POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

A Reachability-Based Decoupling Solution to the Routing Problem in Smart Manufacturing

Tesi di Laurea Magistrale in
Automation and Control Engineering - Ingegneria dell'Automazione e del Controllo

Author: **Alfredo Stama**

Student ID: 10748072
Advisor: Prof. Maria Prandini
Co-advisors: Alessandro Falsone, Lucrezia Manieri, Andrea Cataldo
Academic Year: 2022-23

Abstract

We address optimal routing in smart manufacturing according to a model predictive control (MPC) approach that was proposed in a recent PhD thesis. In this PhD work, a mixed logical dynamical (MLD) model of the transport line in a plant is adopted with a cost function that favors those actions which move pallets towards their destinations through the shortest path. Conflicts are avoided by introducing state-dependent constraints on the admissible control actions. The resulting constrained optimization problem is admittedly computationally intensive due to its combinatorial nature, which can slow down the transport line operation and reduce the manufacturing plant throughput as the number of pallets and the MPC prediction horizon grow.

Here, we propose a methodology to defeat such a computational complexity by decoupling the MPC optimization program into multiple smaller dimensional programs that can be solved in parallel. This is achieved using a graph representation of the transport line and partitioning it in sub-graphs associated to the different pallets by resorting to reachability analysis over the MPC prediction horizon. A reduced MLD model is determined for each sub-graph by pruning state and input variables, and eliminating redundant equations and constraints from the complete MLD model. If the MPC solutions computed in parallel on the sub-graphs are not conflicting, then, the planned actions are applied. If instead some conflict is detected, sub-graphs are joined together and the reduced MLD model and associated MPC solution of the joint sub-graph are computed. The same procedure is repeated until all conflicts are solved.

Although general, the proposed strategy is developed with reference to the manufacturing plant studied in the reference PhD thesis. Extensive simulations show the effectiveness of the reachability-based decoupling approach.

Keywords: Mixed Logical Dynamical (MLD) systems, Model Predictive Control (MPC), smart manufacturing, system decomposition and reduction

Abstract in lingua italiana

Questo lavoro affronta il problema della movimentazione dei pallet in sistemi manufattuieri automatizzati secondo l'approccio del controllo predittivo del modello (MPC) applicato ad un modello ibrido "mixed logical dynamical (MLD)" della linea di trasporto con una funzione di costo che favorisce le azioni che spostano i pallet verso la loro destinazione lungo il percorso più breve evitando conflitti. L'approccio descritto è stato proposto in una recente tesi di dottorato dove è stato evidenziata la complessità computazionale del risultante problema di ottimizzazione vincolata a causa della sua natura combinatoria. Ciò può rallentare il funzionamento della linea di movimentazione e ridurre la produttività dell'impianto.

In questo lavoro, proponiamo una metodologia per ridurre la complessità computazionale disaccoppiando il programma di ottimizzazione MPC in più programmi dimensionalmente più piccoli che possono essere risolti in parallelo. Ciò si ottiene rappresentando mediante un grafo la linea di trasporto, e suddividendolo in sottografi associati ai diversi pallet mediante analisi du raggiungibilità sull'orizzonte predittivo dell'MPC.

Viene quindi determinato un modello MLD ridotto per ciascun sottografo eliminando le variabili di stato e di ingresso e le equazioni e vincoli ridondanti dal modello MLD completo. Se le soluzioni MPC calcolate in parallelo sui sottografi non sono in conflitto, vengono applicate le azioni pianificate. Se invece viene rilevato qualche conflitto, i sottografi corrispondenti ai pallet in conflitto vengono uniti insieme, viene calcolato il loro modello MLD ridotto e la soluzione MPC associata. La stessa procedura viene ripetuta finché tutti i conflitti non vengono risolti.

Sebbene generale, la strategia proposta viene sviluppata con riferimento a uno specifico impianto. Simulazioni estensive confermano l'efficacia della decomposizione basata su analisi di raggiungibilità.

Parole chiave: Sistemi MLD (Mixed Logical Dynamical), controllo predittivo, industria manufatturiera intelligente, decomposizione e riduzione di un sistema

Contents

1 | Introduction

1.1. Routing in Smart Manufacturing

Manufacturing industries are typically characterized by a sequence of operations that need to be performed in a specific order. Processing times and throughput of a manufacturing plant can be improved by designing appropriate strategies to operate the transport lines that are driving the goods through the different machines for subsequent processing, avoiding bottlenecks, starvation and congestion problems [9].

Automation can play an important role in improving the routing strategy and in the solution of many significant problems such as lotsizing, scheduling, packing, inventory and resource allocation [7]-[8]. Automatizing routing in smart manufacturing can also help to ensure consistency and quality control, which are critical factors in the market.

Following the PhD thesis in [4], we further investigate the adoption of Model Predictive Control (MPC) for such a purpose. Moreover, applications of MPC to routing problems are also described in [13] and [1]; in particular, [13] considers the management of a baggage handling system, which could be considered as a similar problem, although the allowed paths are basically unidirectional and the final destination of each baggage does not change during the operations. On the contrary, in the plant of Figure 1.3, double directional paths among the machines must be followed and the target machines for the pallets are dynamically changed in a partially unpredictable way, depending on the outcome of the testing machine.

MPC plays an important role for the solution of many control problems. In particular, its success in the process industries is described extensively in [11]. One of the main advantages of the MPC over other control methods is the fact that it can easily handle constraints through the reformulation of a control problem as an optimization program.

Due to its ability to cope with constraints and multiple objective [10], the MPC technology has progressively been extended to a variety of domains, with processes characterized by faster dynamics, thanks to a reduction of the computing times obtained via advancements

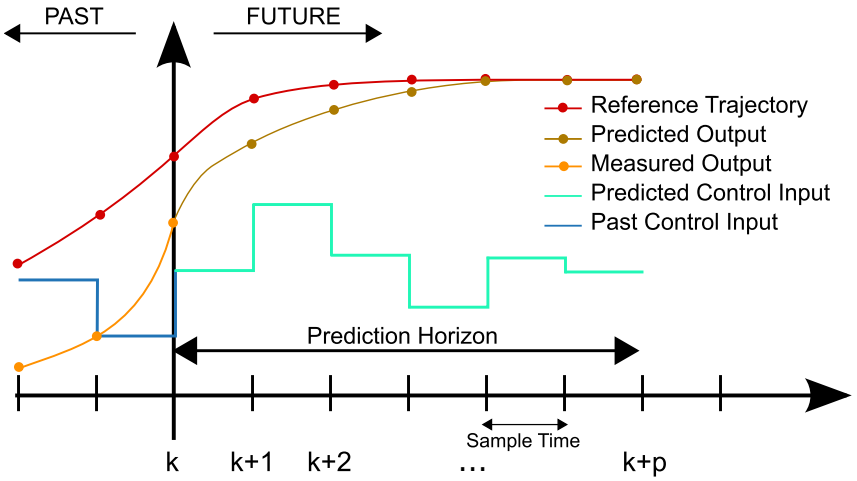

Figure 1.1: MPC strategy for trajectory tracking.

in both technology and optimization theory. However, when the optimization variables take values in a discrete set, then, the computational effort involved in the solution to the resulting integer optimization problem may hamper its applicability. This is indeed the case in manufacturing systems that are typically described as discrete-event systems characterized by integer or Boolean decision variables.

MPC working principle

The MPC strategy (see e.g. [12]) is based on the knowledge of a dynamic model of the system, which allows to compute the future evolution of the controlled variables as a function of the values assigned to the control input. The input sequence can then be computed by minimizing a cost function under state and input constraints.

The basic idea is to turn a control synthesis problem into an optimization problem over some reference prediction horizon. According to the receding horizon implementation scheme in Figure 1.1, at each time k the control input over the whole prediction horizon is computed, but only its first value is applied, to then solve again the finite-horizon optimization problem based on the new state measurement. This results in the MPC implementation scheme in Figure 1.2, where a model of the plant is adopted to predict the future output behavior. The Optimization Unit block is the heart of the scheme as it contains the cost function and the constraints: it takes as an input the prediction errors, so as to produce the control actions through the resolution of an optimization problem, whose size depends on the number of input variables and the value of the prediction horizon.

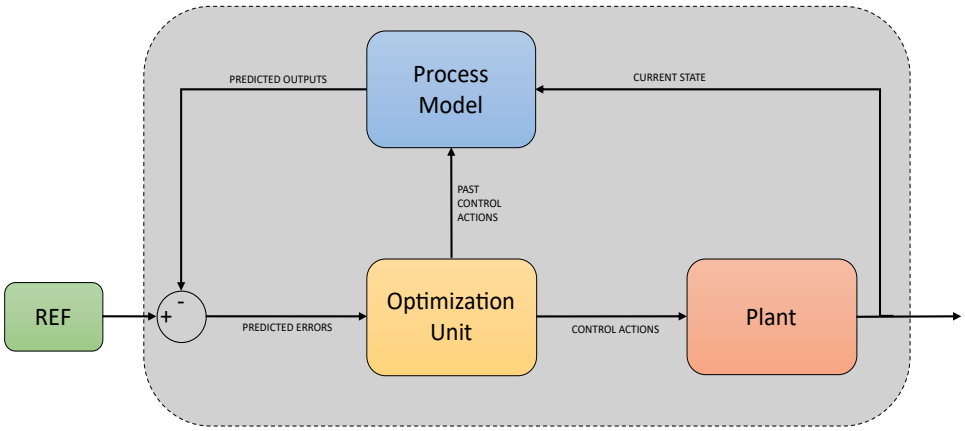

Figure 1.2: MPC reference scheme.

MPC for optimal routing in smart manufacturing

Application of MPC to routing in smart manufacturing was proposed in the PhD thesis [4]. The plant is located in the laboratory at *IT IA - CNR*, Gorgonzola (MI), Italy and was considered as a testbed. The plant, designed for the testing, reparir or disruption of electronic boards is composed by a multi-path transport line and by loading/unloading, testing, repair and discharge machines; its structure and behaviour are extensively described in [6] and [3].

The control system of the plant has a multi-level hierarchical structure:

- at a higher level, MPC is used to coordinate the movement of the pallets along the transportation line in order to optimize the plant performance and to fulfill a set of logical constraints imposed by the transport line structure;
- at a lower level, a set of PLCs, one per transport module, acquires the sensor signals and drive the actuators.

To the purpose of MPC design, the transport line is described as a *Mixed Logical Dynamical* (MLD) system as described in detail in Chapter 2.

MLD systems ([2]) can represent a wide set of models, among which linear hybrid systems, finite state machines, some classes of discrete event systems, constrained linear systems and nonlinear systems whose non-linearities can be expressed by piecewise linear functions. MLD systems are discrete time systems described by linear equations sub-

Figure 1.3: The manufacturing plant in the laboratory at *IT IA - CNR*.

ject to linear mixed-integer inequalities, i.e. inequalities involving both *continuous* and *binary* variables. These include physical/discrete states, continuous/integer inputs, and continuous/binary auxiliary variables.

Main issue of the MPC-based optimal routing

The MPC algorithm proposed in the PhD thesis [4] for the optimal routing in a manufacturing plant modeled as an MLD system shows that the computational effort involved in the MPC solution becomes relevant as the number of pallets and/or the length of the prediction horizon increases. More precisely, the mean value of the time required for the online optimization becomes larger and larger to the point that most of the time of the pallets on the transport line is spent waiting for the next control actions. This highly deteriorates the performance of the manufacturing plant in terms of throughput and, hence, limits the possibility of adopting MPC for optimal routing in smart manufacturing systems.

1.2. Contributions of this thesis

In this thesis, we propose a strategy to alleviate the computational effort involved in determining a solution to the MPC problem for routing suggested in the PhD thesis

[4]. The key idea is to decouple the optimization problem in multiple lower-dimensional problems that can be solved in parallel, each one defined on an appropriate reduced model of the MLD system modeling the overall plant.

To this aim:

1. The transport line is modelled as a direct graph. The graph is then decomposed into smaller sub-graphs according to a reachability analysis performed on every pallet so as to identify the nodes that it can reach within the prediction horizon. This allows to reduce the MLD system into multiple smaller ones.
2. The MPC algorithm is applied in parallel to every reduced MLD system.
3. If the optimal control actions to be applied at the current step do not create any conflict, i.e., the pallets are not driven to the same node, then the computed solution is feasible and can be applied according to the receding horizon strategy. Otherwise, the sub-graphs of the conflicting pallets are joined together and a new MPC problem is defined for the resulting MLD model. This third step is repeated until there are no conflicts.

In order to reduce the number of times the MPC solution is recomputed, the concept of Off-limit zone and Exchange zone are introduced and pallets that are within these zones are grouped together before starting the first iteration.

The underlying idea is that by dividing one complex MLD system into several smaller and simpler ones through a reachability analysis, the overall time to solve the MPC is reduced, especially if the smaller MPC optimization problems are solved in parallel, thus allowing for the adoption of MPC for a transport line with a higher number of pallets and/or larger prediction horizons.

The proposed approach is of general applicability. However, in this thesis it is developed with reference to the *ITIA-CNR* De-Manufacturing plant for mechanical treatment of end-of-life products that was considered in the PhD thesis [4].

1.3. Thesis structure

The rest of the thesis is organized as follows.

In Chapter 2, we recall the strategy proposed in the PhD thesis [4] for the automated routing in a manufacturing plant. In particular, we present the MLD model, which is derived and graph representation of the transport line and the MPC problem formulation for the *ITIA - CNR* plant.

Chapter 3 describes the proposed approach to address the combinatorial complexity of the MPC strategy in the PhD thesis [4] by decomposing the MPC optimization problem into smaller ones that can be solved in parallel. This involves performing reachability analysis to partition the graph of the overall plant into sub-graphs and defining the associated reduced MLD systems. The concept of Off-limit zone and Exchange zone are introduced, which are instrumental to the problem decomposition.

Chapter 4 contains extensive simulation results showing the performance of the proposed MPC approach with respect to the original one in the PhD thesis [4] in terms of computing time and throughput of the plant.

Finally, in Chapter 5 some conclusions are drawn and future possible developments are discussed.

2 | Modeling and MPC Formulation of the Automated Routing Problem

This chapter recalls the description of the *ITIA-CNR* de-manufacturing plant and the MPC formulation of the problem of pallet automated transportation in the PhD thesis [4]. In particular, the graph and MLD system modeling the plant are described together with the adopted MPC finite horizon cost.

2.1. Manufacturing plant description

The considered plant is located in the laboratory of *ITIA-CNR* and is equipped with four machines for de-manufacturing of electronic boards:

- Machine M_1 is the Load/unload robot cell, which load/unload an electronic board onto/from a pallet;
- Machine M_2 is the Testing machine, which identifies the failure mode of the board;
- Machine M_3 is the Reworking machine, which repairs the board;
- Machine M_4 is the Discharge machine, which discharges and destroys non repairable boards.

A transport line composed of fifteen transport modules moves the pallets among the four machines. The overall plant layout is shown in Figure 2.1 and, in a more schematised way, in Figure 2.2.

The sequence of operations for each single board is as follows:

1. the board is **loaded** on the pallet by M_1; the pallet is then placed on the adjacent transport module of the transport line;
2. the pallet is moved to M_2 where the board is **tested**;

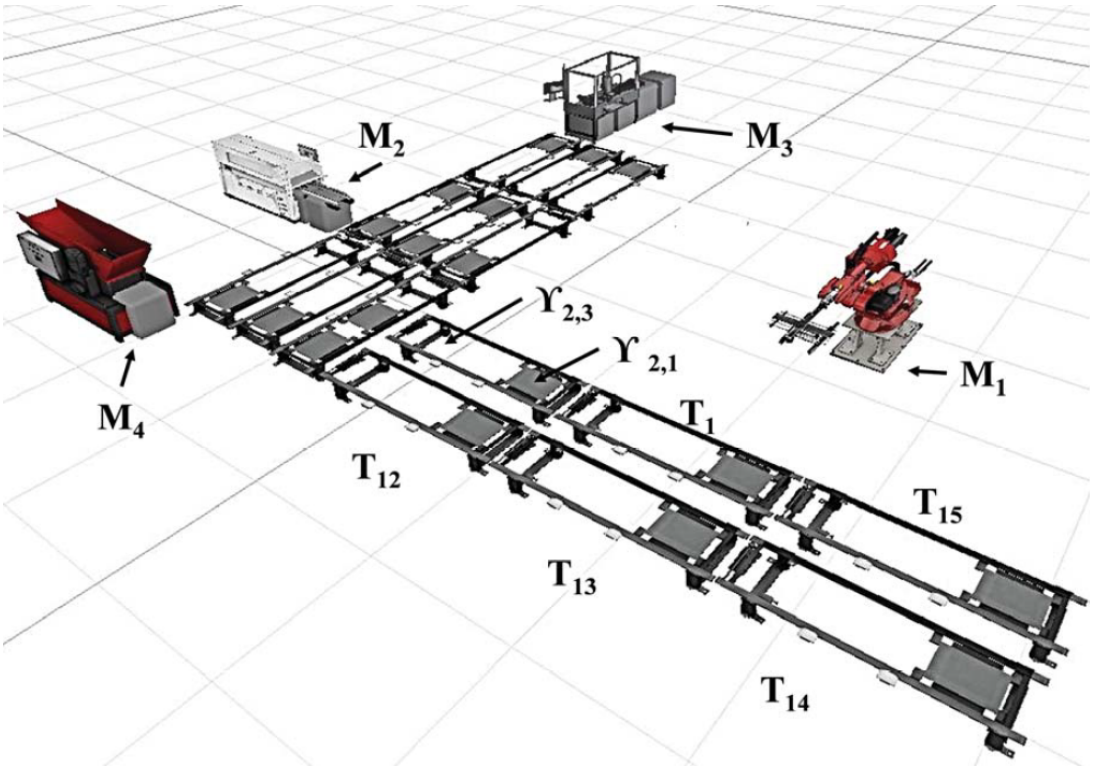

Figure 2.1: Manufacturing plant layout.

3. the pallet is moved to M_3 where the board is **repaired**;
4. the pallet is moved back to M_2 and testing is repeated. If the board works properly, the pallet is sent back to M_1 to be **unloaded**; otherwise it is sent to M_4 where the board is **discharged and destroyed**.

The transport line is equipped with actuators for moving the pallet throughout the different transport modules. Each action is represented in Figure 2.2 as an arrow. On a single transport module, up to three pallets can lay in three adjacent positions called Buffer Zones *(BZ)*. The actual number of BZs available on each transport module depends on its specific layout configuration: some modules will only use one BZ, some modules will use two BZs and some other modules will use all of their three BZs available. The grey BZs in Figure 2.2 are those that are not used.

2.1.1. Directed graph representation

The plant can be represented via a directed graph where the (available) BZs of the transport line are represented by **circular nodes** and the four machines by **square nodes**, for a total of 35 nodes, which are labeled $N_1 - N_{35}$. More specifically, the BZs are labelled $N_1 - N_{31}$ while the machines $M_1 - M_4$ are labelled $N_{32} - N_{35}$.

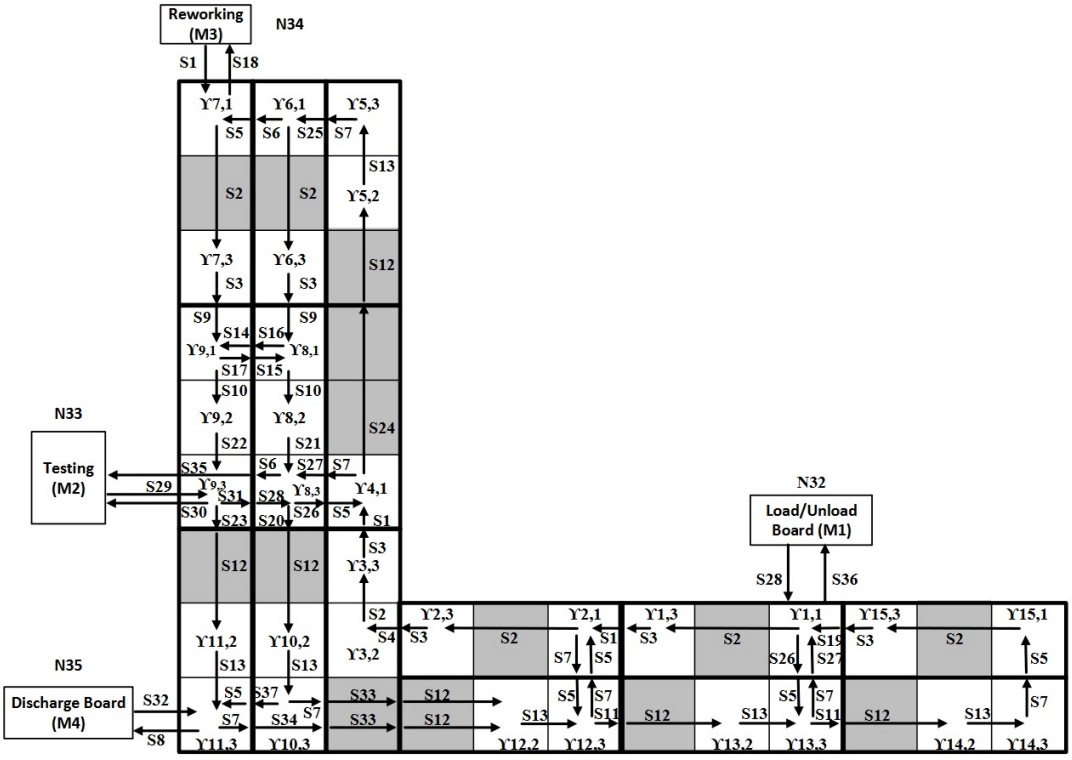

Figure 2.2: Transport modules configuration.

If a pallet can move from node N_i to node N_j then there is an **oriented arch** from node N_i to node N_j, which is labeled by a binary-valued command $u_{i,j}$ that activates/disables the transition ($u_{i,j} = 1/u_{i,j} = 0$).

The directed graph associated to the plant is shown in Figure 2.3.

2.1.2. MLD model

The event-based evolution of the system over the directed graph representing the plant can be described in terms of the *target state* variables Γ_i, $i = 1 \ldots, 35$, associated to the nodes N_i, $i = 1 \ldots, 35$, which take values in $\{0, 1, \ldots, 5\}$:

1. $\Gamma_i(k) = 0$ if the BZ or the machine corresponding to node N_i is empty at k.
2. $\Gamma_i(k) = j$, with $j = 1, 2, 3, 4$ if the BZ or the machine corresponding to the node N_i contains a pallet with a board to be sent, respectively, to the machines M_1 - M_4.
3. $\Gamma_i(k) = 5$ if the BZ or the machine corresponding to node N_i contains a pallet without any target to be reached.

where k denotes the time step associated with the k-th event occurrence.

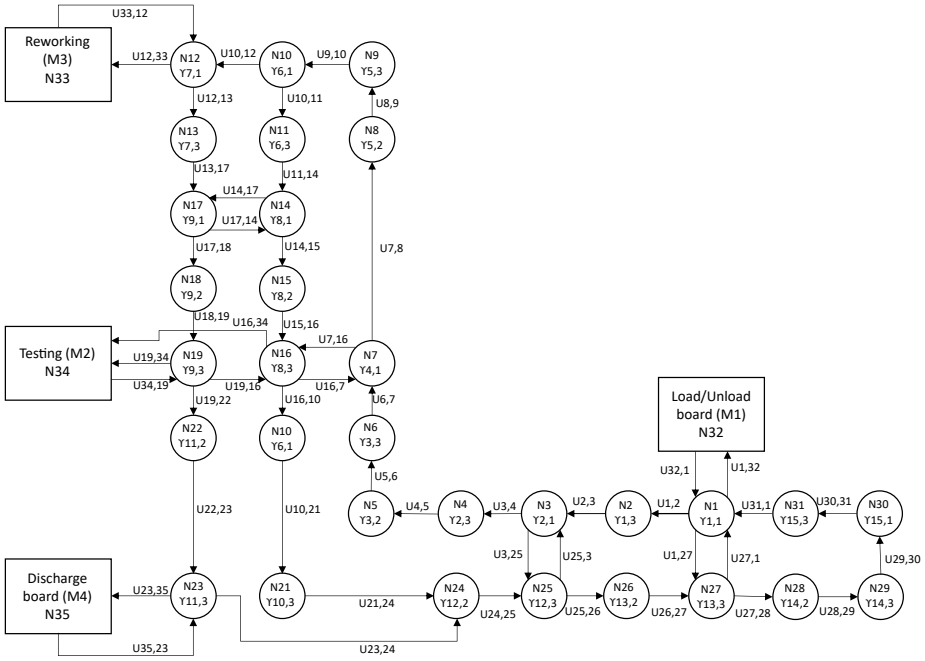

Figure 2.3: Directed graph representation of the plant.

Commands $u_{i,j}$ take values in $\{0, 1\}$:

$$u_{i,j}(k) = \begin{cases} 1 & \text{if the command is active at } k \\ 0 & \text{if the command is not active at } k \end{cases} \tag{2.1}$$

Let

1. $I_{i,in}$ be the set of indices j associated with the commands $u_{j,i}$ which allow to move a pallet to the node N_i from an adjacent node N_j (input commands to node i);
2. $I_{i,out}$ be the set of indices j associated with the commands $u_{i,j}$ which allow to move a pallet from the node N_i to an adjacent node N_j (outut commands from node i).

Then, the fact that only one input command and one output command can be active at a time is translated in the following constraints

$$\sum_{j \in I_{i,in}} u_{j,i}(k) \leq 1, \quad i = 1, ..., 35$$
$$\sum_{j \in I_{i,out}} u_{i,j}(k) \leq 1, \quad i = 1, ..., 35.$$

Also, since when a node is empty, no output commands can be actuated, we have:

$$\Gamma_i(k) = 0 \rightarrow \sum_{j \in I_{i,out}} u_{i,j}(k) = 0, \quad i = 1, ..., 35 \tag{2.2}$$

The target state variables associated with the BZs evolve according to the discrete time equation:

$$\Gamma_i(k+1) = \Gamma_i(k) + \sum_{j \in I_{i,in}} \Gamma_j(k) u_{j,i}(k) - \sum_{j \in I_{i,out}} \Gamma_i(k) u_{i,j}(k), \quad i = 1, ..., 31. \tag{2.3}$$

Each node of the transport line cannot contain more than one pallet. Also, if a node N_i contains a pallet, an input control action $u_{j,i} \in I_{i,in}$ cannot be actuated unless another control action $u_{i,j} \in I_{i,out}$ is activated so as to push out from N_i the loaded pallet. These consitions translate into:

$$\Gamma_i(k) > 0 \wedge \sum_{j \in I_{i,out}} u_{i,j}(k) = 0 \rightarrow \sum_{j \in I_{i,in}} u_{j,i}(k) = 0 \tag{2.4}$$

A distance is associated to each node N_i, $i = 1, ..., 35$, through the following function:

$$\gamma_i(\Gamma_i) = \zeta_{i,1}(\Gamma_i)\phi_{i,32} + \zeta_{i,2}(\Gamma_i)\phi_{i,34} + \zeta_{i,3}(\Gamma_i)\phi_{i,33} + \zeta_{i,4}(\Gamma_i)\phi_{i,35} + (\zeta_{i,0}(\Gamma_i) + \zeta_{i,5}(\Gamma_i))\phi_{i,0}, \tag{2.5}$$

where $\zeta_{i,s}(\Gamma_i), s = 0, ..., 5$ is a binary variable defined as follows:

$$\zeta_{i,s}(\Gamma_i) = \begin{cases} 1 & \text{if } s = \Gamma_i \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

and $\phi_{i,j}$, $j = 32, \ldots, 35$, is the minimum distance between the node N_i and the target machine in node N_j and $\phi_{i,0} = 0$.

This entails that $\gamma_i(\Gamma_i(k)) = 0$ if $\Gamma_i(k) = 0$ or $\Gamma_i(k) = 5$, i.e., the distance will be zero at time k if the node does not contain any pallet or an empty one. In the other cases, $\gamma_i(\Gamma_i(k))$ is equal to the length of the minimal path from N_i to the target machine specified by $\Gamma_i(k)$.

The permanence of the pallet on the transport line must be penalized so as to encourage its movement towards the target machine, thus avoiding deadlocks. This motivates the introduction of a counter η_i, $i = 1, ..., 31$, for each BZ. At each time step, η_i is equal to

the number of steps the pallet in node N_i has been on the line.

In order to define the dynamical model of the counter, two Boolean variables have to be introduced. The first one is $\delta_i(k)$ which is set to one whenever there are no inputs and outputs from/to node N_i, $i = 1, ..., 31$:

$$\delta_i(k) = \begin{cases} 1, & \sum_{j \in I_{i,in}} u_{j,i}(k) = 0 \wedge \sum_{j \in I_{i,out}} u_{i,j}(k) = 0 \\ 0, & \text{otherwise.} \end{cases}$$

The second Boolean variable is $\vartheta_i(k)$ which is set to one whenever the node N_i, $i = 1, ..., 31$, contains a pallet with a board to be sent to one of the machines:

$$\vartheta_i(k) = \begin{cases} 1, & 1 \leq \Gamma_i(k) \leq 4, \\ 0, & \text{otherwise} \end{cases}$$

Thus, the dynamic equation for the counter of the node N_i, $i = 1, ..., 31$, is given by:

$$\eta_i(k+1) = \eta_i(k) + \delta_i(k)\vartheta_i(k) + \sum_{j \in I_{i,in}} [\eta_j(k) + 1]\vartheta_j(k)u_{j,i}(k) - \sum_{j \in I_{i,out}} \eta_i(k)\vartheta_i(k)u_{i,j}(k).$$

The four machines M_1, M_2, M_3 and M_4, corresponding to nodes N_i, $i = 32, \ldots, 35$, are described by a Finite State Machine (FSM) (see Figure 2.4) with the following three Boolean-valued states:

- $x_{i,1}$: idle and empty machine;
- $x_{i,2}$: manufacturing;
- $x_{i,3}$: end manufacturing, loaded machine;

The transitions between the states of the FSM are determined by these conditions:

- when N_i is in state $x_{i,1}$, it is idle and its counter is set to zero counter ($n_i = 0$). Whenever a pallet sits inside a node N_j adjacent to the machine and the control action $u_{j,i}$ is fired, its state switches from $x_{i,1}$ to $x_{i,2}$. In order to model this transition, the following implication must be implemented:

$$x_{i,1}(k) \wedge \sum_{j \in I_{i,in}} u_{j,i}(k) = 1 \rightarrow \begin{cases} x_{i,1}(k+1) = 0 \\ x_{i,2}(k+1) = 1 \end{cases} \tag{2.7}$$

- While N_i is in state $x_{i,2}$, the counter n_i is increased at every step until it reaches a

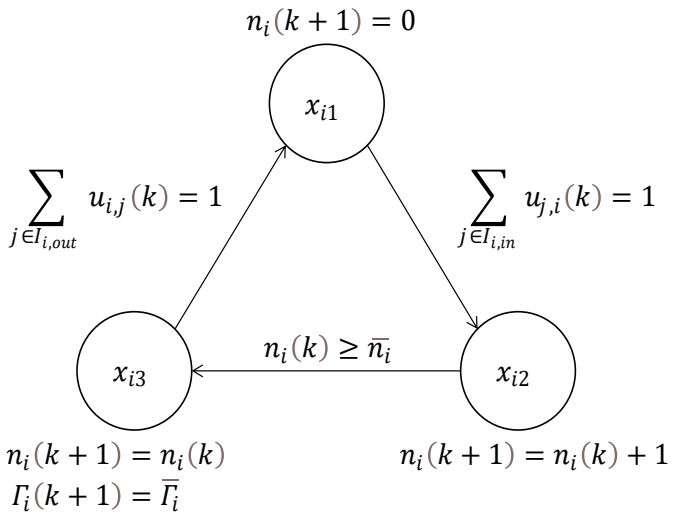

Figure 2.4: FSM model of a machine.

given threshold $\bar{n}_i$, whose value corresponds to the end of the working phase. The machine then switches from x_{i2} to x_{i3}. This is implemented through the following implication:

$$x_{i,2}(k) \wedge (n_i(k) \geq \bar{n}_i) \rightarrow \begin{cases} x_{i,2}(k+1) = 0 \\ x_{i,3}(k+1) = 1 \end{cases} \tag{2.8}$$

- When N_i is in state $x_{i,3}$, the counter is kept constant at the maximum reached value $\bar{n}_i$ and a new target value $\bar{\Gamma}_i$ is assigned to the pallet. In particular, $\bar{\Gamma}_{32} = 2$, $\bar{\Gamma}_{34} = 3$ or 4 or 5 (3: the reworking has to be repeated; 4: the board cannot be repaired and it must be discharged and destroyed; 5: the board is properly working and it can be unloaded from the pallet and stored in the warehouse), $\bar{\Gamma}_{33} = 2$, and $\bar{\Gamma}_{35} = 0$. The values of $\bar{n}_i$ for the four machines are: $\bar{n}_{32} = 11$, $\bar{n}_{33} = 11$, $\bar{n}_{34} = 10$ and $\bar{n}_{35} = 9$. As soon as the control action $u_{i,j}$ in $I_{i,out}$ is activated, the pallet is moved out from the machine to the adjacent BZ N_j of the transport line and the FSM state switches from x_{i3} to x_{i1}, the counter $n_i(k)$ is reset.

$$x_{i,3}(k) \wedge \sum_{j \in I_{i,out}} u_{i,j}(k) = 1 \rightarrow \begin{cases} x_{i,2}(k+1) = 0 \\ x_{i,3}(k+1) = 1 \end{cases} \tag{2.9}$$

Finally, there are additional constraints that must be imposed so as to let the machine operate as intended.

A pallet cannot enter a machine if it is not in $x_{i,1}$, i.e., the control action $u_{j,i}$ in $I_{i,in}$ must be disabled if the machine is processing or is still loaded after finishing the operations:

$$x_{i,1}(k) = 0 \rightarrow \sum_{j \in I_{i,in}} u_{j,i}(k) = 0, \quad i = 32, ..., 35 \tag{2.10}$$

A pallet cannot exit the machine if it is not in $x_{i,3}$, i.e., the control action $u_{i,j}$ in $I_{i,out}$ must be disabled if the machine is idle or is manufacturing:

$$x_{i,3}(k) = 0 \rightarrow \sum_{j \in I_{i,out}} u_{i,j}(k) = 0, \quad i = 32, ..., 35 \tag{2.11}$$

Letting $\delta_{i,23}(k)$ be a Boolean variable representing the logic condition associated to the transition from $x_{i,2}$ to $x_{i,3}$, i.e.:

$$x_{i,2}(k) \wedge (n_i(k) \geq \bar{n_i}) \leftrightarrow \delta_{i,23}(k) \tag{2.12}$$

we can express the dynamic equations regarding the pallet target $\Gamma_i(k)$ and the counter $n_i(k)$ associated with the generic machine $N_i, i = 32, ..., 35$, as follows:

$$\Gamma_i(k+1) = \Gamma_i(k) + \sum_{j \in I_{i,in}} \Gamma_j(k) u_{j,i}(k) - \sum_{j \in I_{i,out}} \Gamma_i(k) u_{i,j}(k) + \delta_{i,23}[\bar{\Gamma}_i - \Gamma_i(k)]$$
$$n_i(k+1) = [n_i(k) + x_{i2}(k)][1 - x_{i1}(k)].$$

By using the HYSDEL tool [14], the dynamic model of the plant described in this section can be translated into a MLD system of the following form

$$\begin{cases} x(k+1) = Ax(k) + B_u u(k) + B_\delta \delta(k) + B_z z(k) \\ y(k) = Cx(k) + D_u u(k) + D_\delta \delta(k) + D_z z(k) \\ E_x x(k) + E_u u(k) + E_{aux} w(k) \leq E_{aff} \end{cases} \tag{2.13}$$

where x is the vector of the state variables, u is the vector of the control actions, δ is the vector of Boolean auxiliary variables and z is a vector of continuous auxiliary variables (see the PhD thesis [4] for more details).

2.2. MPC Formulation

According to the MPC approach proposed in the PhD thesis [4] to automatize the transport line, the following optimization problem is solved at every time step k:

$$\text{minimize } J_k \tag{2.14}$$

$$\text{subject to: } (2.13) \tag{2.15}$$

where J_k is a finite-horizon cost over some prediction horizon of length N_{RH}:

$$\begin{aligned} J_k = \sum_{h=1}^{N_{RH}} \{ \underbrace{\sum_{i=1}^{35} \gamma_i(\Gamma_i(k+h))}_{(a)} + \underbrace{\sum_{i=32}^{35} q_{xi} x_{i3}(k+h)}_{(b)} + \underbrace{\sum_{i=1}^{31} q_{\eta_i} \eta_i(k+h)}_{(c)} + \\ + \underbrace{\sum_{(i,j)\in I_u} q_{u_{i,j}} u_{i,j}(k+h-1)}_{(d)} + \underbrace{\sum_{(m,r,i,j)\in\Psi} \lambda_{m,r}\sigma_m(k+h-1)u_{i,j}(k+h-1)}_{(e)} \}. \end{aligned} \tag{2.16}$$

The 5 contributions to the cost J_k represent:

(a) the distance of the pallets from their targets;

(b) the permanence of the pallets in the states $x_{i,3}, i = 1, ..., 4$ of the machines;

(c) the permanence of the pallets on the transport line;

(d) the control actions;

(e) the permanence of a pallet in the nodes adjacent to $M_1 - M_4$.

The last term (e) is introduced to allow the manufactured pallets to exit the machine. It includes the variables σ_m, $m = 32, \ldots, 35$, which are defined as:

$$\begin{cases} \sigma_{32} = 1 \leftrightarrow (\Gamma_1(k) = 1 \vee \Gamma_{32}(k) = 5 \vee \vartheta_{32}(k) = 1) \\ \sigma_{33} = 1 \leftrightarrow (\Gamma_{12}(k) = 3 \vee \vartheta_{33}(k) = 1) \\ \sigma_{34} = 1 \leftrightarrow (\Gamma_{19}(k) = 2 \vee \vartheta_{34}(k) = 1) \\ \sigma_{35} = 1 \leftrightarrow (\Gamma_2 3(k) = 4 \vee \Gamma_{35}(k) = 5 \vee \vartheta_{35}(k) = 1) \end{cases} \tag{2.17}$$

Summation is taken with indices ranging in the set
$\Psi = \{(32, 1, 27, 1), (32, 2, 31, 1), (34, 1, 7, 16), (34, 2, 15, 16), (34, 3, 8, 19), (33, 1, 10, 12), (35, 1, 22, 23)\}$.

Each term in the cost (2.16) is weighted by a coefficient. Large values of q_{xi} significantly

penalize pallets staying inside a machine after the processing is done. Small positive values of q_{η_i} are useful to include the integral effect on the permanence of a pallet on the transport line. Giving small values to $q_{u_{i,j}}$ is likely to result in the reduction of useless commands to the actuators. In order to avoid deadlocks in the nodes adjacent to the machines $M_1 - M_4$, the values of $\lambda_{32,1}, \lambda_{32,2}, \lambda_{34,1}, \lambda_{34,3}, \lambda_{33,1}$ and $\lambda_{35,1}$ must be chosen large enough.

The value of the prediction horizon N_{RH} should be selected large enough to avoid possible deadlocks due to conflicting paths of the pallets and less than or equal to the number of steps $\bar{n_i}$ required by the machines M_i to work the pallets otherwise the optimization problem would not activate the command to load the machines due to the high penalty on their states x_{i3}.

At each time k, the cost (2.16) is minimized subject to the constraint of the MLD system (2.13). This results in a Mixed Integer Linear Program (MILP) with dimension that grows with the prediction horizon length and also with the number of pallets which do not increase the dimension of the system but make the problem more complex to be solved.

3 | Reachability-based decoupling solution

In this chapter, we explain the proposed strategy to address the combinatorial complexity of the MPC approach to the routing problem in the PhD thesis [4].

3.1. Description of the proposed approach

Starting from the observation that when the pallets are occupying different areas of the plant, they will not interfere, we introduce a reachability-based method to decouple the MPC optimization problem in multiple lower-dimensional problems that can be solved in parallel, each one defined on an appropriate reduced model of the MLD system modeling the overall plant.

To translate this idea in practice, we shall adopt the following strategy:

1. perform reachability analysis on the directed graph representing the plant so as to identify the nodes that each pallet can reach within the prediction horizon;
2. determine the reduced MLD systems modeling the sub-graphs by pruning variables and constraints from the complete MLD system;
3. compute the MPC solution for every reduced MLD system;
4. check whether or not the optimal control actions to be applied at the current time step create any conflict. If no conflict is found, then the computed actions are applied according to the receding horizon strategy. Otherwise, the sub-graphs of the conflicting pallets are joined together and a new MPC problem is defined for the resulting MLD model. This step is repeated until there are no conflicts.

Certain zones of the plant are prone to deadlock situations, we shall then group together the pallets that are within these *critical zones* and perform reachability analysis for them jointly since the very first iteration of the decomposition algorithm.

In the rest of the chapter, we shall detail step 1 on reachability analysis, including the definition of the critical zones, and step 2 on the MLD reduced system associated with a sub-graph.

3.2. Reachability analysis

In order to perform reachability analysis, we shall consider the directed graph representation of the plant presented in Chapter 2 and reported in Figure 3.1 for ease of reference, with additional self-loops on each node modeling the fact that a pallet can remain in that node more than 1 time step.

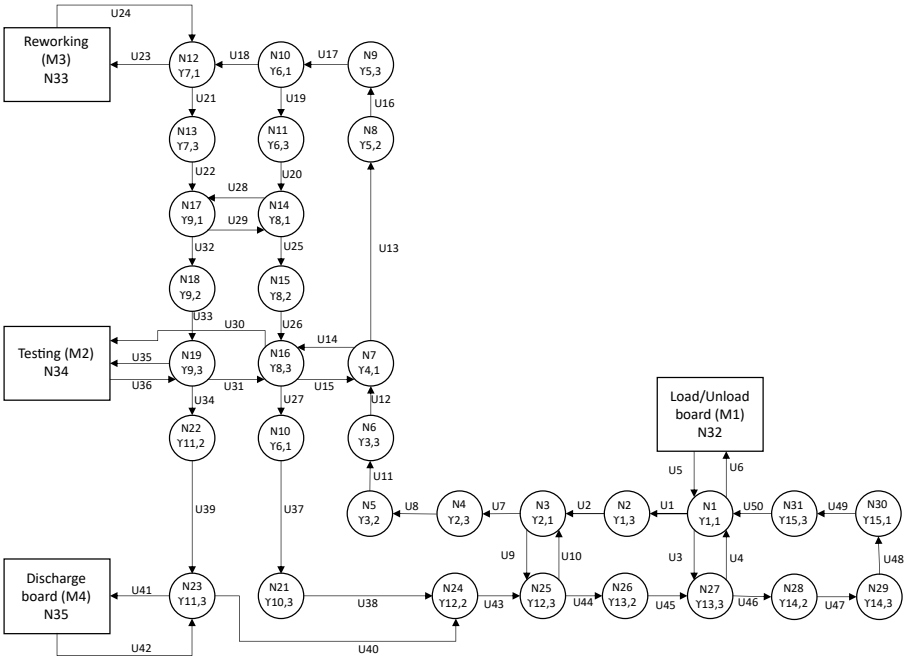

Figure 3.1: Directed graph of the plant.

We shall associate to this graph with $n = 35$ nodes a 35-by-35 *adjacency matrix* A whose elements satisfy

$$a_{i,j} = \begin{cases} 1, & \text{if there is an arrow from } N_j \text{ to } N_i \\ 0, & \text{otherwise.} \end{cases}$$

Due to the self-loops at each node, we then have that $a_{i,i} = 1$, $i = 1, \ldots, 35$.

In order to determine what are the nodes that a subset of m_{sub} pallets out of a total of $m \geq m_{sub}$ can possibly reach over the prediction horizon N_{RH}, we perform the following steps:

- define a column vector X_0 with all elements equal to zero except those corresponding

to the position of the pallets that are set equal to 1

- compute

$$X_{N_{RH}} = A^{N_{RH}} X_0$$

The nodes that can be reached by some of the considered pallets in the look-ahead time horizon of length N_{RH} are those corresponding to the elements in $X_{N_{RH}}$ that differ from zero.

For the sake of clarity, we shall next present an example. Consider the graph in Figure 3.2 related to a zone around machine M_3.

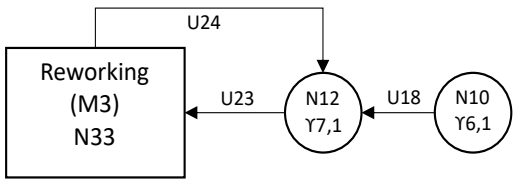

Figure 3.2: Graph of a zone around M_3.

Its adjacency matrix is reported in (3.1), where the first column refers to N_{10}, the second to N_{12}, and the third to N_{33}, the same for the rows.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \tag{3.1}$$

Now, let

$$X_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.2}$$

be the initialization vector of the considered graph when there is only a pallet in N_{10}. By multiplying the adjacency matrix A in (3.1) of the considered graph with the initialization vector (3.2), we get vector X_1 showing all the nodes that can be reached after one step

from the initial starting node N_{10}:

$$X_1 = AX_0 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

thus revealing that in one step the pallet can either stay in N_{10} or move to N_{12}. In 2 steps instead, we get

$$X_2 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \tag{3.3}$$

and hence the pallet can either stay in N_{10} or N_{12}, move from N_{10} to N_{12} or move from N_{12} move to N_{33}.

Reachability analysis needs to be performed for multiple subsets of pallets. It can also be done in parallel by defining X_0 as a matrix instead of a vector, with as many columns as the number of considered subsets.

When decomposing into smaller sub-graphs, there is a broad range of cases that could happen, the two extreme ones being:

- the best case scenario, where each pallet acts independently, so that one big optimization problem can be decomposed into m smaller optimization problems to be solved in parallel, thus significantly improving the computing time;
- the worst case scenario, where all pallets on the transport line are grouped together in one single reachability sub-graph and one single MPC problem is formulated.

Notably, also the worst case scenario can save computing time with respect to the original problem, since variables and constraints tied to the non reachable nodes will be removed from the original MLD system modeling the plant and a simplified MPC optimization will be formulated.

Before dividing the problem into smaller ones, a preliminary analysis must be conducted. Indeed, given certain conditions that will be thoroughly explained in the following section, two or more pallets will need to be grouped together in the first place, without trying first to keep them in separate sub-graphs, in order to avoid deadlocks and/or save computational resources.

3.3. Preliminary grouping for deadlock avoidance

We shall next identify some specific clusters of nodes that are likely to create a deadlock situation when occupied by two or more pallets. All the pallets in a critical zone will then be grouped together when decoupling the MPC problem in multiple smaller ones.

Off-limit zones

When a machine has completed the processing operations, it should be able to take the pallet carrying the board to its output node, which then must be empty. However, from Figure 3.1 one can see that every output node for a machine is also an input node for that same machine. If the pallets move independently, then some pallet could end up being stuck in the output node of a machine while it is busy in its working state; this positioning could prevent the machine from ever dispatching the processed pallet and, at the same time, could prevent the pallet from ever entering the machine.

As an example, suppose that at $k = 1$ a pallet, called pallet A, sits in N_{19} and wants to enter machine M_2 which in turn is occupied by another pallet, called pallet B. Machine M_2 is in its processing state $x_{34,2}$. At $k = 2$ pallet A tries to enter M_2, causing a collision with pallet B inside. Thus, both pallets will be coupled and their trajectories will be calculated by the same MPC, which is going to let pallet A sit still in N_{19} while M_2 finishes its operations. Then, at $k = 3$, both pallets restart to act independently: M_2 finishes its processing operations and enters the state $x_{34,3}$, waiting to eject pallet B at the next time instant. Meanwhile, pallet A tries again to enter M_2, but since it is still occupied, a conflict will arise; the MPC will calculate the trajectories for both pallet A and pallet B and no pallet is be moved in this time instant. Then, at $k = 4$, pallet B is now ready to exit the machine, while pallet A tries again to enter M_2 but, in order to swap places, both pallets would need to cross each other outside of their respective nodes: this is not feasible in the real transport line due to the way that the transport modules are built. Only one pallet at a time can cross the space between two nodes, thus making a swap unfeasible. Since the only combination of control actions that can resolve this situation is unfeasible, this state must not be reached.

In order to solve this problem, *off-limit zones* must be implemented. These zones are pre-defined sub-graphs within the transport line that need to be preserved in the MLD reduced model. These zone are already described in the PhD thesis [4] which implements them in the MLD constraints of the whole system. The issue is that when the system is reduced, the off-limit zones as implemented by Cataldo are lost in the pruning process (described in

Chapter 3). To avoid this, each off-limit zone must be included in a single MPC problem with all the nodes that it contains. This allows to avoid losing important variables and constraints in the pruning process and to avoid situations as the one described in the previous example, where two pallets are blocking each other in front of a machine. The off-limit zones described in the PhD thesis [4] involve the following nodes:

- Off-limit zone of machine M_1 involves N_1, N_{27}, N_{31}, N_{32}
- Off-limit zone of machine M_2 involves N_7, N_{15}, N_{16}, N_{18}, N_{19}, N_{34}
- Off-limit zone of machine M_3 involves N_{10}, N_{12}, N_{33}
- Off-limit zone of machine M_4 involves N_{19}, N_{22}, N_{23}, N_{35}

In our approach, the off-limit zone of machine M_1 is modified so as to include three more BZs: N_{28}, N_{29}, N_{30}. These three nodes, together with node N_{31} act as a buffer for the machine M_1, this means that one or more pallets will wait here to be processed by the machine M_1. If these pallets were to act independently one from another, they would generate a conflict with the pallet in front of them at each time instant; this is caused by each pallet trying to reach its destination (i.e. M_1) and ignoring the line. This conflict forces the grouping of the conflicting pallets, requiring a new iteration of the MPC in order to manage them. To prevent one or more buffer collisions and thus save up some run time, these three nodes are included in the off-limit zone of M_1 as well.

Regarding the off-limit zone of the machine M_2, a similar reasoning applies and the nodes N_4, N_5 and N_6 are added because they act as a buffer for the pallets that need to reach machine M_2. It is also important to observe that nodes N_{19} and N_{16} both act as input nodes for the machine M_2, so it is of paramount importance to coordinate traffic and to avoid clutter between the buffer and the input nodes as efficiently as possible.

The off-limit zone of the machine M_3 is fairly straightforward. Its goal is to avoid deadlocks around the output node N_{12}.

Finally, it must be noted that the off-limit zone of the machine M_4 shares a node with the off-limit zone of the machine M_2, that is node N_{19}. This is due to the fact that the nodes N_{19} and N_{22} act as a buffer for the machine M_4. So, the off-limit zones of the machines M_2 and M_4 could be joined together in order to avoid clutter in a very critical section of the plant.
Moreover, node N_{20} needs to be included as well. Suppose that there are two pallets acting independently, one in N_{23} and one in N_{20} and they both need to go to the machine M_1. In the real physical plant, the control action U_{40} which connects N_{23} and N_{24} passes through N_{21} and reaches N_{24} in the same time step. If U_{37} fires in the same step, the check

to see if there is a collision would give a negative response since at the end of the step, the pallet from N_{23} would be in N_{24} while the pallet from N_{20} would be in N_{21}. But this would not be possible in the real physical plant, since one of the two pallets has to stop in its node for a single time step in order for the other pallet to move. If the two pallets happen to be controlled by the same MPC, then the transit over N_{21} can be regulated without incurring in any unfeasible movement.

Figure 3.3 shows all of the nodes which are included inside of an off-limit zone. These nodes are circled in red.

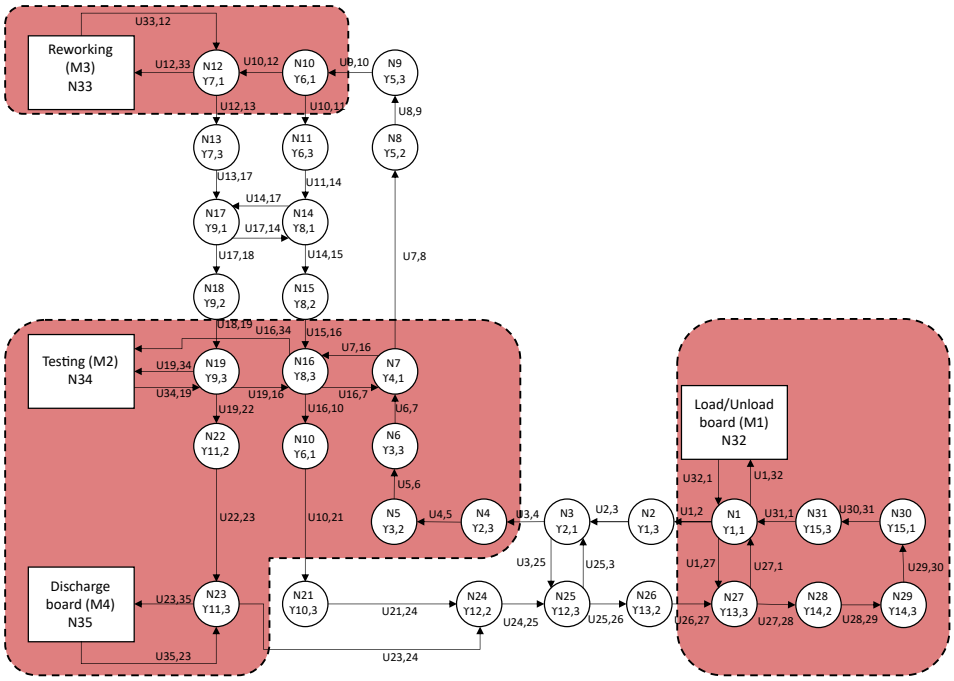

Figure 3.3: Visual representation of the off-limit zones.

Exchange zones

An *exchange zone* is defined as a sub-graph containing a couple of adjacent nodes where a pallet is able to move from one node to the other and vice versa. There are several examples of exchange zones in the plant: each machine and their adjacent node constitutes an exchange zone, since it is possible to travel from the node to the machine and vice versa. Let us take N_3 and N_{25} as another example: these BZs are adjacent to each other and it is possible for a pallet to travel from N_3 to N_{25} and vice versa, thus qualifying this area as an exchange zone.

Suppose, as an example, that there are two pallets: one in N_7 which wants to move

towards N_{16} and one in N_{16} which wants to move towards N_7. If they move at the same time independently one from the other then what would happen is that they would cross each other outside of the nodes, thus creating a collision outside of the graph. This kind of behaviour has to be prevented somehow: in order to avoid the pallets from swapping positions simultaneously between the two nodes, each exchange zone will have its MPC which coordinates the movement. As a matter of fact, whenever two pallets fall into the same exchange zone, their trajectories are computed by the same MPC algorithm, so as to avoid a collision from happening outside of the nodes. All of the exchange zones are highlighted in orange in Figure 3.4.

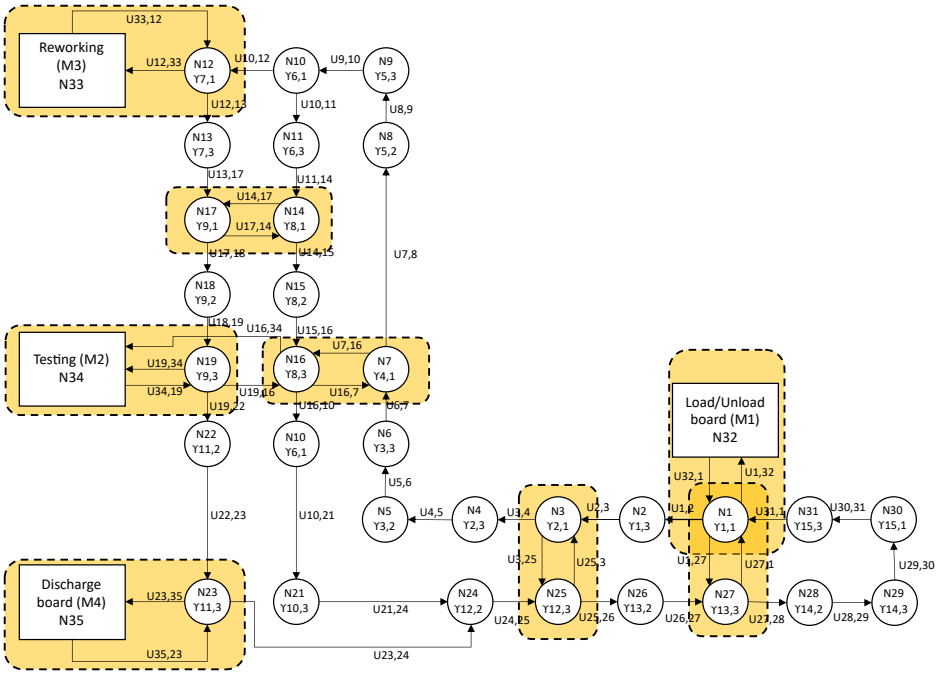

Figure 3.4: Exchange zones of the plant.

Most of the exchange zones happen to be inside of the already established off-limits zones, so the exchange zones within an off-limits zone are discarded, since there is already an MPC coordinating the traffic inside an off-limits zone. Two zones happen to be outside of every off-limits zone and are listed as follows:

- Exchange zone 1 involves the nodes N_3 and N_{25}
- Exchange zone 2 involves the nodes N_{14} and N_{17}

An accurate depiction of both the exchange zones and the off-limit zones is illustrated in Figure 3.5 so as to properly show the nodes where the pallet will act independently from another and the zones where the pallets will act in a coordinated fashion.

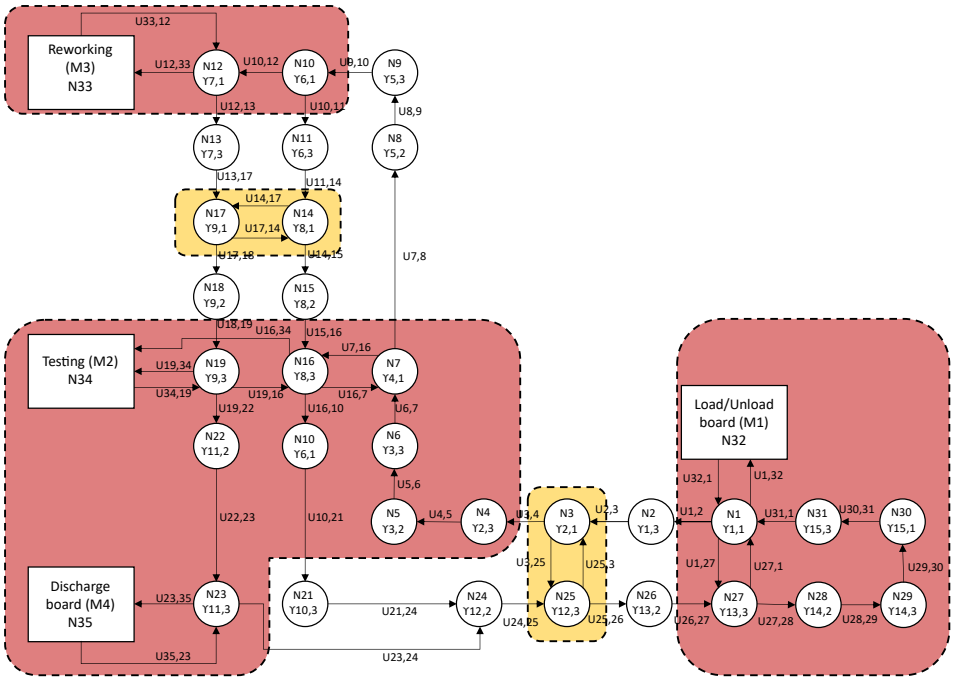

Figure 3.5: Exchange zones (yellow) and off-limit zones (red) of the plant.

Let us consider the case when there are three pallets in the transport line, respectively located in N_1, N_7 and N_{10}. In this configuration, each one will be independent of the others, and its trajectory will be computed by the associated MPC. If, instead, the pallets are placed in nodes N_1, N_3 and N_{25}, then the pallet in N_3 is independent and moved by its corresponding MPC, while the trajectories of the two pallets in the same exchange zone (N_3 and N_{25}) are handled in the same optimization problem.

3.4. Deriving the MLD system of a sub-graph

The goal of this section is to describe how to tie every variable and constraint of the system to the node they refer to, such that when the reachable set of nodes of every sub-graph with at least one pallet is identified, the variables and constraints to be pruned from the MLD system (2.13) modeling the overall plant will be known as well. This will then lead to the derivation of the reduced MLD system of the sub-graph.

The association of plant nodes with their respective variables, within a table, is a rather simple process when considering the input, state and output variables, as they are relatively few (there are a total of 82 state variables, 50 input variables and 35 output variables and a total of 35 nodes). The same process becomes more complex when auxiliary variables and system constraints are brought into play: there are respectively 607

auxiliary variables and 2904 constraints and in order to build a table to associate them with a node, an algorithm is needed.

Once the associations with the nodes have been performed, it is necessary to derive the MLD model of the reduced systems. The process begins by considering the matrices of the MLD system, as defined in Equation 2.13. Once the reachable nodes are known, the matrices are reduced by removing the rows and the columns associated to the non-reachable variables, thus reducing the size of the matrices. Moreover, all of the control actions of the non-reachable nodes are set to zero.

3.4.1. State variables

The 82 state variables are subdivided into four main categories:

- Target state variables Γ_i, $i = 1, ..., 35$. Every node N_i of the plant is associated with its own target which is represented by an integer number ranging from 0 to 5: $\Gamma_i = 0$ if the node is empty; $\Gamma_i = j$, $j = 1, 2, 3, 4$, if the node contains a board to be sent respectively to the j-th machines $M_1 - M_4$; $\Gamma_i = 5$ if the node contains a pallet without any target.
- State variables η_i, $i = 1, ..., 31$, for the counter associated to each BZ. One single counter is defined for each BZ and its value is set, at each time instant, to the number of instants in which the corresponding pallet has been in the transport line. The counter resets when the pallet enters a machine, since its target has been reached.
- FSM binary state variables related to the M_i. x_{i1} is related to the idle state, x_{i2} is related to the working state and x_{i3} is related to the waiting state.
- State variables n_i, $i = 1, ..., 4$, for the event counter.

Starting from N_1, the state variables associated with it are chosen as follows: first, the target state variable associated to BZ_1, that is Γ_1, is picked; then, the state variable for the counter associated to BZ_1, that is η_1. Since the node N_1 is a BZ and not a machine, there is no event counter, nor for the three Boolean state variables of the target automata modelling.
The same reasoning must be applied to the remaining BZs of the plant. So, each N_i for every $i = 1, ..., 31$ will be associated to two variables: Γ_i and η_i. Since there are a total of 31 BZs, this will make up for a total of 62 state variables. The 20 remaining state variables are all divided between the four machines of the transport line.

As for the nodes representing the machines, let us consider the machine M_1 as an example.

N_1	$\Gamma_1,$	η_1
N_2	$\Gamma_2,$	η_2
...	...	...
N_{31}	$\Gamma_{31},$	η_{31}

Table 3.1: Mapping the BZs with their state variables

First of all, the target state variable Γ_{32} associated to N_{32} is picked. Then, the event counter variable n_{32} associated to N_{32} is picked as well. Finally, the three state variables for the Target Automata modelling ($x_{32,1}$, $x_{32,2}$, $x_{32,3}$) associated to N_{32} are chosen. So, there is a total of five state variables associated to a machine. This process can be generalized for each one of the four machines. Thus, the node N_i, when $i = 32, ..., 35$, will be associated to five state variables, the pallet target Γ_i, the event counter n_i and the three state variables for the target automata modelling $x_{i,1}$, $x_{i,2}$, $x_{i,3}$.

N_{32}	$\Gamma_{32}(k),$	$n_{32}(k),$	$x_{32,1},$	$x_{32,2},$	$x_{32,3}$
N_{33}	$\Gamma_{33}(k),$	$n_{33}(k),$	$x_{33,1},$	$x_{33,2},$	$x_{33,3}$
N_{34}	$\Gamma_{34}(k),$	$n_{34}(k),$	$x_{34,1},$	$x_{34,2},$	$x_{34,3}$
N_{35}	$\Gamma_{35}(k),$	$n_{35}(k),$	$x_{35,1},$	$x_{35,2},$	$x_{35,3}$

Table 3.2: Mapping the nodes of the machines with their respective state variables

A table associating each node with their variables can then be built. Once all the nodes reached within N_{RH} steps are known (based on the reachability analysis described in Section 3.2), it is straightforward to identify which state variables to preserve and which ones to prune.

3.4.2. Input variables

Each node is associated with a subset of the control actions. Since there are a total of 50 input variables for only 35 nodes, each node is associated with one or more control actions. It is necessary to build a table linking each node of the graph to their respective control actions. For example, there are three control actions associated to N_1: U_1 is directed towards N_2, U_3 is directed towards N_{27} and U_6 is directed towards M_1, that is N_{32}. So, by repeating this for every node of the graph, Table 3.3 can be obtained.

Once that the table is built, all the input associated to the nodes reached within $N_{RH} - 1$ steps will be preserved.

N_1	U_1, U_3, U_6
N_2	U_2
N_3	U_7, U_9
...	...
N_{35}	U_{42}

Table 3.3: Mapping the nodes with their respective input variables.

3.4.3. Output variables

The output variables are defined in Equation 2.5 as a piecewise affine function of the BZ's distance from the target to be reached. Thus, each node has its own output variable and there is no need to build a table to associate an output variable to a node, since the connection is straightforward.

In order to prune the output variables, it is sufficient to know the nodes that will be reached within N_{RH} steps from the initial position. The only output variables that will not be pruned are the ones tied to the reachable nodes.

3.4.4. Auxiliary variables

Before pruning the auxiliary variables, they must be associated with the node they refer to. First, the complete list of auxiliary variables, together with their declaration, can be found within an HYSDEL file, where the MLD system is defined. Once the auxiliary variable list is obtained from HYSDEL, it is important to understand their counterpart in the MLD model.

By searching in the HYSDEL file for all occurrences of a certain auxiliary variable name within the constraints and equations, it is possible to deduce the corresponding variable in the MLD model and, finally, the node in the graph of the plant transport line to which the variable refers (e.g. the auxiliary variable `dTpNotFree15` is associated to $\theta_{15}(k)$, which is in turn associated to N_{15}).

In the appendix it is possible to find a table where the associations between the auxiliary variables in HYSDEL and their counterpart in the MLD model are shown, together with the nodes to which they refer.

Following this reasoning, it is possible to create a map associating one or more auxiliary variables to each node of the plant.

3.4.5. Constraints

Given the large number of constraints (total of 2904) that characterize this system, there is a need for a quick and efficient method to parse the constraints that must be pruned.

The previous sections demonstrate how to associate the state, input, output and auxiliary variables to each node of the system. Once the reachable nodes are known, all the variables associated with these nodes must be collected into a single set, called V_r. Then, a set called V_i with $i = 1, \ldots, 2904$ is defined as the set of all the variables involved in the i-th constraint.

For example, the 1853rd constraint is:

$$\Gamma_1(k) \leq 1 + 4\theta_1(k)$$

Then, its associated set is $V_{1852} = \{\Gamma_1(k), \theta_1(k)\}$.

Whenever $V_r \cap V_i \neq \emptyset$ for $i = 1, \ldots, 2904$, the i-th constraints must not be pruned because it contains and affects one or more of the variables associated to the reachable nodes. Once the intersections have been evaluated, we must select all the indices for which we have a non-zero intersection. These indices allow us to know which constraints are not to be pruned, keeping only those essential to the functioning of the subsystem involved.

4 | Simulation Results

In this chapter, we analyse the performance of the approach described in the previous chapters and compare it with the original MPC method in the PhD thesis [4] via extensive simulations. To this purpose, we adopt the same MLD system of the overall plant and MPC problem formulation.

The *online computation time* is used to compare the two approaches and assess the improvement obtained by the reachability-based decomposition approach proposed in this thesis. We also determine the number of *machined pallets* in a certain time interval to verify that the throughput of the manufacturing system is unchanged.

The YALMIP toolbox for MATLAB has been used for the MILP problem definition and IBM ILOG CPLEX solver for its solution. The software implementation of the plant is discussed in [5].

4.1. Online computation times

Simulations are run for different values of the prediction horizon N_{RH} and of the number m of pallets on the transport line. The fastest machine of the transport line is M_4 which completes the discharge of an electronic board in about 45 s. Now, it is very important to keep in mind that on average the actuation of the pallets requires about 5s to move them from a BZ to an adjacent one. If the online optimization does not exceed 10 s, the overall time required to compute and actuate the control action is about 15 s, allowing for a total of circa 3 movements during a cycle of the machine M_4 (10 s of computation and 5 s of pallet movement). The goal is to try and reduce the overall time required to compute the control actions such that more moves can be done in the same amount of time.

For each pair (N_{RH}, m), MPC is applied for 100 steps. The average computation time per step for the approach proposed in this thesis and for that in PhD thesis [4] are reported in Tables 4.1 and 4.2, respectively.

As for the time per step of the approach in this thesis, we need to recall that the input

applied at each step is determined by possibly performing a few iterations where the system is decomposed into smaller ones whose MPC solution is computed in parallel, and smaller systems are joined, if needed, to eliminate conflicts by recomputing the MPC solutions in parallel. As a result, the computation time per step is given by the sum over all iterations needed to eliminate conflicts of the worst time for the parallel MPC computations.

N. Pallets	$N_{RH} = 4$	$N_{RH} = 5$	$N_{RH} = 6$	$N_{RH} = 7$
3	0.22	0.31	0.43	0.58
4	0.26	0.39	0.58	0.96
5	0.28	0.43	0.76	1.65
6	0.28	0.45	0.82	1.94
7	0.29	0.46	0.90	2.18
8	0.33	0.55	2.19	4.11
9	0.47	0.74	2.40	4.18
10	0.78	1.55	5.89	10.67

Table 4.1: Online computation time (in seconds) as a function of the number of pallets and the prediction horizon for the proposed approach.

N. Pallets	$N_{RH} = 4$	$N_{RH} = 5$	$N_{RH} = 6$	$N_{RH} = 7$
3	0.42	0.54	0.71	0.90
4	0.47	0.65	0.96	1.62
5	0.50	0.77	1.34	3.11
6	0.54	0.92	2.07	5.81
7	0.56	1.03	2.58	7.96
8	0.60	1.24	4.82	16.06
9	0.65	1.66	6.87	35.02
10	0.81	2.98	11.17	> 70

Table 4.2: Online computation time (in seconds) as a function of the number of pallets and the prediction horizon for the baseline approach described in the PhD thesis [4].

In both tables the more pallets are on the line and/or the higher the value of the prediction horizon N_{RH}, the slower becomes the MPC algorithm to compute an optimal solution. As the prediction horizon grows, average times in Table 4.2 start to grow fast in the number of pallets, making computations more than one minute long for $N_{RH} = 7$ and 10 pallets. This is not the case for the algorithm proposed in this thesis.

The four sub-figures in Figure 4.1 show the average online computation times for the four values of the prediction horizon as a function of the number of pallets for both approaches 4.1 (blue line) and 4.2 (orange line).

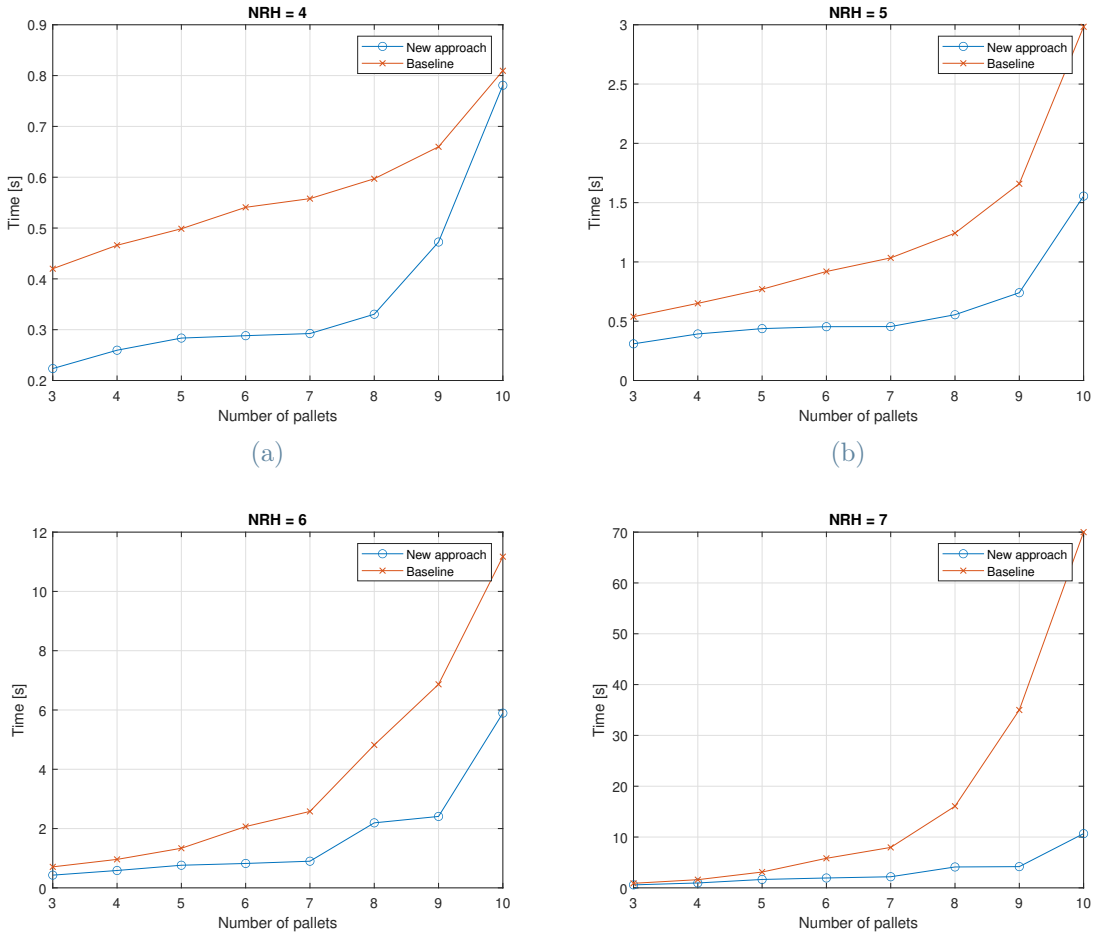

Figure 4.1: Average computation time as a function of the number of pallets for every tested value of N_{RH} for the MPC baseline approach in the PhD thesis [4] (orange line) and for the new approach in this thesis (blue line).

The top left sub-figure corresponding to $N_{RH} = 4$, shows a limited difference between the new approach and the baseline approach which decreases as the number of pallets increases. Nonetheless, with the new approach, the online computing time is smaller for each one of the considered number of pallets.

The same result can also be observed in the remaining cases; in the upper right and lower left cases, respectively the $N_{RH} = 5$ and the $N_{RH} = 6$ cases, the plots have a very similar shape, although the difference is larger. Lastly, in the lower right case ($N_{RH} = 7$), it can be seen that as the number of pallets increases, the gap increases proportionally, with a difference of 60 seconds when there are 10 pallets in the transport line.

This difference is significant and shows that the efficacy of the baseline approach is hampered by the excessive computational complexity incurred for a large number of pallets and a large value of the predictive horizon.

4.2. Number of machined pallets

The second index that is employed for the comparison is the *number of machined pallet*. The purpose of this index is to compare the number of pallets processed with the two approaches over a long period of time.

The approach developed in the PhD thesis [4] and the approach developed in this paper are launched for a total of 2000 time steps. The goal is to count the total number of pallets that are processed by each machine after 2000 steps and compare them.

In order to carry out the test, the output data of the machines was collected. In particular, the test was performed with a predictive horizon equal to $N_{RH} = 4$ whereas the total number of pallets is equal to $m = 9$. The initial state has been set the same for both approaches.

Each plot reports the number of pallets completed by a certain machine, as a function of the time step index. The orange line represents the baseline approach of the PhD thesis [4], whereas the blue line represents the new approach described in this thesis.

It is very interesting to see that in the cases of M_1, M_3 and M_4 the number of machined pallets is the same at every time step. The only exception is M_2: it does not behave the same way in the two approaches. In particular, as can be seen from Figure 4.2, the orange line does not coincide with the blue one. To explain this difference, one must remember that the target assigned to the pallet that has finished processing in machine M_2 can take on different values ($\bar{\Gamma}_{33} = 3$ or 4 or 5).

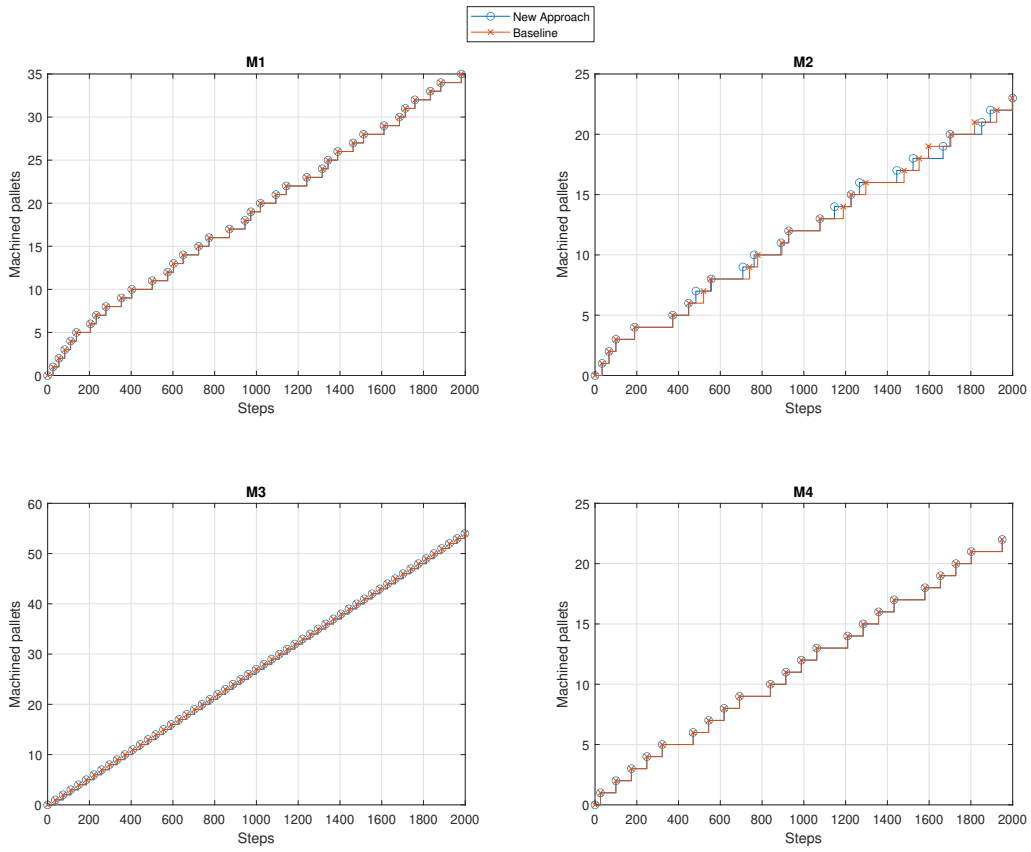

Figure 4.2: Number of machined pallets performed by M_1 (top left), M_2 (top right), M_3 (bottom left), and M_4 (bottom right).

The value assigned to $\bar{\Gamma}_{34}$ depends on the outcome of the test phase; in order to simulate this behaviour on MATLAB, the output of machine M_2 is decided randomly. Consequently, the difference between the two lines in the top right plot of Figure 4.2 is simply due to the fact that we end up with a conveyor line containing pallets which take on different targets in the long term, leading to a lag between the processing times.

Despite this difference, it should be noted that the number of pallets processed after 2000 steps with the new approach is not less than the number of pallets processed in the approach considered as a baseline, thus demonstrating that the new approach does not suffer from a decrease in throughput in the long run nor it does incur in deadlocks.

5 | Conclusions

Motivated by the need of automatic routing in smart manufacturing and inspired by a recent MPC approach proposed in the literature, we developed in this thesis a reachability-based approach for decoupling the MPC optimization problem for pallet routing into multiple smaller ones to be solved in parallel. This reduces the computational complexity of the MPC problem, especially in the cases where both the prediction horizon N_{RH} and the number of pallets m on the transport line assume large values, thus making computing times compatible with the transport system dynamics. Extensive simulations show that the proposed approach effectiveness compared with the original MPC strategy in terms of computing time and plant throughput. Test on the real plant are to be done in a follow-up work.

Although the proposed approach was developed with reference to a specific manufacturing plant, it can be applied to the transport line of a generic manufacturing plant, subject to the availability of:

- a graph and an MLD system model of the plant, the latter possibly obtained using the HYSDEL tool on an easier to derive discrete hybrid automaton model of the system
- a finite horizon cost function for the MPC problem formulation, embedding the minimal distances of all possible locations of a pallet in the transport line from the operating machines.

In order to get a more efficient implementation, however, one should find a methodology to automatically identify those critical zones in the plant that call for a grouping of the pallets that are located therein. A "learn by doing" approach could be adopted by simulating the system and observing possible deadlock conditions or recurrent grouping of pallets located in certain areas. This is left open for future investigation.

Bibliography

[1] A. Alessandri, C. Cervellera, and M. Gaggero. Predictive control of container flows in maritime intermodal terminals. *IEEE Transactions on Control Systems Technology*, 21(4):1423–1431, 2012.

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

[3] A. Brusaferri A, M. Colledani, G. Copani, M. S. N Pedrocchi, T. A. Tolio, et al. Integrated de-manufacturing systems as new approach to end-of-life management of mechatronic devices. In *Proceedings of the 0th Global Conference on Sustainable Manufacturing Towards Implementing Sustainable Manufacturing*, pages 332–339, 2012.

[4] A. Cataldo. *Model predictive control in manufacturing plants*. PhD thesis, Politecnico di Milano, Italy, 2017.

[5] A. Cataldo and R. Scattolini. Logic control design and discrete event simulation model implementation for a de-manufacturing plant. *Automazione-plus on-line Journal*, 2014.

[6] M. Colledani, G. Copani, and T. Tolio. De-manufacturing systems. *Procedia CIRP*, 17:14–19, 2014.

[7] J. Fuh, Y. Wong, C. Yee, L. Zhuang, and K. Neo. Modelling, analysis and simulation for the design of a robotic assembly system. *Computer Integrated Manufacturing Systems*, 9(1):19–31, 1996.

[8] L. Kerbache and J. M. Smith. Multi-objective routing within large scale facilities using open finite queueing networks. *European Journal of Operational Research*, 121 (1):105–123, 2000.

[9] A. R. Lázaro and C. L. Pérez. Dynamic analysis of an automobile assembly line considering starving and blocking. *Robotics and Computer-Integrated Manufacturing*, 25(2):271–279, 2009.

[10] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[11] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control engineering practice*, 11(7):733–764, 2003.

[12] 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.

[13] A. N. Tarau, B. De Schutter, and H. Hellendoorn. Model-based control for route choice in automated baggage handling systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(3):341–351, 2010.

[14] F. Torrisi and A. Bemporad. HYSDEL - a tool for generating computational hybrid models for analysis and synthesis problems. *IEEE Transactions on Control Systems Technology*, 12(2):235–249, 2004.

A | Appendix

The adjacency matrix A for the plant.

$$\begin{bmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&1&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1
\end{bmatrix}$$

MATLAB Implementation

This chapter provides the Matlab implementation of the approach proposed in this thesis. Algorithm A.1 describes in pseudo-code the main steps involved with reference to implementation of MPC for `NSA` steps.

The first part of the code, that is from line 1 to line 3, shows the initialization part. Then, a `FOR` cycle starts to loop for a total of `NSA` steps: its value it is totally arbitrary and it can be changed based on the desired length of the simulation.

Then, the goal of the code between line 5 and line 9 is to produce a two-dimensional array

Algorithm A.1 Main pseudo-code

```
m // define the number of pallets
X = X_0
NSA // number of simulation steps
for k=1 to NSA do
  pos = find(X)
  [ids, g] = offlimitsCoupling(m)
  ids = rmNullElems(ids)
  df = difference(pos,ids) //indexes of pallets outside of OL/EZ zones
  groups = concatenate(ids,df)
  for j=1 to g do
    X_{N,j} = reachabilityAnalysis(A, NRH, groups[j])
    R_j = pruning(X_{N,j}, S)
    X_{RH,j} = MPC(R_j)
  end for
  collisions = collisionCheck(X_RH)// Boolean
  while !equal(collisions,id(g)) do
    for i=1 to g do
      if linesum(collision(i,:))>1 then
        index = nnz(collision(i,:))
        R = joinColliding(R,index)
        X_{RH,i} = MPC(R_i)
      end if
    end for
    collisions = collisionCheck(X_RH)
    g = size(collision,1)
  end while
  X = stateStitch(X_RH)
end for
```

called `groups`. This array contains the indexes of all the pallets on the transport line; if the i-th element of the `groups` array has more than one element, it means that they are in the same off-limits/exchange zone and they must be coordinated by the same MPC.

From line 10, another `FOR` loop begins: this one has the purpose of executing the reachability analysis, the pruning and launching the MPC as many times as the number of groups into which the pallets are subdivided. The function `reachabilityAnalysis()` executes the reachability analysis, as its name suggests; the function `pruning` executes the pruning of the variables associated to non reachable nodes. The `MPC()` function takes the reduced system R_j as an input and produces an array for each one of the pallets/groups of pallets. The pseudo-code for the `MPC()` function is omitted since the function is pretty straightforward: the *performance index* J is calculated and then minimized by calling a MATLAB toolbox called YALMIP. Then, according to the receding horizon approach, only the first step of the optimized path is considered by returning it as an output array called $X_{RH,j}$.

This evolution is not definitive though, since on line 15 it is checked whether or not there were collisions. This is done by the `collisionCheck()` function which takes all of optimal solutions together into one single array X_{RH} as an input and produces a square matrix as the output. The output matrix called `collisions` is a symmetrical matrix that indicates which pallets/groups of pallets are colliding. In case there are not any collision, the matrix is an identity matrix. An example of the `collisionMatrix` - here called C_m - is given in Equation A.1.

$$C_m = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.1}$$

In the example from Equation A.1, the first pallet/group of pallets collides with the third pallet/group of pallets, thus before controlling the agents, these two will have to be grouped together.

If there is at least one collision, the `WHILE` loop on line 16 is entered. For every iteration of the `WHILE` loop, a `FOR` loop is launched: this internal loop's goal is to group together and calculate the MPC for the colliding parties. In order to check whether or not a pallet/group of pallet collided with something else, the sum of the i-th row of the `collision matrix` is checked: if its value is greater than one, it means that the i-th element collided. Taking the example from Equation A.1, the sum of the elements from the first line is equal to two, indicating that a collision is present.

Now, in order to spot the colliding parties, the function `nnz()` is employed. Given an array as input, it returns the indexes of the non-null elements, which are returned and written in `index`.

The function `joinColliding()` on line 20 takes as input arguments the structure `R` containing the sub-systems and the array `index` of the colliding parties. The function produces a new sub-system from the colliding ones, removing the older ones and replacing them inside the structure `R`. It is now possible to launch the MPC for the new subsystems.

Once all of the new evolutions are calculated, it is time to check again if there are any collision: the function `collisionCheck()` is launched again so as to overwrite the previous matrix `collisions`. Then, in line 25, the new value of g is calculated by measuring the number of rows (or columns, since it is a square matrix) of the `collisions` matrix. If there are still some collisions (i.e. the `collisions` matrix is different from $I_d(g)$), another iteration of the `WHILE` loop is launched. Otherwise, the while loop is exited.

Finally, the function `stateStitch()` unites the indexes of the new positions together with the other state variables; in order to do so, it takes the evolution of all of the sub-systems X_{RH} as an input. The output of this function is the evolution of the whole system.

MATLAB Auxiliary Variables

Table A.1 shows the 607 auxiliary variables. The first column represents the type of the variable; the second column represents its name in the MATLAB script; the third column the nodes where that variable is involved and lastly, the MLD column refers to the name of that variable within the plant mathematical model. So, as an example, the variable of the first row, whose MATLAB name is `dji`, refers to the $\zeta_{i,s}$ variable of the MLD model; there are a total of 175 of these variables implemented on MATLAB, 5 for each node of the plant.

Type	Name	Nodes	MLD
BOOL	dji	j=1,...,35; i=1,...,5	$\zeta_{i,s}$
REAL	Z_bzj_uk	j=1,...,35; k=1,...,50	$\Gamma_j(k)u_{j,i}$
BOOL	d_bzj_NotFree_uk	j=1,...,35; k=1,...,50	$\theta_j(k)u_{j,i}(k)$
REAL	Zica_bzj_uk	j=1,...,31	$\eta_j(k)\theta_j(k)u_{j,i}(k)$
REAL	Tp_c_bzj	j=1,...,35	$\gamma_i(\Gamma_i(k))$
BOOL	d_Tp_bzj	j=1,...,35	$\Gamma_i(k) > 0$
BOOL	d_bzj_out	j=1,...,35	$\sum_{j\in I_{i,out}} u_{i,j}(k) = 0$
BOOL	d_bzj_in	j=1,...,35	$\sum_{j\in I_{i,in}} u_{j,i}(k) = 0$
BOOL	dTpNotFreej	j=1,...,35	$\theta_i(k)$
BOOL	dica_np_bzj	j=1,...,31	$\delta_i(k)$
BOOL	dica_np_bz_NotFreej	j=1,...,31	$\delta_i(k)\theta_i(k)$
REAL	Z_bzj_2_3	Mj, j=1,...,4	$\delta_{i,23}\Gamma_i(k)$
REAL	Z_Tp_bzj_x1x2	Mj, j=1,...,4	$x_{i2}(k)x_{i1}(k)$
REAL	Z_n_bzj	Mj, j=1,...,4	$\eta_i(k)x_{i1}$
BOOL	d_Tp_bzj_2_n	Mj, j=1,...,4	$\delta_{i,23}$
BOOL	d_Tp_bzj_1	Mj, j=1,...,4	x_{i1}
BOOL	d_Tp_bzj_2	Mj, j=1,...,4	x_{i2}
BOOL	d_Tp_bzj_3	Mj, j=1,...,4	x_{i3}
BOOL	d_Tp_bzj_n	Mj, j=1,...,4	$n_i(k) \geq \bar{n}_i$
BOOL	U30_35	BZ16, BZ19	$U_{30} = 1 \wedge U_{35} = 1$
BOOL	d_Disch	BZ23, BZ35	$\sigma_{35}(k)$
BOOL	d_Rew	BZ19, BZ33	$\sigma_{33}(k)$
BOOL	d_Load	BZ1, BZ32	$\sigma_{32}(k)$
BOOL	d_Test	BZ12, BZ34	$\sigma_{34}(k)$

Table A.1: Auxiliary variables textual names

List of Figures

List of Tables

Ringraziamenti

Desidero iniziare i miei ringraziamenti con le persone che hanno svolto un ruolo fondamentale nella mia vita e nel raggiungimento di questo importante traguardo.

Prima di tutto, desidero esprimere la mia gratitudine ai miei genitori. Senza il loro sostegno costante, la loro fiducia in me e il loro amore incondizionato, non sarei mai arrivato fino a qui. Grazie per avermi dato le fondamenta su cui costruire il mio percorso accademico e per essermi stati accanto in ogni momento, incoraggiandomi a dare il massimo.

Voglio ringraziare anche Stefania. Hai dimostrato una pazienza infinita, un sostegno costante e una comprensione profonda durante questo percorso. Sei stata la mia fonte di ispirazione, il mio punto di forza e il mio raggio di luce nei momenti di difficoltà. Grazie per avermi sostenuto con tutto il tuo cuore e per aver creduto in me quando avevo bisogno di incoraggiamento.

Un ringraziamento speciale va ai miei amici più cari. In particolare, un "D20" di ringraziamenti a Filippo, Alessandro, Riccardo e Giovanni: ogni sessione, ogni personaggio e ogni dado lanciato hanno reso la mia vita più ricca e stimolante. Avventurieri, che le vostre future imprese siano piene di successi, colpi critici e tante risate.

Vorrei estendere la mia gratitudine anche alla professoressa Maria Prandini. Grazie per avermi guidato e ispirato lungo questo percorso di ricerca. I suoi consigli preziosi e la sua esperienza sono stati fondamentali per il mio successo. Sono profondamente grato per il tempo e l'energia che ha dedicato alla mia formazione.

Infine, vorrei ringraziare Alessandro Falsone, Lucrezia Manieri e Andrea Cataldo che mi hanno supportato nel corso di questi studi. Grazie per la vostra pazienza e competenza. Le vostre indicazioni e il vostro supporto sono stati fondamentali per il completamento di questa tesi.

A tutte queste persone straordinarie, desidero esprimere il mio sincero apprezzamento e riconoscimento. Senza di voi, tutto questo non sarebbe stato possibile. Grazie di cuore a tutti.