SAPIENZA
UNIVERSITÀ DI ROMA

Sapienza University of Rome

Department of Computer, Control and Management Engineering (DIAG)
Automatic Control, Bioengineering and Operations Research (ABRO),
Curriculum in Automatic Control (XXXV cycle)

Thesis For The Degree Of Doctor Of Philosophy

Model Predictive Control of Cyber-Physical Systems

Thesis Advisor
Prof. Francesco Delli Priscoli

Co-advisor
Prof. Francesco Liberati

Candidate
Manuel Donsante
1525802

Academic Year 2023-2024

Left blank intentionally

Abstract

Cyber-Physical Systems (CPS) represent a groundbreaking technological advancement that integrates physical processes with computational resources and networking capabilities, heralding a significant leap in efficiency, functionality, and adaptability across various applications. From revolutionizing transportation through self-driving cars to enhancing energy distribution via smart grids, CPS are poised to be pivotal in the fourth industrial revolution, fundamentally altering daily life and work in a manner akin to the transformative impacts of the internet and the World Wide Web. Originating from the concept of merging digital and physical realms, CPS aim to create systems that are inherently intelligent, adaptive, and resilient, extending beyond traditional embedded systems by leveraging advancements in computing, communication, and control.

These systems are characterized by a core architecture comprising physical components (sensors and actuators), cyber elements (computational and communication infrastructure), and control mechanisms (algorithms and software), working in unison through a feedback loop to dynamically interact with and respond to their environment. In this respect, the work done in this thesis is the application of Model Predictive Control (MPC) framework to CPS with the aim to an increase in operational efficiency, an increase in optimality with respect to resource allocation, and an increase in general responsiveness and adaptability of such systems to ambient variability. This thesis attempts to manifest the future implications in applying MPC to transform the management and control of these sophisticated cyber-physical systems within their crucial sectors via theoretical development and practical implementation of case studies.

The control methodology discussed in this thesis regards the application of MPC in three case studies: in the frame of Power Systems, Smart Cities and Industry 4.0 in the space sector.

The first work deals with the emerging complexities in modern transmission and distribution grids that arise through integration with distributed energy resources such as electric energy storage systems, renewable energy plants, and plug-in electric vehicles. The new issues are the intermittency in power generation from renewable sources and in the demand from electric vehicles present a new challenge to grids requiring advancement in grid control and optimization. Considering these challenges, in this work the candidate proposes a novel reconfiguration algorithm based on MPC for the dynamic configuration and re-configuration (topology) of the grid to minimize losses and to improve operational resilience in the presence of adverse events like faults or (cyber-)attacks. The algorithm progresses over the existing methods by removing the necessity of constantly connected grids to let autonomous grid islands be formed that can dynamically get connected and disconnected from the main grid. This research provides a critical review of existing network reconfiguration strategies, spanning between classic optimization-based methods, heuristics ((meta)heuristics), and machine learning-based solutions with their respective advantages and limits. It is hence observed that while the classic optimization methods actually give optimum solutions, they are afflicted by high computational costs. (Meta)heuristics are computationally efficient, though void of guarantees about the optimality of solutions. Machine learning based approaches, in particular Reinforcement Learning, promise policies that are near optimal but come at an enormously high demand for computational resources during training and also offer serious concerns about safety. In such a way, the proposed MPC-based solution combines the features of optimal control at a lower computational

cost and adaptability for real-time applications. This means to be the breakthrough approach in network reconfiguration, bridging the gaps that exist within today's available methodologies and thereby offering a powerful, robust, efficient, flexible solution to meet challenges posed by today's modern, dynamic grid environment.

The second work addresses a crucial challenge that urban greenhouse gases (GHG), primarily produced by buildings and transportation, with a focus on optimization of the intelligent traffic light (TL) control systems in mitigating road congestion. Given the global climate change efforts like the 2016 Paris Agreement and the EU 2019 Green Deal, the study would emphasize the need for viable urban traffic management strategies that could lead to significant GHG emission reductions, as a majority of such emissions originate from urban settings. Although an extensive literature on Intelligent TL controls is available today, it is found that there is a gap in adaptability and efficiency, mainly in real-time traffic conditions. A novel model predictive control strategy based on mixed-integer optimization has been proposed in this thesis to enhance the timings of TLs at intersections by an original approach different from classical fixed-timing strategies without any real-time reaction. The main contribution of this thesis lies in proposing an integrated MPC controller which determines both the optimal signal timing for the TLs and optimal trajectories for Automatically Driven Vehicles (ADVs), while modelling also Manually Driven Vehicles (MDVs) dynamics, leading to significant reduction of queue length and waiting times. In these terms the controller is adaptive, allowing it to operate in mixed scenarios. In addition, several innovative constraints that have been introduced within the MPC formulation allow recursive feasibility to be ensured in constraint-activating events, for instance, when a vehicle approaching the TL during red signal could bring the problem towards infeasibility because some constraints cannot be violated.

About Industry 4.0, the most important challenge this thesis tackles is the optimization of task scheduling and controlling in the spaceport within the dynamically changing space industry, which previously limited to governmental entities is now expanding to include private companies. This research was carried out within the framewor of the H2020 SESAME project–partnership led by ArianeGroup–that aims to enhance the schedule of assembly operations of space vehicles to maximize the launch throughput at the Guiana space center in Kourou. In the literatyre they are referred to as Assembly Line Balancing Problems (ALBP) and the key contribution of this work is the development of a scalable MPC algorithm, integrated with a Mixed-Integer Linear Program (MILP) model, to optimize campaign planning in real-time leverages both static and dynamic data, addressing scalability, flexibility and the ability to manage complex constraints, and real-time disturbances. Simulation results confirm the merit of proposed efficient task scheduling algorithm which retains the characteristics of standard MPC and outperforms state-of-the-art optimal scheduling heuristics maintaining similar speed which makes it suitable for real-time implementation.

Keywords: Power System, Smart City, Industry 4.0, Model Predictive Control, Cyber-Physical Systems

Contents

List of Figures

List of Tables

Nomenclature

ADV Automatically Driven Vehicles

ALBP Assembly Line Balancing Problem

AOM Adaptive Operations Module

CPS Cyber-Physical System

DG Distributed Generation

DNR Distribution Network Reconfiguration

DRL Deep Reinforcement Learning

ESS Energy Storage System

GHG Greenhouse gas

GRPW Greatest Rank Positional Weight

IDEs Intelligent Electronic Devices

KPI Key Performance Indicator

MDV Manually Driven Vehicles

MILP Mixed-Integer Linear Programming

ML Machine Learning

MLD Mixed Logical Dynamical system

MPC Model Predictive Control

NEMA US National electrical Manufacturing Association

PEVs Plug-in Electric Vehicles

PMU Phase Measurement Units

RESs Renewable Energy Sources

RL Reinforcement Learning

TL Traffic Light

Chapter 1

Introduction to Cyber-physical Systems

1.1 Introduction

The Cyber-Physical Systems (CPS) will be an innovative next-generation technology where effective physical calibration takes place with computational resources and networking capabilities. This enables them to combine the physicalities and complexities of the physical world with the processing ability in allowing efficiency, functionality and adaptability in various applications. From self-driving cars on busy city streets to smart grids managing the flow of electricity, CPS are set to play a central role in the fourth industrial revolution changing the way people live and work just as the internet did in the early 1990s and the World Wide Web did a decade later.

Genesis and Evolution of CPS: The idea of CPS emanates from the insight that it might be possible to braid the two dimensions of digital and physical worlds seamlessly not only for the purpose of acquisition and control over the processes occurring in the physical domain but the development of systems whose characteristics are by nature more intelligent, adaptive and resilient. Most importantly, this integrating also holds out the prospect of extending capabilities of traditional embedded systems. It will permit these embedded systems to take advantage of advances in computing, communication, and control to enable their autonomous adaptation to new conditions, use new data to learn, and even self-correct or optimize their performance.

Core Components and Architecture: At the heart of CPS are three core components: the physical system (sensors and actuators), the cyber system (computation and communication infrastructure), and the control system (algorithms and software). In this interaction, the components work together (and with the cloud, enabling real-time data collection, analysis, and decision-making) with a feedback loop where sensors collect data from the environment, the controller makes decisions out of this data and sends commands to actuators that act upon physical systems therefore changing the behaviour of the system. This integration not only enhances efficiency and productivity but also opens up new possibilities for innovation and services.

The creation and application of Cyber-Physical Systems present challenges like guaranteeing security, privacy, and dependability, especially considering their crucial function in key services and infrastructure. However, the advantages they offer in improving efficiency, sustainability, and economic development position the progress of CPS as a central concern for researchers, engineers, and decision-makers.

In the long run, the integration of CPS with production, logistics, and the service sector will revolutionize the existing industry practice and transition factories into the Industry 4.0 paradigm, indicating the considerable economic potential. Nevertheless, given that CPS is still an emerging area, a lot of emphasis is needed in coming up with a well-defined structure and methodology to guide its implementation in the industry. In response to this, a unified system framework supported by specific algorithms and technologies that belong to each system layer has been proposed to ensure the realization of the functionalities desired, thus improving equipment efficiency and reliability, together with product quality.

In this respect, [69] introduces the concept of 5C architecture levels for Cyber-Physical Systems (CPS) which includes the following levels:

1. *Smart Connection*: This foundational level is concerned with the collection of reliable and accurate data from machines and their components. Data can be derived from the sensors directly or sourced from controllers and enterprise manufacturing systems like ERP, MES, SCM, and CMM. Key considerations would be those of ensuring that data acquisition and transfer to a central server is in a seamless and tether-free manner, specialising with protocols such as MTConnect in terms of effectiveness, and ensuring the type and specification of sensors are the appropriate type.

2. *Conversion from Data into Information*: The unprocessed data is transformed into helpful information with the use of certain tools and methodologies at this level of conversion. Very high effort has been focused on developing algorithms for Prognostics and Health Management applications. This procedure comprises health values computation, machine remaining useful life, and in such a way bringing self-consciousness into the machines.

3. *Cyber*: This is the central level of information in which all information of all the connected machines is gathered to form a network. Individual machines within the fleet interact with each other to share insights about their status through specific analytics, which grants self-comparison capabilities. These machines compare their performance with others in the fleet and predict future behavior based on historical data.

4. *Cognition*: It is the level initiating the informative perception of the monitored system, assisting in making informed decisions. It includes presenting the knowledge obtained to expert users in a way supporting right decision-making, prioritizing tasks to reach the optimal process of maintenance. Proper infographics used to deliver knowledge to users are crucial.

5. *Configuration*: Being the feedback loop from the cyber space to the physical space, at this level, supervisory control gives way to machines that could self-configure and adjust on the basis of insight derived at the cognition level. It is a resilience control system (RCS) by applying corrective and preventive decisions to the monitored system, which helps give assurance that the system can adapt and respond to changes effectively.

Every tier of the 5C architecture has a vital role in efficiently implementing CPS, to ensure that the systems are interconnected and simultaneously intelligent, which can manage their own selves and get adopted to performance and maintenance.

Ahead, we delve into CPS applications, challenges, and future direction in a bid to illustrate their transformative potential, and the considerations therein needed in a renewed approach.

Smart Grids: CPS harbors a smart grid concept to revolutionize the way electricity is generated, distributed, and consumed in the energy sector. The systems integrate renewable energy sources, manage demand using real-time pricing, and augment grid resilience against faults and cyber-attacks. The smart meters and sensors over the grid provide the data for supply balancing with demand through dynamic processing of the same, besides providing for predictive maintenance and an effective way to distribute energy.

Healthcare: CPS has proved a game-changer in healthcare via wearable devices, remote monitoring, and telemedicine. For wearable, sensors could collect real-time vital signs for proactive management of health and care personalized to one's needs. For hospitals, CPS could enable streamlined operations from patient flow management to critical care so that resources are optimally used to enhance the patients' care [133].

Manufacturing (Industry 4.0): Nowadays, the manufacturing sector is leading exponential CPS applications growth through its smart factory creation of Industry 4.0 [69]. The factories use information from the sensors and machines in order to attain ideal production parameters, effective supply chain optimization, and predictive maintenance that results in the improvement of productivity, collaboration, resilience and flexibility in the process of manufacturing [59].

Autonomous Vehicles and Intelligent Transportation: Similar to composite systems, autonomous vehicles and the intelligent transportation systems are based upon the foundation of CPS mainly. These systems use sensors, commutation technologies along with the advanced algorithms in order to have safe navigation and interact safely with other vehicles, infrastructures, as well as adaptations to changing conditions of traffic in achieving secure and efficient transportation networks [26], [25].

Agriculture: In agriculture, CPS technologies facilitate precision farming as in the utilization of sensors to monitor soil moisture, crop health status, and the prevailing weather [95]. The automated machine is directed the same ensuring that irrigation, fertilization, and harvesting are done at the points which are optimum leading to an increased crops product and sustainable agribusiness practice.

These examples represent just a subset of the diverse sectors within the realm of cyber-physical systems.

Challenges and Future Directions

Promising applications notwithstanding, CPS faces several challenges to realize its full potential. **Interoperability** implies that different systems should have a smooth integration across domains without having to put in place alternative measures for the same objective because of more than one systems being involved. Primary among them, there is the related **security and privacy** concerns that underline both the sensitivity and potential impact of data breaches to the physical infrastructure. Another challenge is **scalability** in particular since the collection by CPS of enormous quantity regarding data through a growing number of devices and sensors.

In the horizon lying ahead, the next wave of CPS will be driven by artificial intelligence (AI), machine learning as well as edge computing. This further enhances decision support in CPS to enable more autonomous and intelligent systems. The integration of AI into CPS promises systems that are able to learn from data and predict future states while learning and adapting to new

situations with minimal human intervention.

In addition, the development in **5G and beyond wireless technology** will enable the required high speed and reliable communication infrastructure towards real-time operation of CPS. Advancement in these technologies will present new frontiers from ultra-reliable low-latency communications, for industrial automation to massive machine type communications for smart cities applications among many others.

Cyber-Physical Systems, as integration between the physical and digital worlds, approach us to improve efficiency, sustainability, and quality of life in diverse domains. They have a bright future since they constantly evolve and assert also in science and society their dominance.

As we embark further in the meanings of Cyber-Physical Systems (CPS), there emerges a direction that their infusion on our daily fabric and industries transcends to be more than just technological enhancement but touching to critical social ramifications, the ethical dimensions, and especially multi-level disciplinary capability for their development and governance.

Societal Impacts of CPS

The impact that full-scale adoption of CPS is likely to exert on the society, however, is far-reaching. First, they provide phenomenal benefits such as increased efficiency in industries, improved quality of life through smart healthcare and homes, besides achieving environmental sustainability through optimized energy usage as well as precision agriculture. On the other hand, they express following concerns: displacement of jobs due to automation, digital divide between "privileged" and "not privileged", those who have access to advanced technology and others who don't, and potential increased surveillance and loss of privacy.

Urban Living and Smart Cities: Indeed, CPS optimizes the services of traffic management, public safety, as well as utilities supporting the citizens' well-being in order to enhance the urban living spaces to become smart cities. While this promises a more livable environment, it also requires careful consideration with data privacy and equitable spreading of those technology benefits not to widen already present social inequalities.

Workforce Transformation: In the industrial sector, automation through CPS may displace some of the jobs which will require re-skilling and upskilling of the workforce. This transition must be thoughtfully managed so that the high-tech jobs of the future are available to these workers enabling increased productivity which accompanies this digital transformation to benefit everyone broadly.

Ethical Considerations

CPS has various ethical issues, from data privacy, security to the fact that they ensure no harm in their systems. Significantly, there are concerns seeing that CPS are associated with valuable infrastructure and personal data:

Data Privacy and Security: With CPS representing a bulk of data collected and processed, the privacy and security aspects related to the stored information become an imperative part. Thus, strong cybersecurity practices and ethical ways of handling data required to be in effect to procure them from unauthorized accession and possible misuse of personal and sensitive data.

Autonomy and Accountability: Being that CPS will make autonomous decisions, concerns are bound to arise in respect to accountability, especially where such decisions lead to unintended

consequences. There is a need that there are clear guidelines as well as the definition of frameworks under which responsibility may be traced if in case of any eventualities arising from such initiatives to build trust for these technologies.

1.2 Objectives of the thesis

This thesis is intended as an exploration and application of Model Predictive Control (MPC) for cyber-physical systems (CPS). As a significant contribution, this thesis explores the use of MPC in three most-valued areas: Industry 4.0, power systems, and smart cities. Through the incorporation of MPC in these spheres, the study aims at improving operational efficiency, optimality of resource allocation, and the general responsiveness and adaptability of such systems to ambient variability. This work seeks to demonstrate future implications in using MPC to transform the management and control of these sophisticated cyber-physical systems within their crucial sectors through theoretical development as well as practical implementation of case studies.

It is through the management of CPS that **smart city** urban environments are turned into predominantly clever, intelligent ecosystems where almost everything is interconnected [86]. CPS utilize the Internet of Things, using networks of sensors, IoT devices, and advanced city communication infrastructures to collect real-time data. Real-time data is also adopted in the optimization of signal timings to reduce congestion, particularly in traffic management systems, and ensure efficient trash collection through sensor-monitored bins, as applied in smart waste management systems. This innovations in the town will not only manage all city services into a single system but also improve their efficiency as well as quality of life reduction in pollution, streamlined public transportation, and ensuring public safety with example intelligent surveillance systems. However, deploying or implementing such systems give birth to highly critical considerations related to data privacy, security, and digital divide, wherein a balanced approach of the utilization and governance of technology is indispensable.

Moving on ahead towards the **Industry 4.0**, CPS is setting forth as a prime revolutionizer in the production and manufacturing by far [68]. By integrating sensors, data analytics and automation, CPS allow the implementation of smart factories whereby machinery calling to one another and interacting with each other exchange information with lesser human intervention. This connectivity allows monitoring and controlling of manufacturing processes in real time, leading to better efficiency, less downtime, and improved product quality. Maturity-wise, CPS are one of the key elements on the highest level of the factory automation pyramid. The predictive maintenance functionality can predict equipment failures using CPS before they occur, resulting in a much smaller number of unplanned outages and considerably lessening the maintenance costs. However, the growing move towards highly automated systems serves only to underscore the need for workforce adaptation and reskilling as the traditional roles within manufacturing have transformed towards catering for an industry that is increasingly digitalized.

CPS in this regard orchestrates crucial functions advancing smart grids towards the **power system and energy distribution** development efficiency [47]. They integrate renewable energy sources with demand response functionalities and distributed energy resources addressing electrical grid reliability, sustainability, and resilience. Smart meters and grid sensors facilitate the tactful exchange of information between the consumer and utility in a two-way mechanism for dynamic

pricing, better predictions of demands, and effective integration of renewable energy resources. CPS driven smart grids are a milestone towards making the carbon footprints diminish to considerable levels and making the energies sustainable. However, to the extent that these systems are integrated and digitized, their complexity and security is second to none, thereby making them prime targets for cyber attack that necessity the exigency of robust cybersecurity cover to protect critical infrastructure.

In conclusion, the incorporation of CPS into smart cities, Industry 4.0 and power systems epitomizes what can be referred to as the translative potential of merging the digital with the physical. These systems offer unheard potential for optimization, efficiency and sustainability in different sections. However, the successful deployment of CPS depends on properly dealing with its attendant security, privacy and social adaptation challenges - ensuring that it operates to benefit all the society has a whole.

1.3 Motivation

The burgeoning intrigued in considering Cyber-Physical Systems (CPS) through the focal point of Model Predictive Control (MPC) is driven by an critical have to be upgrade the strength, proficiency, and security of these interconnected frameworks in an time where advanced progressions are inseparably connected to physical operations. The inspiration behind this center is twofold: the inborn capacity of MPC to expect and ideally react to energetic conditions in real-time, and the raising risk scene stamped by modern cyber-attacks that have the potential to compromise or indeed annihilate physical infrastructures.

By leveraging prescient models to figure framework behaviors and potential disturbances, MPC can proactively alter control actions, ensuring the steadiness and unwavering quality of CPS indeed within the confront of unforeseen disturbances or malicious attacks. Besides, the capacity of MPC to consolidate imperatives and oversee multi-variable frameworks makes it especially suited to exploring the complex interdependencies inborn in CPS, where cyber and physical components must be harmonized to protect against and react to cyber dangers viably.

This consider is propelled by the squeezing got to invigorate CPS against cyber-physical attacks and the potential of MPC techniques to revolutionize how these frameworks are controlled and secured. By investigating the crossing point of MPC and CPS, this investigate points to contribute to the improvement of more secure, strong, and proficient systems capable of withstanding the advancing scene of cyber dangers, in this manner guaranteeing the judgment and progression of basic frameworks in our progressively interconnected world.

This thesis is dedicated to exploring the advancement of Model Predictive Control (MPC) methodologies within the realms of Industry 4.0, power systems, and traffic light control in smart cities. As the fourth industrial revolution unfolds, the integration of digital technologies into manufacturing, energy distribution, and urban infrastructure is not just an option but a necessity for sustainable development and enhanced efficiency. The core of this thesis lies in harnessing the predictive and optimization capabilities of MPC to navigate the complexities and dynamic nature of these sectors. By centering on MPC, this investigate points to contribute to the shrewdly robotization of fabricating forms, the solidness and effectiveness of control frameworks, and the optimization of activity stream, in this manner addressing some of the foremost pressing challenges within the

move towards more astute, more associated situations. Through a rigorous exploration of MPC applications in these areas, the proposition of this thesis looks for to supply imaginative arrangements that can adjust to and expect the advancing requests of Industry 4.0, the energy sector, and urban mobility.

1.4 Structure of the thesis

This thesis is is organised as follows. In Chapter 2 is presented the Model Predictive Control (MPC) methodology and its applications in various industries, explaining strength and drawbacks. Chapter 3 presents the application of MPC in power systems, in particular the proposed reconfiguration algorithm is illustrated to increase efficiency and resiliency of smart grids. Chapter 4 shows an integrated MPC controller with the aim of optimising traffic in smart cities by determining optimal signal timings of the traffic lights. Chapter 5 presents a MILP formulation for modelling the optimal task scheduling and control problem in Industry 4.0. The study under consideration regards the effective planning and control (i.e. task scheduling) of the various tasks and resources in an assembly of a space vehicle. Finally, Chapter 6 highlights conclusions and future works.

It is important to underline that Chapters 3, 4 and 5 are a reworking of my papers written during the PhD years, in which further details and insights have been added. The related papers, published or submitted, are the following:

- [37] *Donsante, M.; Tortorelli, A.; Di Giorgio, A.; Liberati, F. "A Model Predictive Control Algorithm for the Reconfiguration of Radially Operated Grids with Islands". Electronics 2023, 12, 4982.*
- [74] *Liberati, F.; Donsante, M; Tortorelli, A. "Single Intersection MPC Traffic Signal Control in Presence of Automated Vehicles". IEEE Transactions on Control Systems Technology. Submitted.*
- [73] *Liberati, F.; Donsante, M; Cirino, C; Tortorelli, A. "Fast Task Scheduling with Model Predictive Control Integrating a Priority-based Heuristic", in IEEE Access. Submitted.*

Chapter 2

Model Predictive Control Methodology

Model Predictive Control (MPC) is an advanced control strategy utilized in different businesses to optimize the control of complex dynamic systems. It is a model-based approach that includes anticipating the long run behavior of a framework, defining an optimization problem, and after that applying the calculated control activities in the prediction horizon. The control inputs are calculated by solving an optimization problem that seek to minimize a cost function while satisfying system constraints. Figure 2.1 shows the MPC-based control loop, where **u** is the optimal control variable determined by solving a constrained optimization problem, the cost function is formulated in such a way that the system output **y** tracks a given reference **w**, and **z** is the disturbance of the system.

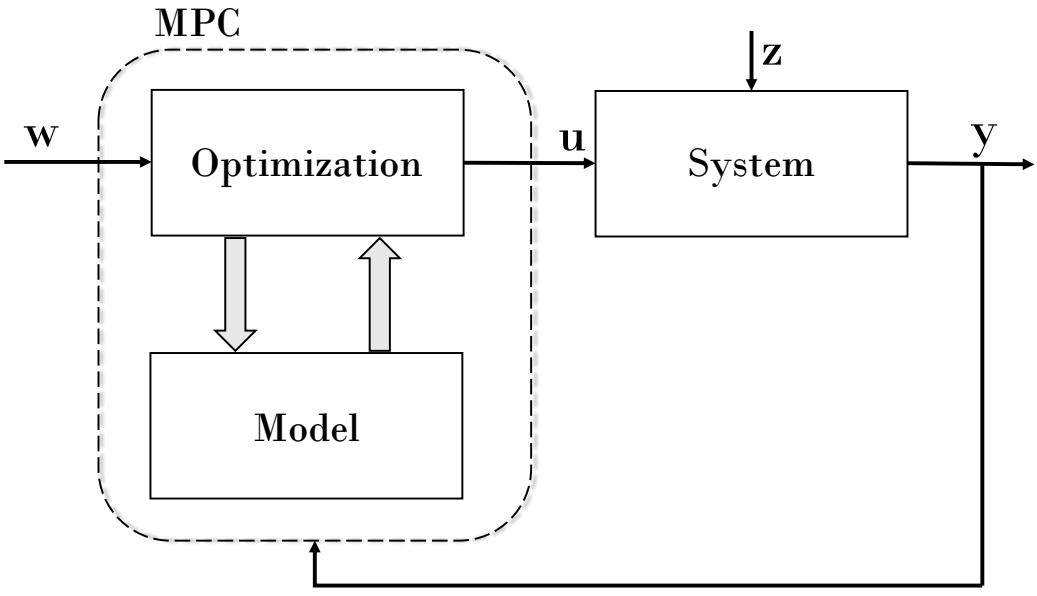


Figure 2.1: Simplified MPC-based control loop, adapted from [108].

MPC relies on a dynamic model that describes how the system evolves over time; the control horizon is a subset of the prediction horizon that decides how distant into long term the control inputs are applied; the cost function is a mathematical expression that quantifies the performance goals of the control problem and it incorporates terms related to desired setpoints, state deviations,

and control efforts.

For the control problem, given the current state of the system and model, expectations are made around the system's future behavior over the prediction horizon. An optimization problem is solved utilizing the predicted trajectories. The objective is to find the control inputs that minimize the cost function and only the first set of the calculated control inputs is applied to the system (the rest are discarded). At that point the system evolves according to the applied input and, and new measurements are obtained. Finally, the process is repeated by moving the control horizon ahead. For this reason, the MPC is also called receding horizon control. The key concepts of MPC are hence Prediction Horizon and Control Horizon. The prediction horizon is the time interval over which future forecasts are made. It's typically partitioned into discrete time steps. The control horizon is a subset of the prediction horizon that decides how distant into the future the calculated control inputs will be applied to the system. Longer prediction horizon capture more future behavior but might introduce uncertainty, whereas shorter prediction horizon might not capture important dynamics. Figure 2.2[1] shows the MPC receding horizon concept.

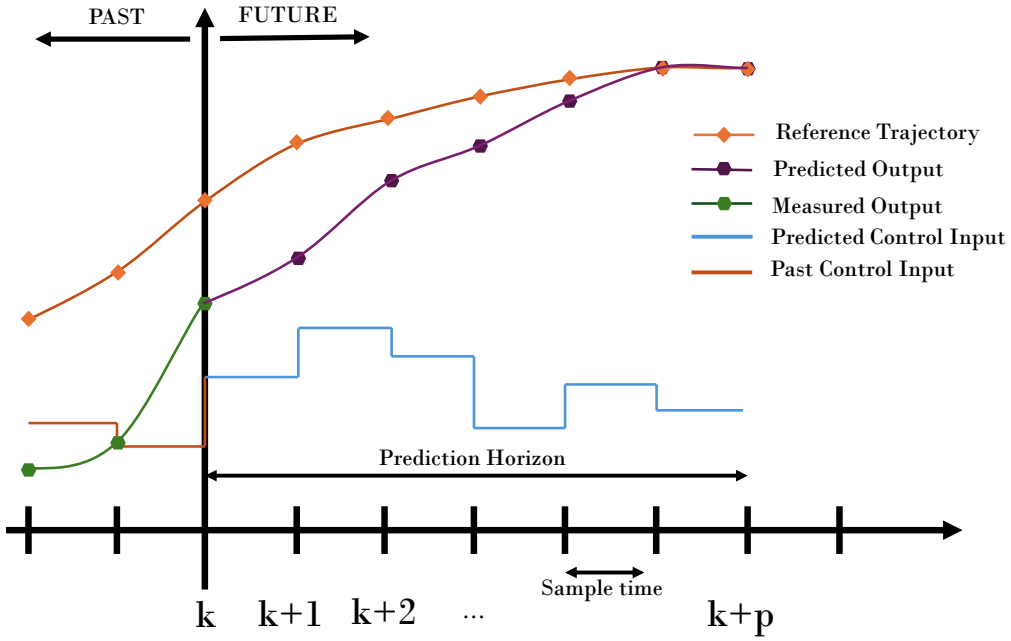


Figure 2.2: The receding horizon concept of Model Predictive Control, adapted from Wikimedia Commons by Martin Behrendt.

In a nutshell what is the Model Predictive Control? With reference to Fig. 2.1, the MPC framework *uses a (simplified) dynamical model of the process to predict its (likely) future evolution and choose a good (instead "the best") control action*[2].

The MPC problem formulation is to find the best control sequence over a future horizon of N

[1] Adapted from `https://commons.wikimedia.org/wiki/File:MPC_scheme_basic.svg`

[2] Model Predictive Control course of prof. Alberto Bemporad. Link: `http://cse.lab.imtlucca.it/~bemporad/mpc_course.html`, Lecture slide "Linear MPC".

steps:

$$\begin{aligned} \min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} ||y_k - r(t)||_2^2 + \rho||u_k - u_r(t)||_2^2 \\ \text{s.t.} \quad & x_{k+1} = f(x_k, u_k) \\ & y_k = g(x_k) \\ & u_{min} \leq u_k \leq u_{max} \\ & y_{min} \leq y_k \leq y_{max} \\ & x_0 = x(t), \end{aligned} \tag{2.1}$$

where x is the state of the system, the variable with the subscript r refer to the reference variables, $r(t)$ is the reference trajectory, and u and y represent the control and the output of the system, respectively, and functions $f(\cdot)$ and $g(\cdot)$ are generic nonlinear functions. This is a numerical optimization problem and for all time step t, it must estimate the current state $x(t)$, optimize with respect to $u_0, \dots, u_{N-1}$ and then only apply the optimal u_0 as input $u(t)$.

The advantages of MPC are that it can handle complex, nonlinear, and time-varying systems; it can handle multi-variable systems; it optimizes control actions over a finite time horizon, driving to better long-term performance. On the other hand, the disadvantages are that it is computational complex, i.e. solving the optimization problem requires powerful algorithms and hardware; the effectiveness of MPC depends on the accuracy of the model used for prediction and the controller has to be implemented in real time, which can be challenging due to the computational requirements.

Despite these challenges, the advantages of MPC, counting its ability to optimize performance while respecting operational constraints, make it a powerful tool in modern control engineering.

Model Predictive Control (MPC) encompasses a variety of techniques tailored to specific system requirements, ranging from Linear MPC for systems well-represented by linear equations, to Nonlinear MPC for handling nonlinear dynamics, and Robust MPC designed to cope with uncertainties in model parameters or operational environment. In the following, several MPC variants are illustrated.

Linear Time-Varying (LTV) MPC adapts to systems whose parameters or dynamics change over time, maintaining a linear relationship between inputs and outputs. This point shows how it is appropriate for systems that incur time-dependent changes. The challenge with LTV MPC lies in accurately predicting the time-varying parameters to ensure optimal control performance.

Nonlinear Model Predictive Control (NMPC) is a control technique meant for handling nonlinear systems that usually exhibit behaviors like saturation, hysteresis, or dynamic instabilities that linear model cannot capture. Although NMPC offers a more applicable model for such systems, it is demanding computationally, considering the fact that real-time online solutions of nonlinear optimization problems can be computationally expensive. Advanced computational techniques and hardware are often required to implement NMPC effectively.

Quadratic Programming (QP) MPC strikes a balance between the best performance and computational complexity, especially for especially in linear or mildly nonlinear systems. The cost function is quadratic and constraints are linear leading to a convex optimization problem, which guarantees a unique global minimum, making the solution process more straightforward and reliable.

A particular variant is the explicit MPC (eMPC) in which the solution to the MPC control

problem formulated as optimization problem is pre-computed offline allowing fast evaluation of the control [18]. In particular conditions, the implementation of the eMPC does not require significant computational resources but the drawback is the exponential complessity growth with respect some key parameters (e.g. system states) and then it is computationally expensive [64].

Hybrid MPC includes systems that switch between various modes of operation or a combination of continuous processes with on/off control actions. This approach can be applied to large complex systems, such as robotics, automotive, and energy systems, where both continuous and discrete decisions are crucial. The critical issue with Hybrid MPC algorithms is the combinatorial complexity in mixed-integer optimization, which may very well be NP-hard in extremely large-scale applications.

The stochastic MPC is, however, very useful for any cases in which the future is uncertain: financial markets, or more generally any application where the system is subject to random disturbances. The approach aims to find a policy that performs well on average, rather than optimizing for a specific, deterministic future. The complexity of Stochastic MPC comes from the need to model the probability distributions of uncertainties and solve optimization problems that incorporate these stochastic elements.

The Learning-based Model Predictive Control is at the forefront of integrating control theory with artificial intelligence. It is particularly suited for environments that are difficult to model accurately due to their complexity or for systems that operate in changing conditions where the ability to learn and adapt can significantly enhance performance. The integration of learning mechanisms within MPC frameworks can lead to controllers that not only adapt in real-time to changes but also improve their performance as they accumulate more data. The challenge lies in ensuring stability and robustness of the control system, as learning algorithms can introduce unpredictability and require rigorous validation.

Each of these variations is an extension of the applicability of the basic framework of MPC to a wider array of systems and conditions, reflecting developments in control theory that mirror the demands of a changing technological world.

Here in the following, some example of MPC application in real (industrial) systems.
Regarding MPC for autonomous driving/driver-assistance systems application, the paper [50] presents a control system for autonomous driving, including a path planner and a linear time-varying model predictive controller to coordinate torque request and steering to achieve safe and comfortable autonomous driving with no collisions. It's tested in various scenarios such as obstacle-free curved road tracking and adaptive cruise control (ACC) coupled with obstacle avoidance (OA). It's used a stochastic prediction models used to account for uncertainty (other vehicles/pedestrians, driver's requests).

In the application of MPC in a gasoline turbocharged engines, the authors in [16] describe the development of a constrained MPC system designed for torque tracking in turbocharged gasoline engines slated for production by General Motors starting from 2018. Collaboratively developed by General Motors and ODYS, the system comprises a set of real-time scheduled linear MPC controllers that are based on engine operating conditions and identified through engine data. These controllers coordinate throttle, wastegate, intake and exhaust cams to match a desired torque profile, optimizing fuel efficiency and considering constraints on variables. Actuator commands are computed by solving optimization problems at each time step, and each MPC uses Kalman filter to to reconstruct the system state from available measurements. Compared to traditional controls,

this MPC system enhances fuel economy and drivability through better actuator coordination while maintaining robustness against noise and variations. The modular MPC software, developed by ODYS, is adaptable to different engine architectures and was integrated and validated by General Motors for deployment in production engine control units (ECUs).

Application of MPC in powertrain with continuously variable transmission (CVT) are studied for example in [15]. This paper describes the design of a supervisory multi variable constrained MPC system for driver requested axle torque tracking with real-time fuel economy optimization scheduled for production by General Motors and ODYS started from 2018.

MPC in Aerospace field could be used for different purposes. For example in unmanned aerial vehicles (UAVs) cooperation, authors in [19] propose a hierarchical MPC approach to stabilization and autonomous navigation of a formation of unmanned aerial vehicles (UAVs), under constraints on motor thrusts, angles and positions, and under collision avoidance constraints. UAVs are stabilized by linear time-invariant (LTI) MPC controller at lower level and commanded by desired set-points generated at the higher level by a linear time-varying (LTV) MPC controller.

In spacecraft application, newly developed real-time embedded MPC guidance and control techniques hold significant promise for the next wave of high-performance reusable European Space Agency (ESA) launch vehicles and guidance navigation and control (GNC) systems. The work done in [91] focuses on these cutting-edge technologies for real-time embedded MPC, which encompass thrust vectored control. These methods show potential for effective utilization during both the ascent phase and powered descent, facilitating precise pinpoint landing capabilities.

The analysis of mobility through computer multi-body simulation of wheeled rovers is pivotal in planetary exploration endeavors. Given the often largely unknown environments in space missions, incorporating stochastic methods into numerical analyses proves advantageous. The work done in [48] showcases outcomes derived from employing nondeterministic techniques for mobility analysis of a planetary rover under conditions of uncertain terrain properties. The uncertainty of the terrain is determined by experimental measurements of soil properties using specialized equipment. Subsequently, this soil uncertainty is propagated through a multi-body model to evaluate the uncertainty of the rover's position when traversing soft soil. Both probabilistic and non-probabilistic approaches are conducted and contrasted in the study.

Another possible application of MPC regards the field of Smart Electricity Grids which consists in the dispatch power in smart distribution grids. The challenges in this case are account for dynamics, network topology, physical constraints, and stochasticity (of renewable energy, demand, electricity prices). In [92], the authors propose a scenario-based stochastic MPC algorithm to solve the real-time market-based optimal power dispatch problem. In fact, prices, demands and intermittent power injections are considered as stochastic processes. The goal is to compute power injections (for the power generators), charge and discharge levels for the storage units and exchanged power with the rest of the grid that minimize operating and trading costs. In this way the generation company that manages the power system can place bids on the real-time energy market (the so-called regulating market) in order to balance its loads and/or to make profit.

Interesting application of MPC are also on the operational control of water networks. In [106] it is proposed a Stochastic model predictive control (SMPC) applied and validated on the water network of the city of Barcelona.

To showcase the potential of MPC, we cannot overlook its applications in the world of finance.

Derivative contracts necessitate replicating the product through a dynamic portfolio comprising simpler, more liquid securities. In [14] the authors propose a solution for a wide range of options in financial engineering to address the challenge of identifying a hedging portfolio which use discrete-time stochastic model predictive control and receding horizon optimization.

In summary, MPC is an advanced control strategy that utilizes a predictive model, optimization, and receding horizon concept to achieve optimal control of complex dynamic systems while considering constraints and performance objectives. Due to its flexibility and to its ability to handle multi-variable control problem, MPC is broadly utilized in different businesses, for example Process industry (chemical, petrochemical, etc.), Automotive, Robotics (trajectory tracking, motion planning), Energy management (smart grids, renewable energy), Aviation (flight control, UAVs), in chemical plants MPC optimizes reactor temperatures and pressures for improve efficiency, etc.

2.1 Mixed Logical Dynamical System

When we talk about controlling physical systems, it should be kept in mind that they are described by physical laws, logical rules and operational constraints. An important work was done in [17], where the authors proposes a framework for modelling and controlling systems with the characteristics mentioned above, denoted as Mixed Logical Dynamical (MLD) systems. These types of systems are characterized by linear dynamic equations subject to linear inequalities, with both real and integer variables. MLD systems include different systems such as linear hybrid systems, finite state machines, specific classes of discrete event systems, constrained linear systems, and nonlinear systems which can be approximated by piecewise linear functions. The concept of MDL is the one on which this thesis is based. Originally, the concept of model of a system is associated with differential or difference equations, which are typically derived from physical laws governing the dynamics of the system under consideration and therefore the the control theory and tools have been developed for such systems, i.e. for systems whose evolution is described by smooth linear or nonlinear state transition functions. In reality however, the systems we want to control are described by logic (i.e. on/off switches, gears or speed selectors). For this reason, the author proposes a framework for modeling and controlling models of systems described by interacting physical laws, logical rules and operating constraints. Propositional logic is transformed into linear inequalities involving integer and continuous variables which naturally leads to the definition of mixed logical dynamical (MLD) systems, described by linear dynamic equations subject to linear mixed-integer inequalities (inequalities with both continuous and binary variables). The MLD systems allow to generalize many set of models among which finite state machines, linear hybrid systems, some classes of discrete event systems, nonlinear systems (approximated by piecewise linear functions) and constrained linear systems.

A propositional logic problem can be solved by means of a linear integer program, by translating the original compound statements into linear inequalities involving logical variables δ_i. Let's assume that x_i represent statements, e.g. $x_i \geq 0$ and it is commonly referred to as *literal*, and has a *truth value* of either "T" (true) or "F" (false). Using the classical notation for operators for logical variables $\neg$ (NOT), $\vee$ (OR), $\wedge$ (AND), $\oplus$ (XOR), and $\rightarrow$ (implies), $\leftrightarrow$ (if and only if), the following equations

are equivalent:

$$x_1 \vee x_2 \text{ is equivalent to } \delta_1 + \delta_2 \geq 1, \tag{2.2}$$
$$x_1 \wedge x_2 \text{ is equivalent to } \delta_1 = 1, \delta_2 = 1, \tag{2.3}$$
$$\neg x_1 \text{ is equivalent to } \delta_1 = 0, \tag{2.4}$$
$$x_1 \rightarrow x_2 \text{ is equivalent to } \delta_1 - \delta_2 \leq 0 \tag{2.5}$$
$$x_1 \leftrightarrow x_2 \text{ is equivalent to } \delta_1 - \delta_2 = 0 \tag{2.6}$$
$$x_1 \oplus x_2 \text{ is equivalent to } \delta_1 + \delta_2 = 1, \tag{2.7}$$

and

$$x_1 \rightarrow x_2 \text{ is equivalent to } \neg x_1 \vee x_2, \tag{2.8}$$
$$x_1 \rightarrow x_2 \text{ is equivalent to } \neg x_2 \rightarrow \neg x_1, \tag{2.9}$$
$$x_1 \leftrightarrow x_2 \text{ is equivalent to } (x_1 \rightarrow x_2) \wedge (x_2 \rightarrow x_1). \tag{2.10}$$

This technique is used to model the logical parts of processes. Also products of logical variables, and of continuous and logical variables, can be transformed in terms of linear inequalities as follows

$$\delta_3 = \delta_1 \delta_2 \text{ is equivalent to } \begin{cases} -\delta_1 + \delta_3 \leq 0 \\ -\delta_2 + \delta_3 \leq 0 \\ -\delta_1 + \delta_2 - \delta_3 \leq 1. \end{cases} \tag{2.11}$$

In this way it is possible to transform propositional logic into linear mixed-integer inequalities and then modeling and controlling systems described by both dynamics and logic, and subject to operating constraints, denoted as mixed logical dynamical (MLD) systems. Additional details and techniques to transform logical facts involving continuous variables into linear inequalities can be found in [17].

2.2 MILP application in Cyber-Physical systems

A particularly powerful variant is the Mixed-Integer Linear Programming (MILP) formulation of MPC, which stands out for its ability to incorporate both logical and physical constraints into the control problem [17]. This capability makes MILP-based MPC especially suited for cyber-physical systems, where the interplay between digital decisions (logical) and physical processes must be carefully managed. The strength of MILP lies in its versatility; it can handle on/off decisions, mode selections, and other discrete choices within a predictive control framework, while still ensuring that physical constraints, such as energy limits or mechanical capacities, are respected. This dual capability enables the design of highly efficient and flexible control strategies that can optimize performance in complex, hybrid systems where traditional continuous control approaches might fall short.

The application of Mixed-Integer Linear Programming (MILP) in cyber-physical systems (CPS) is vast, covering areas such as energy systems, manufacturing, and smart grids, where the integration of computational and physical processes is critical. Here are some examples of MILP applications

in CPS:

- Smart Grids and Energy Management: MILP is used in smart grids for optimal scheduling and dispatch of distributed energy resources (DERs), including renewable energy sources, storage devices, and demand response. This involves solving problems that balance energy supply and demand, considering both physical constraints (like storage capacities and power flow limits) and logical constraints (such as on/off states of DERs) [34].
- Industrial Automation and Manufacturing: In manufacturing systems, MILP models are used for production planning and scheduling, where the objective is to optimize the production process by determining the sequence of operations, allocation of resources, and timing of tasks, while adhering to physical constraints related to machinery and logical constraints such as task ordering [85].
- Autonomous Vehicles and Robotics: MILP formulations are used in the path planning and collision avoidance systems of autonomous vehicles and robots, integrating logical decisions (e.g., choosing paths at intersections) with physical constraints (like vehicle dynamics and obstacle avoidance) [99].
- Water Resource Management: In water networks, MILP is applied to manage the distribution of water resources efficiently, addressing challenges such as the optimal operation of pumps and valves (logical decisions) and maintaining desired pressures and flow rates within the network (physical constraints) [24].
- Traffic Management and Control: MILP is used in optimizing traffic signal timings and coordinating multiple intersections in urban traffic networks. This application involves logical decisions, such as the sequence of traffic lights, and physical constraints, like vehicle flow and pedestrian safety, to improve traffic flow and reduce congestion [75].
- Healthcare Operations: In healthcare systems, MILP can optimize the scheduling of surgeries, allocation of resources (e.g., operating rooms, medical staff), and patient flow management, taking into account both the logical aspects of scheduling and the physical constraints related to healthcare delivery and patient care [60].
- Supply Chain and Logistics: MILP is applied in supply chain management for optimizing logistics operations, including warehouse location selection, inventory management, and transportation routing, where decisions must balance cost, efficiency, and service level constraints against physical limitations like capacity and delivery times [81].
- Energy-Efficient Building Design: In the context of smart buildings, MILP is utilized to optimize the design and operation of heating, ventilation, and air conditioning (HVAC) systems, considering energy consumption, occupant comfort, and varying environmental conditions. This involves a combination of discrete decisions (e.g., HVAC on/off status) and continuous physical constraints (e.g., temperature ranges) [77].

These examples illustrate the versatility and power of MILP in solving complex optimization problems in cyber-physical systems by seamlessly integrating logical and physical constraints to ensure

efficient and effective operations across various domains. Moreover, MILP formulations are extensively applied across various domains such as power systems, Industry 4.0, and smart city traffic optimization, due to their capability to handle complex decision-making processes that involve both discrete and continuous variables and Mixed-Integer Linear Programming (MILP) plays a pivotal role in optimizing the operational decisions and infrastructure development to enhance efficiency, reliability, and sustainability. Here are examples from each of these domains:

- *Power Systems.* In power systems, MILP is particularly useful for the unit commitment problem, where the objective is to determine the on/off status of power generation units and their respective output levels to meet demand at minimum cost, subject to operational and physical constraints. This problem is crucial for the efficient operation of both traditional and renewable energy-based power systems. An example could be the optimal dispatch of distributed generation in microgrids [84], considering renewable energy sources, storage units, and demand response programs. The goal is to minimize operational costs while ensuring reliability and meeting physical constraints like generation limits and network constraints.

- *Optimal Distribution Feeder Reconfiguration.* Distribution feeder reconfiguration involves altering the open/closed status of switches within a distribution network to reroute power, minimize losses, balance loads across feeders, and improve reliability. The MILP formulation for this problem integrates logical decisions (switch open/close) with physical constraints (like voltage limits, branch current limits, and power flow equations) to find the optimal configuration that minimizes power losses and ensures a reliable supply to all customers. The objective is to minimize the total power losses in the distribution network while ensuring adequate service to all demand points and adhering to operational constraints such as radial network topology, voltage levels within permissible limits, and branch current not exceeding maximum capacities. Some of constraints are: radial structure maintenance to ensure there are no loops and every customer is served by exactly one path from the substation; voltage drop constraints to maintain the voltage at each node within regulatory limits, ensuring power quality and equipment safety; current carrying capacity constraints to prevent any line from carrying more than its rated maximum current, avoiding overheating and potential damage; demand satisfaction constraints to ensure the configuration supplies adequate power to meet the demand at each node. The types of variables are Binary variables representing the open/closed status of each switch in the network and continuous variables for the power flow, voltage at each node, and power loss on each line.

  An example application: implementing an MILP-based feeder reconfiguration strategy can lead to significant operational benefits in a smart grid environment [13]. For instance, during peak load conditions or in the event of a line fault, the network can be reconfigured in real-time to redistribute loads, minimize losses, and maintain service without interruption. Additionally, integrating renewable energy sources and energy storage systems into the distribution network adds complexity that MILP can handle effectively by optimally managing these resources to support the main grid.

  This example underscores the capability of MILP to address the intricate challenges of electricity distribution, providing a systematic approach to optimize network configurations for

improved operational efficiency and resilience in the face of dynamic demand and supply conditions.

- *Industry 4.0.* In the context of Industry 4.0, MILP is used to optimize manufacturing processes, supply chains, and production planning, integrating smart technologies and data analytics for improved efficiency and flexibility. This involves scheduling of production lines, maintenance, and logistics operations to minimize costs and meet demand. An example is the optimization of a smart manufacturing system for dynamic job-shop scheduling [53], where the goal is to allocate tasks to machines in real-time, considering machine capabilities, maintenance schedules, and job priorities, to minimize production time and costs.

- *Optimization of Traffic in Smart Cities.* In smart cities, MILP is employed to enhance traffic flow and reduce congestion through intelligent traffic signal control, route planning, and infrastructure utilization. This involves optimizing traffic light timings and coordinating traffic signals across multiple intersections, considering the flow of vehicles, pedestrians, and cyclists. Example: dynamic traffic signal control in an urban network where the objective is to minimize overall travel time and congestion by adjusting signal timings based on real-time traffic data, taking into account constraints like minimum green time and pedestrian crossing times [134].

- *Optimization in traffic light scheduling* within smart cities is a critical aspect of urban traffic management, aiming to improve traffic flow, reduce congestion, and enhance safety for all road users. The application of Mixed-Integer Linear Programming (MILP) in this context allows for the systematic handling of the complex interplay between traffic lights at various intersections and the dynamic nature of traffic flow. An example is the adaptive Traffic Light Scheduling. The goal of adaptive traffic light scheduling is to adjust the signal timings dynamically in response to real-time traffic conditions. This involves determining the optimal duration of green, yellow, and red phases for each traffic light at an intersection and coordinating these phases across multiple intersections to ensure smooth traffic flow. The objective is to minimize the overall travel time and delay for vehicles and pedestrians, reduce the number of stops, and improve the efficiency of traffic flow throughout the urban network. Some of constraints could be: cycle time constraints to ensure the sum of green, yellow, and red times for each signal phase equals the total cycle length.; green time constraints to provide a minimum green time for each phase to ensure safety and efficiency; inter-green time (yellow plus all-red time) constraints to account for the clearance interval between successive phases, ensuring the intersection is clear of vehicles before a new phase starts; capacity constraints to prevent oversaturation of intersections and ensure that the queue length does not exceed available storage space on the approaching lanes; coordination constraints for adjacent intersections, especially along arterial roads, to create green waves that minimize stops and delays for platoons of vehicles.

  Example Application: In a smart city scenario, traffic light scheduling can be integrated with traffic detection systems (such as cameras and sensors) that provide real-time data on traffic density, flow, and speed. This data feeds into the MILP model, which then adjusts the traffic light timings to optimize flow and reduce congestion. For example, during peak hours, green

times can be extended on major arterials to accommodate higher volumes, while off-peak periods might see more responsive adjustments to individual vehicle detections, optimizing for reduced wait times [113].

This approach highlights the potential of MILP in enhancing urban mobility and sustainability by providing a flexible and responsive traffic light scheduling system that adapts to changing traffic conditions, ultimately contributing to the overarching goals of smart city initiatives.

These examples highlight the adaptability of MILP in addressing complex, multi-faceted optimization problems in power systems, Industry 4.0, and smart city infrastructure, showcasing its ability to integrate a wide range of operational constraints and objectives for enhanced decision-making and system performance.

Chapter 3

A Reconfiguration Algorithm for Increasing Efficiency and Resiliency of Smart Grids

This chapter discusses a reconfiguration algorithm, based on centralized Model Predictive Control (MPC), of transmission and distribution grids topology to increase efficiency and resiliency in case of fault and adverse events/(cyber-)attacks.

The following sections regard the research activities and results in Power Systems application in which the candidate produced the paper:

[37] *Donsante, M.; Tortorelli, A.; Di Giorgio, A.; Liberati, F. A Model Predictive Control Algorithm for the Reconfiguration of Radially Operated Grids with Islands. Electronics 2023, 12, 4982.*

3.1 Introduction

Transmission and distribution grids are experiencing a transformation marked by the integration of distributed energy resources, including electric energy storage systems (ESS), renewable energy plants, and plug-in electric vehicles. This departure from the traditional passive and centralized energy systems of the past introduces several challenges due to increased operational complexity. Notably, factors such as the variability in power production from renewable plants and the intermittent demand from plug-in electric vehicles necessitate continuous grid control and optimization.

Amid various available control mechanisms, grid reconfiguration stands out as a valuable tool, providing flexibility to balance grid load, reduce losses, and swiftly restore service following grid disruptions, as highlighted in [11]. Unlike the historical practice of performing reconfiguration offline in passive grids, the current landscape allows for dynamic reconfiguration throughout the day, adapting to evolving grid boundary conditions.

Here is present a reconfiguration algorithm based on Model Predictive Control (MPC), designed to dynamically adjust the grid and minimize losses during adverse events such as faults. Building upon the work [72], this iteration eliminates the constraint requiring constant grid connection. Furthermore, the algorithm now facilitates the formation of grid islands, anticipating a future scenario where microgrid islands can dynamically connect and disconnect from the main grid.

Network reconfiguration algorithms play a pivotal role in improving the efficiency and resilience of contemporary electric grids. The alteration of grid topology enables the reduction of power losses, enhancement of voltage profiles and stability, load balancing, and mitigation of the impact of faults and attacks [83]. Various approaches to network reconfiguration are found in the literature, categorized into classic optimization-based methods, (meta)heuristics (including evolutionary algorithms), and machine learning-based solutions.

Classic optimization-based algorithms offer advantages such as finding optimal solutions, explicit consideration of constraints, and optimization based on predefined criteria. However, these benefits often come at the expense of high computational costs. Conversely, (meta)heuristics are characterized by lower computational costs but do not guarantee optimality in the reconfigured topology.

Machine Learning (ML), particularly Reinforcement Learning (RL), is a promising avenue for large-scale network reconfiguration due to its ability to infer near-optimal policies. For instance, in [66], a Deep Reinforcement Learning (DRL) algorithm addresses dynamic distribution network reconfiguration (DNR), focusing on minimizing active energy loss and manipulating switching devices. While these ML techniques eliminate the need for exact models of the distribution network, they may require significant computational effort during the training phase.

Considering the challenges posed by DRL algorithms, this work proposes a solution based on Model Predictive Control (MPC), an optimization-based technique retaining the advantages of optimal control approaches. The proposed MPC approach ensures low computational costs, making it suitable for real-time applications. Despite its proven effectiveness in various industrial sectors, MPC has not been extensively applied in network reconfiguration [11].

In [96], a Stochastic MPC is employed to minimize operating costs and increase robustness in a distribution network with multiple feeders, accounting for the variability of Renewable Energy Sources (RESs) and prediction errors. Similar methodologies are applied in [43] and [97], where Stochastic MPC is used to schedule operations, reconfigure switches, and manage Plug-in Electric Vehicles (PEVs) charging.

In [132], a centralized MPC optimizes power flow in the distribution network while ensuring thermal comfort in buildings. An intriguing distributed MPC implementation is explored in [123] for networked cyber-physical systems, offering insights for future developments. The feasibility of control actions in large networked systems is addressed in [55] through a distributed MPC approach.

This study makes a significant contribution by exploring grid reconfiguration while considering the dynamic formation of islands within the grid. Building upon the formulation in the prior work [72], the authors extend it by introducing new radiality constraints that facilitate the formation of islands. Additionally, the integration of power flow equations follows the conic programming approach from [57] and [58]. Modifications are made by introducing simple constraints (3.14) and (3.15), emphasising symmetric and anti-symmetric properties of two variables in the power flow equations. These additional constraints enhance the solver's ability to identify the correct solution.

Concerning radiality constraints, various approaches have been proposed in the literature. In [46], the requirement for radiality involves ensuring that the sum of switch states along each loop in the network is one, designating an open switch. This forces the network into a radial configuration. Another method, presented in [98] for a single substation, computes all possible paths from each

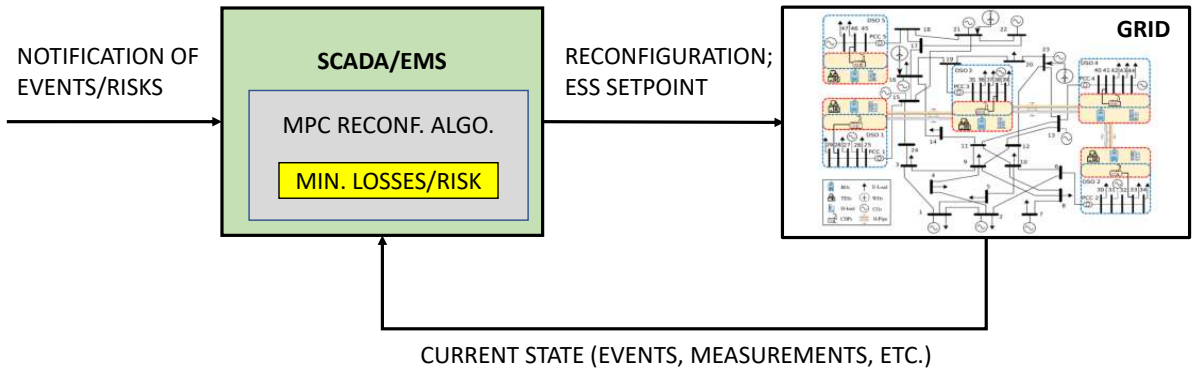


Figure 3.1: Reference scenario.

node to the substation. Radiality is enforced by restricting each node to have only one active path, with the added condition that if a path is active, all paths within it must also be active.

In [101], Romero-Ramos et al. propose a simple equation constraining the sum of link statuses to be equal to the number of non-substation buses in the network. However, this approach fails to ensure radiality in the presence of non-injection buses, leading to potential disconnections and loop formation. To address this, constraints are introduced to guarantee the existence of at least one active path between every zero-injection bus and a non-zero-injection bus. A similar strategy is employed in [43], where additional constraints are added to enforce each node having one parent node.

An alternative method for verifying the radiality of a strongly connected grid is presented in [125], based on calculating the determinant of the connection matrix. This method is utilized in [97]. Notably, this work extends radiality constraints to also function in islanded cases, allowing the formation of radial islands in the grid. Another work considering dynamic island formation is [8], which introduces different and more intricate equations compared to those presented in this work.

3.2 Reference Scenario and Proposed MPC Approach

The focus of the reference scenario is on controlling an electricity grid with a dynamic topology, aiming to enhance network performance (e.g., minimize losses) and address adverse events like faults.

The illustration of the reference scenario is provided in Fig. 3.1. The grid reconfiguration is managed using a centralized Model Predictive Control (MPC) algorithm. MPC is widely embraced across industries, primarily due to its capability to handle constrained control and optimize key performance indicators by designing the objective function. For an in-depth discussion of MPC control techniques, readers are directed to authoritative references such as [65]. A concise depiction of the MPC control logic is presented in Algorithm 1 and elaborated upon below.

The control operates in discrete time, with T representing the sampling time. At the onset of each time step k, the MPC algorithm gathers information about the current state of the grid, encompassing the energy level of storage units and the existing configuration. Subsequently, an optimal reconfiguration optimization problem (referred to as "the MPC iteration") is formulated and solved. The objective function captures the goal of minimizing energy losses, while constraints ensure the admissibility of the reconfiguration actions and the adherence of variables (e.g., voltage

magnitudes, angles) to safety limits. The formulation of this optimization problem, detailing the objective function and constraints, is presented in the next section. At each time step, the optimization problem solution provides optimal commands for the grid, encompassing switch statuses and power injection/consumption for storage units.

The nomenclature used in the work is listed in Tab. 3.1.

Table 3.1: Nomenclature.

Symbol	Explanation
a_{ij}	Status of the switch at line (i, j) ($a_{ij} = 1$ if line is connected, zero otherwise).
b_{ij}	Series susceptance in the π-model of line ij.
α, β, δ	Weights in the objective function.
θ_i	Voltage phase angle at bus i.
θ_{ij}	Difference between the phase angle at bus j and the phase angle at bus i.
$\mathcal{E}$	Set of links in the electricity grid.
F_N	Objective function.
$\mathcal{G}$	Graph modelling the electricity grid.
g_{ij}	Series conductance in the π-model of line ij.
h	Time index.
i, j	Bus indices.
k	Current time.
N	MPC control horizon.
$\mathcal{N}_i$	Set of buses directly connected to bus i.
P_i, Q_i	Active and reactive power injection into the grid at bus i.
P_{ij}, Q_{ij}	Active and reactive power flowing from bus i to bus j.
P_i^{SB}	Power generated by the generator at substation SB.
P_i^{load}	Power load demand.
C_i^{ESS}	ESS capacity [kWh] at bus i.
P^{ESS}, Q^{ESS}	Active/reactive power of the ESS at bus i.
R_{ij}	Auxiliary variable.
$\mathcal{S}$	Subset of $\mathcal{V}$.
SOC_i	SOC of the ESS at bus i.
SOC_i^{ref}	Reference SOC level of the ESS at bus i.
T_{ij}	Auxiliary variable.
T	MPC sampling time.
u_i	Auxiliary variable.
u_i^{ij}	Auxiliary variable.
$\mathcal{V}$	Set of buses in the electricity grid.
V_i	Voltage level at bus i.

Algorithm 1 MPC charging sessions control algorithm.

1: **for** $i = k, k+1, ...$ **do**
2: Acquire the current status of the grid (i.e., the current topology and the energy level of the storage devices).
3: Build and solve an optimization control problem aimed at minimizing network losses and the overall network risk level (the MPC iteration is presented in Section 3.3).
4: Actuate the new computed topology (if different from the one at the previous time), and send the power setpoint to the ESS.
5: **end for**

3.3 Proposed MPC Reconfiguration Algorithm

In the subsequent discussion, we provide a detailed mathematical formulation of the objective function and constraints of the optimization problem solved by the MPC controller at each time instant k (as outlined in step 3 of Algorithm 1) over the time interval $[k, k+N-1]$, commonly known as the control horizon. The standard power system notation is utilized here:

- P_i denotes the power injection (positive) or withdrawal (negative) at bus i,
- P_{ij} and Q_{ij} represent the active and reactive line power flows on the line connecting buses i and j,
- V_i is the voltage at bus i,
- θ_i signifies the voltage angle at bus i,
- θ_{ij} expresses the voltage angle difference between buses i and j.

Additionally, power injections and withdrawals at each bus are denoted as P_i^{SB}, P_i^{DG}, P_i^{ESS}, and P_i^{load}. The objective function and constraints of the optimization problem are presented in the subsequent sections, outlining the key parameters and relationships involved in the formulation.

3.4 Objective Function

The objective function is constructed to achieve three main goals: minimize system power losses, minimize switching actions, and maintain the state of charge of Energy Storage Systems (ESSs) close to a reference value to ensure they always have an operational margin. The index $h \in [k, k+N-1]$ designates any specific time instant within the control horizon.

The objective function is expressed as follows:

$$\begin{aligned} F_N(k, \mathbf{x}_k, \mathbf{u}_k) = \sum_{h=k}^{k+N-1} \Bigg\{ & \alpha(h) \sum_{i \in \mathcal{V}} P_i(h) \\ & + \delta(h) \sum_{i \in \mathcal{V}^{ESS}} [SOC_i(h) - SOC_i^{ref}]^2 \\ & + \beta(h) \sum_{i,j \in \mathcal{V}, j>i} \big[a_{ij}(h)(1 - a_{ij}(h-1)) + a_{ij}(h-1)(1 - a_{ij}(h)) \big] \Bigg\}. \end{aligned} \tag{3.1}$$

The subscript N in $F_N(k, \mathbf{x}_k, \mathbf{u}_k)$ signifies that the optimal control problem is defined over N prediction time intervals. Here, $\mathbf{x}_k$ represents the state of the controlled variable at instant k, and $\mathbf{u}_k$ is the set of control variables, encompassing the ESS power injection and the switching actions a_{ij}.

The parameters α, δ, and β are weights assigned to different components of the objective function. The first term is the summation of power injected/withdrawn at every bus in the grid, representing the total power loss in the network that needs to be minimized. The second term acts as a tracking component, aiming to maintain the state of charge of the ESS close to a reference value on average. This is implemented to prevent situations where ESSs are either fully charged or fully depleted, as these states are undesirable. The third term is introduced to regulate the frequency of feeder connections/disconnections, preventing excessive reconfigurations.

3.5 Power Flow Constraints

The following constraints are added to the above objective function for $h \in [k, k+N-1]$. Starting from the well known nonlinear power flow equations (e.g., [57]), the active and reactive power flows from node i to node j are:

$$P_{ij}(h) = a_{ij}(h)[G_{ij}V_i^2(h) - G_{ij}V_i(h)V_j(h)\cos(\theta_{ij}(h)) - B_{ij}V_i(h)V_j(h)\sin(\theta_{ij}(h))] \tag{3.2}$$

$$Q_{ij}(h) = a_{ij}(h)[-B_{ij}V_i^2(h) + B_{ij}V_i(h)V_j(h)\cos(\theta_{ij}(h)) - G_{ij}V_i(h)V_j(h)\sin(\theta_{ij}(h))] \tag{3.3}$$

where $\theta_{ij}(h) = \theta_i(h) - \theta_j(h)$. Elements G_{ij} and B_{ij} are the line series conductance and susceptance, i.e., the real and the imaginary parts of the (i, j) line series admittance Y_{ij}. Lines' shunt elements are neglected (a reasonable assumption in distribution networks [67]). Equations (3.2) and (3.3) are formulated for a line (i, j) when $a_{ij} = 1$. To achieve precise linearization of these equations, supplementary auxiliary variables and constraints are introduced. Specifically, variables $u_i(h)$ for $i \in \mathcal{V}$, and $R_{ij}(h)$ and $T_{ij}(h)$ for each line (i, j) are introduced as outlined in [58]. These additional variables are defined as follows

$$\begin{aligned} u_i(h) &= V_i(h)^2/\sqrt{2} \\ R_{ij}(h) &= V_i(h)V_j(h)\cos\theta_{ij} \\ T_{ij}(h) &= V_i(h)V_j(h)\sin\theta_{ij}. \end{aligned} \tag{3.4}$$

As previously mentioned, the variables R_{ij} and T_{ij} are associated with each line (i, j), while u_i and u_j pertain to the network nodes i and j. For the variable R_{ij}, it is imperative to enforce non-negativity, given its intended representation as the product of voltages times the cosine of the angle θ_{ij} (as evident in (3.4)). The non-negativity constraint is expressed as follows:

$$R_{ij} \geq 0. \tag{3.5}$$

Following [58], to model the configuration variables within the power flow equations it is necessary to introduce two additional control variables u_i^{ij} and u_j^{ij} for each distinct line (i, j), and the following

constraints:

$$0 \leq u_i^{ij}(h) \leq \frac{V_{i\,\max}^2}{\sqrt{2}} a_{ij}(h) \tag{3.6}$$

$$0 \leq u_j^{ij}(h) \leq \frac{V_{j\,\max}^2}{\sqrt{2}} a_{ij}(h) \tag{3.7}$$

$$0 \leq u_i(h) - u_i^{ij}(h) \leq \frac{V_{i\,\max}^2}{\sqrt{2}} (1 - a_{ij}(h)) \tag{3.8}$$

$$0 \leq u_j(h) - u_j^{ij}(h) \leq \frac{V_{j\,\max}^2}{\sqrt{2}} (1 - a_{ij}(h)). \tag{3.9}$$

The variables $u_i^{ij}(h)$ and $u_j^{ij}(h)$ exhibit specific behaviors based on the connection status of the line (i, j). When the line is disconnected (i.e., $a_{ij} = 0$, as per equations (3.6) and (3.7)), these variables are set to zero. Conversely, when the line is connected (i.e., $a_{ij} = 1$, as per equations (3.8) and (3.9)), they are set respectively to $u_i(h)$ and $u_j(h)$. The relationship among these variables is expressed by the following equations (refer to equation (33) in [58]):

$$2u_i^{ij}(h)u_j^{ij}(h) \geq R_{ij}(h)^2 + T_{ij}(h)^2. \tag{3.10}$$

After introducing the new variables and performing the necessary manipulations (for detailed steps, please refer to [58]), the power flow equations (3.2) and (3.3) take on a linearized form in terms of these newly introduced variables:

$$P_{ij}(h) = \sqrt{2} G_{ij}(h) u_i^{ij}(h) - G_{ij} R_{ij}(h) - B_{ij} T_{ij}(h), \tag{3.11}$$

$$Q_{ij}(h) = -\sqrt{2} B_{ij}(h) u_i^{ij}(h) + B_{ij} R_{ij}(h) - G_{ij} T_{ij}(h). \tag{3.12}$$

Note that from the above equations when a line is disconnected it is $P_{ij}(h) = 0$ and $Q_{ij}(h) = 0$, as expected.
For each node, in terms of the new variables, the voltage limits are:

$$\frac{V_{i\,\min}^2}{\sqrt{2}} \leq u_i(h) \leq \frac{V_{i\,\max}^2}{\sqrt{2}}. \tag{3.13}$$

The following relations are introduced, derived from (3.4), and are crucial to ensure the consistency of the transformation:

$$R_{ij}(h) = R_{ji}(h) \tag{3.14}$$

$$T_{ij}(h) = -T_{ji}(h). \tag{3.15}$$

For each substations we set [57]

$$u_i = V_i/\sqrt{2}, \quad u_i \geq 0 \quad i = 1, 2, 3, \tag{3.16}$$

where V_i is set at 1 pu. The ESS are modeled by the following equations:

$$P_i^{ESS,\min} \leq P_i^{ESS}(h) \leq P_i^{ESS,\max} \tag{3.17}$$

$$Q_i^{ESS,\min} \leq Q_i^{ESS}(h) \leq Q_i^{ESS,\max} \tag{3.18}$$

$$SOC_i^{\min} \leq SOC_i(h) \leq SOC_i^{\max} \quad \forall i \in \mathcal{V}^{ESS} \tag{3.19}$$

$$SOC_i(h+1) = SOC_i(h) - T \cdot P_i^{ESS}(h) \tag{3.20}$$

where P_i^{ESS} and Q_i^{ESS} denote the active and reactive power of the ESS connected at bus i. SOC_i represents the measured energy stored in the ESS, T is the discretization time step, and the initial conditions at each instant k are given by:

$$SOC_i(k) = SOC_i^k \quad \forall i \tag{3.21}$$

where SOC_i^k is the measured ESS energy stored at instant k. The power balance equations of all the nodes $i \in \mathcal{V}$ are:

$$\begin{aligned} P_i(h) &= P_i^{SB}(h) + P_i^{ESS}(h) - P_i^{load}(h) + P_i^{DG}(h) \\ &= \sum_{j \in \mathcal{N}_i} P_{ij}(h) + G_{ii} V_i(h)^2, \end{aligned} \tag{3.22}$$

$$\begin{aligned} Q_i(h) &= Q_i^{SB}(h) + Q_i^{ESS}(h) - Q_i^{load}(h) + Q_i^{DG}(h) \\ &= \sum_{j \in \mathcal{N}_i} Q_{ij}(h) - B_{ii} V_i(h)^2, \end{aligned} \tag{3.23}$$

where $P_i^{SB}(h)$ represents the power injected by the HV/MV substations, and $P_i^{DG}(h)$ is the power injected by all distributed generations at the respective node. The set $\mathcal{N}_i$ denotes the neighbour nodes of node i. To ensure non-injection buses are not isolated, the following constraint is introduced:

$$\sum_{i \in \mathcal{V}} a_{ij}(h) \geq 1 \quad \forall j, \tag{3.24}$$

which means that each node has to be connected at least with another node. The control variables are $a_{ij}(h)$ and $P_i^{ESS}(h)$.

3.6 Radiality constraints

The radiality constraints have been revised compared to [72]. These modifications are essential to guide the reconfiguration algorithm effectively, ensuring that the proposed configurations are free of loops—an undesirable characteristic in current network operation practices by system operators. While enforcing radiality, this work allows algorithms to facilitate the formation of islands when necessary, such as during fault resolution.

To guarantee that every island in the network $\mathcal{G}_k$ adheres to radiality, the following constraints are introduced:

$$\sum_{i \in \mathcal{S}, j \in \mathcal{S}} a_{i,j} \leq |\mathcal{S}| - 1, \quad \forall \mathcal{S} \in \mathcal{V} \tag{3.25}$$

Here, $\mathcal{S}$ represents any subset of $\mathcal{V}$ with a cardinality greater than 2. The constraint is formulated

to ensure that a connected graph with $|\mathcal{S}|$ nodes contains no loops unless there are precisely $|\mathcal{S}|$ edges in the graph, as elaborated in [5]. Merely enforcing this constraint for $\mathcal{V}$ is insufficient, as it permits the formation of islands with loops. However, by applying the constraint $\forall \mathcal{S} \in \mathcal{V}$ (i.e., for every subset in $\mathcal{V}$), we ascertain that every island, viewed as a subset of nodes in $\mathcal{V}$, remains loop-free.

It is worth noting that utilizing (3.25) may have a potential drawback, namely, the exponential growth of constraints with the network's size. The number of constraints to be added is proportional to the number of subsets of nodes in $\mathcal{G}$ with a cardinality greater than 2 (bearing in mind that subsets with only two nodes cannot contain loops):

$$\sum_{i=3}^{|\mathcal{V}|} \binom{|\mathcal{V}|}{i} = \sum_{i=3}^{|\mathcal{V}|} \frac{|\mathcal{V}|!}{(|\mathcal{V}|-i)!i!}. \tag{3.26}$$

In practice, the number of constraints (3.25) to be added is considerably lower than (3.26). This reduction stems from the fact that it is unnecessary to consider subsets $\mathcal{S}$ that are disconnected in the physical topology or those connected by a number of links in the physical topology lower than $|\mathcal{S}| - 1$ (since forming loops in such cases is not possible).

To summarize, constraint (3.25) is exclusively added for subsets $\mathcal{S}$ satisfying the condition:

$$\sum_{(i,j)\in\mathcal{E}:i,j\in\mathcal{S},i\geq j} 1 > |\mathcal{S}| - 1, \tag{3.27}$$

implying that it is only applied to subsets of nodes connected (in the physical topology) by a number of links greater than $|\mathcal{S}| - 1$.

3.7 Simulations

We conduct a simulation spanning one day of network operation to illustrate how the integrated utilization of Energy Storage Systems (ESSs) and reconfiguration actions can effectively manage the grid under normal conditions and in the face of adverse scenarios, including simulated faults or (cyber-)attacks.

3.7.1 Simulation setup

The simulations were conducted using Julia 1.8.5 [20], and the Model Predictive Control (MPC) iterations were solved with the Gurobi solver [52]. The simulations were performed on a PC equipped with an Intel i7-8565U @1.8 GHz processor and 16 GB RAM, running Windows 11.

The selected grid is the 16-bus three-feeder distribution network, and the network data was sourced from [32]. Additionally, generation and Energy Storage System (ESS) devices were incorporated, along with additional branches to enhance the network's configuration capability (refer to Fig. 3.2). Specifically, the buses types are as specified in Tab. 3.2 and network electric parameters are summarized in Tab. 3.3. As detailed in [58], the substation voltages are fixed at 1.0 pu, and the minimum allowed load voltage is set to 0.95 pu. The sampling time is configured as $T = 15$ minutes, and the Model Predictive Control (MPC) prediction horizon is established at $N = 3$. The weights in equation (3.1) were chosen empirically, with $\alpha = 10$, $\delta = 10^{-3}$, and $\beta = 10^{-3}$. Notably,

Table 3.2: Bus types.

Bus types	**Bus Numbers**
Substations	1, 2, 3.
Hosting ESS	3, 7, 9.
Loads buses	5, 6, 7, 8, 9, 10, 11, 13, 14, 16.
Hosting DG	4, 12, 15.

Table 3.3: Conductance and susceptance in per-unit notation

Link	**G**	**B**
L1	0.225	0.3
L5	0.24	0.33
L4	0.27	0.54
L7	0.12	0.12
L2	0.33	0.33
L9	0.24	0.33
L10	0.33	0.33
L13	0.33	0.33
L11	0.24	0.33
L3	0.33	0.33
L15	0.27	0.36
L14	0.24	0.33
L16	0.24	0.33
L6	0.12	0.12
L12	0.12	0.12
L8	0.27	0.36
L17	0.12	0.12
L18	0.12	0.12
L19	0.12	0.12
L20	0.27	0.36

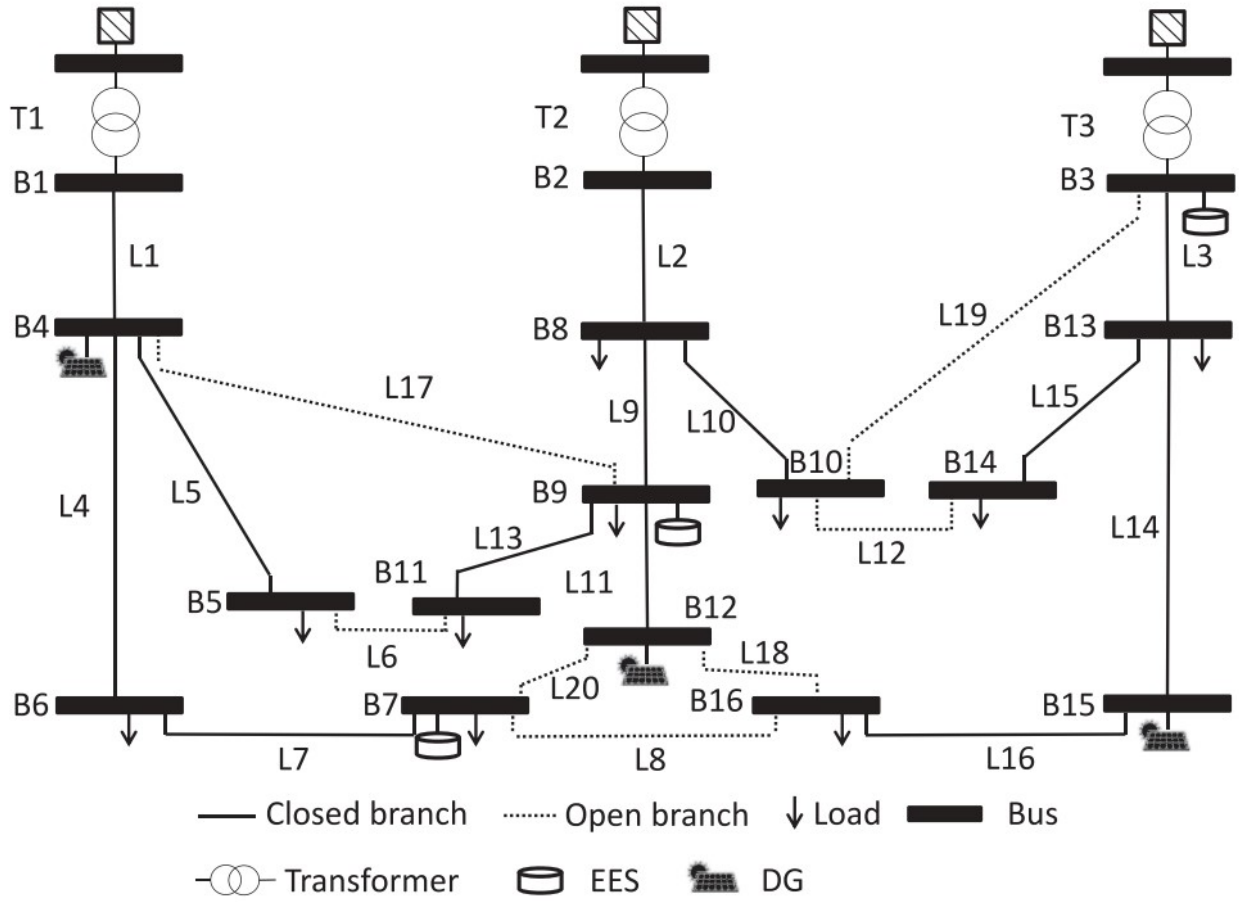


Figure 3.2: 16-bus distribution network [72]. During simulations all branches are considered configurable.

the orders of magnitude differ significantly between α and the other weights to emphasize a control focus on power loss reduction. The parameter δ was chosen sufficiently small to ensure effective activation of Energy Storage Systems (ESSs). Increasing δ would steer the controller toward maintaining ESS stored energy close to the reference value, discouraging their activation. The value of β was determined to balance performance while constraining the number of configurations.

The MV busbars of the HV/MV substations exhibit a small and constant power consumption of 10 kW, reflecting the devices' power requirement for network operation.

The simulated power generation at the buses hosting distributed generation is illustrated in Fig. 3.3, showing a peak of 750 kW during the hottest hours of the day and temporally shifted to simulate different geographical positions realistically. The simulated bus loads exhibit a stepwise shape and vary only in magnitude, as depicted in Fig. 3.4. This choice of load functions is intentional, aiming to subject the system to abrupt load changes. Such load profiles are valuable for testing under non-nominal conditions, allowing us to assess the robustness of the mathematical model and evaluate the system's responsiveness to adverse events. All measures are expressed in per-unit to mitigate numerical issues. To convert all electrical values in per-unit, we established the base power P_{base} as 62 MW and the base voltage V_{base} as 24 kV, representing the nominal power and voltage of the equipment, respectively.

At each bus, the maximum and minimum voltage values are set to $V_{\max} = 24$ kV and $V_{\min} = 22.8$ kV (i.e., 95% of $V_{\max}$). The transmission power constraints of substations are chosen as $P_{\max}^{SB} = 62$

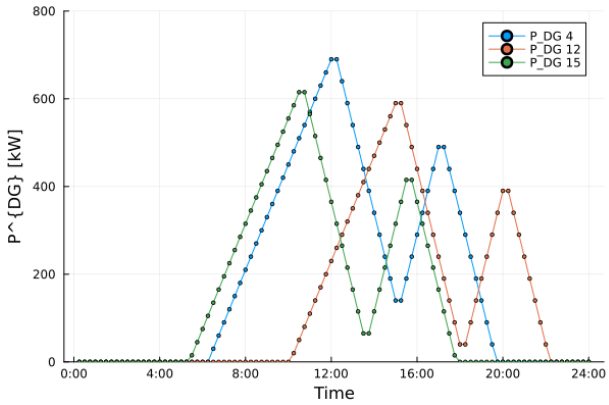


Figure 3.3: Power generation from the distributed energy resources throughout a day. Maximum peak of 750 kW.

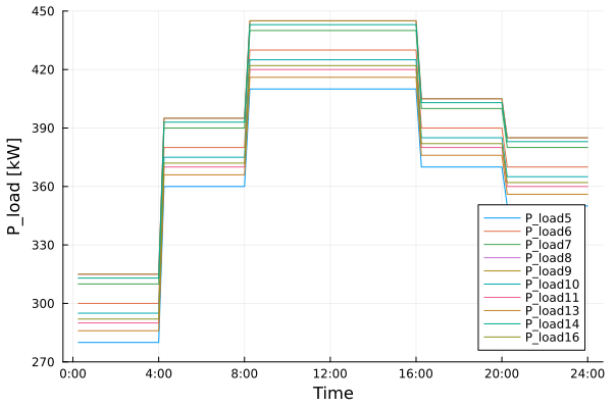


Figure 3.4: Load variations throughout a day.

MW, $P^{SB}_{\min} = -62$ MW, $Q^{SB}_{\max} = 60$ MVA, and $Q^{SB}_{\min} = -60$ MVA for active and reactive power, respectively. Power bus constraints are $P^{bus}_{\max} = 10$ MW, $P^{bus}_{\min} = -10$ MW, $Q^{bus}_{\max} = 10$ MVA, and $Q^{bus}_{\min} = -10$ MVA. The line power constraints are $P^{l}_{\max} = 10$ MW, $P^{l}_{\min} = -10$ MW, $Q^{l}_{\max} = 10$ MVA, and $Q^{l}_{\min} = -10$ MVA.

The Energy Storage System (ESS) energy reference is set to 80% of the total ESS capacity. All ESSs have a maximum capacity of 2 MWh.

All branches are assumed to be configurable, i.e., equipped with switches. It's worth noting that in a real case, this assumption could be relaxed because the network might have fixed and non-configurable branches.

Equation (3.25) is implemented using a customized version of the Depth-First Search (DFS) algorithm. The computational time of this algorithm in the worst case is linear in the size of the graph, denoted as $\mathcal{O}(|\mathcal{V}| + |\mathcal{E}|)$, where $|\mathcal{V}|$ is the number of vertices and $|\mathcal{E}|$ is the number of edges. Additionally, it is linear in space in the number of vertices, expressed as $\mathcal{O}(|\mathcal{V}|)$ in the worst case.

3.7.2 Simulation of Normal Conditions

In this section, simulations under normal conditions are presented, where no faults occur. Figure 3.5 illustrates the computed configurations in the case of nominal conditions. The initial configuration (graph topology) was set with all branches open. In the first iteration, the algorithm chooses to close some branches, as shown at time 0:00. Throughout the 24-hour simulation period, four reconfigurations are computed. The power flow for all lines is depicted in Fig. 3.6. Here, a generic

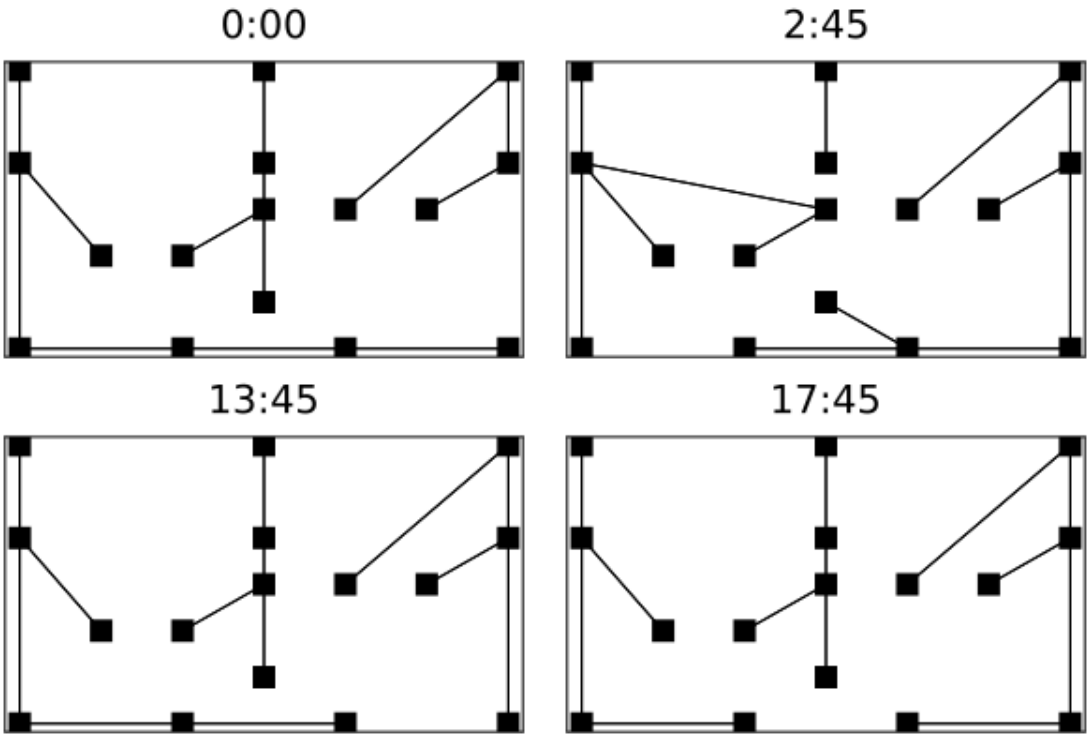


Figure 3.5: Starting configuration with all opened branches. At the first iteration (namely 0:00) the topology is calculated.

line Lk refers to the connection between bus i and bus j, where $i < j$. For instance, $L16$ represents the line connecting bus 15 and 16. Figure 3.7 provides a detailed view of the power flow on line 16. In this case, both flows described by the power flow equations are shown: from bus i to bus j and vice versa. The behaviors of power and energy stored by ESSs are depicted in Figures 3.8 and 3.9, respectively. ESSs at bus 7 and 9 consistently play an active role in the network. Firstly, they are located in critical parts of the network. ESS 7 needs to supply power to the network tree that is disconnected from any DG, and it has to power a large subtree. Additionally, it is situated far from the primary substation. After 12:00 and 18:00, its role is to supply power to the largest subtree in the network. ESS 3 is never activated, as expected, because it is directly connected to the HV/MV substation.

Simulations were also carried out by setting $\beta = 0$ and obtaining 79 reconfigurations.

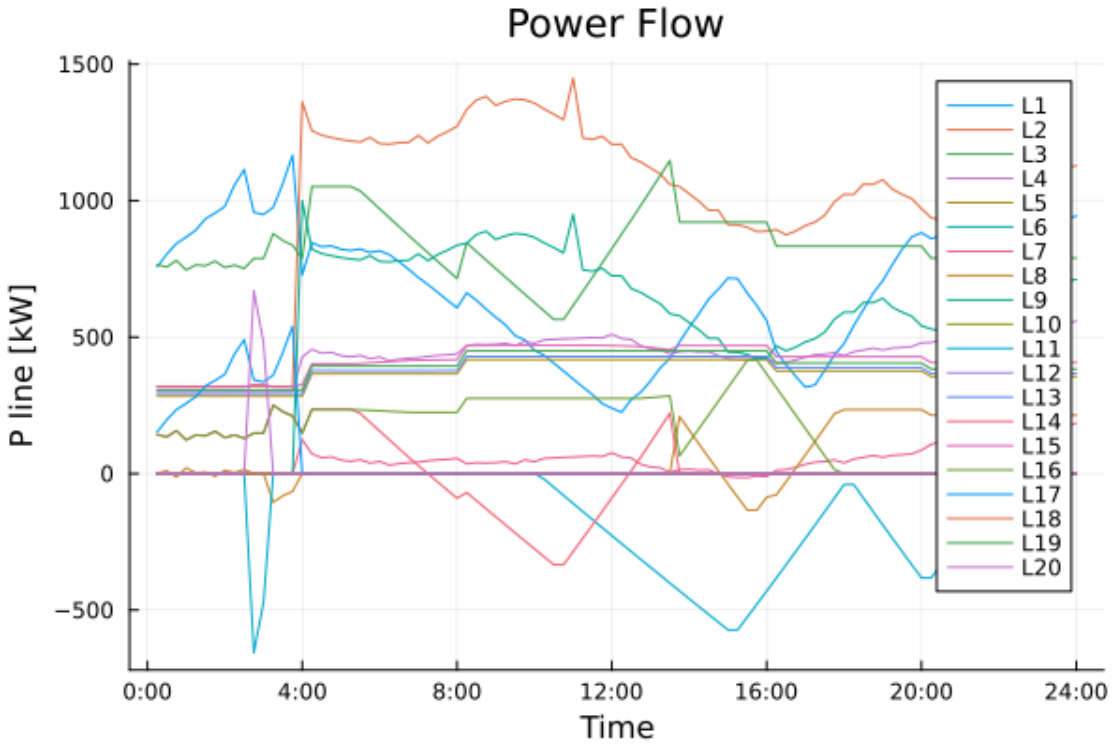


Figure 3.6: Power flow in normal conditions. Each line represent P_{ij}, i.e. the power flow from bus i to bus j, with $i < j$. For example line 13 (L13) connects bus 9 (B9) and bus 11 (B11) and its power flow is $P_{9,11}$.

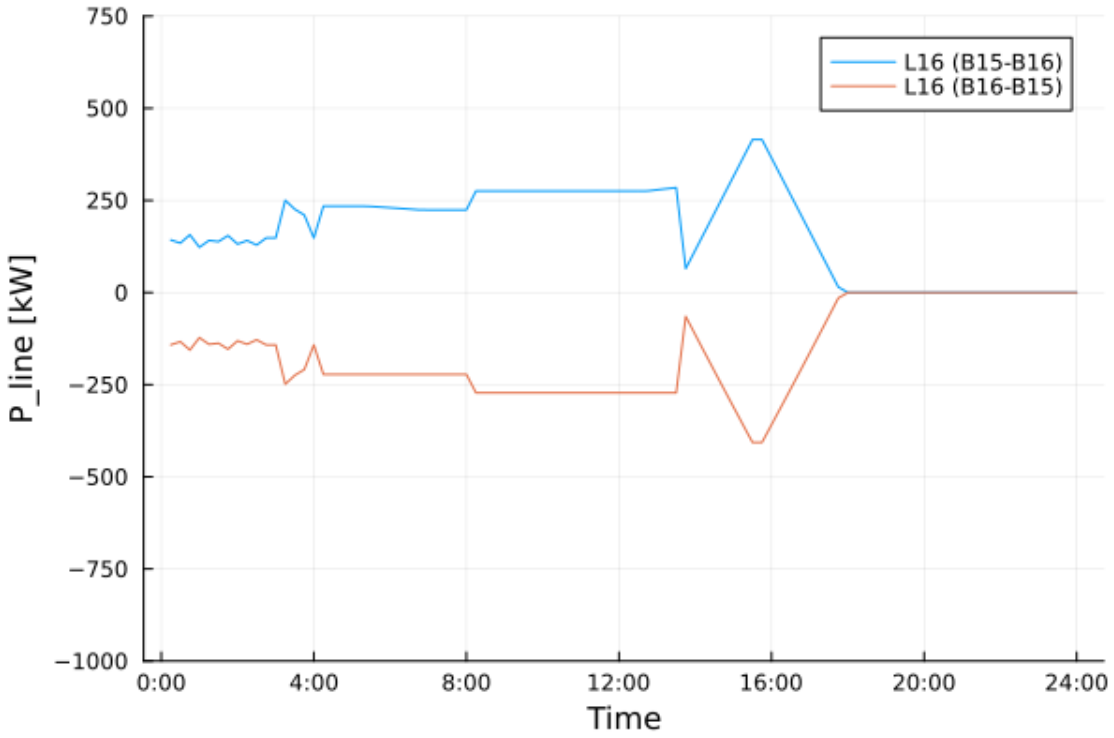


Figure 3.7: Power flow of line 16 (L16) in normal conditions. The power flow is plotted both looking the line from bus 15 (B15) to bus 16 (B16), i.e. $P_{15,16}$, and vice versa ($P_{16,15}$).

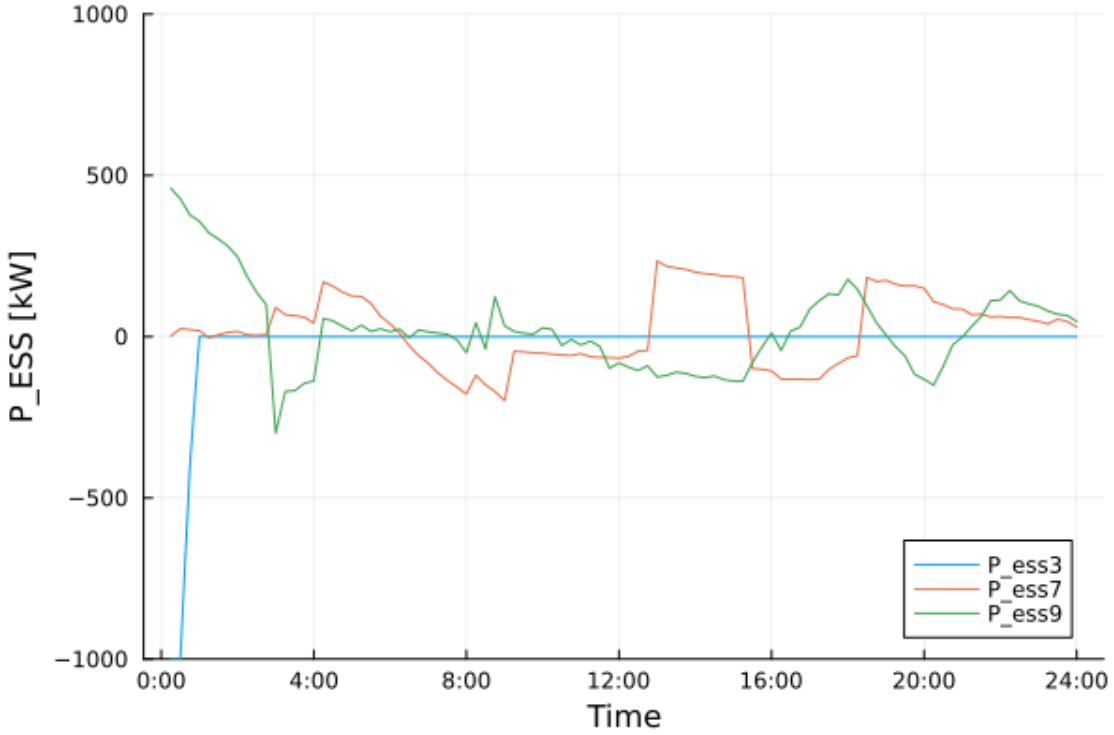


Figure 3.8: ESS power in normal conditions. Lower bound at -1000 kW, maximum value of ≈ 220 kW reached by ESS 7.

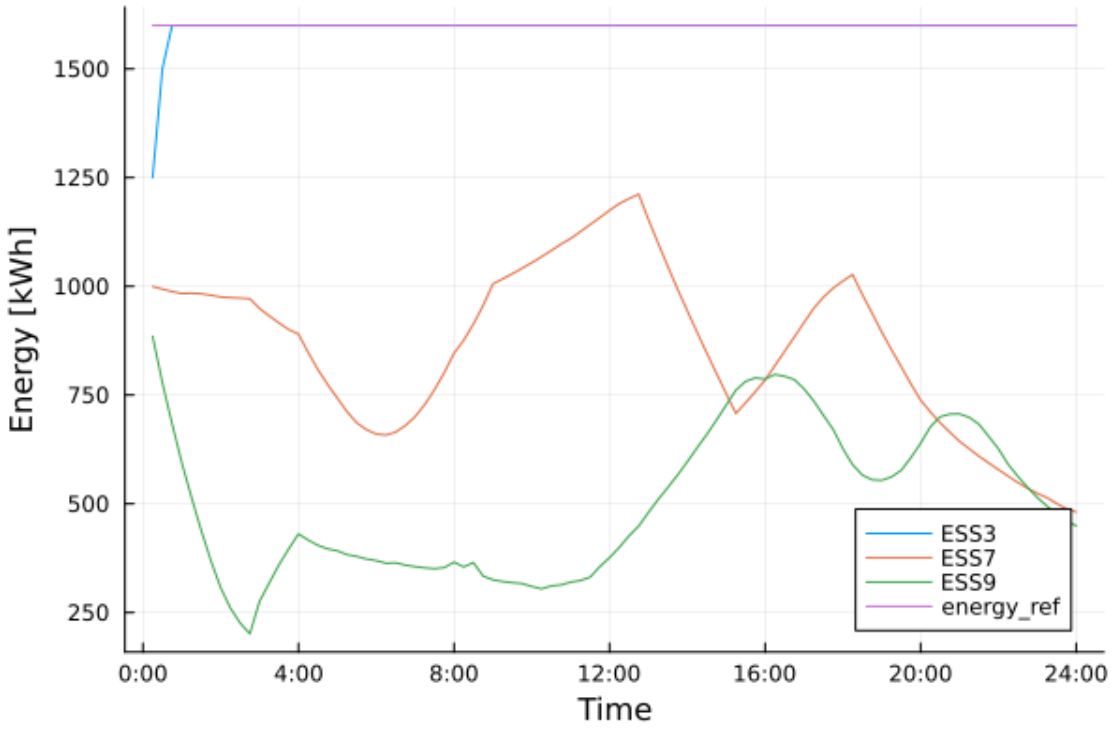


Figure 3.9: ESS energy in normal conditions. Reference value is 80% of the maximum capacity, i.e. 1600 kWh.

3.7.3 Simulation of Adverse Event

This section presents a scenario in which faults/(cyber-)attacks occur at some point in time, prompting the algorithm to calculate a new configuration to counteract the impact of the faulty lines or other adverse events. The faulted lines are detailed in Table 3.4, indicating that a faulty line is disconnected from the network, and its recovery is assumed to take no less than 24 hours. In other words, the fault is considered permanent for the entire duration of the simulation.

Table 3.4: Lines Fault

Fault time	Line
8:45	L18
14:45	L11
20:00	L4 and L8

Seven reconfigurations take place, with some of them depicted in Figure 3.10. The configuration computed in the first iteration is not plotted and remains unchanged until 7:30 when a new configuration is calculated. It's important to note that the aim is to induce the formation of an island to simulate the failure of lines as described in Table 3.4. The island is formed at 20:00. Figures

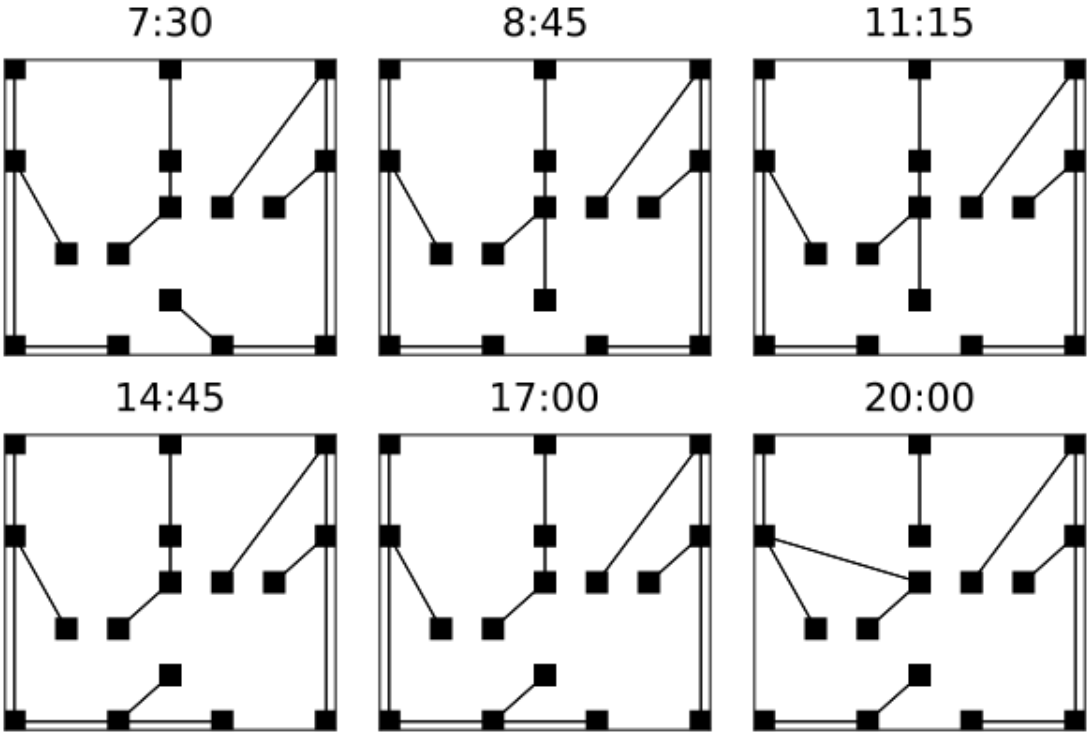


Figure 3.10: Faults occur at 8:45 for L18, 14:45 for L11 and at 20:00 for L4 and L8. The island is formed at 20:00.

3.11 and 3.12 illustrate the energy and power profiles of each ESS, with vertical lines indicating the time steps when faults occur. In this case as well, the weights of the objective function were selected to incentivize the use of ESSs. As anticipated, ESS 3 remains inactive due to its proximity to the HV/MV substation. At 20:00, ESS 7 becomes more actively involved as it needs to supply power to the island formed after the faults. Figure 3.13 displays the power flows of line 16 (L16),

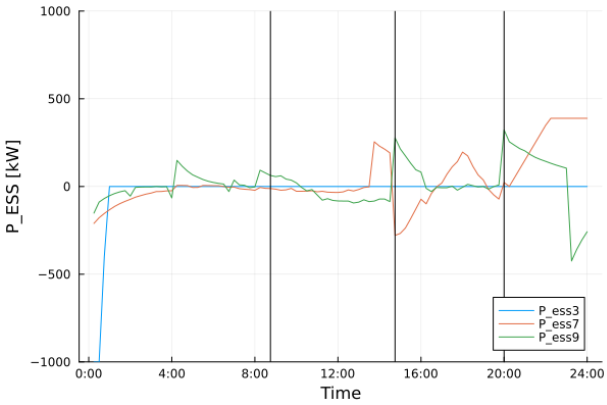


Figure 3.11: ESSs power during the simulation of adverse event. Vertical bars indicate line faults.

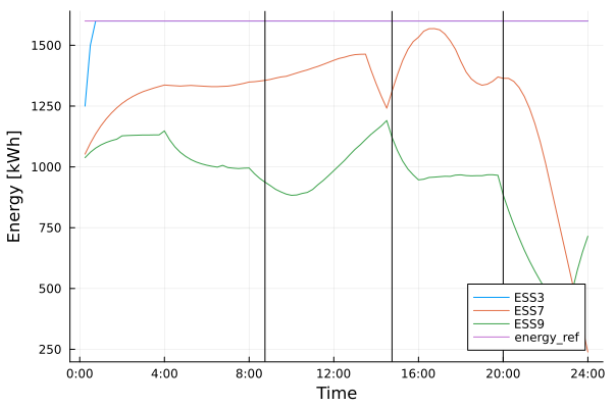


Figure 3.12: ESSs energy during the simulation of adverse event. Vertical bars indicate line faults. Reference value is 80% of the maximum capacity, i.e. 1600 kWh.

which becomes inactive due to the reconfigurations from 14:45 until just before 20:00, as shown in Figure 3.10. A comprehensive overview of the power flows along all the lines is presented in Figure 3.14. It is noteworthy that the only difference from the normal scenario is that, in this case, the simulations were conducted with a constant load of $50kW$ connected to bus 7. This adjustment was made because the island is powered by only an ESS and a DG, resulting in limited available power to counteract the load. In this particular case, a high consumption renders the problem unfeasible, as anticipated. To demonstrate the effectiveness of the third term in the objective function, simulations were also conducted with $\beta = 0$, resulting in 63 reconfigurations.

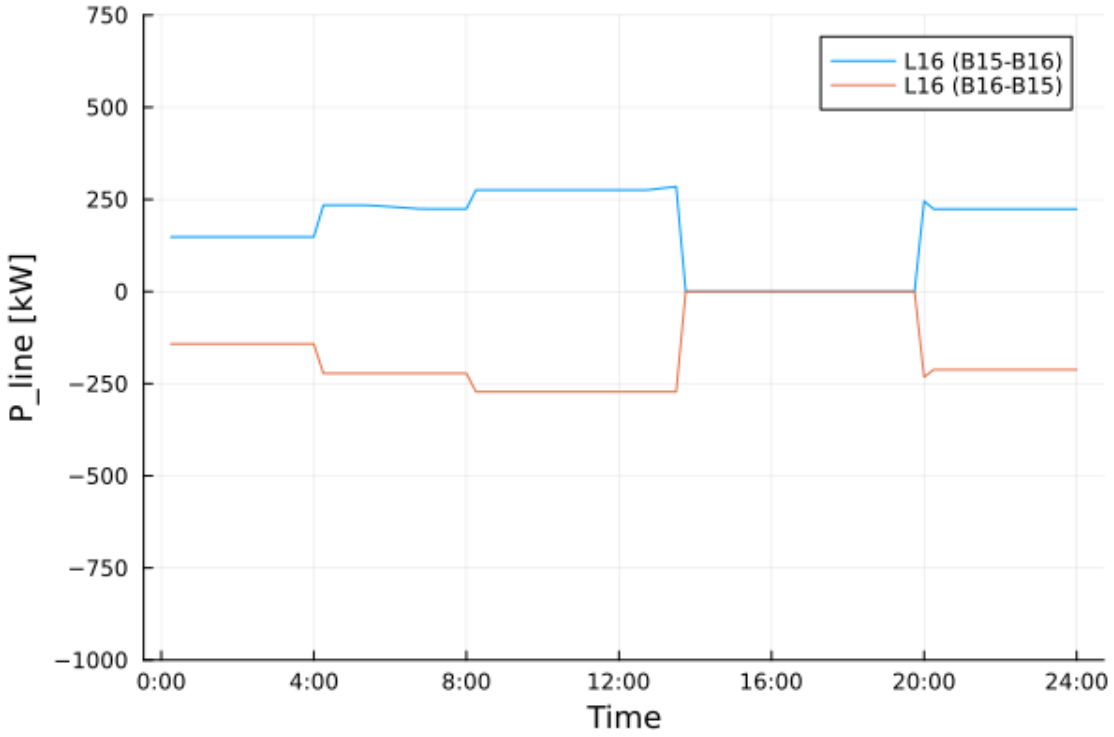


Figure 3.13: Power flow of line L16 in fault conditions. The power is plotted both looking at the line from bus 15 to bus 16 and vice versa. The power flow is zero when the line is disconnected due the reconfiguration from 14:45 to just before 20:00.

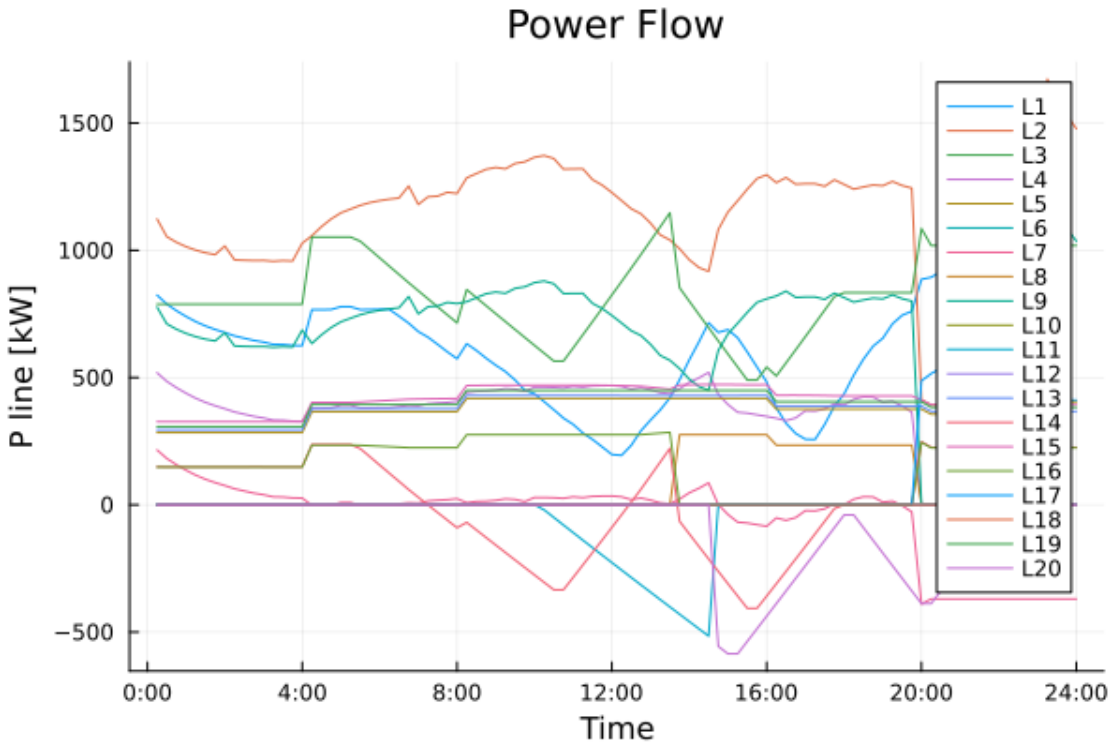


Figure 3.14: Power flow of all lines in fault conditions. Each line represent P_{ij}, i.e., the power flow from bus i to bus j, with $i < j$. For example line 13 (L13) connects bus 9 (B9) and bus 11 (B11).

3.8 Benchmark and Power Losses considerations

The last consideration pertains to power losses, and the simulations in Sections 3.7.2 and 3.7.3 are compared with the case in which the network is static, i.e., no reconfiguration occurs. In this scenario, the reference is the one depicted in Fig. 3.2, where links with solid lines are considered fixed, and the ones with dotted lines are always considered open ($a_{ij} = 0$). Losses are simply calculated as the sum of the power injections at each bus. Figures 3.15 and 3.16 show losses in the static scenario, with the total mean loss being 4.9%.

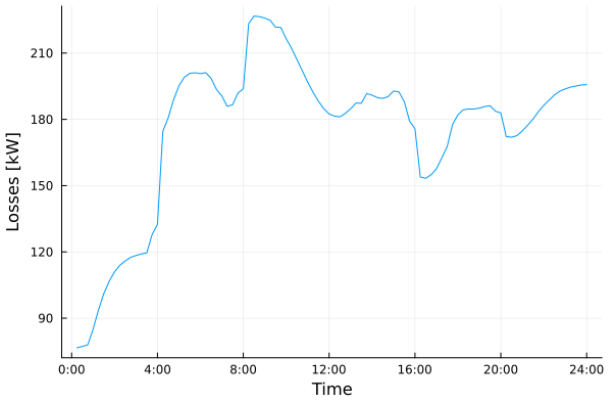


Figure 3.15: Losses in static scenario.

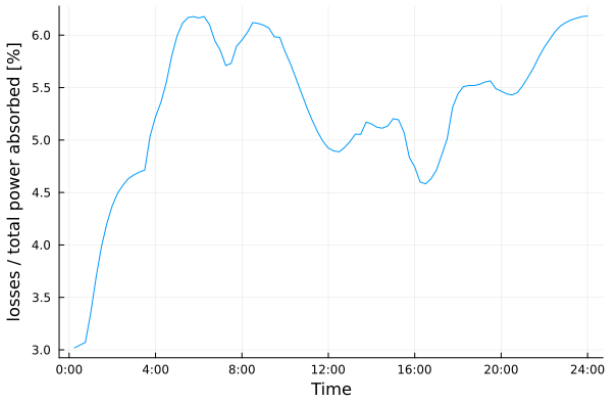


Figure 3.16: Ratio of losses / total power absorbed in static scenario. The total mean loss is 4.9%.

Figures 3.17 and 3.18 show losses in case of normal conditions scenario and the total mean loss is 4.6%. Figures 3.19 and 3.20 show losses in case of adverse events scenario and the total mean loss

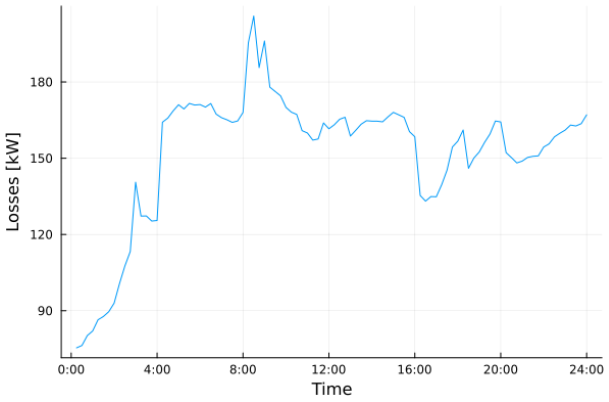


Figure 3.17: Losses in normal conditions scenario.

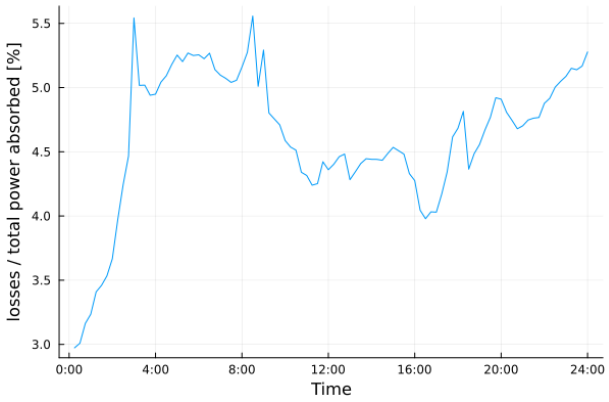


Figure 3.18: Ratio of losses / total power absorbed in normal conditions scenario. The total mean loss is 4.6%.

is 5.01%. The results clearly demonstrate that the current reconfiguration algorithm outperforms the benchmark solution represented by a completely static network in terms of loss optimization.

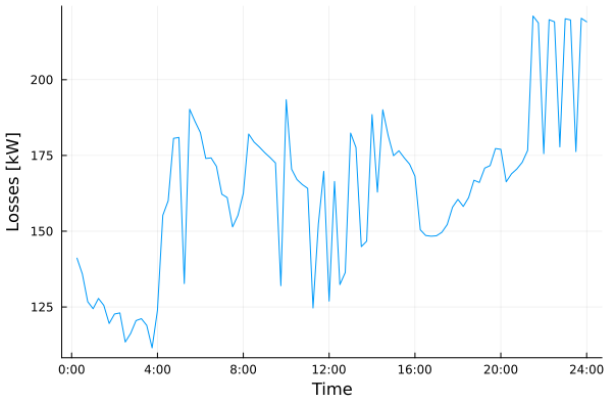


Figure 3.19: Losses in adverse events scenario.

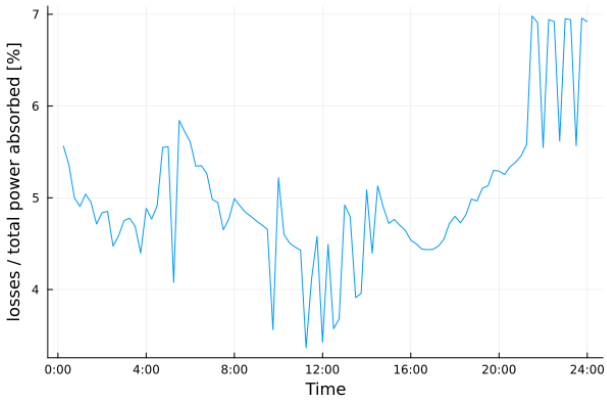


Figure 3.20: Ratio of losses / total power absorbed in adverse events scenario. The total mean loss is 5.01%.

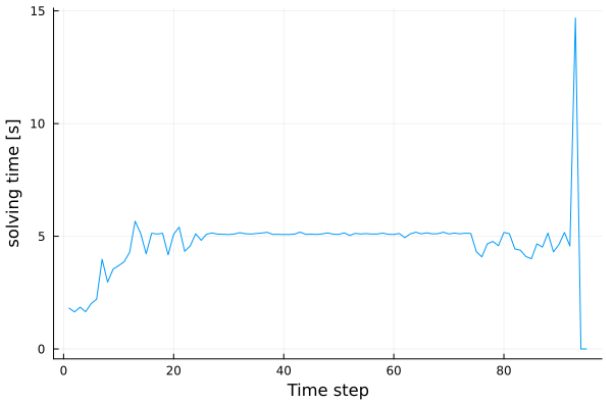


Figure 3.21: Elapsed time in normal conditions scenario.

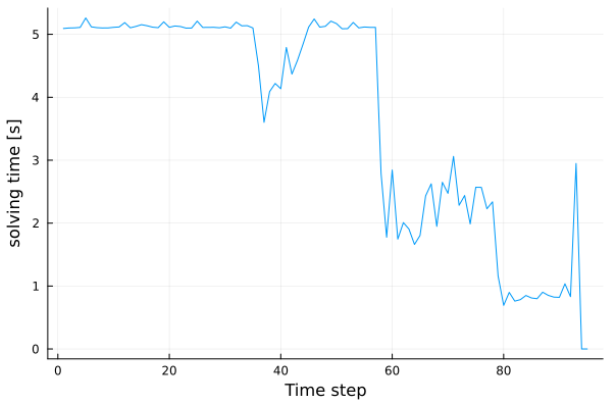


Figure 3.22: Elapsed time in adverse events scenario.

3.9 Computational Complexity

The computational time for each MPC iteration is presented in Figs. 3.21 and 3.22.

3.10 Conclusions and Future Works

This paper introduces a Model Predictive Control (MPC) framework for the dynamic reconfiguration of electricity grids. The extended mathematical formulation allows for the dynamic formation of islands, enhancing the grid resilience to faults and adverse events. The inclusion of dedicated radiality constraints ensures radial operation during both normal and islanded conditions. Simulations demonstrate the algorithm effectiveness in minimizing losses, dynamically reacting to faults, and forming islands when needed.

Future work will focus on extending the algorithm to incorporate risk notifications for network components, enabling proactive responses to cyberattacks. Real-time simulations will also be conducted to account for the network's dynamics during switching, evaluating the controller's performance in real-time implementations, including the impact of MPC computation time on the sampling time.

Chapter 4

Optimization of Traffic in Smart Cities

This chapter contains the work done by the candidate about his research activity in the field of Smart Cities. The findings detailed in the subsequent sections contributed to the recently accepted publication:
[74] *Liberati, F.; Donsante, M; Tortorelli, A. "Single Intersection MPC Traffic Signal Control in Presence of Automated Vehicles". IEEE Transactions on Control Systems Technology.* Submitted.

4.1 Introduction

Over the past decade, global efforts have intensified to address the challenges posed by climate change, culminating in the 2016 Paris Agreement, which set legally binding objectives for limiting global warming. The agreement aims to reduce greenhouse gas (GHG) emissions by 43% by 2030 with the ultimate goal of limiting global temperature rise to $1.5°C$ by 2100. Building on this, the European Union's 2019 Green Deal set even more ambitious targets, declaring its intention to achieve climate neutrality by 2050. Given that over 70% of GHG emissions occur in urban areas [100], effective strategies to reduce emissions in cities are paramount.

Urban GHG emissions primarily stem from buildings and transportation, with road transportation contributing to more than 70% of transportation-related emissions, and private cars accounting for over 60% of that share [44]. Notably, studies (e.g., [21, 22, 88]) emphasize the impact of road congestion on GHG emissions. Intelligent Traffic Light (TL) control systems emerge as a promising solution to address this issue [90]. Beyond environmental benefits, improved traffic management can significantly enhance the quality of life by reducing travel time and congestion. This work presents a predictive control scheme designed to optimize TL timings at a single intersection, contributing to more efficient urban traffic management.

4.2 Related Works

Intelligent Traffic Light (TL) control has been a focal point of extensive research. The traditional control approach involves fixed TL timings, optimized based on statistical properties of the traffic flows they regulate [23]. While simple to implement and not reliant on additional sensing or computing equipment, this strategy lacks adaptability to real-time traffic conditions, potentially resulting in inefficiencies during deviations from nominal conditions. Recognizing these limitations,

the literature has seen the proposal of various enhanced TL control strategies. Comprehensive reviews of these strategies can be found in works such as [110], [30], and [78]. Recently proposed advanced techniques for Traffic Light (TL) control encompass a range of methodologies, including Model Predictive Control (MPC), fuzzy control, evolutionary algorithms, game theory, and deep reinforcement learning, among others. Among these, MPC-based approaches have demonstrated significant efficacy in addressing TL control challenges due to their ability to explicitly consider constraints, formulate predictive control actions by incorporating future time instants, and maintain lower computational costs compared to optimal control-based methods [128]. On the other hand, fuzzy control, evolutionary algorithms, and (deep) reinforcement learning techniques offer opportunities to further reduce computational demands, albeit with varying degrees of optimality and without guaranteed satisfaction of safety constraints in all scenarios. In this context, this work aims to present an MPC-based control strategy with a low computational cost, making it feasible for real-time implementation to address the TL control problem adaptively and predictively.

Building upon the groundwork laid in prior works, specifically [70] and [56], this article introduces a Model Predictive Control (MPC) strategy based on mixed-integer optimization. Pertinent literature in this domain includes [62] and [63], where the authors present an MPC framework for intelligent traffic signal control using mixed-integer linear programming (MILP). The primary focus of these works is to reduce the total number of vehicles in the network by minimizing the sum of queues. In contrast, the approach presented in this article adopts a more comprehensive objective. The proposed MPC framework aims not only to minimize queues but also to reduce and balance waiting times. This is achieved through the utilization of a quadratic objective function. Furthermore, compared to [63], the present paper introduces a more refined queue model. In [61], the authors introduce a model-free adaptive predictive control approach for managing phase splits in urban traffic networks, with the objective of minimizing the total number of vehicles in the network. The proposed method involves deriving a nonlinear traffic network dynamical model, which is then linearized to formulate a simplified Model Predictive Control (MPC) based signal split strategy. The linearization process entails computing the Jacobian of the traffic network nonlinear model at each control step. In contrast, the present work focuses on developing precise queue equations that can be exact linearized through the incorporation of additional auxiliary constraints and Boolean variables. A similar control strategy incorporating relevant rules from the US National Electrical Manufacturing Association (NEMA) standards for the sector is presented in [121]. In [41], the authors extend previous research by introducing a distributed Model Predictive Control (MPC) framework, specifically based on the dual decomposition method, for urban traffic network control. The proposed approach demonstrates a notable reduction in computation times and queue lengths. Future work will explore a similar decentralized version of the presented methodology for application in traffic network control scenarios. Compared to the mentioned article, the present work provides a more accurate modelling of the queue balance equation. Additionally, the solution proposed in this article incorporates variable cycle times (i.e., the sum of traffic light phases' durations), dynamically adjusted by the controller to best match prevailing traffic conditions. In [126], the authors propose a distributed event-triggered Model Predictive Control (MPC) strategy for traffic signal control in urban traffic networks. The primary objective of the proposed solution is to minimize the total time that cars spend in the network. Unlike traditional MPC, where re-optimization occurs periodically based on time-driven triggers, the controller in this article triggers re-optimization

(i.e., it computes a new control solution) only when specific conditions are met, thereby avoiding redundant computation and communication efforts. However, such approach is not applicable to the algorithm proposed in this article since the present controller also provides reference trajectories to automated vehicles. In [1], the authors introduce an MPC framework for controlling a single intersection, considering pedestrian requests and in [2], a refined version of this approach is proposed. The latter article presents a decentralized traffic light (TL) control based on an efficient linear formulation of the TL control problem, allowing for real-time control of intersection signals. The authors emphasize the advantages of decentralized MPC methods over centralized and hierarchical ones. The proposed queue dynamics model evolves by considering a fixed departure rate for each intersection. In contrast, the current article models the trajectory of each vehicle in each queue in detail, resulting in a more intricate nonlinear queue dynamics. The estimation of queue lengths in urban traffic intersections is a crucial aspect for TL control problems. The proposed approach in this article has been developed to provide a detailed queue model, capturing the actual status of the intersection in a fine-grained manner. An interesting work in this regard is presented in [107], where the authors investigate different strategies to estimate queues. In [94], the authors present a distributed Model Predictive Control (MPC) strategy to optimize various performance indices related to the traffic network. In this approach, each transport network link utilizes only locally available information. In [54], the authors develop both a centralized and a decentralized control framework to address congested roads. The distinctive feature of this work is its focus on emergency vehicles, and the proposed approach can also manage their priority, aligning with the capabilities of the present algorithm.

In [49], the authors tackle a scenario where both self- and human-driven vehicles coexist. The objective is to compute an optimal sequence for vehicles leaving the intersection, formulated as an integer-constrained nonlinear optimization problem. The proposed solution, relaxed to consider only newly arrived vehicles for scalability, demonstrates improved performance over the simple first-in-first-out rule. The development of a control scheme capable of handling both human-driven and self-driven vehicles, as well as integration with self-driven vehicles, aligns with the approach presented in this work. In comparison to [49], this paper provides a more integrated and simplified formulation, where a single controller harmonizes TL signal timing with vehicle trajectories, encompassing the trajectories of both Manually Driven Vehicles (MDV) and the computed trajectories for Automatically Driven Vehicles (ADV). Additionally, a more detailed modeling of queue dynamics is presented, accounting for the backward propagation of TL signal switching effects on arriving vehicle queues. The controller fully incorporates the generalizations discussed in [49], including reaction times, yellow-signal dynamics, and a more precise discussion of stopping sequences.

In [42], an interesting solution is presented, focusing on optimizing the throughput of interconnections by leveraging connected vehicles. The authors discuss the results concerning different connected vehicle arrival rates. Another application scenario is explored in [115], considering a multi-modal environment with private and public vehicles, as well as light rails. The authors use a cell transmission model to describe queue dynamics and solve the optimization problem using the particle swarm optimization algorithm. Addressing the TL control problem in large transportation networks, [120] proposes an adaptive linear quadratic regulator control framework. The objectives include minimizing traffic delays and limiting significant variations in the control variable. A similar self-tuning MPC controller is introduced in [130], adapting to changing scenarios and utilizing linear

and quadratic programming optimization techniques.

4.3 Paper Contributions

The paper main contribution lies in proposing an integrated MPC controller that incorporates an adaptive and detailed model of intersection dynamics, accounting for vehicle interactions with each other and with the TL.

The proposed controller jointly determines and harmonizes the optimal signal timing for the TLs at the intersection, and the computation of optimal trajectories for ADVs, while modelling also MDVs dynamics, leading to significant reduction of queue length and waiting times.

Moreover, the controller integrates a detailed model of vehicle/queue dynamics and their interaction with TL signal timing logic. It is adaptive, meaning that specific constraints vary based on the vehicle type (manual vs automated), allowing the controller to operate in mixed scenarios.

The formulation also includes unique constraints to ensure recursive feasibility, a non-trivial aspect given that approaching vehicles naturally push the problem towards constraint activation, potentially leading to infeasibility. For instance, consider a vehicle approaching the TL during a red signal; this situation brings the system closer to a constraint that must not be violated (i.e., the vehicle cannot pass the intersection with a red signal).

In contrast to the earlier works [70] and [56], the current paper extends the MPC formulation to encompass the dynamics of each vehicle position, velocity, and waiting time at the intersection. This extension results in a more accurate and detailed model of the queues, whereas the mentioned papers solely presented an aggregate model of the queues.

4.4 Reference Scenario

The reference scenario revolves around a single intersection where the flow of vehicles is regulated by traffic lights (Fig. 4.1). Each traffic light displays only one color (green, yellow - also referred to as amber - or red) at any given time. The proposed algorithm operates in discrete time and aims to make decisions at each time step regarding the traffic light signals, with the objective of enhancing traffic conditions based on desired metrics such as queue lengths and waiting times.

It is assumed that the intersection is equipped with sensors capable of measuring various variables relevant to the control problem. These variables include the number of vehicles in each queue, the position, velocity, and cumulative waiting time of each vehicle in queue, and technical parameters of each vehicle (e.g., length, maximum velocity, and acceleration). Measurements could be obtained directly or estimated by each traffic light, using technologies such as cameras, road-side sensors, or information communicated by connected vehicles. The paper does not delve into the intricacies of measurement or estimation processes.

The proposed Model Predictive Control (MPC) algorithm is designed to operate in the presence of both manually-driven and automated vehicles. Different constraints are applied to each type of vehicle in the controller mathematical formulation, making it adaptable to real-time scenarios ranging from exclusive manual driving to fully automated driving. The controller takes charge of vehicles once they approach a certain distance from the intersection.

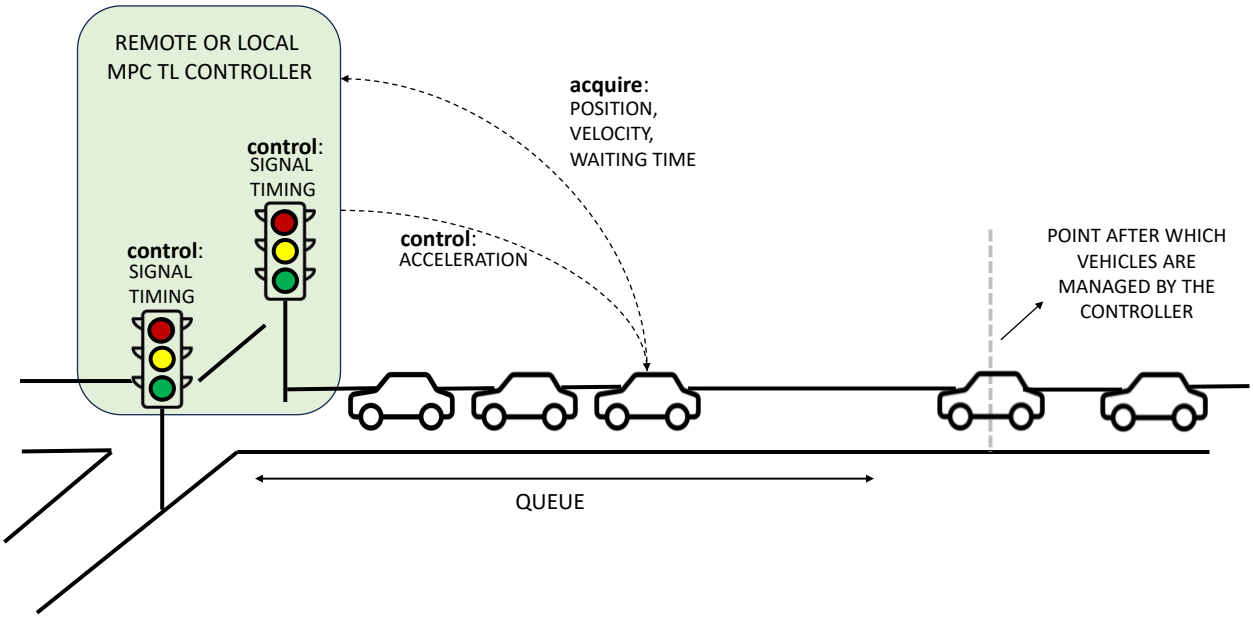


Figure 4.1: Reference scenario showing the interaction of the controller with the TL and the vehicles.

The controller can either reside locally with the traffic lights (TLs) or be deployed remotely, with measurement and control information exchanged through fast and reliable communication. In the event of faults, a traditional control strategy can be implemented.

For simplicity, the model assumes that each road at the intersection has only one lane in each direction, and it considers only vehicle traffic (excluding interactions with pedestrians). These assumptions are made to streamline the mathematical formulation of the MPC problem, but they can be removed with additional modeling efforts.

4.5 Formulation of the Proposed Controller

4.5.1 Notation and nomenclature

Given a set $\mathcal{S}$, the symbol $2^{\mathcal{S}}$ denotes its power set, which includes all possible subsets of $\mathcal{S}$. The symbol $|\mathcal{S}|$ denotes the cardinality of set $\mathcal{S}$. For a natural number n, $\mathcal{I}_n$ represents the set $0, 1, ..., n$. The notation $\lceil a \rceil$ represents rounding the real number a to the nearest upper integer. The symbol $\mathbb{N}_0$ denotes the set of natural numbers including zero. The vector $\mathbb{1}_n$ signifies a column vector of ones with a dimension of n. If x and y are two column vectors, $col(x, y)$ denotes their column concatenation.

The nomenclature used in the remainder of the chapter is summarized in Table 5.1.

Table 4.1: Nomenclature used in Section 4.5.

Symbol	Explanation
$d_{i,v}$	Acceleration of vehicle v at time i.
α_1, α_2	Objective function weight coefficients.
$d_{i,v}$	Position of vehicle v at time i.
$\delta_{i,v}$	Boolean variable equal to 1 if $d_{i,v}$ is positive.
ϵ	Small positive constant.
$f(v)$	The vehicle immediately following vehicle v.
$g_{i,j}$	Boolean variable equal to one if and only if TL j is green at time i.
j	Index to denote a generic TL.
k	Index to denote the current time.
i	Index to denote a generic time in the MPC prediction window.
$y_{i,j}$	Boolean variable equal to one if and only if TL j is yellow at time i.
M	Large positive number.
$n_{i,j}$	Number of vehicles at queue j at time i.
ν_v	Integer variable.
N	Length of the MPC prediction window (in time steps).
$r_{i,j}$	Boolean variable equal to one if and only if TL j is red at time i.
$t^y_{i,j}$	Duration of the yellow time at TL j at time i.
$t^{y,min}_j$	Minimum duration of the yellow sign.
T	Sampling time (seconds).
$v_{i,v}$	Velocity of vehicle v at time i.
$w_{i,j,v}$	Total waiting time in queue of vehicle v at TL j at the *beginning* of time interval i.

4.5.2 MPC logic

The Model Predictive Control (MPC) control logic that forms the basis of the proposed Traffic Light (TL) control algorithm is outlined in Algorithm 2. The problem is formulated in discrete

Algorithm 2 MPC control algorithm

1: **for** every time k (i.e., at the beginning of every sampling interval) **do**
2: Acquire the current state of the intersection and all other relevant inputs that enter the formulation.
3: Build and solve an optimal control problem (call it $\mathcal{P}_k$) to determine the optimal TL control signals to apply during the current time interval k, i.e., $g_{k,j}$, $y_{k,j}$, $r_{k,j}$ (green, yellow, and red signals), and the optimal reference trajectories for ADV. Problem $\mathcal{P}_k$ is defined from time k to the future time $k+N$, where N is the length of the MPC prediction/control window. That is, the controller optimizes the near future performance of the intersection.
4: Actuate the found optimal TL control signals $g_{k,j}$, $y_{k,j}$, $r_{k,j}$ over the time interval k.
5: **end for**

time, where T represents the sampling time in seconds. At each sampling time $k = 1, 2, \ldots,$ the controller collects the current state of the intersection, including the position, velocity, and cumulative waiting time of every vehicle. Subsequently (Step 3), the optimal values of the TL control signals $g_{k,j}$, $y_{k,j}$, $r_{k,j}$ (representing green, yellow, and red, respectively) to be activated are determined by solving a constrained optimization problem ($\mathcal{P}_k$) defined over a time window, known as the *prediction horizon*, of N time steps into the future. In case of ADV, the algorithm can also compute the optimal reference trajectory for vehicles to pass the intersection.

Finally (Step 4), the optimal TL controls obtained by solving $\mathcal{P}_k$ in Step 3 are applied over the current time interval. This process repeats from Step 2 at the end of each time interval.

In the subsequent sections, when describing the formulation of the constrained optimization problem constructed and solved in Step 3, the common Model Predictive Control (MPC) notation with two-time indices will be used. This notation distinguishes between the index representing the current time k (when the MPC iteration is executed) and the time index i used to denote a specific time slot in the prediction window ranging from k to $k+N$. This nomenclature is elucidated in Fig. 4.2.

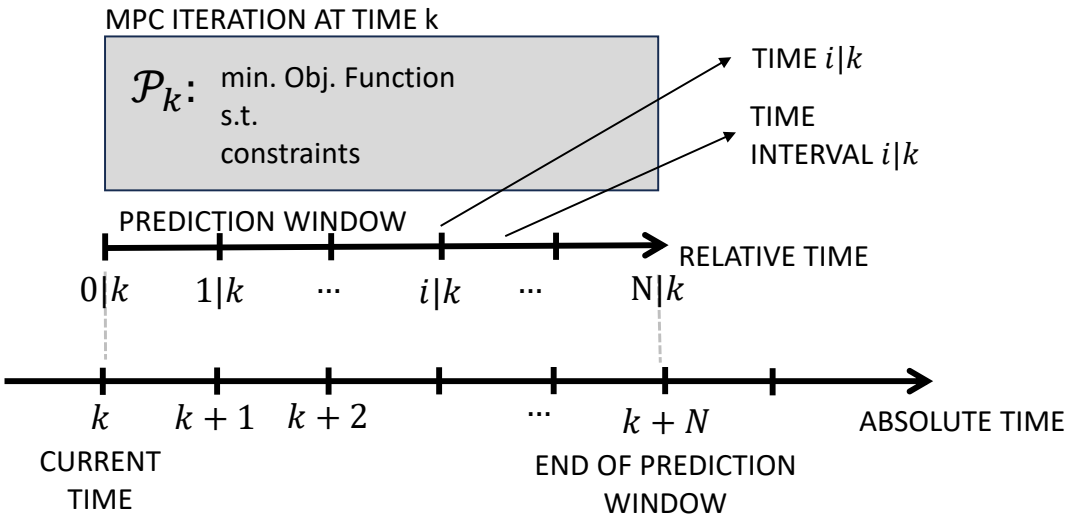


Figure 4.2: MPC time notation.

It's essential to clarify that the state variables in the formulation (e.g., positions and velocities of the vehicles) denote the value that the given variable takes at time $i|k$, i.e., at the beginning of time interval i. For instance, $v_{i|k,v}$ represents the predicted velocity computed by the MPC algorithm

at time k for vehicle v *at the start* of time interval $i|k$. This is a point-in-time information, as the value of the variable may vary during time interval $i|k$. In contrast, the control variables represent the constant value to be applied *throughout* interval i (e.g., $g_{i|k,j} = 1$, indicates that, at time k, the predicted signal for TL j to display during time interval $i|k$ is green).

4.6 Definition of relevan sets

The primary sets of interest include the time steps within the prediction window, the set of TLs, and specific sets of vehicles. The definitions for the first two are as follows:

- $\mathcal{I}_{N-1} := \{0, 1, ..., N-1\}$, set of time steps in the prediction horizon;
- $\mathcal{J}$ the set of TLs. There is a one to one correspondence between a TL and the associated queue of vehicles.

The subsequent sets of vehicles can be calculated at the conclusion of the generic time interval k, relying on observations gathered from the field and/or communication with the vehicles:

- Set $\mathcal{A}_{k,j}$ (arriving vehicles); the set of vehicles that arrive at TL j between k and $k+1$ (i.e., during time interval k);
- Set $\mathcal{D}_{k,j}$ (departing vehicles); the set of vehicles that pass the intersection, leaving queue j, between k and $k+1$;
- $\mathcal{V}_{k,j}$, the set of vehicles present at TL j at time k. Its dynamics is given by:

$$\mathcal{V}_{k+1,j} = (\mathcal{V}_{k,j} \cup \mathcal{A}_{k,j}) \setminus \mathcal{D}_{k,j}. \tag{4.1}$$

It is useful to further partition $\mathcal{V}_{k,j}$ as $\mathcal{V}_{k,j} = \mathcal{V}_{k,j}^{MDV} \cup \mathcal{V}_{k,j}^{ADV}$, where $\mathcal{V}_{k,j}^{MDV}$ and $\mathcal{V}_{k,j}^{ADV}$ denote the MDV and the ADV, respectively.

Hereafter, the mathematical expression of $\mathcal{P}_k$–signifying the optimization problem formulated and resolved at the generic time k–is elucidated. This involves delineating the constraints that guarantee the intersection correct and secure operation, along with the objective function designed to minimize and balance both the queues lengths and the waiting times of drivers.

To maintain conciseness in the formulation, the domain of definition for the variables and constraints is explicitly stated in Section 4.13.

4.7 Constraints on TL signals

The constraints that model a correct transition in TL signals are adopted from the prior study [70], with the exception of constraints (4.5), (4.7), and (4.8), which are simplified here compared to [70].

Introducing Boolean control variables $g_{i|k,j}$, $y_{i|k,j}$, and $r_{i|k,j}$, representing green, yellow, and red signals respectively, they take a value of one if TL j displays the corresponding signal during time interval $i|k$. It is essential to note that each TL must exhibit only one signal at any given time:

$$g_{i|k,j} + y_{i|k,j} + r_{i|k,j} = 1. \tag{4.2}$$

Subsequently, it is imperative to prevent simultaneous green or yellow signals for two or more *conflicting* TLs, referring to TLs on intersecting streets, to avert potential collisions. Let $\mathcal{C} \subseteq 2^{\mathcal{J}}$ denote the set that encompasses all sets of conflicting lines, where each element of $\mathcal{C}$ represents a set of conflicting lanes. Therefore, for every $c \in \mathcal{C}$, the following condition must be satisfied:

$$\sum_{j \in c} (g_{i|k,j} + y_{i|k,j}) \leq 1. \tag{4.3}$$

The proper order of TL color transition is enforced by the following constraints:

$$r_{i+1|k,j} \leq 1 - g_{i|k,j}, \tag{4.4a}$$
$$y_{i+1|k,j} \leq 1 - r_{i|k,j}, \tag{4.4b}$$
$$g_{i+1|k,j} \leq 1 - y_{i|k,j}. \tag{4.4c}$$

For example, (4.4a) prevents the prohibited transition from a green signal in the current time to a red status in the subsequent time.

Ultimately, a constraint is essential to guarantee, for safety considerations, that a minimum duration $t_j^{y,min}$ for the yellow signal is adhered to. To address this, an auxiliary variable $t_{i|k,j}^{y}$ is introduced, signifying the duration of the yellow signal at time $i|k$ for TL j. The evolution of $t_{i|k,j}^{y}$ is expressed as follows:

$$t_{i+1|k,j}^{y} = (t_{i|k,j}^{y} + T) y_{i|k,j}. \tag{4.5}$$

Therefore, the ensuing constraint assures the adherence to the minimum duration of the yellow signal:

$$y_{i|k,j} - \frac{t_{i+1|k,j}^{y}}{t_j^{y,min}} \leq y_{i+1|k,j}. \tag{4.6}$$

Certainly, if the current signal is yellow (i.e., $y_{i|k,j} = 1$), and the duration of the yellow signal at the conclusion of time interval $i|k$ is strictly less than the minimum duration (i.e., $t_{i+1|k,j}^{y} / t_j^{y,min} \in (0, 1)$), then the left-hand side of (4.6) becomes a value between zero and one. Consequently, the variable $y_{i+1|k,j}$ is appropriately compelled to be equal to one.

To linearize the product $t_{i|k,j}^{y} y_{i|k,j}$ in (4.5), it can be replaced with an auxiliary real variable, denoted as $t_{i|k,j}^{y,\delta}$. This variable is equivalent to $t_{i|k,j}^{y} y_{i|k,j}$ when the following additional constraints are incorporated:

$$\begin{aligned} 0 \leq t_{i|k,j}^{y,\delta} &\leq M y_{i|k,j}, \\ t_{i|k,j}^{y} - (1 - y_{i|k,j}) M \leq t_{i|k,j}^{y,\delta} &\leq t_{i|k,j}^{y}. \end{aligned} \tag{4.7}$$

So (4.5) can be replaced by the linear constraint:

$$t_{i+1|k,j}^{y} = t_{i|k,j}^{y,\delta} + T y_{i|k,j}. \tag{4.8}$$

In practice, modern solvers are able to internally resolve the nonlinearity of (4.5) as explained in the above[1], so that there is no practical gain in considering the latter two equations in place of (4.5).

[1]See, e.g., the PreQLinearize functionality of the Gurobi solver: `https://www.gurobi.com/documentation/current/refman/preqlinearize.html`.

Similarly, the duration of the green and red signals could be modeled to explicitly introduce constraints on their maximum duration, ensuring a minimum guaranteed rotation of signals. However, this is not implemented in the following, allowing the algorithm to autonomously determine the TL signal timing to balance waiting times and queue lengths.

It's worth noting that the MPC strategy provides real-time information on the remaining time for the currently displayed phase, which can enhance the interaction with individuals manually driving the car.

The constraints discussed so far pertain solely to TLs. In the subsequent section, constraints governing the dynamics of vehicles and their interaction with TL signals will be addressed. Notably, certain constraints vary depending on whether the vehicle is manually or automatically driven, and these nuanced distinctions are discussed on a case-by-case basis.

4.8 Dynamics of vehicles' position

Constraints on vehicles dynamics are introduced to model the movement of vehicles and, at an aggregate level, the dynamics of the queues.

The variable $d_{i|k,v}$ represents the distance of vehicle v from the relevant TL at time $i|k$. This distance is positive when the vehicle has not reached the TL yet, zero when it is at the TL, and negative when the vehicle has passed the TL.

Furthermore, the velocity and acceleration variables for each vehicle are denoted by $v_{i|k,v}$ and $a_{i|k,v}$, respectively.

Assuming constant acceleration during the generic time interval from $i|k$ to $i+1|k$, the dynamics of velocity can be approximated as:

$$v_{i+1|k,v} = v_{i|k,v} + Ta_{i|k,v}, \tag{4.9}$$

with the initial condition:

$$v_{0|k,v} = v_{k,v}, \tag{4.10}$$

given by the measured velocity $v_{k,v}$ at k.

The evolution of the variable $d_{i|k,v}$ is determined through integration, specifically, $d_{i+1|k,v} = d_{i|k,v} - \int_{iT}^{(i+1)T} v_{i|k,v}(\tau)d\tau$. Substituting (4.9) into this equation and performing straightforward calculations yield the familiar equation of motion with uniform acceleration:

$$d_{i+1|k,v} = d_{i|k,v} - v_{i|k,v}T - \frac{1}{2}a_{i|k,v}T^2, \tag{4.11}$$

with the initial condition:

$$d_{0|k,v} = d_{k,v}, \quad i = 0, 1, ..., N-1 \tag{4.12}$$

given by the measured/estimated position at time k. The initial conditions can be transmitted by the ADV at each time step k, whereas they must be estimated or measured through sensors for MDV.

In the aforementioned equations, $d_{i|k,v}$ and $v_{i|k,v}$ function as state variables, while $a_{i|k,v}$ serves as a control variable. The position and velocity dynamics, when expressed in matrix form, establish

the following linear system:

$$\begin{bmatrix} d_{i+1|k,v} \\ v_{i+1|k,v} \end{bmatrix} = \underbrace{\begin{pmatrix} 1 & -T \\ 0 & 1 \end{pmatrix}}_{A} \begin{bmatrix} d_{i|k,v} \\ v_{i|k,v} \end{bmatrix} + \underbrace{\begin{pmatrix} -T^2/2 \\ T \end{pmatrix}}_{B} a_{i|k,v}, \tag{4.13}$$

whose well-known explicit solution is:

$$\begin{bmatrix} d_{i|k,v} \\ v_{i|k,v} \end{bmatrix} = A^i \begin{bmatrix} d_{0|k,v} \\ v_{0|k,v} \end{bmatrix} + \sum_{j=0}^{i-1} A^j B a_{i-1-j,v}, \tag{4.14}$$

with $A^j = \begin{pmatrix} 1 & -jT \\ 0 & 1 \end{pmatrix}$.

The constraints in this section hold for both MDV and ADV. However, the meaning of variable $a_{i|k,v}$ is potentially different. For ADV, the variable can be seen as a setpoint sent by the TL to the vehicle, to guide it to optimally pass the intersection. Through onboard lower-level controls then, the vehicles can track the given trajectory, while interacting in real time with the surrounding environment. For MDV instead, it represents the acceleration that the driver would typically command depending on the particular situation.

Additional constraints are included in the following to model the behaviour of vehicles in queue. These constraints vary depending on whether the vehicle is a MDV or an ADV. The main difference lays in the fact that ADV are more efficient in terms of reaction times (e.g., negligible starting delays after a TL signal change), and also have access to the near future TL signal schedule computed by the TL, so that their trajectory is efficiently adjusted accordingly, in a predictive way. MDV instead are purely reactive, and experience delays, being driven by humans. The above is discussed in the following subsections. Before that, the next section discusses two propositions that will be useful in the following.

4.8.1 Propositions on stopping sequences

The forthcoming propositions are useful to characterise the vehicles that can or cannot stop in time before reaching the intersection. These propositions will be employed to model vehicle dynamics during the yellow signal and ensure the recursive feasibility of the algorithm.

Proposition 4.8.1 (Fastest stopping sequence)**.** *The deceleration sequence which halts the vehicle in the shortest possible time and space (i.e., the "fastest stopping sequence") is:*

$$col(a_v^{min} \mathbb{1}_B, a_v^{term}) \tag{4.15}$$

where: $B = \lfloor A/T \rfloor$, $A = -v_{0|k,v}/a_v^{min}$, $a_v^{term} = -D/T$, *and* $D = v_{0|k,v} + a_v^{min} B$. *The position reached when halting (i.e., the stopping position) is given by:*

$$d_{0|k,v} - \frac{1}{2}(v_{0|k,v} B + CD). \tag{4.16}$$

Proof. The fastest stopping sequence is easily derived by analysing the velocity plot in Fig. 4.3. Letter A marks the fastest stopping time one would have by modelling the system in continuous

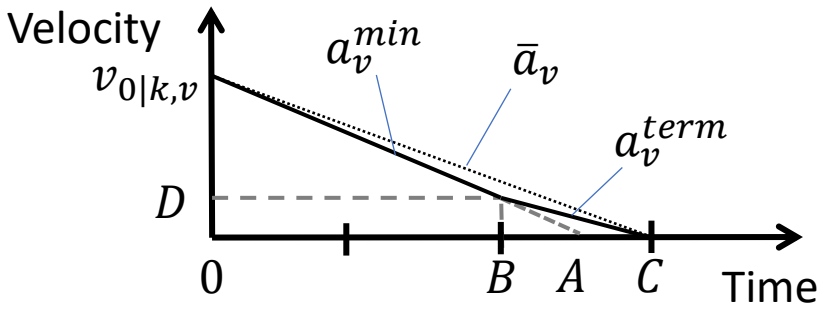


Figure 4.3: Velocity plot to derive the fastest stopping sequence.

time. It is obtained by breaking at the maximum intensity a_v^{min}, and it is given by $-v_{0|k,v}/a_v^{min}$. Letter B marks the larger sampling time that is smaller or equal than time marked by A, and it is given by $B = \lfloor A/T \rfloor$. When the system is modelled in discrete time, stopping by breaking at maximum intensity is not feasible in general, since A in general does not coincide with a sampling time, so that breaking between times B and C at maximum intensity would violate (4.19a). Hence, the fastest breaking sequence foresees to brake at maximum intensity until time B, and then to brake with intensity $a_v^{term} = -D/T$ for one last time step, where D is the velocity reached at time B, given by $D = v_{0|k,v} + a_v^{min} B$. By construction, a_v^{term} is feasible, i.e., $a_v^{term} \geq a_v^{min}$.

That (4.15) is the fastest possible stopping sequence follows again from the velocity diagram. In fact, the integral of the velocity plot represents the space travelled until stopping. The velocity profile having the smallest integral is indeed the one connecting points $(0, v_{0|k,v})$, (B, D), and $(C, 0)$.

The stopping position can be computed from (4.14) and (4.15), or from Fig. 4.3, by computing the distance travelled by the vehicle until stop, which is given by the area of the two lines connecting connecting points $(0, v_{0|k,v})$, (B, D), and $(C, 0)$, i.e., the sum of a trapezoid and a triangle. After some calculations, this is given by $(v_{0|k,v}B + CD)/2$, from which the stopping position (4.16) follows.

Finally, notice from Fig. 4.3 that breaking at the constant intensity $\bar{a}_v = -v_{0|k,v}/C$, with $C = A + 1$, stops the vehicle in the same time, but the vehicle travels more distance until stopping, since the integral of the velocity line with slope $\bar{a}_v$ is higher. This is relevant for the following. The stopping sequence with constant breaking intensity $\bar{a}_v$ is referred to in the following as the "fastest constant stopping sequence". The associated stopping position is:

$$d_{0|k,v} - v_{0|k,v} C. \tag{4.17}$$

□

Proposition 4.8.1 implies that a vehicle with the current position $p_{0|k,v}$ can come to a halt before crossing the intersection if and only if the stopping position given by (4.16) is greater than or equal to zero.

For the sake of simplicity, we will consider stopping sequences with constant deceleration instead of the fastest stopping sequence. It's important to note, as per Proposition 4.8.1, that in some cases, the fastest stopping sequence brings the vehicle to a stop before crossing, while the fastest constant stopping sequence does not. This occurrence typically arises when the vehicle is very close to the intersection with a high velocity.

Proposition 4.8.2 (Vehicles that can stop before crossing, by breaking with constant intensity)**.** *A vehicle can stop before crossing the intersection if:*

$$\exists\ \nu \in \mathbb{N}_0 : \begin{cases} p_{0|k,v} - \frac{1}{2} v_{0|k,v} \nu T \geq 0, \\ -v_{0|k,v}/(\nu T) \geq a_v^{min}. \end{cases} \tag{4.18}$$

Proof. Consider a vehicle which brakes at a constant, feasible, deceleration, until halting in νT seconds, with $\nu \in \mathbb{N}_0$. From (4.9), the constant deceleration is given by $-v_{0|k,v}/(\nu T)$. From (4.11), considering νT in place of T and $-v_{0|k,v}/(\nu T)$ as the acceleration, the position reached by the vehicle is $p_{0|k,v} - \frac{1}{2} v_{0|k,v} \nu T$. Hence the conclusion follows, where the second condition in (4.18) states that the deceleration must be feasible.

Notice that, since the system is modelled in discrete time, condition (4.18) is more elaborated than the one found in continuous time, which is simply $p_{0|k,v} + \frac{1}{2} v_{0|k,v}^2 / a_v^{min} \geq 0$, obtained by considering a vehicle breaking at the maximum intensity a_v^{min}. Notice also that this condition is only sufficient, as explained after Proposition 4.8.1 (i.e., a vehicle for which (4.18) is not satisfied, might be able to stop before crossing the intersection with the fastest stopping sequence). □

4.8.2 Constraints on velocity and acceleration

Initially, the velocity variables are constrained to be within the range of zero and a maximum permissible value. Simultaneously, the acceleration variables are restricted to fall within the range of a minimum negative value (representing the maximum possible deceleration) and a maximum positive value:

$$0 \leq v_{i|k,v} \leq v_v^{max} \tag{4.19a}$$

$$a_v^{min} \leq a_{i|k,v} \leq a_v^{max}. \tag{4.19b}$$

These limits are generally contingent on the specific vehicle and the speed limits applicable to the given street. Constraints (4.9)-(4.19) are applicable to both MDV and ADV. It is noteworthy from (4.19a) that the velocity is always greater than or equal to zero, meaning that the vehicles are not allowed to back up.

4.8.3 Constraint on position (collision avoidance)

There must be no collision between any two consecutive vehicles:

$$d_{i|k,f(v)} \geq d_{i|k,v} + l_v, \tag{4.20}$$

where $f(v)$ is the vehicle immediately following the generic vehicle v, and l_v is the length of vehicle v. Similarly, a minimum safety distance greater than l_v can be imposed.

Incorporating only (4.20) typically results in an MPC algorithm that lacks recursive feasibility. This is because (4.20) solely guarantees the absence of collisions during the prediction horizon. However, the solution computed at time k may lead to initial conditions (in terms of position and velocity) at the next time step, $k+1$, that no longer allow compliance with the collision avoidance constraint along the prediction horizon starting from $k+1$. For instance, consider a scenario where

a vehicle is rapidly approaching a queue of stationary vehicles. Relying solely on (4.20) (which does not involve velocity) could lead to a situation where, at a certain iteration, the problem becomes infeasible since a collision with the queue becomes physically unavoidable.

To address this challenge, appropriate constraints involving velocity must be added. Although the constraints differ for MDV and ADV, their underlying principle is the same: ensuring that the following vehicle maintains a safety margin to avoid collision even in the event of the leading vehicle coming to a stop. For ADV, the following constraints are introduced to guarantee the existence of a constant deceleration for the leading vehicle and a constant deceleration (potentially different) for the following vehicle, ensuring no collision occurs at the stopping position.

$$u_v, u_{f(v)} \in \mathbb{N}_0, \tag{4.21a}$$

$$p_{N|k,f(v)} - \frac{1}{2} v_{N|k,f(v)} u_v T \geq \tag{4.21b}$$

$$p_{N|k,v} - \frac{1}{2} v_{N|k,v} u_v T + l_v, \tag{4.21c}$$

$$-v_{N|k,v} \geq a_v^{min} u_v T, \tag{4.21d}$$

$$-v_{N|k,f(v)} \geq a_v^{min} u_{f(v)} T. \tag{4.21e}$$

The aforementioned constraint cannot be incorporated if one or both of the vehicles are MDV, as MDV lack information on the stopping trajectory of the other vehicle. Even if such information were available, it is not realistic to expect a human driver to utilize it. In this scenario, analogous to how human drivers instinctively navigate, a worst-case constraint must be included to ensure collision avoidance even when the leading vehicle brakes at the maximum intensity. In this case, the maximum possible decelerations are $-v_{N|k,v}/T$ and $-v_{N|k,f(v)}/T$, respectively (as they bring the velocity to zero in one time step). Therefore, (4.20) must continue to hold even after both vehicles decelerate at the maximum intensity, i.e.:

$$d_{N|k,f(v)} - \frac{1}{2} v_{N|k,f(v)} T \geq d_{N|k,v} - \frac{1}{2} v_{N|k,v} T + l_v. \tag{4.22}$$

4.8.4 Acceleration delay

A significant factor that can adversely impact the performance of the intersection is the restarting delay of MDV halted in a queue. This delay can be modeled by enforcing that if the velocity of two vehicles in line is zero, then the acceleration of the following vehicle remains zero for the next θ seconds, where θ represents the restarting delay. To implement this, a Boolean variable $q_{i|k,v}$ is introduced, taking a value of one if and only if $v_{i|k,v} + v_{i|k,j,v+1} > 0$. This is obtained with good approximation by the following constraint (cf. (4e) in [17]):

$$v_{i|k,v} + v_{i|k,v+1} \leq (v_v^{max} + v_{v+1}^{max}) q_{i|k,v}, \tag{4.23a}$$

$$\epsilon q_{i|k,v} \leq v_{i|k,v} + v_{i|k,v+1}, \tag{4.23b}$$

in which ϵ as a small tolerance. Notice that the second equation forces $q_{i|k,v} = 0$ also when $(v_v^{max} + v_{v+1}^{max}) \in [0, \epsilon)$, which is acceptable in practice if ϵ is taken small.

Then, when two generic consecutive vehicles v and $f(v)$ are halted at time $i|k$ (i.e., $q_{i|k,v} = 0$),

the acceleration of vehicle $f(v)$ is constrained to zero for the next θ seconds:

$$a_{f(v)}^{min} q_{i|k,v} \leq a_{l|k,f(v)} \leq a_{f(v)}^{max} q_{i|k,v}, \tag{4.24}$$

for $h = i, ..., \min\{N, \lceil \frac{\theta}{T} \rceil\}$.

Constraints (4.23) and (4.24) are present only for MDV.

4.8.5 Dynamics with yellow and red sign

Unless it is not feasible (i.e., the vehicles are too fast and too close to the intersection), vehicles must come to a stop with the yellow signal. This is the case when there exists a deceleration sequence capable of reducing the velocity to zero before crossing the intersection.

In light of Proposition 4.8.2, vehicles that do not satisfy (4.18) will be permitted to pass the intersection despite the yellow signal, as these vehicles are physically unable to stop in time. It is noteworthy that, to avoid overly abrupt stops, analogous to human drivers in practice, the second condition in (4.18) could be adjusted to allow for a more moderate maximum deceleration.

For vehicles that do satisfy (4.18) at time k, they must not proceed through the intersection with the yellow signal. This can be achieved by incorporating (4.18) as a terminal constraint at time $N|k$. This ensures that the vehicle remains capable of stopping at the subsequent time step, $k+1$. This constraint is applicable only if the Traffic Light (TL) is currently displaying a yellow signal. Therefore, it can be expressed as:

$$\nu_v \in \mathbb{N}_0, \tag{4.25a}$$

$$p_{N|k,v} - \frac{1}{2} v_{N|k,v} \nu_v T \geq -M(1 - y_{0|k,j}), \tag{4.25b}$$

$$-v_{N|k,v}/(\nu_v T) + M(1 - y_{0|k,j}) \geq a_v^{min}. \tag{4.25c}$$

From the above, the applicable sufficient stopping conditions at $N|k$ are enforced when $y_{0|k,j} = 1$, while they become irrelevant when $y_{0|k,j} = 0$. Constraints (4.25) are valid for both MDV and ADV. It is worth noting that, in contrast to constraint (4.22) addressing collision avoidance, there is no need to introduce a different or worst-case constraint for MDV here, as the driver is aware of and fixed on the position of the TL.

Concerning the dynamics during the red signal, vehicles are simply prohibited from passing through the intersection. Hence, if a vehicle is in the queue at a given time (i.e., $\delta_{i|k,v}^d = 1$) and the signal is red at that time, the vehicle must remain in the queue at the subsequent time:

$$\delta_{i|k,v}^d + r_{i|k,j,v} - 1 \leq \delta_{i+1|k,v}^d. \tag{4.26}$$

The aforementioned constraint is applicable to both MDV and ADV. Similar to (4.21) and (4.22), (4.25) also plays a crucial role in ensuring the recursive feasibility of the algorithm. It's important to note that introducing (4.26) without (4.25) would render the MPC iteration not recursively feasible. In one iteration, a vehicle could have a velocity too high to make it physically possible for it to stop when the signal switches from yellow to red, leading to infeasibility of the MPC iteration due to the violation of (4.26).

4.9 Queues' Length

The count of vehicles in the queues at any given time is equivalent to the number of vehicles with a non-negative distance from the TL (vehicles with a zero distance are still considered in the queue).

For each vehicle, an auxiliary Boolean indicator variable, $\delta^d_{i|k,v}$, is introduced, constrained to be equal to one when the position of the vehicle is positive or equal to zero, i.e., $[d_{i|k,v} \geq 0] \iff [\delta^d_{i|k,v} = 1]$. Following (4e) in [17], this logical double implication is practically equivalent to the following constraints:

$$m(1 - \delta^d_{i|k,v}) \leq d_{i|k,v} \tag{4.27a}$$

$$d_{i|k,v} \leq \epsilon(\delta^d_{i|k,v} - 1) + M\delta^d_{i|k,v}, \tag{4.27b}$$

where M and m are any upper and lower bound of $d_{i|k,v}$. Then, the number of vehicles in queue at $i|k$ is given by:

$$n_{i|k,j} = \sum_v \delta^d_{i|k,v}. \tag{4.28}$$

The above constraints are the same for both MDV and ADV.

4.10 Vehicles' Waiting Time

For every vehicle approaching the intersection, the waiting time initiates at zero and increments by T seconds at each iteration, ultimately returning to zero once the vehicle traverses the intersection (i.e., when $\delta^d_{i|k,v} = 0$):

$$w_{i+1|k,v} = (w_{i|k,v} + T)\delta^d_{i+1|k,v}. \tag{4.29}$$

Analogous to the approach taken in (4.5), the product $w_{i|k,v}\delta^d_{i+1|k,v}$ can be exactly linearized by substituting it with an auxiliary real variable, denoted as $w^\delta_{i|k,v}$. This variable is identical to $w_{i|k,v}\delta^d_{i+1|k,v}$ when the following additional constraints are introduced:

$$\begin{aligned} 0 \leq w^\delta_{i|k,v} &\leq M\delta^d_{i+1|k,v}, \\ w_{i|k,v} - (1 - \delta^d_{i+1|k,v})M \leq w^\delta_{i|k,v} &\leq w_{i|k,v}. \end{aligned} \tag{4.30}$$

So (4.29) can be replaced by the linear constraint

$$w_{i+1|k,j,v} = w^\delta_{i|k,v} + T\delta^d_{i+1|k,v}. \tag{4.31}$$

The above constraints are the same for both MDV and ADV.

4.11 Objective Function

One of the key advantages of MPC lies in its flexibility to customize the objective function according to specific user requirements and goals.

From the point of view of the drivers/passengers, the following objective function formulation is designed to minimize and balance the waiting times of passengers. Specifically, a linear formulation is presented, aiming to minimize the H-infinity norm of the waiting times. Additionally, a second

term is introduced to encourage vehicles to approach the TLs by maximizing their velocity. The overall objective function to minimize is:

$$J = \sum_{i=0}^{N} \left(\alpha_1 w_i^{\infty} - \alpha_2 \sum_{v \in V_{i,j}} \frac{1}{i} v_{i|k,v} \right), \tag{4.32}$$

where $\alpha_1, \alpha_2 \geq 0$, and

$$w_i^{\infty} \geq w_{i|k,v}. \tag{4.33}$$

Further terms can be incorporated into (4.32) to select solutions with additional desirable properties among the optimal solutions or those close to optimality. For example, additional straightforward terms could be introduced to:

1. Balance also the queue lengths;
2. Smooth accelerations (for user comfort);
3. Favour, among the solutions at k, those that are closer to the ones found at $k-1$;
4. To minimize the switching of TL signals;
5. To differently prioritize the regulation to zero of a given queue with respect to others, or to prioritize given vehicles with respect to others.

4.12 Complete Formulation

The complete MPC iteration is:

$$\begin{aligned} \min \quad & (4.32), \\ \text{subject to:} \quad & (4.2) - (4.6), (4.9) - (4.12), \\ & (4.19) - (4.29), (4.33). \end{aligned} \tag{4.34}$$

The above one is a mixed-integer quadratic programming problem which, as discussed after (4.6), can be converted into an equivalent MILP problem. Efficient solvers exist nowadays to solve such problems (see, e.g., [112], [105]).

4.13 Domain of definition and number of variables and constraints

The domain of definition for variables and constraints is outlined in Table 4.2.

Table 4.2: Domain of definition of variables and constraints.

Variables	**Domain of definition**
$g_{i\|k,j}$, $y_{i\|k,j}$, $r_{i\|k,j}$	$\forall i \in \mathcal{I}_{N-1}, j \in \mathcal{J}$
$n_{i\|k,j}$, $t^y_{i\|k,j}$	$\forall i \in \mathcal{I}_N, j \in \mathcal{J}$
$q_{i\|k,v}$	$\forall i \in \mathcal{I}_{N-1}, v \in \mathcal{V}_{k,j}$
$\delta_{i\|k,v}$, $d_{i\|k,v}$, $v_{i\|k,v}$, $w_{i\|k,v}$	$\forall i \in \mathcal{I}_N, v \in \mathcal{V}_{k,j}$
Constraints	**Domain of definition**
(4.2), (4.3)	$\forall i \in \mathcal{I}_{N-1}$
(4.4), (4.5), (4.6)	$\forall i \in \mathcal{I}_{N-2}, j \in \mathcal{J}$
(4.9), (4.11), (4.26), (4.29)	$\forall i \in \mathcal{I}_{N-2}, v \in \mathcal{V}_{k,j}$
(4.19), (4.20), (4.23), (4.27), (4.33), (4.33)	$i \in \mathcal{I}_{N-1}, v \in \mathcal{V}_{k,j}$
(4.21), (4.22), (4.25)	$\forall v \in \mathcal{V}_{k,j}$
(4.28)	$\forall i \in \mathcal{I}_{N-1}, j \in \mathcal{J}$

Based on the table, in each MPC iteration, there are $3N|\mathcal{J}|$ Boolean variables and $2(N+1)|\mathcal{J}|$ real variables associated with the TLs, while there are $(2N+1)|\mathcal{V}_k|$ Boolean variables and $3(N+1)|\mathcal{V}_k|$ real variables associated with the vehicles. There are a total of $(2N-2)(|\mathcal{V}_k|+|\mathcal{J}|)$ constraints. In conclusion, since $|\mathcal{V}_k|$ is usually much larger than $|\mathcal{J}|$, the spatial complexity of the problem is proportional to the product $N|\mathcal{V}_k|$.

4.13.1 Recursive Feasibility

An MPC algorithm is considered recursively feasible if and only if, when the MPC iteration is feasible at a given time, then all the subsequent iterations are guaranteed to be feasible. Recursive feasibility is a crucial property to ensure that a proper control can always be computed, contributing to a safe implementation of the algorithm. To establish recursive feasibility, it is sufficient to demonstrate that if the MPC iteration is feasible at a particular time k, then it is possible to find a feasible control sequence at $k+1$. A common practical strategy for achieving this (see, for example, [65, Section 2.5]) involves attempting to construct a feasible solution at $k+1$ by using the tail of the solution at k (i.e., the solution at k with all the variables referring to time $0|k$ removed) and adding appropriate terminal controls to complete the full control sequence at $k+1$. For the problem at hand, it can be observed from the two propositions in Section 4.8.1 that such a strategy works, for instance, by considering the terminal control (i.e., at time $N-1|k+1$) for vehicles that involves braking at maximum intensity.

4.14 Simulations

In this section, numerical simulations are presented to validate the proposed approach on a real intersection found near Rome, Italy.

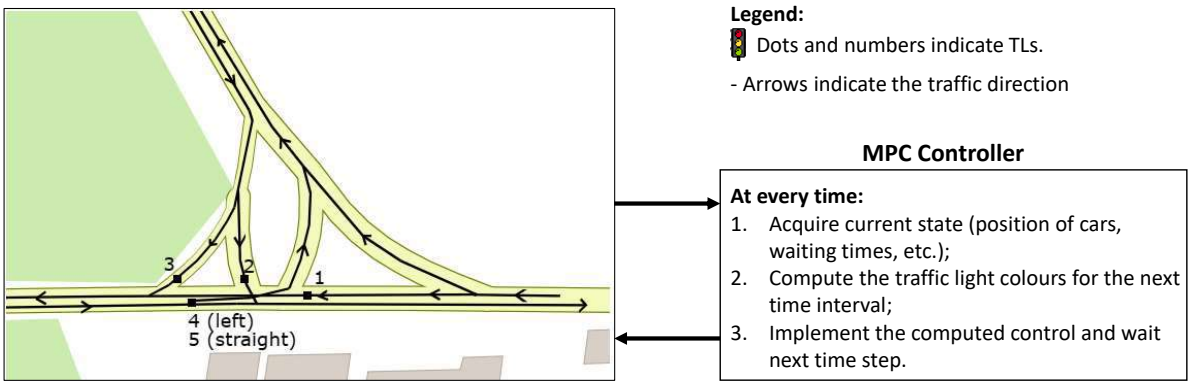


Figure 4.4: Intersection considered in the simulations.

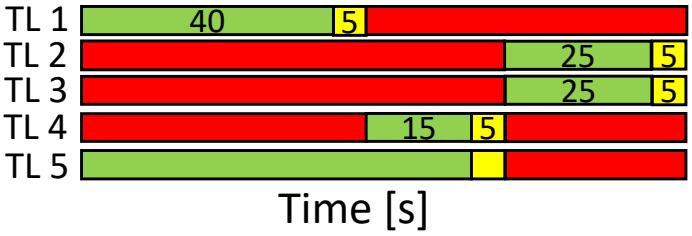


Figure 4.5: Currently implemented fixed signal phases. The number in the box is the duration of the phase in seconds.

4.14.1 Validation Tools and Implementation Details

Simulations have been performed using the Julia programming language (`https://julialang.org/`), version 1.9.3 [20]. The quadratic mixed-integer optimization problem constituting the MPC iteration has been modelled using the Julia library JuMP [39] and solved with the Gurobi solver [52]. The simulations have been performed on a Windows 10 machine 64-bit, equipped with an Intel I7-5500U CPU, 2.40 GHz and 16 GB RAM.

4.14.2 Simulation Scenario

A real road intersection near Rome[2] , Italy, is considered (Fig. 4.4). Presently, the TLs signals adhere to a regular and unchanging cycle lasting for 95 seconds. The cycle is illustrated in Figure 4.5. The vehicles are assumed to arrive in accordance with a Poisson distribution. To keep the figures readable, twenty minutes of real time simulation are reported in the following, testing the system under different arrival rates. The first ten minutes are characterized by medium/low arrival rates, while the last ten minutes by sustained arrival rates. The corresponding arrival rates are detailed in Table 4.3, where λ_j represents the expected number of vehicles arriving at TL j every second.

Table 4.3: Car arrival rates at the TL (*vehicles/s*).

Traffic Intensity	λ_1	λ_2	λ_3	λ_4	λ_5
Minutes 1-10	0.25	0.1	0.1	0.1	0.25
Minutes 10-20	0.4	0.25	0.25	0.2	0.4

[2]The intersection can be visualized at this link: `maps.com`

A single simulation scenario has been established with respect to arriving vehicles, and various comparative experiments have been conducted, as outlined below. The simulation scenario was constructed in the following manner. Initially, the vehicles arriving at the intersection at each sampling time were determined by sampling from a Poisson distribution with arrival rates as specified in Table 4.3. In total, 1393 vehicles arrive at the TL during the twenty minutes (490 during the first slot, and the remainder during the second one). The primary simulation parameters are provided in Table 4.4.

Table 4.4: Simulation parameters.

Parameter	Value	Parameter	Value
T	2 seconds	N	5
a_v^{max}	10 m/s^2	a_v^{min}	-20 m/s^2
v_v^{max}	10 m/s	$t_j^{y,min}$	4 s
α_1	1	α_2	1
ϵ	10^{-3}	-	-

The TL controller takes charge of managing vehicles once they reach a distance of 300 m from the intersection. In the following, four simulations scenarios are presented, starting from the simulation of the currently implemented fixed-timing control strategy. Than, the proposed controller is evaluated in three distinct scenarios: the first involving only MDV, the second with solely ADV, and the third, a mixed scenario. The attained performances, encompassing waiting times, queue lengths, and other metrics, are compared and discussed in Section 4.15.3.

4.15 Scenario 1: fixed cycle with MDV (currently implemented strategy)

Figure 4.6 reports the number of vehicles at the five queues, at each sampling time. The color of each TL is displayed as well, to check for consistency.

Figure 4.7 illustrates the maximum waiting times at the five queues for each sampling time (decreasing as vehicles exit the intersection). The two figures clearly demonstrate that the fixed timing strategy proves highly inefficient in balancing the queues, particularly during congested hours.

4.15.1 Scenario 2: Proposed MPC controller with MDV

The queue length and the maximum waiting times for the five queues are plotted in Figs. 4.8 and 4.9, respectively. In comparison with Scenario 1, a notable enhancement in performance is observed, attributed to the capability of the proposed approach to adjust the TL signal timing based on the prevailing short-term traffic conditions.

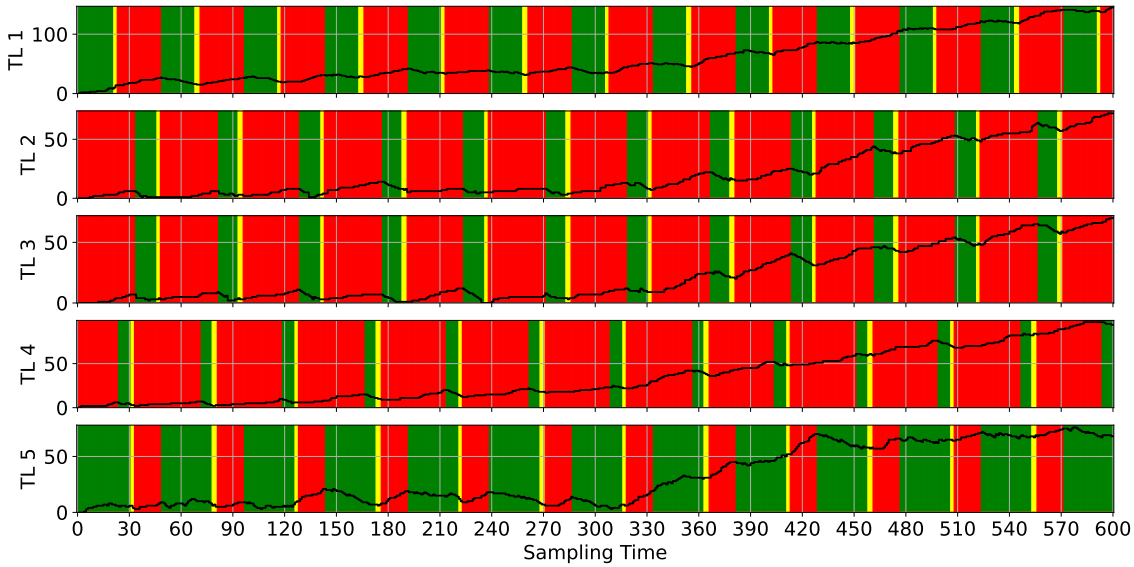


Figure 4.6: Scenario 1: fixed cycle, manual driving.

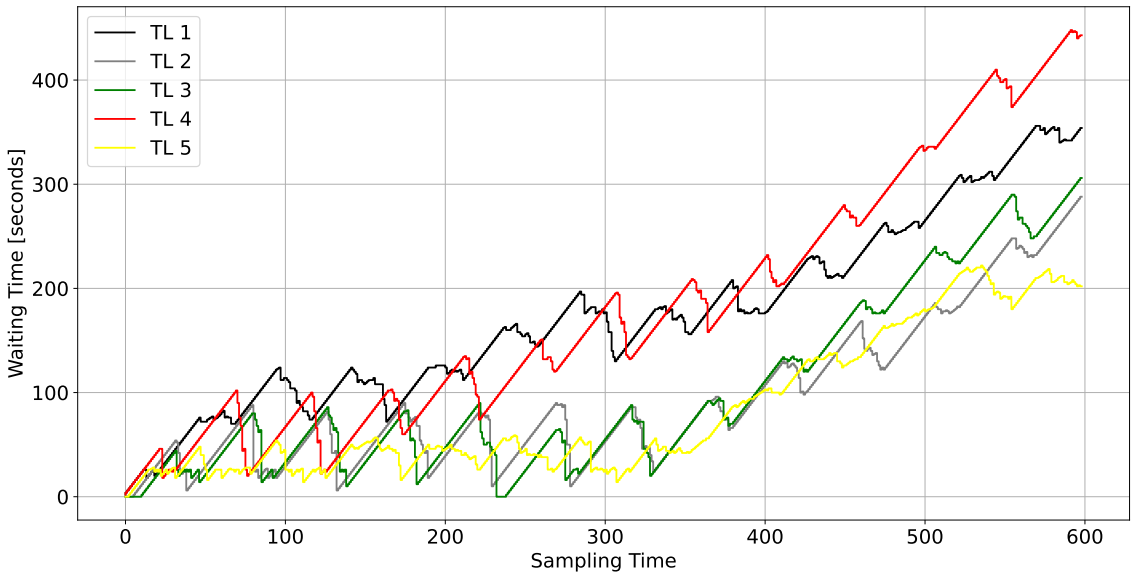


Figure 4.7: Scenario 1: maximum waiting time at each TL.

4.15.2 Scenario 3: Proposed MPC controller with ADV

In the scenario where only ADV are present at the intersection, the performance sees a further improvement. This is attributed to the fact that ADV do not encounter significant delays in their dynamics due to human intervention. Additionally, they can communicate with the controller, gaining knowledge of the near-future TL signal timing, thereby enabling a more fluid and predictive interaction.

This observation is reinforced by the analysis of the results in terms of queue lengths and maximum queue waiting times (depicted in Figs. 4.10 and 4.11, respectively). Concerning waiting times, they closely approach the minimum possible value of 25 seconds, considering that vehicles are tracked by the controller starting from 300 meters from the intersection, and their maximum velocity is assumed to be 12 m/s. Lower values in Fig. 4.11 indicate that the leading vehicle of the queue is still not close to the intersection.

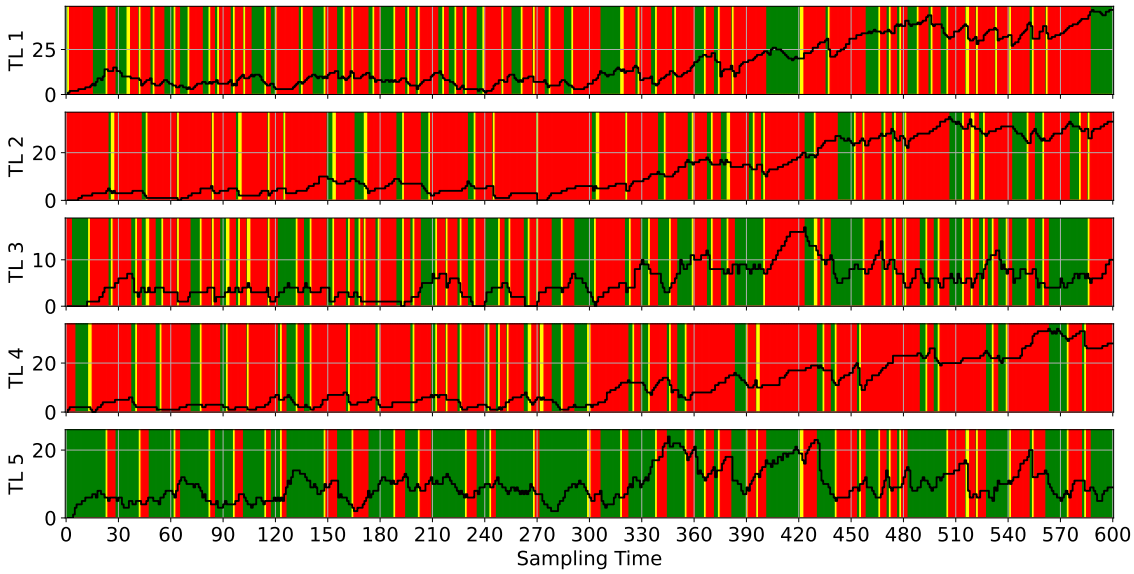


Figure 4.8: Scenario 2: proposed controller, manual driving.

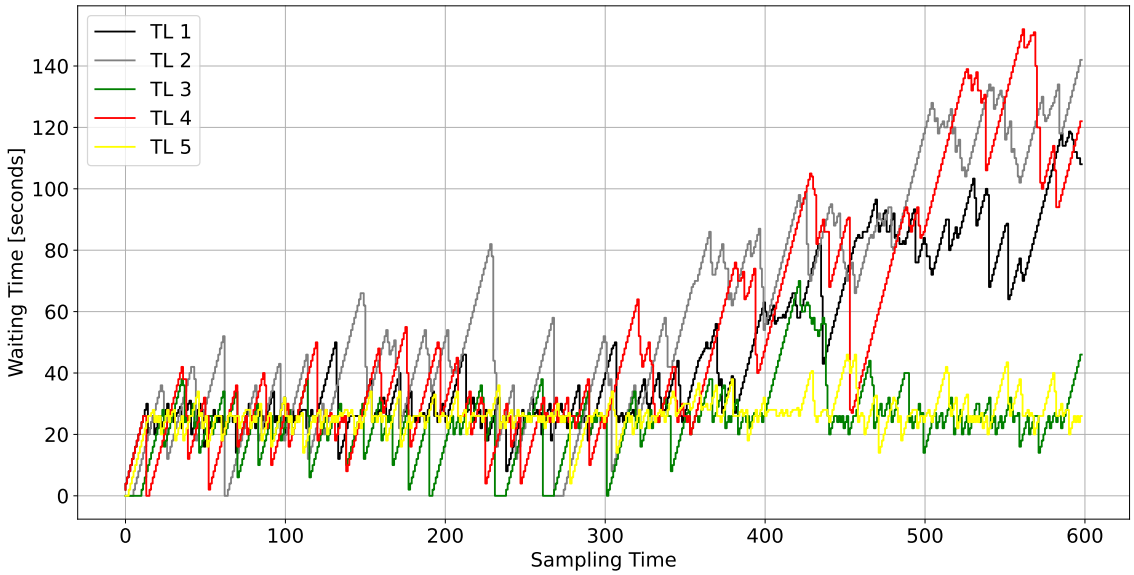


Figure 4.9: Scenario 2: maximum waiting time at each TL.

4.15.3 Scenario 4: Proposed MPC controller with MDV and ADV

In the fourth scenario, both MDV and ADV coexist. Table 4.5 provides a comparison of the performance observed in the four scenarios, focusing on waiting time and queue length. In Scenario 4, increasing shares of ADVs are also considered. As expected, as the share of ADVs increases the traffic becomes more and more fluid.

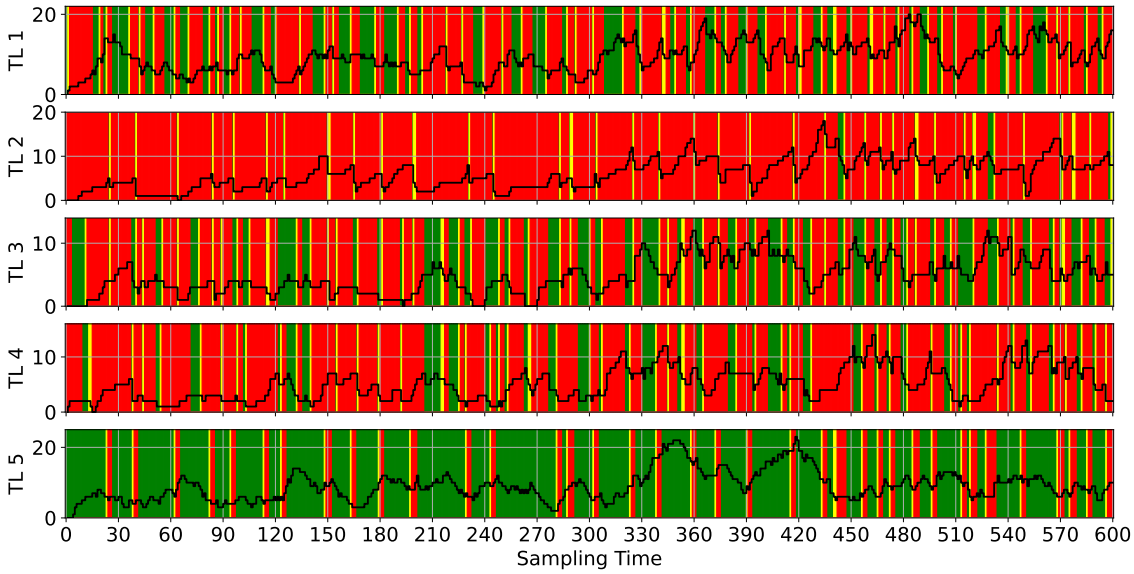


Figure 4.10: Scenario 3: proposed controller, self-driving vehicles.

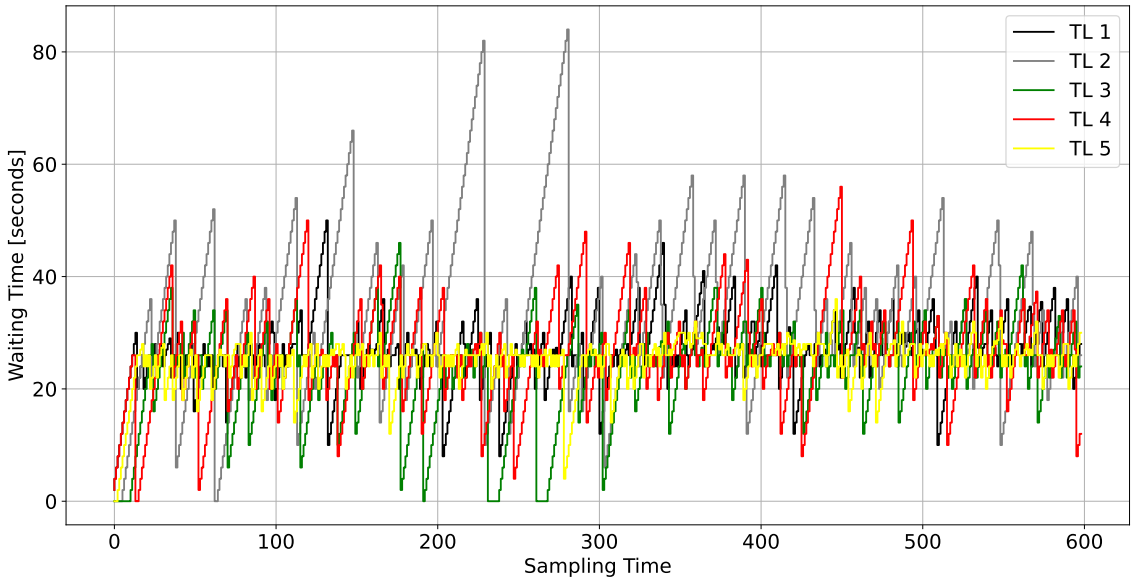


Figure 4.11: Scenario 3: maximum waiting time at each TL.

Table 4.5: Performance comparison.

Scenario	Average Waiting Time [s]	Average Vehicles at Intersection
Scenario 1	143.7	166
Scenario 2 (100% MDV)	47.2	54.9
Scenario 3 (100% ADV)	30	34
Scenario 4 (50% ADV)	31.2	36
Scenario 4 (30% ADV)	32	38
Scenario 4 (10% ADV)	37.6	43

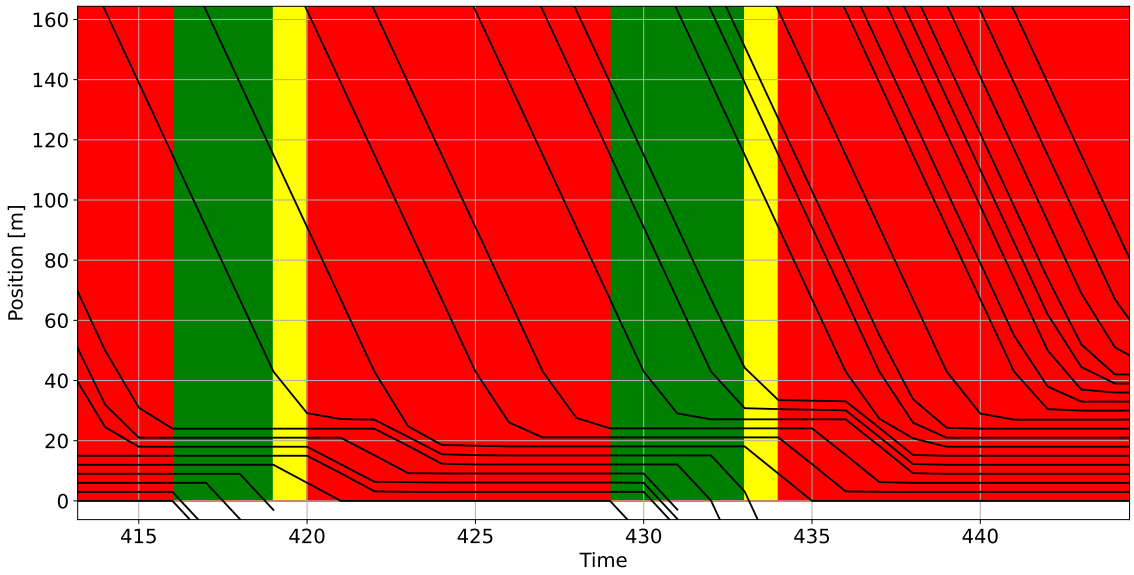


Figure 4.12: Scenario 4: detail of vehicles trajectories.

For comprehensive insight, Fig. 4.12 provides a detailed view of the trajectories of the vehicles at TL2. This example serves to illustrate how vehicles interact with the TL and how the entire queue evolves. From the figure, it is evident that the vehicles adhere to the constraints (distance, TL signal, etc.). Most notably, it is evident the difference of the dynamics of MDVs and ADVs. In fact, ADVs do not encounter the restarting delays that impact MDVs (as observed around time 430, where some vehicles are affected by delays, and others are not). Additionally, during the yellow phase after time 430, one vehicle passes with the yellow signal (as it cannot physically stop), while the subsequent vehicles correctly stop with the yellow signal.

4.16 Computational Complexity

Figure 4.13 presents the total computing time, along with the number of variables and the number of constraints characterizing each MPC iteration for Scenario 1, which is the most computationally demanding. As observed, the computing times show promise for a real implementation of the algorithm, especially considering that they are achieved with a standard personal computer. Computational efficiency can be improved through a variety of means, technical and procedural way. For instance, the tail of the solution obtained at a given time can be used to build a feasible solution to warm start the solver at the next time. This can significantly speed up the computation, since it immediately provides the solver with a feasible solution to improve. A second simple strategy is that the entry of vehicles into the mathematical model is restricted to only the first few in each queue, thus the complexity as a function of the number of vehicles taken into account in each queue is directly affected. Finally, a promising future research foresees decentralising the computation to the TLs, or even to the vehicles, by leveraging recent algorithms focusing on decentralised solution of MILPs (see, e.g., [79]).

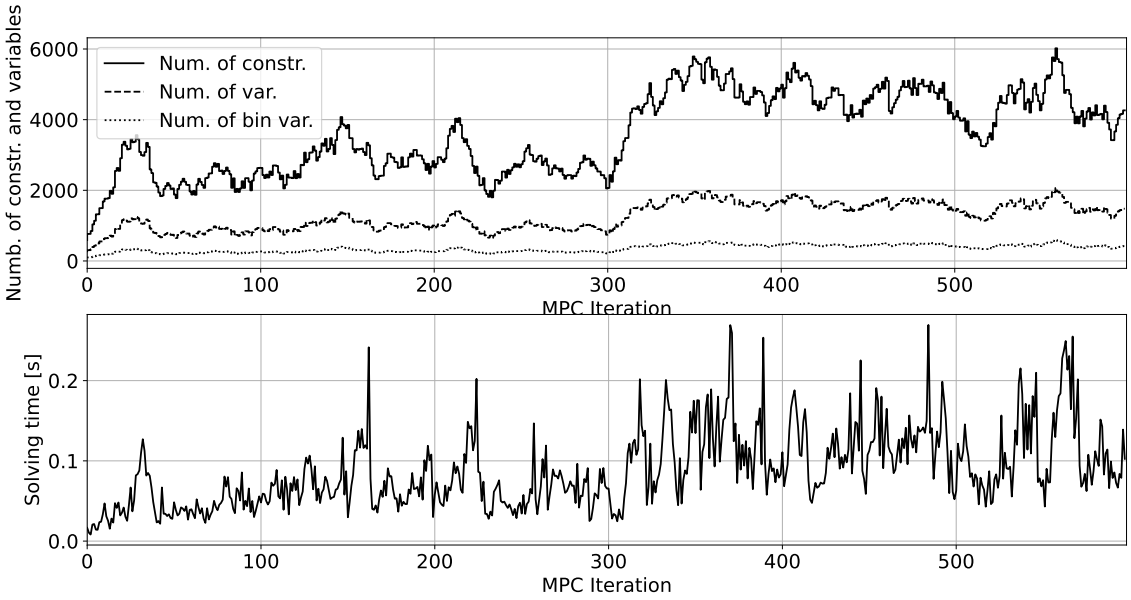


Figure 4.13: Top subplot: number of constraints, total variables, and binary variables at each MPC iterations. Bottom subplot: solving time of every MPC iteration.

4.17 Conclusions and Future Works

This paper has presented a Model Predictive Control approach to the control of the traffic lights at a single road intersection. The proposed controller incorporates a microscopic traffic model, capturing the position, velocity and acceleration of every single vehicle at the intersection. This allows to achieve a detailed modelling of the dynamics of the queues. The proposed controller can be configured to work in scenarios that go from one in which vehicles are manually controlled by the drivers, to ones in which some or all of the vehicles are automatically driven. In the former scenario, the dynamics of the vehicles' variables are intended to mimic the drivers' behaviour, in the latter ones (i.e., semi or fully autonomous driving), automated vehicles' variables are optimal references sent by the traffic light to the vehicles. Numerical simulations on a real intersection, with realistic traffic characteristics are discussed, and results in the scenarios from the manual one to the fully automated, are compared, evaluating the performance in terms of queue length and waiting times. It is shown how the proposed controller alone, and even more with the presence of automated vehicles, can significantly improve the management of the intersection, leading to less traffic.

Ongoing works are to improve the tuning and tailoring of the model, test it on a real time traffic simulator, and to develop a decentralized version of the proposed algorithm, by letting every traffic light, or every vehicle, collaborate to compute the solution. Future works regard the integration of deep learning features in the proposed MPC algorithm, by letting the deep network tune the parameters of the objective function and the model.

Chapter 5

Task Control in an Assembly Line

The fundamental goal of the presented work is that of enhancing the operational efficiency of spaceports. Translating this goal, one means, to reduce the time spent on the integration tasks in the Assembly Halls of the spaceports [10]. One flexible tool is thus required in order to deliver the best schedule for the integration tasks in real time to the spaceport operators, accounting for their availability and for the possible occurrence of unexpected situations. The daily rescheduling of these tasks is indeed a labor-intensive and is susceptible to human errors. This dissertation is focused on the launch campaign of Vega rocket, which includes: Pre-campaign phase (Vega launch site setting up); Campaign phase (movement, assembling, approval, and inspection of rocket and inter-stages); Countdown phase and Post-launch phase (e.g., flight analysis until re-validation of ground systems).

Though the presented algorithm is basically meant for management of the campaign phase, the same is open to application of the same to all the stated phases. Specifically, this is integrated into the Adaptive Operations Module (AOM), a software component responsible for the overseeing and management of the advancement of integration activities. It encompasses both static information (i.e., historical data) and dynamic inputs (e.g., live data from the operations) for offering the spaceport operators a real-time update and optimized plan of a campaign.

The Adaptive Operations Module (AOM) considers the following key factors:

- Tasks: These are the integration activities required throughout the launch campaign, defined by their duration, sequence, and various restrictions.
- Workstations: Designated areas or equipment where tasks are executed. These stations utilize and deplete certain resources to complete tasks.
- Resources: These vary based on the task at hand and may comprise human operators, tools, machinery, energy, consumables, raw materials, and more.

This work will focus on the planning and control of the various tasks and resources in an assembly of a space vehicle. The flexibility within the AOM lies in how tasks and resources are allocated to different workstations.

The content of this chapter is what was taken from the candidate newly submitted article:
[73] *Liberati, F.; Donsante, M; Cirino, C; Tortorelli, A. "Fast Task Scheduling with Model Predictive Control Integrating a Priority-based Heuristic", in IEEE Access.* Submitted.

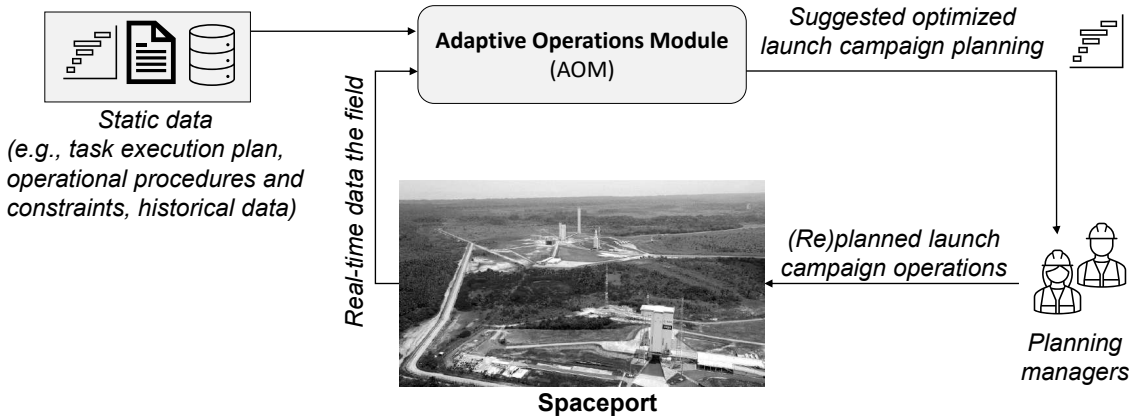


Figure 5.1: Reference architecture.

5.1 Introduction

The space industry is going through a radical change prompted by the adoption of the elements of the Fourth Industrial Revolution. For a long period in the past, the space industry was a prerogative of governmental entities due to the high costs and the skills required. Though, with the development of new technologies and business models, and with easier access to high level education, a number of new business opportunities arose and few private companies have entered in this sector. Cost reduction is one of the focal challenges the sector is looking to fight. This can be through the development of new technologies that would allow re-using the components and optimization of operations which increases the usage of resources and the rate of launches.

The outcome of the research activities carried out within the framework of the H2020 SESAME (Smart European Space Access through Modern Exploitation of data science) project, coordinated by ArianeGroup and co-financed by the European Commission under the H2020 program [33], will be introduced in this context. The project focuses on two use cases:

1. Increasing the throughput of the space vehicles' manufacturing process, by eliminating expensive and time consuming quality checks, through the adoption of advanced predictive maintenance and predictive quality algorithms;

2. Increasing the launch throughput at the Guiana space center in Kourou, by improving the scheduling of space vehicles' assembly operations at the spaceport.

This paper will focus on the second problem and derive a scalable MPC algorithm for the planning and control of the various tasks and resources in an assembly of a space vehicle. The MPC integrates a MILP model of the assembly process.

As reported in the architecture diagram (Figure 5.1), the algorithm–hosted in the "Adaptive Operations Module (AOM)"–takes as input not only static data (e.g., list of planned activities, operational procedures and constraints, historical data) but also dynamic ones (real-time data coming from workstations, operators, etc.) and provides in real time an updated and optimized campaign planning. The variables considered in the optimization process are tasks, workstations and resources. Tasks are the integration activities necessary for the launch campaign, defined by their time requirements, logical, and physical constraints, such as duration, maximum start and end times, and order of operations. Workstations are designated areas or machines where these tasks

take place, utilizing or altering resources such as tools, consumables, components, and vehicles. The decision-making process entails allocating tasks and resources to suitable workstations. The control decisions are given by the assignment of tasks and resources to the workstations. In the next sections, relevant papers from the state of the art are discussed, and then the distinctive features of the present work are highlighted.

5.2 Literature Review

Task scheduling and control is an extremely significant challenge in the industrial realm that has been the subject of much attention over the years. The essence of this area focuses on the attainment of optimum compromises between several objectives, such as the minimization of the total processing time (the makespan), the optimization of workloads in the assembly line, the minimization of costs, and/or the minimization of resources' movements in the factory. In the literature, these types of problems are referred to as Assembly Line Balancing Problem (ALBP), and many instances have been identified and addressed. ALBP have been classified based on: the shape of the assembly line (also known as the factory layout), the characteristics of the products to be processed, the type of constraints involved and the optimization goals [3, 29, 31, 36, 82, 89, 122].

In addressing the ALBP, both exact methods, which aim to find the optimal solution for a given ALBP instance (for example [35, 51, 80]), and heuristic approaches, which seek sub-optimal solutions (e.g. [6, 7, 127]), have been proposed in the literature. Recently, also Machine Learning (ML) approaches, such as [38, 109, 129] and our recent work [117], have been discussed in the literature. While exact methods are known for their potential scalability issues in practical applications, heuristic and ML approaches may not always assure the discovery of the optimal solution. The Model Predictive Control (MPC) strategy introduced in this paper is aligned with the first category and one of the main distinctive features of the presented approach is its scalability.

Model Predictive Control stands out among the exact methods for optimizing industrial processes, primarily because of its adaptable modeling capabilities and the ability to directly integrate constraints and Key Performance Indicators (KPIs) into the problem formulation [124], making it a favored choice in the context of Industry 4.0 advancements.

Surprisingly, MPC has not been extensively adopted in the context of ALBPs, although some interesting applications can be found. For instance, Subramanian et al. present a cooperative MPC algorithm for optimizing task scheduling and inventory management in supply chains [114], where under specific conditions, the presented cooperative MPC algorithm is able to achieve the same performance of a centralized sub-optimal MPC. The key contribution of the mentioned work lies in the development of a scalable distributed MPC algorithm that ensures the closed-loop stability of the controlled process. The MPC algorithm described in the present paper addresses and solves scalability issues as well, but withouth resorting to the theory of distributed optimization. In our work, we employ a more detailed model and incorporate several advanced features not found in [114]. These enhancements include the inclusion of precedence constraints among tasks, of environmental constraints, and the possibility of interrupting activities, e.g., due to failures or maintenance of machines. In [111], Sprodowski et al. present an economic MPC problem for minimizing the inventory expenses in a hospital. Although the presented formulation allows to consider workstations as multi-agent systems, the proposed control architecture is centralized. Indeed, the authors suggest

that further effort should be spent to develop a distributed framework, to improve scalability.

Perea-López et al. tackle the problem of inventory control in supply chains [93]. Although the application is not exactly the same as the one considered in this article, the presented solution has several interesting features. The authors describe a semi-decentralized robust MPC framework able to address errors in the predicted demand and discuss how a properly designed and tuned MPC algorithm is able to provide high quality solutions in highly stochastic manufacturing systems. In [131], Zhang et al. tackle the dynamic task scheduling problem exploiting a digital twin of the plant. Digital twin-based approaches allow to integrate real and simulated data implementing disturbance detection mechanisms, performance evaluation algorithms and machine availability predictions. The authors demonstrated that this digital twin-based strategy can decrease the total processing time (makespan) and increase the average utility rate. Another objective of [131] is to minimize changes in the computed schedules from one iteration to another, which is also be discussed in this article. Compared to [131], we removed the assumptions about the absence of precedence constraints and parallel task execution. Furthermore, the approach presented in this article considers not only the task assignment problem but also the resource constraints.

A key feature of MPC-based approaches consists in the possibility of exploiting feedbacks received from the field between two different iterations, in order to mitigate the effect of disturbances. This capability makes existing literature on machine availability prediction, disturbance detection, and performance evaluation highly relevant to the approach discussed in this article. Long et al., in their work [76], explore these aspects within the context of an aircraft assembly line, and compare the performances of several predictive models. In [119], Wang et al. present a method to estimate deviations between the nominal completion time and the actual one. Similarly to what is present in the next sections, this approach exploits the estimated deviations to trigger a rescheduling of the activities. The activities rescheduling problem has been addressed also in the work done by Wan et al. [118] in which the authors present a context-aware dynamic scheduling model aimed at minimizing the makespan and the energy consumption of the robots. Nevertheless, the proposed scheduling algorithm, based on solving an MILP problem is not scalable to large scenario and not suitable for fast re-scheduling computation (the authors propose to address this point in their future works). The problem of tackling disturbances in a dynamic way has been addressed also by Tliba et al. [116], in which they consider an MILP model to optimize multiple objectives and a 3D model of the plant to be controlled. The latter model allows to take into account difficult elements to modelled (e.g., stochastic behaviours and constraints which are difficult or impossible to be specified in an MILP model). The integration of the MILP and 3D models allows to obtain a digital twin of the plant which can be used for scheduling and simulation activities. In contrast to the approach just described, our formulation presents a more detailed modelling of the scheduling problem (with respect to constraints resources, tasks interruption, precedence relations, etc.) while also extending it to the MPC paradigm. In [45], Fan et al. address the scheduling problem in real world scenarios. The authors point out how an MILP solution method is not feasible for large problem instances. They tackle scalability issues by means of an improved genetic algorithm. However, as discussed by the authors, a significant instance-dependent effort in tuning the parameters of the algorithm is required.

Due to the challenges posed by the scale of real-world problems, various (meta) heuristic strategies have been devised to tackle different instances of the ALBP (e.g., [4, 12, 40, 87]). Although

heuristics can provide acceptable results in a short time, in challenging applications, such as the one considered in this paper, their performance may be too penalizing for the operations.

Indeed, the ambition of the algorithm presented in this work is to combine the advantages of exact and heuristics methods, without being affected by their limitations. Motivated by this, as mentioned in the following and further detailed in Section 5.3, we will integrate in our MPC formulation a priority weight inherited from a state of the art heuristic (see Section 5.5.2).

Relevant works are also present in the literature related to the problem of scheduling task graphs on heterogeneous distributed systems. For example, in [103], Roy et al. tackle the problem of scheduling tasks in a distributed cyber-physical system characterized by heterogeneous processing elements. The proposed solution is based on an Integer Linear Programming problem and has been developed to solve the problem in real time. Scalability issues, arising from the direct connection between the number of processing elements and the number of constraints to be defined, have been dealt with by adopting a non-overlapping approach. Another interesting work in this context is the one of [104], where Roy et al. consider a distributed cyber-physical system in which tasks the same task can be executed by different processing units with different quality levels. The objective of the proposed work is to derive a real time feasible scheduling maximizing the tasks quality. This is achieved by an Integer Linear Programming-based problem complemented by two alternative heuristics aimed at reducing time overheads. Also [102] addressed the optimal scheduling of task graphs problem, in which Roy et al. consider the periodic execution of independent control tasks in real-time. The objective of this work is to minimize the dynamic energy dissipation associated to the tasks to be executed while guaranteeing that deadlines are satisfied. To address scalability challenges due to the NP-complete characteristics of the problem, the authors have broken down the process into three hierarchical stages.

Finally, related works can be found in the literature with respect to Machine Learning based approaches. In [27] for example, Chen et al. describe an hybrid approach combining MPC and a gradient descendent machine learning procedure. The latter is used to estimate system parameters: by means of said estimates the MPC controller is able to improve the performances of an industrial production scheduler. In Section 5.6, this approach will be investigated to extend the work described in the present article. Chen et al. in [28] extend their previous work and present a similar methodology considering the execution of collaborative tasks.

5.2.1 Paper Contributions

The distinguishing contributions of this paper are:

1. A detailed MILP formulation for modelling the optimal task scheduling and control problem is presented. The proposed MILP model is more detailed compared to the literature and allows to capture a number of practical and relevant constraints and requirements. Both physical and logical constraints (i.e., arising from rules, procedures to be observed during task execution, or from safety and environmental conditions) can be included, which leads to a rich and complex model. We could not find the same level of detail in the literature.

2. The paper represents one of the first MPC applications to the problem of task scheduling and control. Beyond this, the plain application of the MPC approach to the scheduling problem might be unfeasible in practice due to the huge dimension that the optimization problem soon

reaches in realistic and large scenarios. For this reason, we propose a simple way of breaking such a potentially huge problem (which we call the "monolithic" MILP problem, presented in Section 5.3.2) into a sequence of smaller MILP problems, which can be solved much faster. This idea makes the proposed scalable MPC algorithm suitable for real time applications. One important example is given by the need of re-scheduling the tasks after an adverse and unexpected event has happened, such as a failure, the sudden unavailability of a resource, delays on tasks execution, etc.

3. A priority criterion taken from one efficient state of the art heuristic scheduling method [9] is integrated into the proposed MPC formulation. Thanks to the innovation mentioned in the previous point, the obtained MPC algorithm is almost as fast as the heuristic method. In addition, our solution is superior to the heuristic in terms of scheduling performance (i.e., makespan reduction), and flexibility. Indeed, as the MPC algorithm is based on solving sequences of MILP problems, it retains the detailed description capabilities of optimization-based algorithms (i.e., it allows to include all the relevant constraints for the problem, and to consider given KPI to be optimized in the objective function). Therefore, the proposed method combines the strengths of the two approaches, while mitigating their drawbacks at the same time. Notice that, in general, other heuristics can be integrated into the proposed MPC algorithm in a similar way as shown for the particular heuristic used in this paper.

5.3 Problem Description and Modelling

5.3.1 Problem Description

Tasks are activities to be completed (like, assembly of two or more parts, performing a check, refuelling a tank, move a component from one place to another, etc.). In general, they can be executed only at specific workstations, and require specific input resources to run. Task scheduling refers to the problem of deciding when each task should be executed, on which workstation, and by using which resources. In the following, let $\mathcal{T}$, $\mathcal{W}$, and $\mathcal{R}$ denote, respectively, the set of tasks, workstations, and resources *types* in the scheduling problem.

A generic task $t \in \mathcal{T}$ is characterised by the following temporal parameters: d_t, the total duration; S_t, the earliest allowed start time; F_t, the deadline. Tasks can start only after the needed resources are allocated to them. Depending on the specific application, tasks need different resource types, including: human operators, tools, machines, energy and other consumables, raw materials, etc. Let $r^{\#}_{t,r}$ denote the amount of resource type r that task t needs to run. More generically, the function $r^{\#} : \mathcal{T} \times \mathcal{R} \to \mathbb{R}$ can be made time varying, to capture the fact that the amount of a certain type of resource needed by a task can vary during the task execution (e.g., consider the case of a time varying power consumption). The proposed MILP formulation can handle this case but, for simplicity, in this paper we assume that a fixed amount of resources must be allocated to a task before it can start.

Often, there are precedence relations to be respected among the tasks. In general, any two tasks can be in one of the following precedence relations: finish to start (F2S), start to start, finish to finish, start to finish. If a generic task t is in a F2S relation with a predecessor task t', then t' has to finish before t can start. The other relations have an analogous meaning. Precedence relations

induce a graph, in which nodes are tasks, and edges denote a precedence relation (every edge can be marked with the type of precedence relation; when no mark is present, it is understood a F2S relation). In general, the precedence graph is non connected, since there can be tasks with no precedence relation with the other tasks.

Tasks are executed on workstations, which represent specific machines or dedicated areas of the industry. $\mathcal{W}_t \subseteq \mathcal{W}$ denotes the workstations where task t can be executed. There is a maximum amount of tasks that can run in parallel on each workstation. Resources and workstations can be assigned an operative status. When a resource or a workstation is not operative (e.g., due to a fault or planned maintenance), it cannot be used.

Tasks, workstations and resources can be further characterized depending on the specific application. For instance, some tasks can be mutually exclusive, meaning that they cannot be executed at the same time (in practice it is common that some dangerous operations, like refueling, require many other operations to stop). These and other constraints are detailed in the next subsection.

5.3.2 Monolithic MILP Formulation

In this section, an MILP formulation for modelling the optimal scheduling problem is presented. The nomenclature used in this paper is reported in Table 5.1.

Symbol	Explanation
$\alpha_j \in \mathbb{R}$	j-th weight in the objective function.
$c_t \in \mathbb{R}$	Occupancy factor of task t.
ΔT	Sampling time of the MPC algorithm.
$d_t \in \mathbb{Z}$	Time left to complete task t at current time k (expressed in number of time steps).
$d_{t,i} \in \mathbb{Z}$	Predicted time left to complete task t at time $i \geq k$ (expressed in number of time steps).
$e_{t,w,i} \in \{0,1\}$	Boolean decision variable equal to one if and only if task t is executing at workstation w at time i.
$e^{and}_{t,w,i} \in \mathbb{R}$	Auxiliary variable.
$env_{t,i} \in \mathbb{R}$	Variable indicating an environmental condition at time i, relevant for task t.
$F_t \in \mathbb{Z}$	Deadline of task t (i.e., latest possible finish time).
$F \in \mathbb{Z}$	Greatest deadline among all the schedulable tasks (i.e., $F := \max_{t \in \mathcal{T}^s_k}\{F_t\}$).
$i \in \mathbb{Z}$	Index to denote the i-th time interval in the MPC prediction window.
$\mathcal{I}_t$	Time interval over which task t can be scheduled, i.e., $\mathcal{I}_t := [\max\{k, S_t\}, ..., F_t]$.
$\mathcal{I}$	Time interval from the earliest start time to the latest finish time, i.e., $\mathcal{I} := [\max\{k, S\}, ..., F]$.
$\mathcal{I}^N$	MPC prediction window, which goes from the current time to N time steps in the future, i.e., $\mathcal{I}^N := [k, k+1, ..., k+N-1]$.

$\mathcal{I}_t^N$	Time interval over which task t can be scheduled in the MPC problem, i.e., $\mathcal{I}_t^N := [\max\{k, S_t\}, ..., \min\{F_t, N\}]$.
$k \in \mathbb{Z}$	Index to denote the current time.
$M \in \mathbb{R}$	Large positive constant.
$N \in \mathbb{Z}$	Length of the MPC prediction horizon.
$o_{w,i} \in \mathbb{R}$	Occupancy level of workstation w at time i.
$o_w^{min} \in \mathbb{R}$	Minimum occupancy level of workstation w.
$o_w^{max} \in \mathbb{R}$	Maximum occupancy level of workstation w.
$\mathcal{P}_k^m$	The monolithic scheduling problem defined at time k.
$\mathcal{P}_k'$	The optimization problem embedded in the proposed MPC scheduling algorithm.
$r \in \mathbb{Z}$	Index to denote a generic resource type.
$r_{t,r}^{\#} \in \mathbb{R}$	Amount of resource of type r that task t needs to run.
$r_{r,i}^a \in \mathbb{R}$	Number of units of resource type r available in the assembly line at time i.
$s_{w,i} \in \{0, 1\}$	Operative status of workstation w at time i (1 if operative, 0 otherwise).
$S \in \mathbb{Z}$	Smallest first possible start time among all the schedulable tasks (i.e., $S := \min_{t \in \mathcal{T}_k^s}\{S_t\}$).
$S_t \in \mathbb{Z}$	First possible start time of task t.
$start_{t,i}$	Boolean variable equal to one if and only if task t is started or re-started at time i.
$t \in \mathbb{Z}$	Index to denote a generic task.
$\mathcal{T}$	Set of all possible tasks.
$\mathcal{T}_k$	Set of all tasks relevant for scheduling at k (i.e., the tasks that at time k are either completed, in execution, or waiting to be executed).
$\mathcal{T}_k^c$	Set of candidate tasks (i.e., tasks that is feasible to start individually at k).
$\mathcal{T}_k^f$	Set of tasks that at time k are finished.
$\mathcal{T}_k^r$	Set of tasks that at time k are running.
$\mathcal{T}_k^s$	Set of tasks that at time k can be scheduled or re-scheduled.
$\mathcal{T}_k^{s,i}$	Set, computed at time k, of tasks that could be potentially active at time $i \geq k$.
$\mathcal{T}_t^{succ}$	Set of tasks that are successors of task t.
$\mathcal{T}_k^w$	Set of tasks that at k are waiting to be scheduled.
V_k	Objective function at current time k.
$V_{j,k}$	j-th term of the objective function.
$w \in \mathbb{Z}$	Index to denote a generic workstation.
$\mathcal{W}$	Set of all the workstations.
$\mathcal{W}_t$	Set of workstations where task t can be executed.
$w_t \in \mathbb{R}$	Weight associated to task t.

Table 5.1: List of symbols used in the paper.

The problem is set up in discrete time: the generic current time is denoted with the index k and the sampling time is denoted with ΔT.

Denote with $\mathcal{T}_k \subseteq \mathcal{T}$ the set of tasks existing at time k (it is a subset of the set of all the possible tasks since new tasks can appear or be removed during time, for example, as a result of faults, checks, etc.). Consider further the following partition of $\mathcal{T}_k$:

- $\mathcal{T}_k^f$: the set of tasks that at time k are finished;
- $\mathcal{T}_k^r$, the set of tasks that at time k are running;
- $\mathcal{T}_k^w$: the set of tasks that at time k are waiting to be scheduled.

According to the approach presented in this section, at every time k, the set of tasks in $\mathcal{T}_k^r$ and $\mathcal{T}_k^w$ are scheduled by building and solving a monolithic MILP problem. Note that, in general, running tasks (i.e., tasks in $\mathcal{T}_k^r$) might be re-scheduled, and therefore they are included in the problem. Define for brevity $\mathcal{T}_k^s = \mathcal{T}_k^w \cup \mathcal{T}_k^r$ the set of *schedulable* tasks at time k.

The formulation of the "monolithic" problem is relevant since it allows to effectively capture the goals and the constraints of the optimal scheduling problem. However, solving it directly might not be possible in general in large-scale scenarios, since the MILP problem is of combinatorial nature. The scalable algorithm presented in the next section breaks the monolithic MILP formulation into several smaller MILP problems, which can be solved much faster.

The MILP problem defined in this section is referred to as $\mathcal{P}_k(\mathcal{T}_k^s)$, to remind the time when the problem is build and solved (the generic current time k), and the set of tasks that enter the problem. In the following, $\mathcal{P}_k$ is detailed, in terms of variables, constraints, and the objective function. The subscript indices used are: i, t, and w, to denote, respectively, a generic time instant $i \geq k$, a generic task, and a generic workstation. The following time domains are defined (they are later used in the definition of the constraints):

- $\mathcal{I}_t := [\max\{k, S_t\}, ..., F_t]$. This is the set of time intervals when the generic task t might run.
- $\mathcal{I}_t^N := [\max\{k, S_t\}, ..., \min\{F_t, N\}]$. Same as above, except for the task deadline.
- $\mathcal{I} := [\max\{k, S\}, ..., F]$, with $S := \min_{t \in \mathcal{T}_k^s}\{S_t\}$ and $F := \max_{t \in \mathcal{T}_k^s}\{F_t\}$. This is the set of time indices over which the scheduling problem is defined (i.e., from the earliest time when a task can start, which can coincide with the current time k, to the latest time when a task can finish).

Figure 5.2 reports a representation of the relevant temporal indices and intervals.

In the next subsection, the constraints and the objective function of the monolithic MILP formulation are discussed. Please refer to Table 5.1 for the meaning of the symbols.

Constraints Related to Tasks

The time left to complete a task evolves as follows:

$$d_{t,i+1} = d_{t,i} - \sum_{w \in \mathcal{W}_t} e_{t,w,i}, \quad \forall t \in \mathcal{T}_k^s, i \in \mathcal{I}_t \setminus \{F_t\} \tag{5.1.1}$$

$$d_{t,k} = d_t \quad \forall t \in \mathcal{T}_k^s. \tag{5.1.2}$$

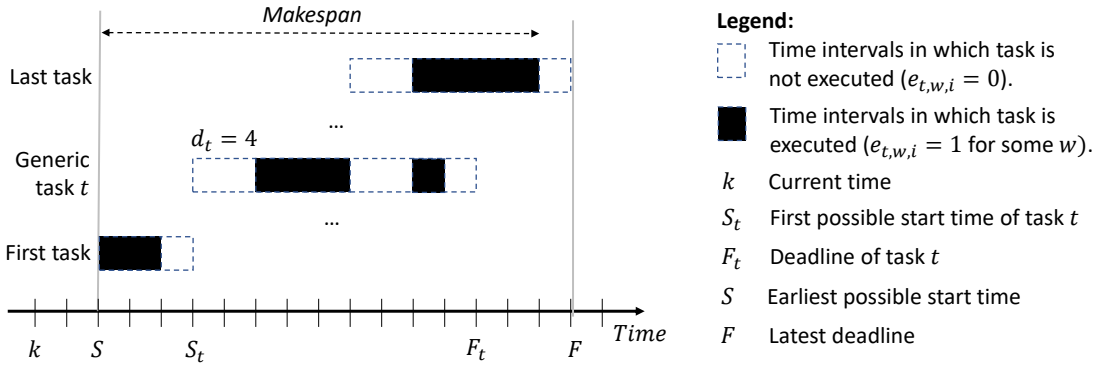


Figure 5.2: Relevant temporal task parameters.

The second equation is the initial condition. If the task is executing at time i (i.e., $\sum_{w \in \mathcal{W}_t} e_{t,w,i} = 1$, see also (5.3)), then $d_{t,i+1} = d_{t,i} - 1$. Notice that (5.1.1) allows to capture and take into account delays occurring in tasks execution: variable d_t is a feedback of the current time left to execute task t, and therefore it incorporates any accumulated delay, which will be taken into consideration for the computation of the task planning (i.e., the computed task planning might change because of delays).

Obviously, the time left to complete a task is a non-negative variable:

$$d_{t,i} \geq 0, \quad \forall t \in \mathcal{T}_k^s, i \in \mathcal{I}_t. \tag{5.2}$$

Every task can be executed only at one workstation at a time:

$$\sum_{w \in \mathcal{W}_t} e_{t,w,i} \leq 1, \quad \forall t \in \mathcal{T}_k^s, i \in \mathcal{I}_t. \tag{5.3}$$

In principle, tasks can change workstations of execution (e.g., after a fault affecting a workstation). This can be controlled with the inclusion of additional constraints and terms in the objective function, as explained next.

Regarding precedence constraints, an F2S relation between tasks t' and t (i.e., t can start only after t' has finished) can be captured by the following constraint:

$$\sum_{w \in \mathcal{W}_t} e_{t,w,i} \leq 1 - \frac{d_{t',i}}{d_{t'}}, \quad \forall t \in \mathcal{T}_k^s, i \in \mathcal{I}_t. \tag{5.4}$$

It is seen from (5.4) that, as long as $d_{t',i} > 0$, the sum $\sum_{w \in \mathcal{W}_t} e_{t,w,i}$ is forced to zero, which means that task t cannot execute. The other types of task dependency relations can be captured in a similar way. In addition, often there are also temporal constraints among tasks. A typical case is to require that the execution of any given two tasks be separated by a given amount of time. Such a constraint can be modelled in a similar way.

The constraint on the task deadline can be expressed as:

$$d_{t,F_t} = 0, \quad \forall t \in \mathcal{T}_k^s. \tag{5.5}$$

In some cases, no feasible schedule exists which satisfies all the problem constraint (for example,

in the case of large delays in tasks execution). To make sure that the algorithm always provides a solution, a proper approach can be to add delay variables, and minimize them through the inclusion of a specific objective function term (so that the solution will foresee delays only if otherwise no feasible schedule exists).

Conditions can be put as well to denote sets of tasks that are mutually exclusive. For instance, (5.6) can be added to express that the execution of tasks in a given set $\mathcal{T}_b$ is forbidden while some of the tasks in the given set $\mathcal{T}_a$ are executing (e.g., some dangerous operations).

$$\sum_{w \in \mathcal{W}_t} e_{t,w,i} \leq 1 - \frac{\sum_{t \in \mathcal{T}_a} \sum_{w \in \mathcal{W}_t} e_{t,w,i}}{|\mathcal{T}_a|}, \quad \forall t \in \mathcal{T}_b, i \in \mathcal{I}_t. \tag{5.6}$$

From (5.6) it is seen that, when one or more of the tasks in $\mathcal{T}_a$ are executing, the right hand side is lower than one (and positive), so that the sum $\sum_{w \in \mathcal{W}_t} e_{t,w,i}, \forall t \in \mathcal{T}_b$ is pushed to zero, which means that no task in $\mathcal{T}_b$ can execute.

Task preemption must be controlled, to make sure that the execution of a task is interrupted –if allowed– only when really necessary for some reason (tasks interruption comes at increased additional costs, setup times and resources, etc.). Preemption can be controlled by introducing an auxiliary variable, $start_{t,i} \in \{0, 1\}$, which is equal to one if and only if task t is started (or re-started) at time i. This definition of $start_{t,i}$ is enforced by adding the following constraint:

$$\sum_{w \in \mathcal{W}_t} (e_{t,w,i} - e_{t,w,i-1}) \leq start_{t,i}, \quad \forall t \in \mathcal{T}_k^s, i \in \mathcal{I}_t. \tag{5.7}$$

If the task is started/restarted at time i, then the start variable is forced to 1. Then, to force the variable to 0 whenever the above constraints does not apply, the term (5.15) in the target function is added. Variable $start_{t,i}$ enables fine-grained control of task preemption. For instance, a constraint on the maximum number of possible task interruptions can be included as a hard constraint (which also allows to model uninterruptible tasks, by setting the number of allowed interruptions to zero), or a term in the objective function can be added to penalize tasks interruptions, so that they will happen only when unavoidable (e.g., in case of failure of a workstation - see comments after (5.15)).

In general, depending on the specific application, the proposed algorithm has the flexibility of integrating very easily many other safety constraints, environmental constraints, constraints on working shifts, or, more in general, any other condition that must be observed before a task can be allocated or a resource assigned. This comes from the flexibility introduced by the adopted modelling framework based on mixed integer programming, which allows to introduce both physical and logical constraints. For instance, safety constraints indeed often appear in the form of logic constraints (e.g., if-then conditions), which can be easily included, case by case, by adding new variables capturing the happening of the specific logic conditions, and additional constraints to capture the effect of the condition on the control/controlled variables. A typical example of a safety constraint which can be easily included is: “if a human being is detected in the proximity of a workstation, then all the activities at the workstation must be stopped”. To provide another simple and concrete example, assume that certain tasks can be executed only when given environmental conditions hold. Denoting with $env_{t,i}$ a signal that is greater than one if and only if the given environmental condition for the execution of task t holds, and between zero and one otherwise, then

the constraint is:

$$e_{t,w,i} \leq env_{t,i}, \quad \forall t \in \mathcal{T}_k^s, w \in \mathcal{W}_t, i \in \mathcal{I}_t. \tag{5.8}$$

Constraints on Workstations

The occupancy level of workstation w at time i is defined as a function of the number of tasks currently running on the workstation:

$$o_{w,i} = \sum_{t \in \mathcal{T}_k^{s,i} : w \in \mathcal{W}_t} c_t e_{t,w,i}, \quad \forall i \in \mathcal{I}, w \in \mathcal{W}, \tag{5.9}$$

where $\mathcal{T}_k^{s,i} \subseteq \mathcal{T}_k^s$ is the set of tasks potentially active at time i, i.e., $\mathcal{T}_k^{s,i} = \{t \in \mathcal{T}_k^s : S_t \leq i \leq F_t\}$. Equation (5.9) can capture different aspects, depending on the choice of the coefficients c_t. For example, it can simply represent the number of tasks executing at the workstation (in which case, it is $c_t = 1$), or the volume occupied, or the weight (in which case c_t is an occupancy factor in terms of volume or weight, etc.). Of course, there are limits in general on the maximum and possibly minimum occupancy levels for every workstation:

$$o_w^{min} \leq o_{w,i} \leq o_w^{max}, \quad \forall i \in \mathcal{I}, w \in \mathcal{W}. \tag{5.10}$$

Finally, no task can be executed on a workstation that is unavailable:

$$\sum_{t \in \mathcal{T}_k^{s,i} : w \in \mathcal{W}_t} e_{t,w,i} \leq M s_{w,i}, \quad \forall i \in \mathcal{I}, w \in \mathcal{W}, \tag{5.11}$$

where $s_{w,i} \in \{0, 1\}$ is the operative status of workstation w at time i (0 if not operative), and M is a large positive constant (i.e., larger than the maximum number of tasks that can simultaneously execute on w). Variable $s_{w,i}$ can be used to integrate real-time information about possible faults at workstations, or even future planned maintenance activities at the workstation. In this case, the algorithm will compute an updated schedule, to take into account the fact that one or more workstations are not available.

Constraints on Resource Consumption

The task assignment must not determine a resource consumption which exceeds the resource availability.

$$\sum_{t \in \mathcal{T}_k^{s,i}} \sum_{w \in \mathcal{W}_t} e_{t,w,i} r_{t,r}^{\#} \leq r_{r,i}^a, \quad \forall i \in \mathcal{I}, r \in \mathcal{R}. \tag{5.12}$$

The left hand side of (5.12) represents the total level of consumption of resource type r at time i, which must be below the available number of units of resource type r, i.e., $r_{r,i}^a$.

Objective Function

The objective function is the convex combination of four terms, reflecting different optimization goals, i.e.,:

$$V_k = \sum_{j=1}^{4} \alpha_j V_{j,k}, \tag{5.13}$$

with $\alpha_j \geq 0$ and $\sum_j \alpha_j = 1$. The term $V_{1,k}$ pushes the minimization of the total time left to complete the tasks:

$$V_{1,k} = \sum_{t \in \mathcal{T}_k^s} \sum_{i \in \mathcal{I}_t} d_{t,i}. \tag{5.14}$$

The term $V_{2,k}$ is added to ensure that $start_{t,i} = 1$ only when forced to one by (5.7), and zero in all other cases:

$$V_{2,k} = \sum_{t \in \mathcal{T}_k^s} \sum_{i \in \mathcal{I}_t} start_{t,i}. \tag{5.15}$$

As a matter of fact, at the times i in which the first member of (5.7) is equal to zero variable $start_{t,i}$ can be either 0 or 1 according to (5.7). But thanks to the inclusion of $V_{2,k}$ at the optimum it is $start_{t,i} = 0$, since $start_{t,i} = 1$ is associated with a higher cost in terms of $V_{2,k}$. This effect is valid for every α_2. However, notice that, as the value of α_2 increases, a second substantial effect is produced, which is the minimization of tasks interruptions. If this is not desired, then α_2 should be selected very small (indeed, the minimization of tasks interruptions might be counterproductive for the objective of minimizing the total makespan).

The term $V_{3,k}$ ensures that the assignment of tasks to workstations changes only when needed (e.g., for faults)[1].

$$V_{3,k} = -\sum_{t \in \mathcal{T}_k^s} \sum_{w \in \mathcal{W}_t} \left[e_{t,w,k|k-1} e_{t,w,k} + \sum_{i=k+1}^{F_t - 1} e_{t,w,i} e_{t,w,i+1} \right], \tag{5.16}$$

Note that $V_{3,k}$ is minimized (notice the minus sign) when the assignment of tasks to workstations do not change in time (i.e., when $e_{t,w,k|k-1} = e_{t,w,k}$ and $e_{t,w,i} = e_{t,w,i+1}$), and there is no task preemption (similar terms are presented in the literature, see, e.g., (16) and (17) in [118]). The nonlinear term $e_{t,w,i} e_{t,w,i+1}$ in (5.16) is *exactly linearized* by introducing the auxiliary variable $e^{and}_{t,w,i}$ and constraints:

$$e^{and}_{t,w,i} \leq e_{t,w,i}, \tag{5.17.1}$$

$$e^{and}_{t,w,i} \leq e_{t,w,i+1}, \tag{5.17.2}$$

$$e_{t,w,i} + e_{t,w,i+1} - 1 \leq e^{and}_{t,w,i}. \tag{5.17.3}$$

With the inclusion of (5.17), $e_{t,w,i} e_{t,w,i+1}$ in (5.16) can be replaced by $e^{and}_{t,w,i}$.

Finally, term (5.18) is added to penalize unnecessary changes in the scheduling of tasks computed in the current MPC iteration at k, compared to the solution computed at the previous MPC iteration

[1] $e_{t,w,k|k-1}$ denotes the variable $e_{t,w,k}$ computed at time $k-1$, i.e., belonging to problem $\mathcal{P}_{k-1}$. This is a customary MPC notation.

at k:

$$V_{4,k} = -\sum_{t\in\mathcal{T}_k^s}\sum_{i\in\mathcal{I}_t}\sum_{w\in\mathcal{W}_t} e_{t,w,i|k-1}e_{t,w,i}. \tag{5.18}$$

Equation (5.18) is minimized (notice again the minus sign) when $e_{t,w,i|k-1} = e_{t,w,i}$, i.e., the decision about time i taken at the previous MPC iteration (i.e., at time $k-1$) is equal to $e_{t,w,i}$, i.e., the decision about time i taken at the current MPC iteration (at time k). Parameters $e_{t,w,i|k-1}$ in (5.16) and (5.18) are initialized at zero in the first iteration.

Notice the subtle difference between (5.16) and (5.18). The aim of (5.16) is to favour that, once a task is assigned to a workstation, it keeps executing on that same workstation for the entire task duration, if possible. Equation (5.18) instead is to favour that, when a re-scheduling is computed, the new task allocation deviates from the previous one only if necessary (e.g., in case of fault of a workstation).

5.4 Proposed Scalable MPC Algorithm for Task Scheduling

The proposed scalable algorithm is based on the idea of breaking the monolithic MILP formulation into smaller MILP problems solved sequentially. We took inspiration from the MPC approach, which can be interpreted as a way to approximate an infinite horizon optimal control problem, e.g., written generically as

$$\begin{aligned} \min_{\{u_1,u_2,...\}} \quad & \sum_{i=1}^{\infty} l(x_i,u_i) \\ \text{s.t.} \quad & f_i(x_i,u_i) \le 0, \quad i=1,2,... \end{aligned} \tag{5.19}$$

with a finite-horizon one, in which the objective function is the sum of the stage cost over a prediction window of N steps, and the omitted infinite-horizon tail is approximated with an additional final cost term, function of the state at the end of the prediction window:

$$\begin{aligned} \min_{\{u_1,u_2,...\}} \quad & \sum_{i=1}^{N} l(x_i,u_i) + Q(x_N) \\ \text{s.t.} \quad & f_i(x_i,u_i) \le 0, \quad i=1,2,...,N \end{aligned} \tag{5.20}$$

It is known that in some cases, if N is large enough and $Q(x_N)$ is properly selected, the two problems yield the same solution (see, e.g., [65, Section 2.3]). We adopt a similar approach, further noticing that the specific constraints present in our problem suggest a natural way to break the monolithic MILP formulation.

Focusing on the precedence constraints, suppose for instance that at a given time k, the status of the tasks is as Fig. 5.3. Obviously, due to the precedence constraints, the next tasks to be started can be selected only among the running tasks (assuming that they can be re-scheduled), the isolated tasks (i.e., tasks with no precedence constraint), and the successors of the finished tasks. Therefore, it is intuitive to consider a reduced problem, defined only over the above said "candidate" tasks. Notice that, in general, not all the candidate tasks can be started simultaneously, due to the presence of the additional constraints on resources, workstations, etc., which may limit the number of tasks that can be started at a given time. We therefore come to the following definition.

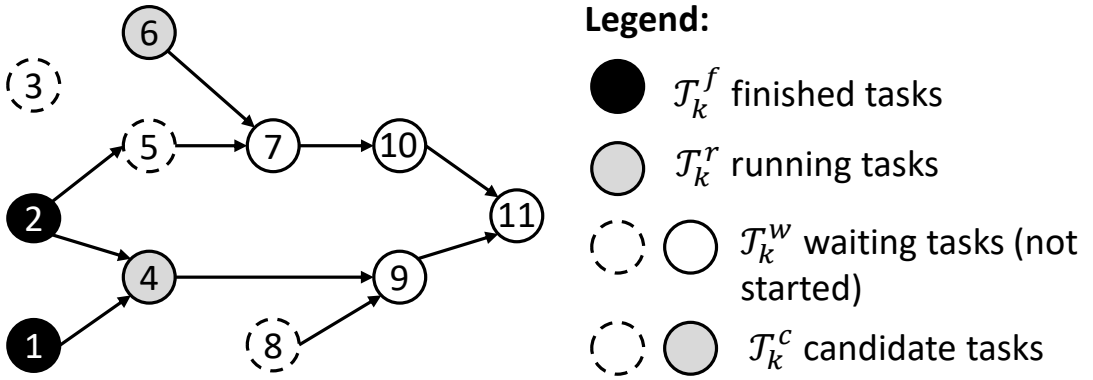


Figure 5.3: Illustrative precedence graph.

Definition 1 (Set of candidate tasks, $\mathcal{T}_k^c$)**.** *A candidate task is a task which can be started at k without violating any constraint. The set of all candidate tasks at k is denoted as $\mathcal{T}_k^c$.*

Notice from the definition that, starting at k any single task in $\mathcal{T}_k^c$ is feasible. However, starting simultaneously more than one of the tasks in $\mathcal{T}_k^c$ might be unfeasible. For this reason, a criterion is needed to decide, at any k, which of the tasks in $\mathcal{T}_k^c$ to start. Notice that it is exactly the decision of which tasks to start or not among the ones in $\mathcal{T}_k^c$ which makes the difference between a sub-optimal schedule, and a globally optimal one. In this paper, we investigate the natural strategy of taking this decision by building and solving an MILP problem like the one defined in Section 5.3, but this time only over $\mathcal{T}_k^c$ (instead of $\mathcal{T}_k^s$).

Since the sequential problems are built only over $\mathcal{T}_k^c$, in general this approach results into a sub-optimal planning. Also, the approach in theory could be unfeasible in some pathological cases (i.e., one of the sequential problems at one point might be unfeasible) even when there exists a feasible solution to the monolithic problem. This resembles the same issue found in the problem of redundancy resolutions in robots, where "local" approaches are in general sub-optimal with respect to global ones, and in some cases could be unfeasible even when a feasible global solution exists.

In practice, during the extensive testing carried out, no instance of unfeasibility was observed. To mitigate both issues (i.e., sub-optimality and unfeasibility), the objective function of the sequential problems is modified (compared to the monolithic one), by including *an additional cost term* (just like in (5.20), compared to (5.19)).

Algorithm 3 presents the pseudo-code of the proposed control strategy. At line 3, the current status of the tasks is updated, to understand what are the finished tasks (set $\mathcal{T}_k^f$), the tasks currently running (set $\mathcal{T}_k^r$), and the tasks waiting to be scheduled (set $\mathcal{T}_k^w$). If there are no more running tasks and waiting tasks (line 4), the scheduling problem is finished (line 5); otherwise, the set of candidate tasks is computed (line 7). If there are no candidate tasks, the algorithm breaks the current iteration of the for loop and goes directly to the next one (lines 8 and 9). Next (line 11), the tasks to be started among the ones in the set of candidate tasks $\mathcal{T}_k^c$ are determined by solving the optimization problem $\mathcal{P}'_k(\mathcal{T}_k^c)$, whose formulation is discussed in Section 5.4.1. Next, the current *for* iteration ends, and the next one is started, until either all tasks are scheduled, or $\mathcal{P}'_k$ is unfeasible for some k, which means that either the problem is unfeasible, or the proposed algorithm is not able to find a feasible solution.

Algorithm 3 Scalable MPC task control

1: **for** every current time $k = 1, 2, ...$ **do**
2: Acquire all the relevant feedback information from the assembly line (i.e., time left to complete the tasks, operative status of workstations, relevant environmental conditions).
3: Update the current status of tasks (i.e., compute $\mathcal{T}_k^f$, $\mathcal{T}_k^r$, and $\mathcal{T}_k^w$).
4: **if** $\mathcal{T}_k^r$ and $\mathcal{T}_k^w$ are empty **then**
5: Exit the for loop, scheduling finished.
6: **else**
7: Compute the set of candidate tasks $\mathcal{T}_k^c$.
8: **if** $\mathcal{T}_k^c$ is empty **then**
9: Skip to the next iteration of the for loop.
10: **else**
11: Build and solve the optimization problem $\mathcal{P}'_k(\mathcal{T}_k^c)$, to compute the set of tasks to start at k.
12: **if** $\mathcal{P}'_k$ is unfeasible **then**
13: Terminate the algorithm: a solution could not be found, the scheduling problem could be unfeasible.
14: **end if**
15: **end if**
16: **end if**
17: **end for**

5.4.1 Mathematical Formulation of $\mathcal{P}'_k$

Problem $\mathcal{P}'_k$, built and solved in line 11 of Algorithm 3, has almost the same formulation as $\mathcal{P}_k$ discussed in the previous sections. The only differences are:

- The problem is defined only over the set of candidate tasks at $k \geq k$ (while $\mathcal{P}_k$ is defined over all the tasks). For this reason, it is denoted as $\mathcal{P}'_k(\mathcal{T}_k^c)$;
- The objective function of $\mathcal{P}'_k$ also includes terminal costs, as detailed in the next subsection;
- Precedence constraints are not included, since the tasks in $\mathcal{T}_k^c$ satisfy them by construction (the tasks satisfy Definition 1, which means they satisfy, among the others, the precedence constraints). Notice that instead the other constraints (e.g., the one on resource consumption) cannot be removed, since starting more than one task in $\mathcal{T}_k^c$ may lead to constraint violation;
- To further reduce the dimension of the problem, a fixed time window for $\mathcal{P}'_k$ can be adopted, of length N, i.e., $\mathcal{I}^N := [k, k+1, ..., k+N-1]$ (instead of using the time horizon induced by the tasks in $\mathcal{T}_k^c$, i.e., $\mathcal{I}^c := [\max\{k, S^c\}, ..., F^c]$, with $S^c := \min_{t \in \mathcal{T}_k^c}\{S_t\}$ and $F^c := \max_{t \in \mathcal{T}_k^c}\{F_t\}$, which might be considerably longer).

5.4.2 Definition of the terminal cost in $\mathcal{P}'_k(\mathcal{T}_k^c)$

A fifth cost term is added to the objective function, to weight differently the tasks in $\mathcal{T}_k^c$:

$$V_{5,k} = \sum_{t \in \mathcal{T}_k^c} \sum_{i \in \mathcal{I}_t} w_t \sum_{w \in \mathcal{W}_t} e_{t,w,i}, \tag{5.21}$$

where the task weight w_t is defined based on the precedence graph and the time left to complete the tasks. Weights w_t are defined as follows:

$$w_t = d_t + \sum_{\tau \in \mathcal{T}_t^{succ}} d_\tau, \tag{5.22}$$

where $\mathcal{T}_t^{succ}$ is the set of successors of the considered task t. An higher weight is associated to the tasks having many successors, with long duration. Such tasks are given a higher priority because, if delayed, they are likely to cause further delays among the successors, delaying the whole planning.

5.4.3 Complexity Analysis

In this section, the space and time complexity of the proposed Algorithm 3 is discussed. As a proxy of size complexity, we consider the number of variables present in the problem $\mathcal{P}'_k(\mathcal{T}_k^c)$ at the generic time k (Algorithm 3, line 11). The variables of the problem are $e_{t,w,k}$ (Boolean), $d_{t,k}$ (real), and $w_{w,k}$ (real). The number of variables $e_{t,w,k}$ present in problem $\mathcal{P}'_k(\mathcal{T}_k^c)$ is $\sum_{t \in \mathcal{T}_k^c} |\mathcal{W}_t||\mathcal{I}_t|$. An upper bound for this number is: $|\mathcal{T}||\mathcal{W}|N$ (this bound is reached in the worst case, which happens when all the tasks are schedulable at k, every task can be run on every workstation, and every task can be run at every time in the MPC control window). Similarly, the number of variables $d_{t,k}$ present at time k is $\sum_{t \in \mathcal{T}_k^c} |\mathcal{I}_t|$ which, in the worst case is $|\mathcal{T}|N$. Finally, the number of variables $o_{w,i}$ present at time i is $|\mathcal{W}|N$.

Therefore, in the worst case, the total number of variables in the problem at the generic MPC iteration is given by:

$$N(|\mathcal{W}| + |\mathcal{T}| + |\mathcal{W}||\mathcal{T}|). \tag{5.23}$$

In practice, the number of variables is much lower, for the following reasons:

- Due to the precedence relations among the tasks, only a fraction of the total tasks are schedulable at the generic time k. That is, $\mathcal{T}_k^c$ is a small subset of $\mathcal{T}$.
- Tasks in general can be run only at specific workstations, i.e., $\mathcal{W}_t$ is a small subset of $\mathcal{W}$.
- The task might not be active for the entire time window.

As far as it concerns the time complexity, the most complex operation of Algorithm 3 is the solution of $\mathcal{P}'_k$ (Algorithm 3, line 11; all the other operations in Algorithm 3 have a time complexity linear in the size of the set of tasks, workstations, and resources). The solution of $\mathcal{P}'_k$ (which is a MILP problem) has a time complexity which is influenced by many factors. Empirically, one of the most important factors is the number of variables present in the problem (which has been discussed in the paragraphs above). As explained, one of the merits of the proposed Algorithm is exactly that of keeping limited the number of variables (compared to monolithic optimization, or standard MPC algorithms), thus reducing the computation time. As a matter of fact, the simulations presented next confirm that the proposed approach can handle the complexity of real-world task control problems.

5.5 Simulations

Numerical tests are presented to validate the proposed approach in different scenarios. The proposed algorithm (Section 5.4) is compared against a state of the art priority-based heuristic [9], the one that has been integrated into the proposed scalable MPC scheme through the definition of the terminal cost (5.21) (notice that, in general, any other heuristic can be integrated into the proposed MPC algorithm in the same way).

Simulations have been performed on a 8GB RAM computer running Windows 10. The proposed algorithm and the simulation environment have been coded in Julia 1.8 [20]. The MILP optimization problems at the base of the MPC iterations have been modelled by using the JuMP library [39], and solved by using the Gurobi solver [52].

5.5.1 Simulation Scenarios

Several artificial, yet realistic, scenarios have been used to test the algorithm. A scenario is characterized by the following parameters:

- Parameter A: The number of tasks to be scheduled;
- Parameter B: The number of resource types available in the assembly line;
- Parameter C: The number of workstations in the assembly line;
- Parameter D: The maximum capacity of the workstations;
- Parameter E: The number of resource items existing in the assembly line for every resource type;
- Parameter F: The duration of every task;
- Parameter G: The first possible start date and the deadline for every task;
- Parameter H: For every task, the types of resources it needs to run. This is controlled by specifying the probability that the task needs a given resource type. For simplicity, this probability is taken the same for all resource types. Parameter H is represented by such a probability. Then, for every resource type, a random number is sampled uniformly between 0 and 1, and if it is higher than the above probability, it means that the task needs that resource type to run;
- Parameter I: For every task and for every resource type needed by the task, the amount of resources of that type that the task needs to run;
- Parameter L: The number of predecessors of any given task. This is controlled by varying the probability that any two nodes of the precedence graph are connected by a link. Parameter L is represented by such a probability.

Each scenario has been generated by varying the above parameters as detailed in Table 5.2. The sampling time is $\Delta T = 15$ minutes. The resulting scenarios are of a high complexity. However, as it will be evident from the simulation results, the algorithm could scale even to larger scenarios.

Parameter	Selection criterion
A	Fixed at 90.
B	Uniform integer random number in the interval $[5, 10]$.
C	Fixed at 4.
D	Fixed at 1.
E	Uniform integer random number in the interval $[1, 10]$.
F	Uniform integer random number in the interval $[2, 10]$.
G	Not set (i.e., every task can start immediately, and must finish before the end of the problem).
H	Fixed at 0.6.
I	Uniform integer random number in the interval $[1, 10]$.
L	Connection probabilities: 30%, 40%, 50% and 60%.

Table 5.2: Parameters selection for the generation of the simulation scenarios.

The proposed MPC algorithm is tested on the above scenarios, varying the window N of prediction between 0, 5, and 10 hours. Results are compared against the ones obtained from a state-of-the-art heuristic method, which is detailed in the following.

5.5.2 Benchmark Heuristic method

The heuristic considered for the comparison with the proposed algorithm is taken from [9]. The priority criteria that has been chosen is the GRPW. For every task, the GRPW is calculated as the sum of the duration of the activity of the task and the duration of all its successors (see (5.22)). The heuristic scheduling approach is a parallel one, in the sense that at each time instant it considers all the tasks that can be scheduled in that moment. For each of these tasks the GRPW is computed, to be used as the priority criterion. Then the actual scheduling depends on the availability of the workstations and the resources. If a workstation is free, then a generic task is assigned to it on the base of the GRPW weight associated to it, and depending on the availability of the resources. If the task with the highest GRPW priority cannot be scheduled in the considered instant due to resource constraints, then the next highest priority task from the list is taken in consideration; an so on until all the candidate tasks that could be scheduled at the given time have been evaluated.

5.5.3 Comparison Results

Figure 5.4 reports the results of simulations carried out over 24 randomly generated scenarios, comparing the performance achieved by the proposed algorithm and the heuristic. Each of the four subplots reports the results obtained in six different scenarios, which are generated considering a value of parameter L in Tab. 5.2 (the connection probability, also called connectivity level) of 30%, 40%, 50%, and 60%, respectively. In each subplot of Fig. 5.4, every group of bars (which is composed of four bars) refers to four different tests carried out on the same scenario, characterized by the given connectivity level and number of resource types (reported on the x-axis). The bars

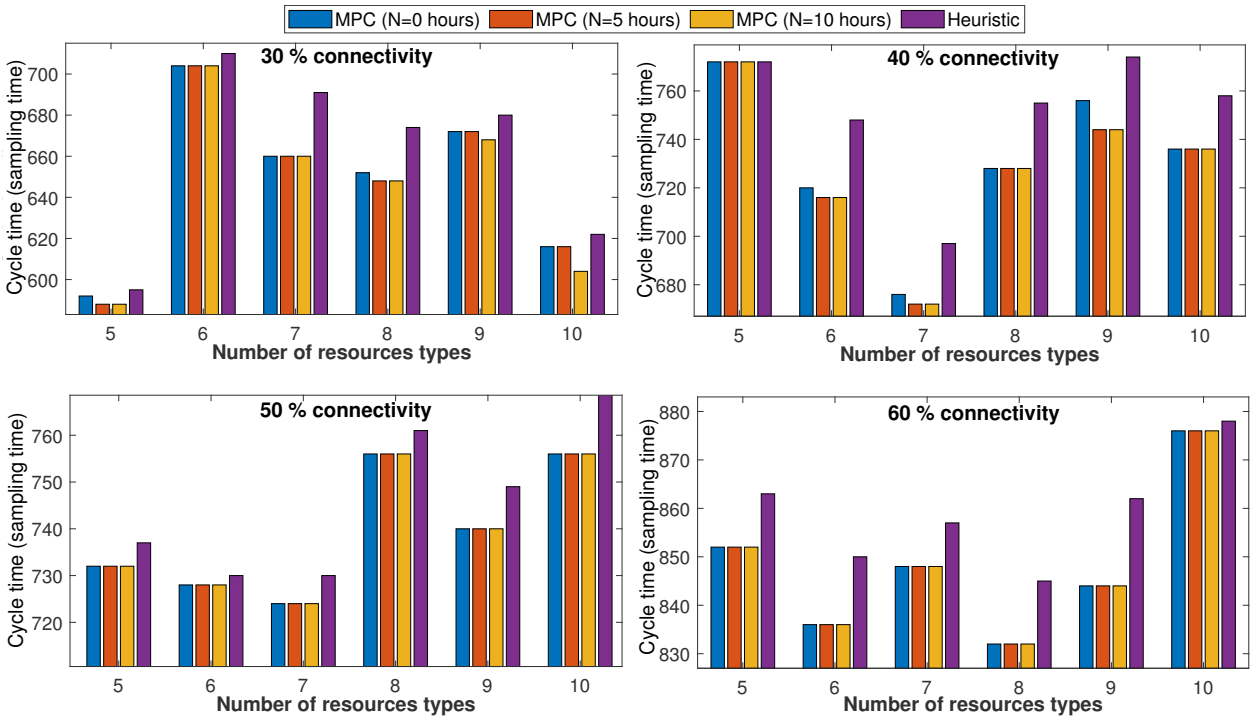


Figure 5.4: Comparison of the makespan achieved by the proposed MPC algorithm (with three different lengths of the prediction window) and the heuristic, in different scenarios. The makespan is reported in number of sampling intervals (the duration of each sampling interval, i.e., the sampling time, is $\Delta T = 15$ minutes).

report the makespan achieved in each simulation. In each group of bars, the first three bars from the left report, respectively, the makespan achieved by the proposed algorithm when $N = 0$ (i.e., MPC without a prediction window), $N = 20$ (i.e., a prediction window of 5 hours), and $N = 40$ (a prediction window of 10 hours). The last bar of each group of bars reports the makespan achieved by the heuristic algorithm.

From the simulations, the following can be observed:

1. The proposed MPC algorithm always provides a better result than the heuristic, except for one case (first bars in the top-right subplot), in which they give the same result;

2. In general, increasing the prediction window for the MPC algorithm leads to a decreased makespan. This is because, with a longer prediction window, the MPC can better "see" future constraints that are relevant to take into account for getting an optimized plan. This effect however seems less evident as the connectivity of the precedence graph increases. This is because, with high connectivity, the large number of precedence constraints present reduces the possible scheduling choices for the algorithm;

3. As the connectivity of the precedence graph increases, the makespan increases. This is expected, since the number of precedence constraints increases, and so the possibility to parallelize tasks decreases.

The average makespan reduction achieved by the proposed algorithm in the above simulation is reported in Tab. 5.3 (in brackets, the percentage increase in the makespan achieved by the proposed

algorithm compared to the monolithic one is reported–blue color–as well as the percentage reduction achieved by the proposed algorithm compared to the heuristic–red colour).

Connectivity	Makespan monolithic algo.	Makespan heuristic algo.	Makespan Proposed algo., N=0	Makespan Proposed algo., N=5h	Makespan Proposed algo., N=10h
30%	641	662	649,3 ($+1,29\%$, $-1,9\%$)	648 ($+1,09\%$, $-2,1\%$)	645,3 ($+0,67\%$, $-2,5\%$)
40%	709,1	750,6	731,3 ($+3,15\%$, $-2,6\%$)	728 ($+2,68\%$, -3%)	728 ($+2,68\%$, -3%)
50%	724	746,5	739,3 ($+2,11\%$, $-0,96\%$)	739,3 ($+2,11\%$, $-0,96\%$)	739,3 ($+2,11\%$, $-0,96\%$)
60%	774,3	859,2	848 ($+11,01\%$, $-1,3\%$)	848 ($+9,56\%$, $-1,3\%$)	848 ($+9,56\%$, $-1,3\%$)
Average values	712,1	754,6	742 ($+4,40\%$, $-1,67\%$)	740,8 ($+4,05\%$, $-1,82\%$)	740,2 ($+3,95\%$, $-1,91\%$)

Table 5.3: Comparison of the average makespan reduction achieved by the proposed algorithm with different prediction horizons, compared to the monolithic algorithm and the benchmark heuristic.

	Min. Time [ms]	Max. Time [ms]
MPC (N=0 h)	4	8
MPC (N=5 h)	13	137
MPC (N=10 h)	59	195
Heuristic	7	20
Monolithic algo.	11 839 000	149 420 000

Table 5.4: Comparison of the average minimum and maximum solving time for a single scheduling iteration.

The proposed algorithm always outperforms the heuristic. The reduced makespan that the proposed MPC algorithm can achieve also implies a better (i.e., higher) utilization of the resources, and a reduced inactivity of the workstations. In relation to this, note that, while in this paper we focus on the reduction of the sole makespan, in fact any additional KPI (utilization rate, inactivity time, and others depending on the specific scenario) could be included in the MPC objective function, for optimization.

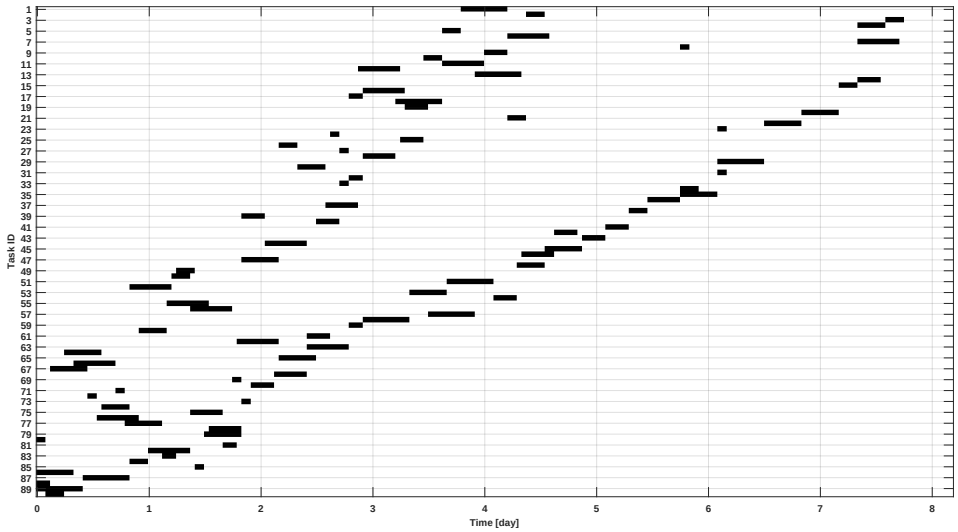
Regarding the computational time, Tab. 5.4 reports the average minimum and maximum solving time of the MPC iterations, over all the scenarios, in milliseconds. For comparison, the table also reports the computing time for the heuristic and the monolithic algorithm. As expected, the proposed MPC algorithm with a prediction window of 10 hours requires the highest computing time. However, this time is compatible with a real-time implementation of the algorithm, and comparable with the speed of the heuristic. The monolithic algorithm takes several hours to run, confirming that, although it provides the best possible solution (useful for comparison), it cannot be used as is in practice.

Finally, for completeness and to have a visual representation of the results on one of the test cases, the Gantt charts resulting from the application of the proposed MPC algorithm and the heuristic, in one of the scenarios above presented, is reported in Fig. 5.5. The Gantts refer to the scenario with a number of resource types equal to 9, and a 40% connectivity of the precedence graph. This is a good representative scenario to visualize the difference in performance between the MPC, with different prediction windows and the heuristic (see the related group of bars in Fig. 5.4, top-right subplot).

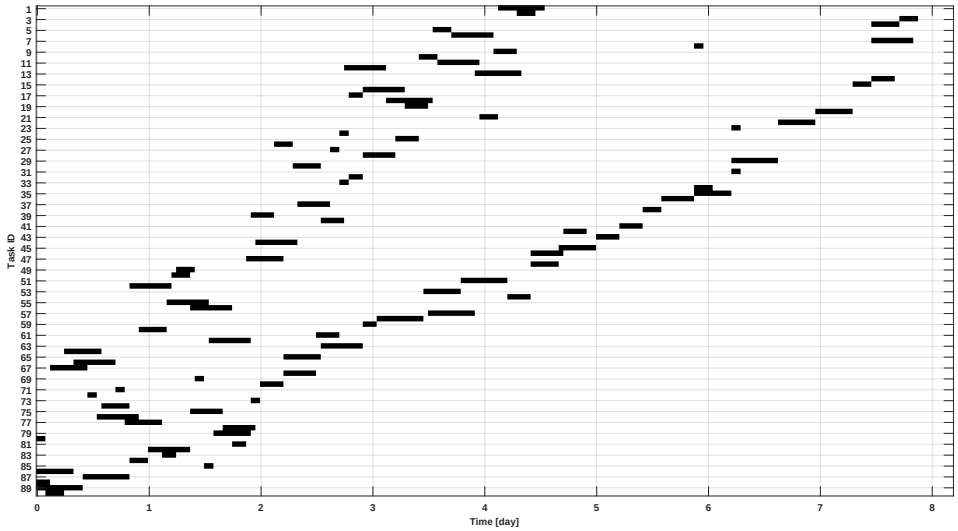
The three Gantt charts in Fig. 5.5 result, respectively, from the proposed MPC algorithm with no prediction window, from the proposed MPC algorithm with a 10 hours prediction window, and from the heuristic algorithm. In this case, it is evident that the MPC algorithm performs in general better than the heuristic, but also it is evident that a longer prediction window can result in a shorter makespan. This is expected since, as it is also true for MPC in general, as the prediction horizon increases, the problem becomes more and more like the monolithic one, which provides the best possible solution. However it is interesting to notice that, thanks also to the inclusion of the term (5.22), the proposed algorithm can perform very well even with low prediction horizons, which are associated with lower computing times.

From the analysis of the Gantt charts it is also evident why the proposed MPC algorithm outperforms the heuristic. Since the heuristic is not based on optimization, but rather on a simple ranking of the tasks according to a priority criterion, it might happen that at some instants it takes decisions that later on reveal to be sub-optimal ones.

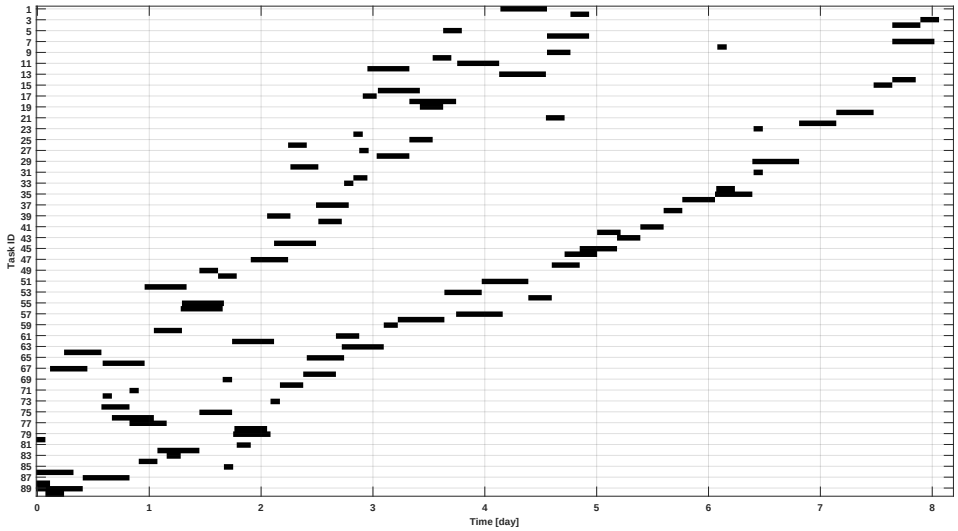
In addition, an optimization-based scheduling algorithm, such as the proposed MPC-one, allows



(a) Proposed MPC algorithm ($N = 10$ hours).



(b) Proposed MPC algorithm ($N = 0$ hours).



(c) Heuristic algorithm.

Figure 5.5: Comparison of Gantt charts resulting from the proposed MPC algorithm and the heuristic.

to integrate into the scheduling problem also additional relevant factors, such as the optimization of energy management, like it has been proposed in [71] for a monolithic MPC scheme, and more in general, resource optimization. For this reason, the proposed MPC scheduling algorithm allows to combine the flexibility of optimization-based algorithms (including MPC approaches) with the speed of heuristic approaches.

About the MPC sampling time, the chosen time step of 15 minutes is well suited for the tackled application, which involves operations lasting for several days, with tasks that in general last for several tens of minutes, or hours. Therefore, the chosen sampling time provides a good balance between the granularity of the control and the computation time. However, the speed of the algorithm allows for reducing the sampling time, if needed (e.g., if detailed optimization of the energy flows is desired). Of course, reducing the sampling time while keeping the same time prediction window will increase the number of variables in the problem (since N will increase), which will slow down the computation. To have an idea of this, consider from Figure 5.4 and Table 5.4 that the proposed algorithm with $N = 10$ gives almost the same performance achieved with $N = 5$. Therefore, by halving the sampling time and the prediction window, it is possible to keep almost unaltered the performance and the computation time of the algorithm.

In conclusion, the proposed algorithm provides a powerful tool to planning and scheduling operators, supporting them in the fast scheduling and re-scheduling of tasks, in real time. This can be particularly useful, e.g., in case faults or other unexpected events happen, making necessary to compute a new scheduling solution in a very short time. Being feedback-based, the proposed algorithm can handle disturbances and model inaccuracies in a transparent way, through re-optimization. As a matter of fact, at every time (see lines 1 and 2 of Algorithm 3) all the relevant information from the assembly line is acquired (i.e., time left to complete the tasks, operative status of workstations and resources, relevant environmental conditions), so that the newly computed task scheduling can take into account any disturbance or unexpected event happened in the assembly line, guaranteeing the operator with a solution that trough time is optimal and aligned with the status of the line.

5.6 Conclusions and Future Works

In this paper, a scalable MPC approach has been proposed for the efficient task scheduling and re-scheduling. The proposed algorithm computes the complete task planning by sequentially solving optimization problems defined each one over the set of current tasks candidate to be executed. This strategy speeds up computation and makes the algorithm suitable for real time implementation. A priority criterion inherited from a state of the art scheduling approach is integrated into the MPC objective function, with the purpose of improving makespan minimization, by associating a higher priority to the more "strategic" tasks, i.e., the ones having more successors. Simulation results confirm the merit of the algorithm, which retains the characteristics of standard MPC (i.e., the flexibility in modelling the problem and capturing the optimization of KPI), and outperforms state of the art scheduling heuristics (with an average reduction of $1,8\%$ in the makespan), while having a similar speed, which makes it suitable for real time implementation. At the same time, the performance of the proposed algorithm is only slightly lower (with nearly 4% increase in the makespan) than the one achieved by global optimization, which however is not suitable for real time implementation due to excessive computation times (hours or days, in the tested scenarios, Tab.

5.4).

Future works will address two main points. First, the formulation will be extended in order to include also the optimization of energy-related KPI (e.g., minimization of energy costs, optimal management of local storage and renewable energy production, minimization of the power flow between the industry and the grid). A second interesting direction for future works concerns the integration of deep learning features into the proposed MPC scheme. It is envisages to integrate a neural networks which, at every MPC iteration, will seek to provide as output an optimal value for the weight w_t in (5.21) (which in this paper is computed instead as in (5.22)). It is expected that this integration will allow to obtain optimal or close to optimal schedules even with a small MPC prediction horizon.

Chapter 6

Conclusions and Perspectives

This thesis contains three of the main research activities completed by the candidate during the years of his PhD. This research demonstrates the broad applicability and effectiveness of Model Predictive Control (MPC) in Cyber-Physical Systems (CPS) across diverse domains. The candidate had the opportunity to delve into the most recent innovative sectors such as Power Systems, Smart cities and Industry 4.0 in the space sector.

In the power systems sector, the research focused on transmission and distribution grids, in which with the recent integration of distributed energy resources such as electric energy storage systems (ESS), renewable energy plants, and plug-in electric vehicles are experiencing a transformation marked. These new factors necessitate continuous grid control and optimization and one of the possible control mechanism is the grid reconfiguration, providing flexibility to balance grid load, reduce losses, and swiftly restore service following grid disruptions. In this respect, the candidate presents a reconfiguration algorithm based on Model Predictive Control (MPC) framework, designed to dynamically configure the grid topology and minimizing at the same time losses during adverse events such as faults or (cyber-)attacks. The proposed algorithm facilitates the formation of (radial) grid islands with a complex mathematical formulation extending the radiality constraints, anticipating a future scenario where microgrid islands can dynamically connect and disconnect from the main grid.

The work done in the Smart Cities context focused on developing a strategy based on Model Predictive Control (MPC) for operating traffic lights incorporating a microgrid traffic model. This approach is unique because it uses a detailed traffic model that considers the position, speed, and acceleration of each vehicle, allowing for accurate modelling of queue dynamics. The controller is versatile, capable of handling various situations, from vehicles manually controlled to those with partially or fully autonomous vehicles, by sending optimized driving instructions to the vehicles. The study tested the controller's performance through simulations of an intersection with real traffic conditions, covering scenarios from entirely manual to fully automated traffic management. The findings show that the controller can significantly improve intersection management, as seen in shorter queue lengths and wait times, particularly in scenarios with autonomous vehicles. Additionally, in this research is discusses ongoing improvements to the model, such as real-time traffic simulation experiments and the creation of a decentralized algorithm that enables traffic lights or vehicles to collaboratively decide on the best control approach. Future work may involve incorporating deep learning to dynamically fine-tune the MPC algorithm's settings and model, potentially

leading to further advancements in traffic light management efficiency.

The context of space sector is of greatest interest in this historical period. In particular the candidate took the opportunity to deepen his research in this sense and the research presented in this thesis introduces a scalable MPC methodology for effective planning and control (i.e., task scheduling) of the various tasks and resources in an assembly of a space vehicle. The problem of how to optimize the performance of an assembly line, namely in the definition of the most efficient schedule for carrying out the individual operations, is well known and a widely studied problem, namely Assembly Line Balancing Problem (ALBP). The algorithm progresses by iteratively solving optimization problems for the current set of tasks awaiting execution, thus improving computational efficiency and making it suitable for real-time application. A unique feature of our approach is the incorporation of a priority criterion inherited from a conventional state of the art scheduling method into the MPC objective function, prioritizing "strategic" tasks which could have physical and logical constraints, to minimize overall makespan duration. The algorithm performance is validated through simulations, showcasing its ability to retain the characteristics of MPC methodology, i.e. the flexibility in modelling the problem and capturing the optimization of Key Performance Indicators (KPIs). It outperforms state of the art scheduling heuristics while maintaining speed suitable for real-time applications. However, the proposed algorithm experiences a slight decrease in performance compared to global optimization methods, which are impractical for real-time use due to their extensive computational requirements. Future research will focus on two main improvements: expanding the model to include the optimization of energy-related KPIs, such as minimizing energy costs and optimizing the use of local storage and renewable energy sources minimizing the power flow between the industry and the grid. Additionally, we aim to integrate deep learning techniques into the MPC framework, to determine optimal values for certain parameters in each MPC cycle. This integration seeks to achieve optimal or near-optimal scheduling outcomes, even with limited prediction horizons.

In conclusion, the research work results of this PhD thesis encompassed not just a control theory point of view but also delved into the practical implementability of the suggested methods. The study focused both on emerging and high-impact sectors and taking into consideration the needs of the industrial world. It is important to highlight the practical value of the proposed solutions, which stand out for their implementability and innovative character, offering the industry useful tools to stay updated and competitive.

Bibliography

[1] F. Abbracciavento, F. Zinnari, S. Formentin, A. G. Bianchessi, and S. M. Savaresi. Real-time optimal traffic management in signal-controlled intersections: a receding-horizon approach. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 1947–1952. IEEE, 2021.

[2] F. Abbracciavento, F. Zinnari, S. Formentin, A. G. Bianchessi, and S. M. Savaresi. Multi-intersection traffic signal control: A decentralized mpc-based approach. *IFAC Journal of Systems and Control*, page 100214, 2022.

[3] N. B. Abdelkrim R. Yelles-Chaouche, Evgeny Gurevsky and A. Dolgui. Reconfigurable manufacturing systems from an optimisation perspective: a focused review of literature. *International Journal of Production Research*, 59(21):6400–6418, 2021.

[4] H. Aguilar, A. García-Villoria, and R. Pastor. Heuristic and metaheuristic procedures for the buffer sizing problem in parallel assembly lines balancing problem with multi-line workstations and different cycle times. *Computers & Operations Research*, 157:106285, 2023.

[5] H. Ahmadi and J. R. Martí. Mathematical representation of radiality constraint in distribution system reconfiguration problem. *International journal of electrical power & energy systems*, 64:293–299, 2015.

[6] S. Akpinar, A. Elmi, and T. Bektaş. Combinatorial Benders cuts for assembly line balancing problems with setups. *European Journal of Operational Research*, 259(2):527–537, 2017.

[7] M. H. Alavidoost, M. H. Fazel Zarandi, M. Tarimoradi, and Y. Nemati. Modified genetic algorithm for simple straight and U-shaped assembly line balancing with fuzzy processing times. *Journal of Intelligent Manufacturing*, 28:313–336, 2017.

[8] M. Alizadeh and M. Jafari-Nokandi. A bi-level resilience-oriented islanding framework for an active distribution network incorporating electric vehicles parking lots. *Electric Power Systems Research*, 218:109233, 2023.

[9] B. F. Almeida, I. Correia, and F. Saldanha-da Gama. Priority-based heuristics for the multi-skill resource constrained project scheduling problem. *Expert Systems with Applications*, 57:91–103, 2016.

[10] arianespace. Vega-C user manual. issue 1.0, Sept 2020.

[11] T. Aziz, Z. Lin, M. Waseem, and S. Liu. Review on optimization methodologies in transmission network reconfiguration of power systems for grid resilience. *International Transactions on Electrical Energy Systems*, 31(3):e12704, 2021.

[12] H. Babazadeh and N. Javadian. A novel meta-heuristic approach to solve fuzzy multi-objective straight and u-shaped assembly line balancing problems. *Soft Computing*, 23:8217–8245, 2019.

[13] M. Baran and F. Wu. Network reconfiguration in distribution systems for loss reduction and load balancing. *IEEE Transactions on Power Delivery*, 4(2):1401–1407, 1989.

[14] A. Bemporad, L. Bellucci, and T. Gabbriellini. Dynamic option hedging via stochastic model predictive control based on scenario simulation. *Quantitative Finance*, 14(10):1739–1751, 2014.

[15] A. Bemporad, D. Bernardini, M. Livshiz, and B. Pattipati. Supervisory model predictive control of a powertrain with a continuously variable transmission. Technical report, SAE Technical Paper, 2018.

[16] A. Bemporad, D. Bernardini, R. Long, and J. Verdejo. Model predictive control of turbocharged gasoline engines for mass production. Technical report, SAE Technical Paper, 2018.

[17] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[18] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[19] A. Bemporad and C. Rocchi. Decentralized linear time-varying model predictive control of a formation of unmanned aerial vehicles. In *2011 50th IEEE conference on decision and control and European control conference*, pages 7488–7493. IEEE, 2011.

[20] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[21] S. Bharadwaj, S. Ballare, Rohit, and M. K. Chandel. Impact of congestion on greenhouse gas emissions for road transport in mumbai metropolitan region. *Transportation Research Procedia*, 25:3538–3551, 2017. World Conference on Transport Research - WCTR 2016 Shanghai. 10-15 July 2016.

[22] D. L. Bleviss. Transportation is critical to reducing greenhouse gas emissions in the united states. *WIREs Energy and environment*, 10, 2021.

[23] T. R. Board, E. National Academies of Sciences, and Medicine. *Signal Timing Manual - Second Edition*. The National Academies Press, Washington, DC, 2015.

[24] J. Burgschweiger, B. Gnädig, and M. Steinbach. Optimization models for operative planning in drinking water networks. *Optimization and Engineering*, 10:43–73, 03 2009.

[25] W. Chang, S. Burton, C.-W. Lin, Q. Zhu, L. Gauerhof, and J. McDermid. *Intelligent and Connected Cyber-Physical Systems: A Perspective from Connected Autonomous Vehicles*, pages 357–392. Springer International Publishing, Cham, 2020.

[26] B. Chen, Z. Yang, S. Huang, X. Du, Z. Cui, J. Bhimani, X. Xie, and N. Mi. Cyber-physical system enabled nearby traffic flow modelling for autonomous vehicles. In *2017 IEEE 36th International Performance Computing and Communications Conference (IPCCC)*, pages 1–6, 2017.

[27] Y. Chen and Y. Zhou. Machine learning based decision making for time varying systems: Parameter estimation and performance optimization. *Knowledge-Based Systems*, 190:105479, 2020.

[28] Y. Chen, Y. Zhou, and Y. Zhang. Machine learning-based model predictive control for collaborative production planning problem with unknown information. *Electronics*, 10(15):1818, 2021.

[29] Chetna, N. R. Chauhan, and V. Chawla. A review on mixed assembly line balancing, sequencing and scheduling in an industry. *IOP Conf. Series: Materials Science and Engineering*, 691, 2019.

[30] A. H. Chow, R. Sha, and S. Li. Centralised and decentralised signal timing optimisation approaches for network traffic control. *Transportation Research Part C: Emerging Technologies*, 2019.

[31] P. Chutima. A comprehensive review of robotic assembly line balancing problem. *Journal of Intelligent Manufacturing*, 2020.

[32] S. Civanlar, J. Grainger, H. Yin, and S. Lee. Distribution feeder reconfiguration for loss reduction. *IEEE Transactions on Power Delivery*, 3(3):1217–1223, 1988.

[33] S. Consortium. Smart european space access through modern exploitation of data science, March 2022.

[34] E. Dall'Anese, H. Zhu, and G. B. Giannakis. Distributed optimal power flow for smart microgrids. *IEEE Transactions on Smart Grid*, 4(3):1464–1475, 2013.

[35] A. De Concini and J. Toth. The future of the European space sector. How to leverage Europe's technological leadership and boost investments for space ventures. Technical report, EIB, 2019.

[36] F. M. Defersha and F. Mohebalizadehgashti. Simultaneous balancing, sequencing, and workstation planning for a mixed model manual assembly line using hybrid genetic algorithm. *Computers & Industrial Engineering*, 119:370–387, 2018.

[37] M. Donsante, A. Tortorelli, A. Di Giorgio, and F. Liberati. A model predictive control algorithm for the reconfiguration of radially operated grids with islands. *Electronics*, 12(24), 2023.

[38] J. Dornheim, N. Link, and P. Gumbsch. Model-free Adaptive Optimal Control of Episodic Fixed-horizon Manufacturing Processes Using Reinforcement Learning. *Int. Journal of Control, Automation and Systems*, 18(6):15931604, 2020.

[39] I. Dunning, J. Huchette, and M. Lubin. Jump: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[40] C. V. Eduardo Álvarez Miranda, Jordi Pereira and M. Vilà. Variable-depth local search heuristic for assembly line balancing problems. *International Journal of Production Research*, 61(9):3103–3121, 2023.

[41] R. Eini and S. Abdelwahed. Distributed model predictive control for intelligent traffic system. In *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 909–915. IEEE, 2019.

[42] A. Emami, M. Sarvi, and S. A. Bagloee. Network-wide traffic state estimation and rolling horizon-based signal control optimization in a connected vehicle environment. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):5840–5858, 2022.

[43] S. Esmaeili, A. Anvari-Moghaddam, S. Jadid, and J. M. Guerrero. A stochastic model predictive control approach for joint operational scheduling and hourly reconfiguration of distribution systems. *Energies*, 11(7), 2018.

[44] European Environment Agency. Transport and environment report 2021 | decarbonising road transport: the role of vehicles, fuels and transport demand. *EEA Report No 2/2022*, 2022.

[45] J. Fan, C. Zhang, Q. Liu, W. Shen, and L. Gao. An improved genetic algorithm for flexible job shop scheduling problem considering reconfigurable machine tools with limited auxiliary modules. *Journal of Manufacturing Systems*, 62:650–667, 2022.

[46] J.-Y. Fan, L. Zhang, and J. D. McDonald. Distribution network reconfiguration: single loop optimization. *IEEE transactions on Power Systems*, 11(3):1643–1647, 1996.

[47] H. Farhangi. The path of the smart grid. *IEEE Power and Energy Magazine*, 8(1):18–28, 2010.

[48] A. Gallina, A. Gibbesch, R. Krenn, T. Uhl, and B. Schäfer. Multibody simulation of planetary rover mobility in condition of uncertain soft terrain. *Procedia IUTAM*, 13:118–126, 2015.

[49] A. Ghosh and T. Parisini. Traffic control in a mixed autonomy scenario at urban intersections: An optimal control approach. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):17325–17341, 2022.

[50] M. Graf Plessen, D. Bernardini, H. Esen, and A. Bemporad. Spatial-based predictive control and geometric corridor planning for adaptive cruise control coupled with obstacle avoidance. *IEEE Transactions on Control Systems Technology*, 26(1):38–50, 2018.

[51] C. Gröger. Building an Industry 4.0 Analytics Platform. *Datenbank Spektrum*, 18:5–14, 2018.

[52] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.

[53] J. W. Herrmann. *Handbook of production scheduling*, volume 89. Springer Science & Business Media, 2006.

[54] M. Hosseinzadeh, B. Sinopoli, I. Kolmanovsky, and S. Baruah. Mpc-based emergency vehicle-centered multi-intersection traffic control. *IEEE Transactions on Control Systems Technology*, pages 1–13, 2022.

[55] B. Hou, S. Li, and Y. Zheng. Distributed model predictive control for reconfigurable systems with network connection. *IEEE Transactions on Automation Science and Engineering*, 19(2):907–918, 2022.

[56] M. Imran, R. Izzo, A. Tortorelli, and F. Liberati. Comparison of traffic control with model predictive control and deep reinforcement learning. In *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 989–994, 2023.

[57] R. Jabr. Radial distribution load flow using conic programming. *IEEE Transactions on Power Systems*, 21(3):1458–1459, 2006.

[58] R. A. Jabr. Minimum loss network reconfiguration using mixed-integer convex programming. *IEEE TRANSACTIONS ON POWER SYSTEMS*, 27, 2012.

[59] N. Jazdi. Cyber physical systems in the context of industry 4.0. In *2014 IEEE International Conference on Automation, Quality and Testing, Robotics*, pages 1–4, 2014.

[60] A. Jebali, A. B. Hadj Alouane, and P. Ladet. Operating rooms scheduling. *International Journal of Production Economics*, 99(1-2):52–62, February 2006.

[61] S. Jin, Z. Hou, R. Chi, and X. Bu. Model free adaptive predictive control approach for phase splits of urban traffic network. In *2016 Chinese Control and Decision Conference (CCDC)*, pages 5750–5754, 2016.

[62] M. Kamal, J. Imura, T. Hayakawa, A. Ohata, and K. Aihara. Traffic signal control in an mpc framework using mixed integer programming. *IFAC Proceedings Volumes*, 46(21):645 – 650, 2013. 7th IFAC Symposium on Advances in Automotive Control.

[63] M. A. S. Kamal, J.-i. Imura, T. Hayakawa, A. Ohata, and K. Aihara. Traffic signal control of a road network using milp in the mpc framework. *International journal of intelligent transportation systems research*, 13(2):107–118, 2015.

[64] M. Klaučo, M. Kalúz, and M. Kvasnica. Real-time implementation of an explicit mpc-based reference governor for control of a magnetic levitation system. *Control Engineering Practice*, 60:99–105, 2017.

[65] B. Kouvaritakis and M. Cannon. Model predictive control. *Switzerland: Springer International Publishing*, page 38, 2016.

[66] O. Kundačina, Vidović, P.M., and M. Petković. Solving dynamic distribution network reconfiguration using deep reinforcement learning. *Electrical Engineering*, 104:1487–1501, 2022.

[67] P. S. Kundur and O. P. Malik. *Power system stability and control*. McGraw-Hill Education, 2022.

[68] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6(4):239–242, aug 2014.

[69] J. Lee, B. Bagheri, and H.-A. Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3:18–23, 2015.

[70] F. Liberati. Model predictive control of a road junction. *Smart Cities*, 3(3):806–817, 2020.

[71] F. Liberati, C. M. F. Cirino, and A. Tortorelli. Energy-aware model predictive control of assembly lines. In *Actuators*, volume 11, page 172. MDPI, 2022.

[72] F. Liberati, A. Di Giorgio, A. Giuseppi, A. Pietrabissa, and F. D. Priscoli. Efficient and risk-aware control of electricity distribution grids. *IEEE Systems Journal*, 14(3):3586–3597, 2020.

[73] F. Liberati, M. Donsante, C. Cirino, and A. Tortorelli. Fast task scheduling with model predictive control integrating a priority-based heuristic. *IEEE Access*.

[74] F. Liberati, M. Donsante, and A. Tortorelli. Single intersection mpc traffic signal control in presence of automated vehicles. *IEEE Transactions on Control Systems Technology*.

[75] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn. Fast model predictive control for urban road networks via milp. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):846–856, 2011.

[76] T. Long, Y. Li, and J. Chen. Productivity prediction in aircraft final assembly lines: Comparisons and insights in different productivity ranges. *Journal of Manufacturing Systems*, 62:377–389, 2022.

[77] Y. Ma, F. Borrelli, B. Hencey, B. Coffey, S. Bengea, and P. Haves. Model predictive control for the operation of building cooling systems. *IEEE Transactions on Control Systems Technology*, 20(3):796–803, 2012.

[78] K. Małecki, P. Pietruszka, and S. Iwan. Comparative analysis of selected algorithms in the process of optimization of traffic lights. In *Asian Conference on Intelligent Information and Database Systems*, pages 497–506. Springer, 2017.

[79] L. Manieri, A. Falsone, and M. Prandini. A novel decentralized approach to large-scale multi-agent milps. *IFAC-PapersOnLine*, 56(2):5919–5924, 2023.

[80] C. Martínez-Olvera and J. Mora-Vargas. A Comprehensive Framework for the Analysis of Industry 4.0 Value Domains. *Sustainable Engineering and Science*, 11(10), 2019.

[81] M. Melo, S. Nickel, and F. S. da Gama. Facility location and supply chain management – a review. *European Journal of Operational Research*, 196(2):401–412, 2009.

[82] J. Miltenburg. U-shaped production lines: A review of theory and practice. *International Journal of Production Economics*, 70(3):201–214, 2001.

[83] A. Mishra, M. Tripathy, and P. Ray. A survey on different techniques for distribution network reconfiguration. *Journal of Engineering Research*, 2023.

[84] J. M. Morales, A. J. Conejo, and J. PÉrez-Ruiz. Short-term trading for a wind power producer. *IEEE Transactions on Power Systems*, 25(1):554–564, 2010.

[85] C. A. Méndez, J. Cerdá, I. E. Grossmann, I. Harjunkoski, and M. Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers Chemical Engineering*, 30(6):913–946, 2006.

[86] P. Neirotti, A. De Marco, A. C. Cagliano, G. Mangano, and F. Scorrano. Current trends in smart city initiatives: Some stylised facts. *Cities*, 38:25–36, 2014.

[87] M. G. Nejad and A. H. Kashan. An effective grouping evolution strategy algorithm enhanced with heuristic methods for assembly line balancing problem. *Journal of Advanced Manufacturing Systems*, 18(3):487–509, 2019.

[88] R. Neufville, H. Abdalla, and A. Abbas. Potential of connected fully autonomous vehicles in reducing congestion and associated carbon emissions. *Sustainability*, 14(11), 2022.

[89] J. M. Nilakantan, Z. Li, Q. Tang, and P. Nielsen. MILP models and metaheuristic for balancing and sequencing of mixed-model two-sided assembly lines. *European Journal of Industrial Engineering*, 11(3):353–379, 2017.

[90] C. Osorio and K. Nanduri. Urban transportation emissions mitigation: Coupling high-resolution vehicular emissions and traffic models for traffic signal optimization. *Transportation Research Part B: Methodological*, 81:520–538, 2015. Optimization of Urban Transportation Service Networks.

[91] C. A. Pascucci, S. Bennani, and A. Bemporad. Model predictive control for powered descent guidance and control. In *2015 European control conference (ECC)*, pages 1388–1393. IEEE, 2015.

[92] P. Patrinos, S. Trimboli, and A. Bemporad. Stochastic mpc for real-time market-based optimal power dispatch. In *2011 50th IEEE conference on decision and control and European control conference*, pages 7111–7116. IEEE, 2011.

[93] E. Perea-López, B. E. Ydstie, and I. E. Grossmann. A model predictive control strategy for supply chain optimization. *Computers & Chemical Engineering*, 27(8-9):1201–1218, 2003.

[94] V. H. Pham, K. Sakurama, S. Mou, and H.-S. Ahn. Distributed control for an urban traffic network. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):22937–22953, 2022.

[95] C.-R. Rad, O. Hancu, I.-A. Takacs, and G. Olteanu. Smart monitoring of potato crop: A cyber-physical system architecture model in the field of precision agriculture. *Agriculture and Agricultural Science Procedia*, 6:73–79, 2015. Conference Agriculture for Life, Life for Agriculture.

[96] M. Rahmani-Andebili. Dynamic and adaptive reconfiguration of electrical distribution system including renewables applying stochastic model predictive control. *IET Generation, Transmission & Distribution*, 11(16):3912–3921, 2017.

[97] M. Rahmani-Andebili and M. Fotuhi-Firuzabad. An adaptive approach for pevs charging management and reconfiguration of electrical distribution system penetrated by renewables. *IEEE Transactions on Industrial Informatics*, 14(5):2001–2010, 2018.

[98] E. R. Ramos, A. G. Expósito, J. R. Santos, and F. L. Iborra. Path-based distribution network modeling: application to reconfiguration for loss reduction. *IEEE Transactions on power systems*, 20(2):556–564, 2005.

[99] A. Richards and J. How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, volume 3, pages 1936–1941 vol.3, 2002.

[100] H. Ritchie, P. Rosado, and M. Roser. Emissions by sector: where do greenhouse gases come from? *Our World in Data*, 2020. https://ourworldindata.org/emissions-by-sector.

[101] E. Romero-Ramos, J. Riquelme-Santos, and J. Reyes. A simpler and exact mathematical model for the computation of the minimal power losses tree. *Electric Power Systems Research*, 80(5):562–571, 2010.

[102] S. K. Roy, R. Devaraj, and A. Sarkar. Safla: scheduling multiple real-time periodic task graphs on heterogeneous systems. *IEEE Transactions on Computers*, 2022.

[103] S. K. Roy, R. Devaraj, A. Sarkar, K. Maji, and S. Sinha. Contention-aware optimal scheduling of real-time precedence-constrained task graphs on heterogeneous distributed systems. *Journal of Systems Architecture*, 105:101706, 2020.

[104] S. K. Roy, R. Devaraj, A. Sarkar, and D. Senapati. Slaqa: Quality-level aware scheduling of task graphs on heterogeneous distributed systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 20(5):1–31, 2021.

[105] N. V. Sahinidis. Mixed-integer nonlinear programming 2018, 2019.

[106] A. K. Sampathirao, P. Sopasakis, A. Bemporad, and P. P. Patrinos. Gpu-accelerated stochastic predictive control of drinking water networks. *IEEE Transactions on Control Systems Technology*, 26(2):551–562, 2017.

[107] A. Sassella, F. Abbracciavento, S. Formentin, A. G. Bianchessi, and S. M. Savaresi. On queue length estimation in urban traffic intersections via inductive loops. In *2023 American Control Conference (ACC)*, pages 1135–1140. IEEE, 2023.

[108] M. Schwenzer, M. Ay, T. Bergs, and D. Abel. Review on model predictive control: An engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5-6):1327–1349, 2021.

[109] D. Shi, W. Fan, Y. Xiao, T. Lin, and C. Xing. Intelligent scheduling of discrete automated production line via deep reinforcement learning. *Journal of Manufacturing Systems*, 58(11):3362–3380, January 2020.

[110] M. J. Smith, T. Iryo, R. Mounce, M. Rinaldi, and F. Viti. Traffic control which maximises network throughput: Some simple examples. *Transportation Research Part C: Emerging Technologies*, 107:211–228, 2019.

[111] T. Sprodowski, J. Sagawa, and J. Pannek. Frequency Based Model Predictive Control of a Manufacturing System. *9th Vienna Int. Conf. on Mathematical Modelling*, 51(2):801–806, 2018.

[112] B. Stellato, V. V. Naik, A. Bemporad, P. Goulart, and S. Boyd. Embedded mixed-integer quadratic optimization using the osqp solver. In *2018 European Control Conference (ECC)*, pages 1536–1541. IEEE, 2018.

[113] A. Stevanovic, J. Stevanovic, K. Zhang, and S. Batterman. Optimizing traffic control to reduce fuel consumption and vehicular emissions: Integrated approach with vissim, cmem, and visgaost. *Transportation Research Record*, 2128(1):105–113, 2009.

[114] K. Subramanian, J. Rawlings, C. Maravelias, J. Flores-Cerrillo, and L. Megan. Integration of Control Theory and Scheduling Methods for Supply Chain Management. *Computers & Chemical Engineering*, 51:4–20, 2013.

[115] L. Tang, Q. He, D. Wang, and C. Qiao. Multi-modal traffic signal control in shared space street. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):392–403, 2022.

[116] K. Tliba, T. M. Diallo, O. Penas, R. Ben Khalifa, N. Ben Yahia, and J.-Y. Choley. Digital twin-driven dynamic scheduling of a hybrid flow shop. *Journal of Intelligent Manufacturing*, pages 1–26, 2022.

[117] A. Tortorelli, M. Imran, F. Delli Priscoli, and F. Liberati. A parallel deep reinforcement learning framework for controlling industrial assembly lines. *Electronics*, 11(4):539, 2022.

[118] G. Wan, X. Dong, Q. Dong, Y. He, and P. Zeng. Context-aware scheduling and control architecture for cyber-physical production systems. *Journal of Manufacturing Systems*, 62:550–560, 2022.

[119] C. Wang and P. Jiang. Manifold learning based rescheduling decision mechanism for recessive disturbances in rfid-driven job shops. *Journal of Intelligent Manufacturing*, 29(7):1485–1500, 2018.

[120] H. Wang, M. Zhu, W. Hong, C. Wang, G. Tao, and Y. Wang. Optimizing signal timing control for large urban traffic networks using an adaptive linear quadratic regulator control strategy. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):333–343, 2022.

[121] Q. Wang and M. Abbas. Optimal urban traffic model predictive control for NEMA standards. *Transportation Research Record*, 2673(7):413–424, 2019.

[122] C. Weckenborg, K. Kieckhäfer, C. Müller, M. Grunewald, and T. S. Spengler. Balancing of assembly lines with collaborative robots. *Business Research*, 13(1):93–132, 2020.

[123] Y. Wei, S. Li, and Y. Zheng. Enhanced information reconfiguration for distributed model predictive control for cyber-physical networked systems. *International Journal of Robust and Nonlinear Control*, 30(1):198–221, 2020.

[124] P. Wenzelburger and F. Allgöwer. Model predictive control for flexible job shop scheduling in industry 4.0. *Applied Sciences*, 11(17), 2021.

[125] P. Wright. On minimum spanning trees and determinants. *Mathematics magazine*, 73(1):21–28, 2000.

[126] N. Wu, D. Li, Y. Xi, and B. de Schutter. Distributed event-triggered model predictive control for urban traffic lights. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–11, 2020.

[127] M. S. Yang, L. Ba, Y. Liu, H. Y. Zheng, J. T. Yan, X. Q. Gao, and J. M. Xiao. An improved genetic simulated annealing algorithm for stochastic two-sided assembly line balancing problems. *Int. Journal of Simulation Modelling (IJSIMM)*, 18(1):175–186, 2019.

[128] B.-L. Ye, W. Wu, K. Ruan, L. Li, T. Chen, H. Gao, and Y. Chen. A survey of model predictive control methods for traffic signal control. *IEEE/CAA Journal of Automatica Sinica*, 6(3):623–640, 2019.

[129] T. Yu, J. Huang, and Q. Chang. Optimizing task scheduling in human-robot collaboration with deep multi-agent reinforcement learning. *Journal of Manufacturing Systems*, 60:487–499, July 2021.

[130] L. C. Zammit, S. G. Fabri, and K. Scerri. Self-tuning model predictive control for signalized traffic junctions. In *2021 European Control Conference (ECC)*, pages 2585–2590, 2021.

[131] M. Zhang, F. Tao, and A. Nee. Digital twin enhanced dynamic job-shop scheduling. *Journal of Manufacturing Systems*, 58:146–156, 2021.

[132] Y. Zhang, T. Qian, and W. Tang. Buildings-to-distribution-network integration considering power transformer loading capability and distribution network reconfiguration. *Energy*, 244:123104, 2022.

[133] Y. Zhang, M. Qiu, C.-W. Tsai, M. M. Hassan, and A. Alamri. Health-cps: Healthcare cyber-physical system assisted by cloud and big data. *IEEE Systems Journal*, 11(1):88–95, 2017.

[134] I. H. Zohdy and H. A. Rakha. Intersection management via vehicle connectivity: The intersection cooperative adaptive cruise control system concept. *Journal of Intelligent Transportation Systems*, 20(1):17–32, 2016. Cyber Transportation Systems and Connected Vehicle Research.

Acknowledgements

(Left blank intentionally)