we address the partitioning problem that consists in finding the optimal subdivision of a network into subsystems for non-centralized operation. We propose a generalized partitioning strategy based on the selection of fundamental systems followed by an aggregation procedure minimizing a global performance metric, and complement these results with a systematization of the existing literature and the novel concepts of multi-topological network representations and predictive partitioning.

In the second part, we introduce a multi-level control framework for nonlinear stochastic systems subject to complex safety-critical specifications, integrating abstraction-based policy synthesis through a novel interval Markov decision process (IMDP) formulation with online optimization through a hybrid MPC controller. This approach yields superior control performance while guaranteeing satisfaction of temporal specifications within a predefined probability lower bound.