UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA
DE AUTOMAÇÃO E SISTEMAS

Marcelo Lopes de Lima

DISTRIBUTED SATISFICING MPC

Florianópolis

2013

Marcelo Lopes de Lima

DISTRIBUTED SATISFICING MPC

Thesis presented in partial fulfillment of the requirements for the degree of Doctor in Automation and Systems Engineering.
Advisor: Prof. Eduardo Camponogara
Coadvisor: Prof. Daniel Limón Marruedo

Florianópolis

2013

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

de Lima, Marcelo Lopes
Distributed satisficing MPC / Marcelo Lopes de Lima ; orientador, Eduardo Camponogara ; co-orientador, Daniel Limón Marruedo. - Florianópolis, SC, 2013.
120 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Inclui referências

1. Engenharia de Automação e Sistemas. 2. Model predictive control. 3. Satisfacing MPC. 4. Distributed MPC. 5. Suboptimal MPC. I. Camponogara, Eduardo. II. Marruedo, Daniel Limón. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. IV. Título.

Marcelo Lopes de Lima

DISTRIBUTED SATISFICING MPC

This thesis was accepted in its present form by the Programa de Pós-graduação em Engenharia de Automação e Sistemas as a partial fulfillment of the requirements for the degree of "Doctor in Automation and Systems Engineering".

Florianópolis, 25 of October 2013.

Prof. Jomi Fred Hübner
Course Coordinator

Prof. Eduardo Camponogara
Advisor

Prof. Daniel Limón Marruedo
Coadvisor

Committee Approval:

Prof. Eduardo Camponogara
President

Prof. Amit Bhaya

Prof. Julio Elias Normey-Rico

Dr. Mário César Mello Massa de Campos

Prof. Tito Luís Maia Santos

Prof. Ubirajara Franco Moreno

Para minha amada esposa, Deborah, e nossa querida filha, Júlia.

ACKNOWLEDGMENTS

This thesis is dedicated to my dear wife, Deborah, for her tremendous generosity and love in sharing this project with me, to our daughter, Júlia, my treasure, and to my parents, Marcílio and Terezinha, for their unconditional love.

I would like to thank Professors Eduardo Camponogara and Daniel Limón for the invaluable guidance and availability, for listening to my ideas with patience and interest, encouraging and helping me, despite my many limitations. In particular, I thank Prof. Camponogara for believing that this work would be possible.

I also thank Prof. Júlio Normey Rico for requesting, and Prof. Eduardo Camacho for receiving me and allowing me to stay at the University of Seville during the last year of the Ph.D., a period rich in knowledge and very productive. In particular, I would like to thank Prof. David Muñoz de la Peña who collaborated actively with comments and discussions, particularly with regards to the stability of the controller and its formalization.

I want to thank Petrobras, Petróleo Brasileiro S.A., for funding this work, and Alexandre Müller for supporting and making it possible within the company. I want to thank my colleagues for taking on an additional workload during my absence, and Dr. Mário Campos for the encouragement and suggestions.

Finally, I thank Federal University of Santa Catarina, especially the staff, professors and colleagues in the Department of Automation and Systems Engineering for their friendship and contribution to the development of this work.

RESUMO

Esta tese apresenta uma nova abordagem de controle MPC distribuído, ou seja, onde uma rede de controladores MPC é associada a uma rede de subsistemas, aplicada a problemas convexos e a subsistemas acoplados pelas entradas, sob restrições impostas tanto aos estados quanto às entradas dos subsistemas.

Em geral, para que a interação entre os controladores distribuídos resulte em uma solução global Pareto-ótima, os controladores são obrigados a compartilharem um mesmo custo global fixo e imposto a todos. A imposição de um custo global resulta em um altruísmo *categórico*, fixo, onde cada controlador é obrigado a ceder parte de seu desempenho para a satisfação do outro, qualquer que seja a situação.

Nesta tese, por outro lado, o controlador proposto implementa um altruísmo *situacional*, onde um custo global equivalente, não imposto nem fixo, emerge da interação entre os controladores. O altruísmo situacional é obtido através de uma abordagem satisfatória (o termo "satisficing" foi traduzido aqui por *satisfatório*, embora o termo original seja uma fusão de satisfação com suficiência, satisficing = satisfy + suffice).

Na abordagem satisfatória (SIMON, 1955) é definido um desempenho mínimo, *local a cada controlador*. Toda solução que leva a um desempenho melhor que o mínimo é uma solução válida e satisfatória. Os controladores satisfatórios propostos nesta tese são também altruístas, na medida em que incorporam em sua satisfação também a satisfação dos outros. Os controladores aplicam um algoritmo distribuído de ponto-interior para encontrar uma solução global que, no mínimo, pertence à região onde as soluções são satisfatórias e suficientes para todos eles. Tal solução é alcançada através da otimização na direção do centro analítico da região satisfatória.

Mostra-se que a solução na direção do centro analítico da região satisfatória, mesmo que sujeita às restrições do problema, corresponde à solução ótima de um custo global equivalente que explica o comportamento emergente. Mostra-se também que esse custo global equivalente não é mais fixo e leva a um altruísmo que depende da situação atual -

um altruísmo situacional.

O altruísmo situacional proposto aqui faz com que aqueles controladores com piores desempenhos locais, medidos em relação ao desempenho mínimo, ganhem mais importância em relação àqueles com melhor desempenho. Essa relação de importância é ajustada automaticamente, de forma dinâmica. Espera-se, assim, uma relação mais justa entre os controladores, baseado em critérios locais. A definição de critérios locais dá mais sentido ao que seria um comportamento aceitável; um comportamento global é aceitável quando também é aceitável localmente.

A estabilidade em malha fechada do controlador satisfatório é demonstrada com a inclusão de restrições adicionais que garantem a contração dos custos locais.

Esta tese é organizada em sete capítulos, um apêndice e referências. O Capítulo 1 tem o objetivo de contextualizar e motivar de forma breve o assunto da tese. Os capítulos 2 e 3 resumem o controle distribuído e, mais especificamente, o MPC distribuído cooperativo, onde é definida boa parte da nomenclatura e é apresentado muitos dos aspectos de interesse para o desenvolvimento do MPC satisfatório. As principais contribuições teóricas desta tese são apresentadas nos capítulos 4 e 5. No Capítulo 4, é apresentado o MPC satisfatório sem os ingredientes para a estabilidade garantida, introduzidos apenas no Capítulo 5 onde se prova a estabilidade em malha fechada. No Capítulo 6, três exemplos são utilizados para avaliar os resultados do MPC satisfatório se comparado a um MPC clássico centralizado. Algumas conclusões são discutidas no Capítulo 7, incluindo uma lista de sugestões para trabalhos futuros. No apêndice é apresentado um assunto específico, relacionados à teoria principal. Mostra-se que o altruísmo mínimo necessário para a satisfação de todos os controladores pode ser associado a multiplicadores de Lagrange. Esse assunto foi tratado no apêndice por ser considerado acessório à teoria principal.

Palavras-chave: MPC satisfatório. MPC Distribuído. MPC subótimo. Altruismo situacional. Ponto-interior distribuído. Estabilidade em malha fechada. Solução Pareto-otima. MPC multi-objetivo.

ABSTRACT

To obtain a Pareto-optimal solution, the classical cooperative MPC implements a *categorical* altruism imposed by a fixed global cost shared by all the local controllers. Instead, this thesis implements a *situational* altruism where a global cost, neither imposed nor fixed, emerges from *convex* local costs and local specifications. The satisficing controllers employ a distributed algorithm to find a solution that lies in a convex region that is satisfactory and sufficient for all controllers (satisficing = satisfy + suffice), while optimizing in the direction of the analytic center of such a region. The system is modeled as being a network of linear subsystems, coupled by their inputs, and the algorithm uses a distributed interior-point method to avoid fixed points when the constraints are also coupled. The optimal solution of the satisficing MPC, besides Pareto-optimal, gives more importance to the controllers with a worst performance at the moment. Situational altruism permits a more balanced division of resources, avoiding the exploitation of one controller by the others. The satisficing MPC is shown to be stabilizing even if suboptimal, provided that it is satisficing. To this end, stabilizing constraints are added to the basic formulation.

Keywords: Satisficing MPC. Distributed MPC. Sub-optimal MPC. Situational altruism. Distributed interior-point method. Closed-loop stability. Pareto-optimal solution. Multi-objective MPC.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

LIST OF SYMBOLS

CONTENTS

1 INTRODUCTION

Process plants, like refineries and petrochemical plants, are composed by a number of complex and interconnected units. Complexity arises from strict environmental regulations and product specifications, and also from the fact that it is economically important to take advantage from the opportunities of energy integration inside and among units. Highly integrated plants lead to the need for better control systems that can seamlessly cope with the increasing complexity.

Traditionally, the technology widely used to deal with the complexity that arises from couplings and interactions is based on centralized, multi-variable, model-based controllers (SEBORG, 1994, 1999) known as Model Predictive Control (MPC). Conceptually, MPCs are controllers that use the knowledge of the process to be controlled, represented by a model, to predict what would be the outcome of the action plan they are considering to apply. In possession of a criterion they are able to choose a plan of action in comparison to another. This process of selection is done by an optimization algorithm that chooses the best plan considering the given criterion. In general, every MPC has a prediction model, an objective function as criterion and a process to decide the plan of actions (CAMACHO; BORDONS, 2004). At each control cycle the MPC reads inputs from the unit, calculates a plan of control outputs, applies only the first action of the calculated plan, and starts another cycle respecting a deterministic interval.

Large MPC controllers are difficult to implement and maintain. The overall time to model and implement a large MPC application is considerable, and the process of tuning, that is, the translation of control specification into a consistent set of relative weights and parameters required by an MPC is not a trivial task (QIN; BADGWELL, 2003). Once in operation, the available models can become out of date due to units' aging, or even the structure of the problem can change dynamically because, for example, process variables may become unavailable in real time (QIN; BADGWELL, 2003). We advocate that small MPC controllers are easy to tune, model and maintain than large ones. Small MPCs are also easier to understand and faster to deploy, allowing users to obtain gains earlier when compared to a large deployment of a single MPC system.

In a plantwide control perspective, usually a process unit is controlled by a number of local *independent* MPC controllers so that the global solution is then a collection of local solutions. However, the

strategy of using local independent small controllers, called decentralized control, also has some drawbacks: although the interactions among the MPCs may be treated by each other as disturbances, the global outcome is not optimal and may lead to degraded performance (CUI; JACOBSEN, 2002) or instabilities if the interactions are strong.

An alternative to the decentralized strategy above is to explicitly model the interactions between the controllers and develop an iterative process of solution. This strategy is called distributed and, differently of the decentralized strategy, there must be communication between controllers (SCATTOLINI, 2009). The approach is to decompose the system in a number of local controllers and employ a coordination process in such a way that the emerging global solution resembles the performance obtained when a centralized control is employed. With the use of smaller controllers, the control system can enjoy more flexibility and scalability (CAMPONOGARA; SCHERER, 2011), preserving the performance of a centralized controller.

In this work we show that for a distributed strategy to be optimal it will require some sort of *altruism* between controllers. This is the case of the so called cooperative MPC (STEWART et al., 2010). There, the controllers are forced to abdicate part of their own local objectives in favor of a global objective shared by all. The global objective is the weighted sum of local objectives whose weights define the relative importance of the controllers. The weights are fixed and have to be chosen a-priori by the designer to tune the resulting global performance associated to the obtained Pareto optimal solution. Other examples of cooperative MPCs are (CAMPONOGARA; OLIVEIRA, 2009), (CAMPONOGARA; SCHERER, 2011) and (CAMPONOGARA; LIMA, 2012) that differ from (STEWART et al., 2010) mainly in the protocol they use and in the treatment they give to coupled constraints.

The ad hoc tuning of fixed weights, besides difficult in most cases, results in what Stirling (STIRLING; FROST, 2007) calls *categorical altruism*. Categorical altruism condemns a controller to always please other controllers even if the others are by far accomplishing their objectives. In other words, the categorical altruism fixes the trade off among the controllers ignoring the evolution of their objectives with time or even the real need for altruism. Another issue applicable to the cooperative MPC (and classical MPC in general) is cited by Qin e Badgwell (2003):

> Prett and Garcia (1988) commented (...) : The combination of multiple objectives into one objective (function) does not allow the designer to reflect the true performance requirements. [reference in the original](QIN; BADGWELL,

2003)

This thesis proposes a fair approach called *situational altruism*. In situational altruism the weights, instead of being fixed, are dynamically adjusted at each cycle according to the system's current state. The idea is to define a minimum level of satisfaction for each controller, based on its objective, so that the less satisfied controller will automatically receive more importance.

This research contributes in the sense of developing a theory that allows large applications with small MPCs, but in a "satisficing'[1] perspective (SIMON, 1955; STIRLING, 2003; GOODRICH et al., 1998), where situational altruism is implemented.

1.1 CONTRIBUTIONS

A new linear distributed MPC with input and state coupled constraints and guarantee of stability is proposed, in which the relative importance of the controllers dynamically changes according to the actual conditions. The satisficing controllers have the following characteristics:

- the emerging solution lies in a region that is satisfactory and sufficient for all controllers (satisficing = satisfy + suffice);
- the solution may be optimal or suboptimal if desired, depending, for example, on the time available to pursue an optimal solution. Stability is guaranteed even for suboptimal solutions;
- a Pareto optimal solution exists and is equivalent to that obtained by a classical centralized controller in which the relative importance of controllers is not fixed but varies at each sample time;
- there are no weights to be directly set. Instead, the relative importance of the controllers is a byproduct of the distributed optimization at each sample time. The less satisfied controller automatically receives more importance and the tuning process is much simplified;
- the definition of a minimum level of performance instead of weights leads to a broader range of behaviors.

[1] Satisfacing is a combination of the words satisfy and suffice.

From the point of view of applications, since petrochemical industries and refineries are becoming more complex, the demand for multivariable controllers is increasing. The development of practical techniques concerning *distributed* linear MPCs will permit an immediate and broader application of them. Due to the importance of linear MPC controllers in practical applications, it makes sense to tackle practical aspects concerning linear, as opposed to non-linear MPC. Besides, it may be the first step towards a satisficing non-linear theory.

A distributed satisficing MPC control system could be advantageous:

because it is distributed: a "divide and conquer" strategy may be applied, that is, a strategy that employs a set of small, simple controllers rather than a large, complex controller. Small controllers are easier to model, understand and analyze. Small controllers can be tuned, maintained and operated individually and a failure in one MPC does not compromise the whole system. Small MPCs are also faster to deploy allowing users to obtain gains earlier on and in different phases when compared to a centralized MPC deployment;

because it is satisficing: the satisficing criterion is conceptually easier to understand, the satisficing controller presents adaptiveness features and it is stabilizing.

This thesis is organized in seven chapters, one appendix and references. Chapter 1 aims to briefly contextualize and motivate the subject of this thesis. Chapters 2 and 3 introduces the distributed control and more specifically the cooperative distributed MPC, where most of the nomenclature is defined and many aspects of interest for the development of the satisficing MPC is presented. The main theoretical contributions of this thesis are in chapter 4 and 5. In Chapter 4, the satisficing MPC is presented without the ingredients for stability, introduced in Chapter 5 where stability is proven. In Chapter 6, three examples are used to evaluate the results of the satisficing MPC against a classical MPC. Some concluding remarks are discussed in Chapter 7, including a list of suggestions for future developments.

The appendix presents the minimal altruism necessary for the satisfaction of all controllers, and associates the minimal altruism to Lagrange multipliers. This subject is original, but it was left to the appendix because it was considered accessory to the main theory.

2 DISTRIBUTED CONTROL

In a "divide and conquer" strategy, a large system may be understood as a network of interdependent subsystems, coupled in their dynamics and constraints. A global behavior, then, results from the interaction of the subsystems. The network of subsystems is controlled by a network of controllers $\mathcal{C}_m$, one controller for each subsystem m or a group of subsystems. See, for example, Figure 1 that represents a refinery distillation unit, used to process crude oil, divided into 8 subsystems.

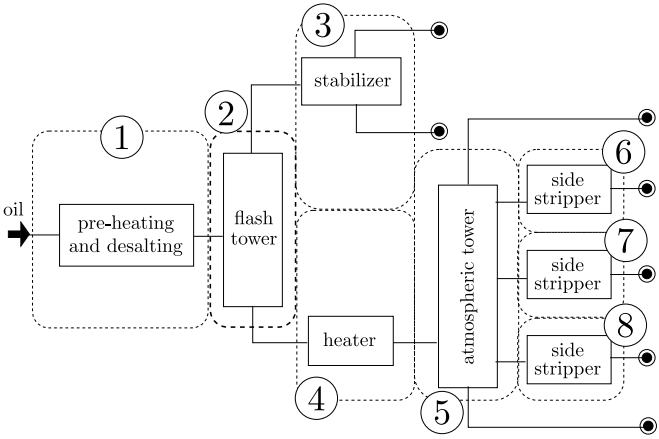

Figure 1 – A distillation unit divided in 8 coupled subsystems.

This strategy has, at least, the following advantages:

1. small controllers are easier to model, understand and analyze;
2. small controllers can be tuned, maintained and operated individually;
3. a failure in one controller does not compromise the whole system;
4. small controllers are faster to deploy allowing users to obtain gains earlier;
5. the number of controllers may grow according to the user's implementation schedule;

6. changes may be done locally;

An overview of schemes applied to the divide and conquer strategy may be found in (SCATTOLINI, 2009) and (CHRISTOFIDES et al., 2013). The most common is to apply a decentralized approach, even though the overall performance is not the same as would be obtained by a centralized controller. In a decentralized approach, a controller responsible for some subsystem m manipulates input u_m in order to control the output y_m, without communication with other controllers regardless of the physical interactions among the subsystems (see Figure 2.a). Because the decentralized controller cannot calculate how the interactions affect the output it is controlling, it can only react to these effects. Although this scheme is widely used, the global outcome is not optimal and may lead to degraded performance (CUI; JACOBSEN, 2002) or instabilities if the interactions are strong.

An alternative is to use a distributed approach, where the interactions between the controllers are explicitly modeled. Differently of the decentralized strategy, there must be communication between controllers (SCATTOLINI, 2009), allowing them to deal with subsystems interactions. Figure 2.b illustrates this scheme.

In the distributed scheme, the emerging global solution resembles the performance obtained by a centralized control but, on the other hand, the amount of communication may be an issue.

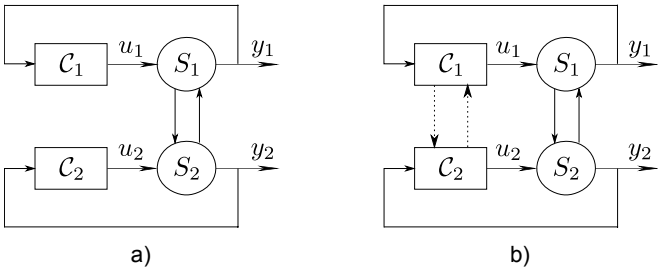

Figure 2 – Two strategies to control coupled subsystems S_1 and S_2: a) Decentralized approach: there is no communication between controllers $\mathcal{C}_1$ and $\mathcal{C}_2$. b) Distributed approach: there is communication between controllers $\mathcal{C}_1$ and $\mathcal{C}_2$, which allows them to explicitly deal with subsystems interactions.

In the following, it is discussed how to model coupled systems in a network of subsystems amenable for distributed control and for the theory presented in this work. The modeling discussed here is

exemplified in Chapter 6 with the use of three examples. A coupled system may be coupled in their dynamics, in their constraints or both.

2.1 COUPLINGS IN SUBSYSTEMS DYNAMICS

A system may be thought of as being a network of subsystems, each subsystem defined by its outputs. To exemplify, let us consider the system depicted in Figure 3 where output y_1 is directly driven by input u_1 and influenced by input u_2 and, on the other hand, output y_2 is directly driven by input u_2 and influenced by input u_1.

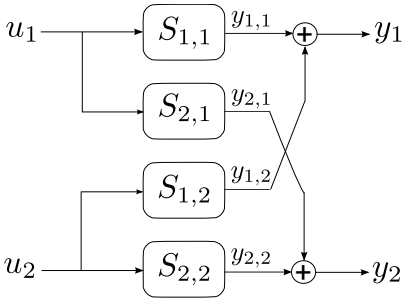

Figure 3 – Example of system interactions. Subsystems $S_{1,1}$ and $S_{2,2}$ model the direct path from u_1 to y_1 and u_2 to y_2 respectively. $S_{1,2}$ and $S_{2,1}$ model interactions.

Each connection is a discrete state space dynamic equation of the form:

$$S_{m,i} : \begin{cases} x_{m,i}(k+1) = A_{m,i}x_{m,i}(k) + W_{m,i}u_i(k) \\ y_{m,i}(k) = C_{m,i}x_{m,i}(k) \end{cases} \tag{2.1}$$

where k is a discrete time and $S_{m,i}$, when $m \neq i$, represents interactions.

Let us define a subsystem as being the collection of all direct and interaction connections needed to define its output. The system of Figure 3, for example, may be divided into two subsystems: one defined by output y_1 and the other defined by output y_2. Subsystem 1 is the collection of all connections needed to define output y_1 (see Figure 4), and the same reasoning is used to define subsystem 2.

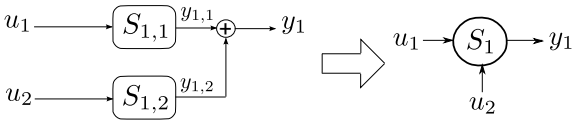

Figure 4 – The input/output representation of a subsystem collects direct and interaction models needed to represent the output.

Subsystem 1 is given by:

$$S_1 : \begin{cases} x_1(k+1) = A_1 x_1(k) + B_{1,1} u_1(k) + B_{1,2} u_2(k) \\ y_1(k) = C_1 x_1(k) \end{cases} \tag{2.2}$$

where

$$x_1(k) = \begin{bmatrix} x_{1,1}(k) \\ x_{1,2}(k) \end{bmatrix}; A_1 = \begin{bmatrix} A_{1,1} & \\ & A_{1,2} \end{bmatrix};$$
$$B_{1,1} = \begin{bmatrix} W_{1,1} \\ 0 \end{bmatrix}; B_{1,2} = \begin{bmatrix} 0 \\ W_{1,2} \end{bmatrix};$$
$$C_1 = \begin{bmatrix} C_{1,1} & C_{1,2} \end{bmatrix}$$

and 0 means matrices of zeros of proper dimensions. Subsystem 2 is, in its turn, given by

$$S_2 : \begin{cases} x_2(k+1) = A_2 x_2(k) + B_{2,1} u_1(k) + B_{2,2} u_2(k) \\ y_2(k) = C_2 x_2(k) \end{cases} \tag{2.3}$$

where

$$x_2(k) = \begin{bmatrix} x_{2,1}(k) \\ x_{2,2}(k) \end{bmatrix}; A_2 = \begin{bmatrix} A_{2,1} & \\ & A_{2,2} \end{bmatrix};$$
$$B_{2,1} = \begin{bmatrix} W_{2,1} \\ 0 \end{bmatrix}; B_{2,2} = \begin{bmatrix} 0 \\ W_{2,2} \end{bmatrix};$$
$$C_2 = \begin{bmatrix} C_{2,1} & C_{2,2} \end{bmatrix}$$

It can be seen that both subsystems have their dynamic equations coupled by their inputs u_1 and u_2.

The definition of subsystems coupled by inputs allows the system of Figure 3 to be represented by a direct graph connecting the

subsystems, as in Figure 5. Inputs and outputs are made implicit in the last figure.

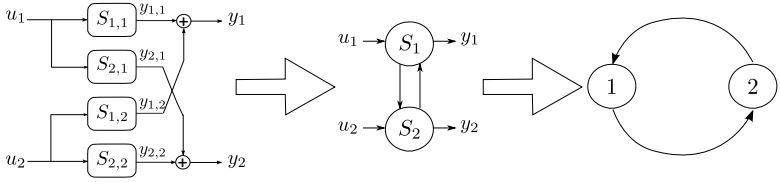

Figure 5 – Each subsystem is coupled with other subsystems by their inputs, forming a network that represents the interactions between them.

Generalizing this procedure, couplings among subsystems dynamics can be represented by a directed graph $\mathcal{G} = (\mathcal{M}, \mathcal{E})$, whose vertex set $\mathcal{M} = \{1, ..., M\}$ denotes the subsystems and whose arc set $\mathcal{E} \subseteq \mathcal{M} \times \mathcal{M}$ represents couplings. An arc $(m, i) \in \mathcal{E}$ indicates that subsystem i affects the state of subsystem m therefore affecting its output.

Formally, let us consider a class of coupled systems in which each subsystem $m \in \mathcal{M}$ is a linear, time invariant state space dynamic system, affected by its own actions and also by the actions of its input subsystems, as follows:

$$x_m(k+1) = A_m x_m(k) + B_{m,m} u_m(k) + \sum_{i \in \mathcal{M} \backslash \{m\}} B_{m,i} u_i(k) \qquad (2.4)$$

where $x_m \in \mathbb{R}^{n_m}$ and $u_m \in \mathbb{R}^{m_m}$ are the state and action of subsystem $m \in \mathcal{M}$, $\mathcal{M}\backslash\{m\}$ are all subsystems excluding m, and A_m, $B_{m,m}$ and $B_{m,i}$ are matrices of proper dimensions. Observe that, unless the subsystems are fully connected by the inputs, some matrices $B_{m,i}$ will be zero.

The dynamic equation above can be stated in a compact form by

$$x_m^+ = A_m x_m + B_m u \qquad (2.5)$$

where $B_m = [B_{m,1}, \cdots, B_{m,m}, \cdots, B_{m,M}]$, and x_m^+ is the successor state when u is applied and the actual state is x_m. The vector[1]

$$u = (u_1, \cdots, u_m, \cdots, u_M)$$

[1] A column vector will be represented by its elements in parentheses.

is the collection of the actual subsystems' inputs. The global system is then represented by the collection of the resulting subsystems' states

$$x = (x_1, \cdots, x_m, \cdots, x_M)$$

that emerges from the interaction of the subsystems.

2.2 COUPLINGS CAUSED BY CONSTRAINTS

The system may have associated constraints that must be satisfied by the states and actions, so that

$$x(k) \in \mathbb{X},\ u(k) \in \mathbb{U}$$

for all instants k. The set $\mathbb{X}$ is convex, $\mathbb{U}$ is convex and compact, and each set has the origin in its interior. The constraints define a domain of valid actions dependent on the actual state x,

$$\mathbb{D}(x) = \{u \mid u \in \mathbb{U}, x^+ \in \mathbb{X}\}$$

These constraints in global level may impose local coupled constraints to the subsystems, as exemplified in Figure 6. With coupled constraints, the constraints of subsystem 1 are dependent on subsystem 2 and vise-versa. The set of valid inputs of subsystem 1 given the actual state and input of subsystem 2, is given by

$$\mathbb{D}_1(x, u_2) = \{u_1 \mid u \in \mathbb{U}, x^+ \in \mathbb{X}\}$$

This set is not the same if subsystem 2 chooses input u_2' or u_2''.

Generalizing, we have that

$$\mathbb{D}_m(x, u_{\neg m}) = \{u_m \mid u \in \mathbb{U}, x^+ \in \mathbb{X}\}$$

in which the parameters are the actual state x and the other inputs $u_{\neg m}$ given by

$$u_{\neg m} = (u_i \mid i \in \mathcal{M} \backslash \{m\})$$

Uncoupled constraints, on the other hand, are characterized by $\mathbb{X} = \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_M}$ and $\mathbb{U} = \mathbb{U}_1 \times \cdots \times \mathbb{U}_M$, the Cartesian product of the individual constraints, where $\mathbb{U}_m$ is only dependent on subsystem m, for all m, so that $\mathbb{D}_m$ is independent of $u_{\neg m}$.

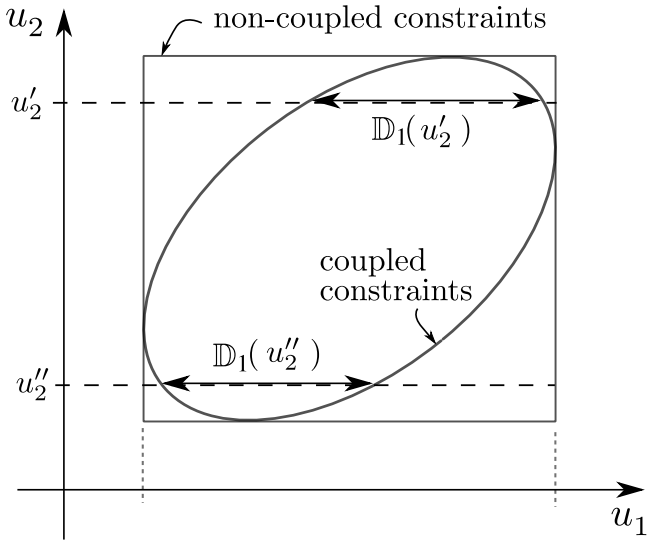

Figure 6 – Coupled constraints. The constraints of subsystem 1 are dependent on subsystem 2 and vise-versa. Input u_1 must be in the set of valid actions $\mathbb{D}_1(x, u_2) = \{u_1 \mid u \in \mathbb{U}, x^+ \in \mathbb{X}\}$.

3 DISTRIBUTED MPC

This chapter presents, succinctly, the main components of a Distributed Model Predictive Control (DMPC) used to control coupled subsystems. It is the basis from which the distributed satisficing MPC is built, incorporating the satisficing ideas discussed in the following chapters.

A DMPC control system is a distributed control scheme where each local controller $\mathcal{C}_m$, for each subsystem m, is a local MPC. Every MPC is a decision maker, composed of three main components: a prediction model, an objective function, and a decision process (CAMACHO; BORDONS, 2004):

- Prediction model: allows the controller to predict the future evolution of the system given a plan of actions;
- Objective function: is the criterion under which the predicted evolution is valued;
- Decision process: is the process by which the controller decides its plan of action, in order to result in a future evolution with a good valued objective function. In general, the decision process is an optimization problem.

At each time k, an MPC decides which plan of action evolves the system in the best way, given its objectives. Once the action plan is decided, the MPC applies only the first action of this plan. The system reacts to the applied action and evolves to its successor state in $k+1$ according to its dynamic model. A next sample is made in time $k+1$ and the process is repeated. This scheme is also known as receding horizon control.

The three components of MPCs are briefly discussed in the following. They will be developed later in the context of the distributed satisficing MPC.

3.1 PREDICTION MODEL

Because the system we are considering is time-invariant, in the rest of this thesis, the time k is always the actual time $k = 0$ and it will be dropped from the variables most of the time. In this way, a variable, for example, $x_m(1)$ is the same as $x_m(k+1)$.

Every MPC controller $\mathcal{C}_m$ has an internal representation of subsystem m, governed by the nominal dynamic equation (2.5)[Page 33], and uses it to predict the evolution of its states

$$\mathbf{x}_m = (x_m(1), \cdots, x_m(N))$$

along a prediction horizon N. This state trajectory $\mathbf{x}_m$ can be obtained from the initial state $x_m = x_m(0)$ and the future input trajectory of all the inputs using the following linear prediction model:

$$\mathbf{x}_m = \widetilde{A}_m x_m + \widetilde{B}_m \mathbf{u} \tag{3.1}$$

where matrices $\widetilde{A}_m$ and $\widetilde{B}_m$ are obtained from the dynamic model (2.5) and the *action profile*

$$\mathbf{u} = (\mathbf{u}_1, \cdots, \mathbf{u}_m, \cdots, \mathbf{u}_M)$$

includes the *action plan* of controllers $\mathcal{C}_m$

$$\mathbf{u}_m = (u_m(0), \cdots, u_m(N-1))$$

and also the action plan of all controllers $\mathcal{C}_i$, $i \in \mathcal{M}\backslash\{m\}$. We may summarize the action plans of the other controllers but controller $\mathcal{C}_m$ in a vector

$$\mathbf{u}_{\neg m} = (\mathbf{u}_i, i \in \mathcal{M}\backslash\{m\})$$

Although the controllers may not have an exact representation of each other, in this thesis, the following assumption is made:

Assumption 3.1. *Perfect information: all controllers know each other's variables perfectly.*

This assumption is reasonable when the communication between controllers is reliable and the controllers are credible. All controllers know each other's action plan correctly.

The evolution of the equivalent global system is then the collection of local state trajectories, being given by

$$\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_m, \cdots, \mathbf{x}_M)$$

Both the state predictions and the input trajectories must respect the constraints along the full prediction horizon, that is, if we define

$$\mathcal{X} \triangleq \{\mathbf{x} \mid x(k) \in \mathbb{X},\ k = 1 \ldots N\}$$

$$\mathcal{U} \triangleq \{\mathbf{u} \mid u(k) \in \mathbb{U},\ k = 1 \dots N\}$$

it is required that

$$\mathbf{x} \in \mathcal{X},\ \mathbf{u} \in \mathcal{U}$$

These constraints define the domain of valid plans

$$\mathcal{D}(x) = \{\mathbf{u} \mid \mathbf{u} \in \mathcal{U},\ \mathbf{x} \in \mathcal{X}\}$$

that imposes local domains given the other plans and actual states

$$\mathcal{D}_m(x, \mathbf{u}_{\neg m}) = \{\mathbf{u}_m \mid \mathbf{u} \in \mathcal{U},\ \mathbf{x} \in \mathcal{X}\} \tag{3.2}$$

for all $m \in \mathcal{M}$.

3.2 OBJECTIVE FUNCTION

MPC controllers are decision makers. They have to decide their action plan, restricted to the domain of valid actions, based on the internal representation they have and always trying to accomplish their objectives. For that, each controller has to have a *total ordering* of its options as explained below: let the symbols $\succeq_m$ and $\sim_m$ denote binary relations meaning "better or as good as" and "as good as", respectively, under the point of view of controller $\mathcal{C}_m$. A total ordering over the domain of $\mathcal{C}_m$ given other controllers options is characterized when the following properties are satisfied:

completeness: $\forall \mathbf{u}_{1m}, \mathbf{u}_{2m} \in \mathcal{D}_m(\mathbf{u}_{\neg m})$,
$\mathbf{u}_{1m} \succeq_m \mathbf{u}_{2m}$ or $\mathbf{u}_{2m} \succeq_m \mathbf{u}_{1m}$;

reflexivity: $\forall \mathbf{u}_m \in \mathcal{D}_m(\mathbf{u}_{\neg m}), \quad \mathbf{u}_m \succeq_m \mathbf{u}_m$;

antisymmetry: $\forall \mathbf{u}_{1m}, \mathbf{u}_{2m} \in \mathcal{D}_m(\mathbf{u}_{\neg m})$,
$\mathbf{u}_{1m} \succeq_m \mathbf{u}_{2m},\ \mathbf{u}_{2m} \succeq_m \mathbf{u}_{1m} \Rightarrow \mathbf{u}_{1m} \sim_m \mathbf{u}_{2m}$;

transitivity: $\forall \mathbf{u}_{1m}, \mathbf{u}_{2m}, \mathbf{u}_{3m} \in \mathcal{D}_m(\mathbf{u}_{\neg m})$,
$\mathbf{u}_{1m} \succeq_m \mathbf{u}_{2m},\ \mathbf{u}_{2m} \succeq_m \mathbf{u}_{3m} \Rightarrow \mathbf{u}_{1m} \succeq_m \mathbf{u}_{3m}$;

and, if a total order is established then the controller would compare all options and choose one among the best of them.

Mathematically, it is convenient to have a function that captures the totally ordered preferences:

Definition 3.1. *A cost is a real-valued function over a domain $\mathcal{D}$, $V : \mathcal{D} \to \mathbb{R}$ such that, for all $\mathbf{u}_1$ and $\mathbf{u}_2 \in \mathcal{D}$, $\mathbf{u}_1 \succeq \mathbf{u}_2$ if, and only if, $V(\mathbf{u}_1) \leqslant V(\mathbf{u}_2)$.*

Any rational controller chooses its decisions so as to minimize its cost. The cost of the controller should reflect the controller's objective so that any action that minimizes the cost is one of its best actions to the accomplishment of the objective.

Usually, to each controller $\mathcal{C}_m$ it is associated a quadratic cost V_m, a function of the actual state x_m and action profile $\mathbf{u}$, given by

$$V_m(x_m, \mathbf{u}) = \sum_{k=0}^{N-1} \ell_m(x_m(k), u_m(k)) \tag{3.3}$$

where the stage cost of controller $\mathcal{C}_m$, $\ell_m(x_m(k), u_m(k))$, is a definite positive function with $\ell_m(0,0) = 0$. In general,

$$\ell_m(x_m(k), u_m(k)) = x_m(k)' Q_m x_m(k) + \alpha_m \cdot u_m(k)' R_m u_m(k)$$

where matrices Q_m and R_m are positive definite and positive semi-definite, respectively, and $\alpha_m \geqslant 0$ is the *sensibility to cost* of controller $\mathcal{C}_m$.

3.3 DECISION PROCESS

A distributed control system has two or more controllers, each one with its own objective. In this case, contrary to single-objective optimization, there is not a single definition of optimum. Usually, the optimum in a distributed control system follows the concept of Pareto optimality.

Normally, in the DMPC framework, a Pareto solution is obtained by a scalarization procedure, when the local objectives are aggregated in a single scalar global objective obtained by the weighted sum of local objectives. This is the approach used in the cooperative distributed MPC (STEWART et al., 2010; CAMPONOGARA; OLIVEIRA, 2009; CAMPONOGARA; SCHERER, 2011) and explained here. Lately, the characteristics of this approach will be compared to the satisficing approach proposed in this thesis.

3.3.1 Pareto Optimality

Once defined the cost functions V_m of the controllers, it is clear that a solution $\mathbf{u}_m$ for controller $\mathcal{C}_m$ is better than $\mathbf{u}'_m$ if $V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) < V_m(\mathbf{u}'_m | x_m, \mathbf{u}_{\neg m})$. If, under the point of view of the other con-

trollers, it is still true that $V_j(\mathbf{u}_m|x_j, \mathbf{u}_{\neg m}) \leqslant V_j(\mathbf{u}'_m|x_j, \mathbf{u}_{\neg m})$ for all $j \in \mathcal{M}\backslash\{m\}$, then the solution $\mathbf{u}_m$ is also better than $\mathbf{u}'_m$ under a collective point of view. In this case it is said that $\mathbf{u}_m$ dominates $\mathbf{u}'_m$. But $\mathbf{u}_m$ does not dominate $\mathbf{u}'_m$ if there exist at least one controller for which $\mathbf{u}'_m$ is better than $\mathbf{u}_m$. Every non-dominated solution is optimal in some sense because no other solution could dominate it (see Figure 7).

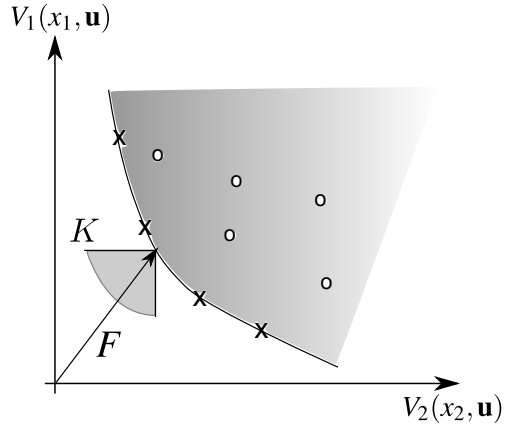

Figure 7 – For every dominated point (O) there is at least one non-dominated point (X) that offer lower costs. All non-dominated points are Pareto-optimal. The cone K defines the Pareto frontier for a given multi-objective optimization problem. A point is Pareto if there is no other point in the cone K with origin in the point.

In a multi-objective optimization framework, all Pareto solutions are obtained by the optimization problem:

$$P_F(x) : \begin{cases} \text{Minimize}_K & \mathbf{F}(x, \mathbf{u}) \\ \text{subject to} & \mathbf{u} \in \mathcal{D}(x) \end{cases} \tag{3.4}$$

where

$$\mathbf{F}(x, \mathbf{u}) = (V_1(x_1, \mathbf{u}_1), \ldots, V_m(x_m, \mathbf{u}_m), \ldots, V_M(x_M, \mathbf{u}_M)) \tag{3.5}$$

is the vector-valued map composed of the controllers' costs, and K is the cone that defines a *partial order* to the problem. A partial order is reflexive, antisymmetric and transitive but does not have the completeness property.

From now on the cone K will be the classical Pareto-cone $\mathbb{R}_+^M$. In the example of Figure 7, the cone $\mathbf{F}(x, \mathbf{u}) - K$ defines a region where any point dominates the point reached by $\mathbf{u}$. There are no points in cone $\mathbf{F}(x, \mathbf{u}) - K$ when $\mathbf{u}$ is Pareto. For a formal treatment of multi-objective optimization see (PAPPALARDO, 2008).

All solutions of problem (3.4) are Pareto-optimal:

Definition 3.2. *A decision vector* $\mathbf{u}^\star$ *is Pareto-optimal if there does not exist another decision vector* $\mathbf{u}$ *in the domain such that* $V_j(x_j, \mathbf{u}) \leqslant V_j(x_j, \mathbf{u}^\star), \forall j = 1, \ldots, M$ *and* $V_m(x_m, \mathbf{u}) < V_m(x_m, \mathbf{u}^\star)$ *for at least one index* m*. In other words, when there is no* $\mathbf{u}$ *such that* $\mathbf{F}(x, \mathbf{u}) \leqslant \mathbf{F}(x, \mathbf{u}^\star)$ *and* $\mathbf{F}(x, \mathbf{u}) \neq \mathbf{F}(x, \mathbf{u}^\star)$.

An overview of methods available to solve problem (3.4) may be found in (GAMBIER, 2008).

3.3.2 Scalarization

Observe that cone K defines a partial order because the completeness property of a total order is lost. A Pareto point can not be compared to another Pareto point.

The classical way, used in the DMPC, to reestablish the total order of options is to define a global cost V for the society of M controllers through a scalarization approach (PAPPALARDO, 2008; LUC, 2008; BOYD; VANDENBERGHE, 2004) according to which the interests of the controllers are aggregated in a global cost shared by all of them:

$$V(x, \mathbf{u}) = \sum_{m=1}^{M} w_m V_m(x_m, \mathbf{u}), \qquad w_m > 0 \tag{3.6}$$

The solution $\mathbf{u} \in \mathcal{D}(x)$ that minimizes this global cost is Pareto-optimal (BOYD; VANDENBERGHE, 2004). The adjustment of weights w_m define a fixed trade off among the controller objectives and, as a consequence, a particular solution in the Pareto set. The adjustment of w_m is done manually and may not be a trivial task.

3.3.3 Cooperative Solution

The decision process in a distributed problem is iterative. The controllers should cooperate in order to reach a Pareto solution. In the so called cooperative DMPC, cooperation is obtained by forcing the

controllers to share the same global cost V. Each cooperative controller solves the following problem over their own variables $\mathbf{u}_m$ given the states x and the variables of the other controllers $\mathbf{u}_{\neg m}$:

$$P_m^{\text{Co}}(x, \mathbf{u}_{\neg m}) : \begin{cases} \text{Minimize}_{\mathbf{u}_m} & V(\mathbf{u}_m | x, \mathbf{u}_{\neg m}^{(E_p)}) \\ \text{subject to} & \mathbf{u}_m \in \mathcal{D}_m(x, \mathbf{u}_{\neg m}^{(E_p)}) \end{cases} \tag{3.7}$$

where the superscript E_p indicates the *readjustment scheme* at each iteration p. Two examples of readjustment schemes are the Gauss-Seidel scheme where $\mathbf{u}_{\neg m}^{(E_p)} = (\mathbf{u}_1^{(p)}, \ldots, \mathbf{u}_{m-1}^{(p)}, \mathbf{u}_{m+1}^{(p-1)}, \ldots, \mathbf{u}_M^{(p-1)})$ and the Jacobi scheme where $\mathbf{u}_{\neg m}^{(E_p)} = \mathbf{u}_{\neg m}^{(p-1)}$.

Definition 3.3. *A* fixed point $\mathbf{u}^\dagger = (\mathbf{u}_1^\dagger, \ldots, \mathbf{u}_M^\dagger)$ *is (globally)* convergent *with respect to a readjustment scheme E_p if it can be obtained as the limit of the iterations:*

$$\begin{cases} \mathbf{u}_m^{(p)} = r_m(\mathbf{u}_{\neg m}^{(E_p)}) \triangleq \arg \sup_{\mathbf{u}_m \in \mathcal{D}_m} V(\mathbf{u}_m | x, \mathbf{u}_{\neg m}^{(E_p)}), \quad \forall m \in \mathcal{M} \\ \mathbf{u}_m^\dagger = \lim_{p \to \infty} \mathbf{u}_m^{(p)} \end{cases} \tag{3.8}$$

where the superscript E_p indicates that the choice $\mathbf{u}_{\neg m}^{(E_p)}$ depends on the readjustment scheme selected.

The readjustment scheme is fundamental to the convergence of the method (BASAR; OLSDER, 1999). One example of simultaneous readjustment is given in (VENKAT et al., 2005, 2006; STEWART et al., 2010). There, the scheme is:

$$\begin{cases} \mathbf{u}_m^{*(p)} = r_m(\mathbf{u}_{\neg m}^{(p-1)}) \\ \mathbf{u}_m^{(p)} = \gamma_m \mathbf{u}_m^{*(p)} + (1 - \gamma_m)\mathbf{u}_m^{(p-1)}, \quad \gamma_m \in (0, 1], \quad \sum_{m \in \mathcal{M}} \gamma_m = 1 \end{cases} \tag{3.9}$$

for all m iterating in parallel, where the reaction function r_m was defined in Equation (3.8) and $\mathbf{u}_m^{(p)}$ is calculated by the convex combination of $\mathbf{u}_m^{*(p)}$ and the previous solution $\mathbf{u}_m^{(p-1)}$.

The authors show that this sequence produces non-increasing global costs and converges to an optimal point in case the costs are convex. These results are repeated here:

Proposition 3.1. *(STEWART et al., 2010) The sequence of joint costs $\{V(x, \mathbf{u}^{(p)})\}_{p \to \infty}$ generated by the simultaneous readjustment scheme* (3.9) *is non-increasing and converges.*

Proof. (Adapted from (STEWART et al., 2010)): From Equation (3.9), where $\sum_{m\in\mathcal{M}}\gamma_m = 1$, and convexity of $V(\mathbf{u}_m|x,\mathbf{u}_{\neg m})$ it follows that

$$\begin{aligned}
V(\mathbf{u}_m^{(p)}|x,\mathbf{u}_{\neg m}^{(p)}) &= V(x,\mathbf{u}^{(p)})\\
&= V\left(x,[\gamma_1\mathbf{u}_1^{*(p)} + (1-\gamma_1)\mathbf{u}_1^{(p-1)},\ldots,\gamma_M\mathbf{u}_M^{*(p)} + (1-\gamma_M)\mathbf{u}_M^{(p-1)}]\right)\\
&= V\left(x,[\gamma_1(\mathbf{u}_1^{*(p)},\ldots,\mathbf{u}_M^{(p-1)}) + \ldots + \gamma_M(\mathbf{u}_1^{(p-1)},\ldots,\mathbf{u}_M^{*(p)})]\right)\\
&\leqslant \sum_{m\in\mathcal{M}}\gamma_m V(\mathbf{u}_m^{*(p)}|x,\mathbf{u}_{\neg m}^{(p-1)})\\
&\leqslant V(\mathbf{u}_m^{(p-1)}|x,\mathbf{u}_{\neg m}^{(p-1)})
\end{aligned}$$

where the first inequality follows from the convexity of V, and the second inequality from the optimality of $\mathbf{u}_m^{*(p)}$. The sequence converges because the cost V is bounded below. □

The requirement $\sum_{m\in\mathcal{M}}\gamma_m = 1$ implies small steps when there are many controllers.

In (CAMPONOGARA; OLIVEIRA, 2009) it is proposed a readjustment scheme sequential by groups of non-neighboring controllers and parallel inside groups. In this scheme there is a sequence of groups $\{K_1,\cdots,K_r\}$ that repeats until convergence, where $K_i \subseteq \mathcal{M}$ and $\bigcup_{i=1}^{r} K_i = \mathcal{M}$. Each group is composed by non-neighboring (non-coupled) controllers that may iterate in parallel. While any controller $\mathcal{C}_m \in K_i$ reacts, all controllers in its neighborhood keep their decisions to the next iteration, that is:

$$\begin{cases}\mathbf{u}_m^{(p)} = r_m(\mathbf{u}_{\nu(m)}^{(p-1)}), & \forall m \in K_i, \quad \nu(m) = \mathcal{M}\backslash K_i\\ \mathbf{u}_{\nu(m)}^{(p)} = \mathbf{u}_{\nu(m)}^{(p-1)}\end{cases} \tag{3.10}$$

before switching to the next group.

This *sequential/parallel* scheme ensures that only non-neighboring controllers will update its decision at iteration p. It leads to a proposition similar to Proposition 3.1:

Proposition 3.2. *The sequence of joint costs* $\{V(x,\mathbf{u}^{(p)})\}_{p\to\infty}$ *generated by sequential/parallel readjustment scheme* (3.10) *is non-increasing and converges.*

Proof. From the optimality of $\mathbf{u}_m^{(p)}$, we have that $V(\mathbf{u}_m^{(p)}|x,\mathbf{u}_{\neg m}^{(p-1)}) \leqslant V(\mathbf{u}_m^{(p-1)}|x,\mathbf{u}_{\neg m}^{(p-1)})$. Because outside groups we have that $\mathbf{u}_{\nu(m)}^{(p)} = \mathbf{u}_{\nu(m)}^{(p-1)}$ then

$V(\mathbf{u}_m^{(p)}|x, \mathbf{u}_{\neg m}^{(p)}) \leqslant V(\mathbf{u}_m^{(p-1)}|x, \mathbf{u}_{\neg m}^{(p-1)})$. The sequence converges because the cost V is bounded below. □

3.3.4 Coupled Constraints

Although all controllers share the same cost function, the distributed optimal solution $\mathbf{u}^\dagger$ may not be coincident with an optimal solution of V. For the readjustment schemes just presented, the convergent distributed solution is guaranteed to be the minimization of V, and thereby Pareto-optimal, only if the constraints are not coupled. This is because coupled constraints may create fixed points other than the minimum of V. See, for example, Figure 8 extracted from (CAMPONOGARA; SCHERER, 2011) that shows a sequential readjustment that leads to a fixed point, where the agents (controllers) can not improve the solution further by using the same readjustment scheme.

To avoid fixed points even with coupled constraints, in (CAMPONOGARA; SCHERER, 2011) is proposed a distributed interior-point method that leads the controllers through a central path exemplified in Figure 8.

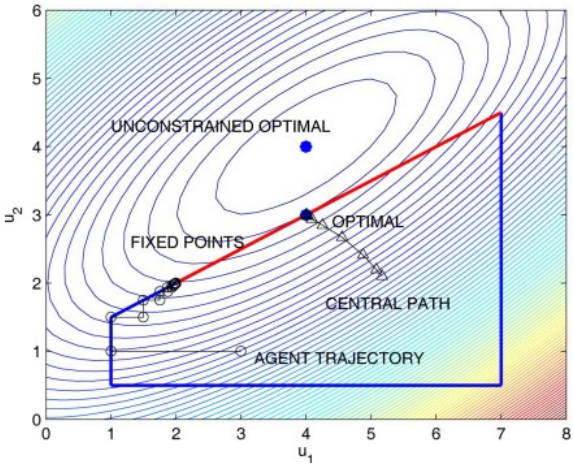

Figure 8 – Extracted from (CAMPONOGARA; SCHERER, 2011), the figure shows the nonequivalence between distributed and optimal solutions under coupled constraints.

4 DISTRIBUTED SATISFICING MPC

This chapter presents an alternative to the cooperative distributed MPC presented in Chapter 3 and constitutes, together with Chapter 5, the main theoretical contribution of this thesis, that is a new distributed controller called Distributed Satisficing MPC (SMPC) (LIMA et al., –).

The classical way of using fixed weights w_m to define a global cost $V(x, \mathbf{u}) = \sum_{m \in \mathcal{M}} w_m V_m(x_m, \mathbf{u})$, where w_m defines a fixed trade-of between the controllers, results in what Stirling (STIRLING; FROST, 2007) calls *categorical altruism*. To explicitly show the altruism embedded in the global cost $V(x, \mathbf{u})$, let us rearrange it as follows:

$$\begin{aligned} V(x, \mathbf{u}) &= w_m V_m(x_m, \mathbf{u}) + \sum_{j \in \mathcal{M} \setminus \{m\}} w_j V_j(x_j, \mathbf{u}) \\ &= w_m \left[V_m(x_m, \mathbf{u}) + \sum_{j \in \mathcal{M} \setminus \{m\}} \lambda_{m,j} V_j(x_j, \mathbf{u}) \right] \end{aligned}$$

where $\lambda_{m,j} = \frac{w_j}{w_m} > 0$. Observe that $\lambda_{m,j}$ is the amount of altruism that controller $\mathcal{C}_m$ assigns to controller $\mathcal{C}_j$ by adding they costs to its own cost.

In the categorical altruism, the controllers are categorically required to subjugate their own welfare, in all situations, in order to benefit the society. This condemns the controller to always please other controllers even if they are by far accomplishing their objectives.

Prett and Garcia, according to Qin e Badgwell (2003), raised another issue:

> Prett and Garcia (1988) commented (...) : The combination of multiple objectives into one objective (function) does not allow the designer to reflect the true performance requirements. [reference in the original](QIN; BADGWELL, 2003)

We advocate that a performance requirement must be defined to each objective, individually. Instead of minimize a rather arbitrary combination of objectives, the controllers will try to satisfy their own requirements, and also the requirements of the others. It will lead to another kind of altruism, called situational.

Situational x categorical altruism: The situational altruism is dy-

namically adjusted at each cycle according to the system's current state, in opposition to the categorical altruism that is fixed and adjusted a priori.

4.1 SATISFICING PROBLEM

The satisficing theory proposed by Simon (1955) defines that each decision maker should decide if a solution is satisfactory and sufficient comparing it against a standard, called aspiration level. If the solution meets the decision maker's aspiration level, the solution is satisficing, that is, it is satisfactory and sufficient (satisficing = satisfy + suffice), and no further optimization is necessary.

In the context of this thesis, the aspiration level will be used to define a set of global solutions that are satisfactory and sufficient to all controllers. Such region is called the satisficing set and is defined as

$$S(x) \triangleq \{\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_M) \mid V_m(x_m, \mathbf{u}) \leqslant \gamma_m,\ \forall m \in \mathcal{M}\}$$

Here, the aspiration level is the *maximal satisficing cost* $\gamma_m > 0$, still satisficing for controller $\mathcal{C}_m$. Any solution in $S(x)$ is locally and globally satisficing. The satisficing controllers, then, try to (distributively) solve the following problem:

$$P^{\mathrm{S}}: \quad \begin{cases} \text{find any} & \mathbf{u} \in \mathcal{D}(x) \\ \text{such that:} & V_m(x_m, \mathbf{u}) \leqslant \gamma_m,\ \forall m \in \mathcal{M} \end{cases} \tag{4.1}$$

where the set

$$\mathcal{D}(x) = \{\mathbf{u} \mid \mathbf{u} \in \mathcal{U}, \mathbf{x} \in \mathcal{X}\}$$

is the problem's domain.

Remark 4.1. *Observe that γ_m is a local requirement, different of the weights w_m that are meaningful only in respect to others. It may be constant or adjusted according to the circumstances and, in some cases, γ_m is associated to physical parameters as is the case of the example in Section 6.1.*

Besides the fact that γ_m is local, another practical advantage of the satisficing problem is that the inclusion of a new controller is performed by adding the cost of the controller and its maximal satisficing cost into the formulation, and reconfiguring only the subsystems with which the new subsystem will be coupled. The definition of γ_m

opens room for what Nikolaou in (NIKOLAOU, 2001) called "achievable targets of performance":

> Similarly to the minimum-work concept in thermodynamics, control theory should provide *achievable targets of performance* and should do so under practical conditions, e.g., in the presence of inequality constraints and model inaccuracies (Morari, 1988) [reference in the original] (NIKOLAOU, 2001, emphasis added).

The satisficing problem P^{S} is a feasibility problem that defines a *set* of possible solutions, all of them satisficing. Any solution $\mathbf{u}$ of P^{S} is in the constrained jointly satisficing set, defined as

$$S_{\mathcal{D}}(x) \triangleq S(x) \cap \mathcal{D}(x)$$

such that $\mathbf{u} \in S_{\mathcal{D}}(x)$ respects the constraints and maintains the cost below or equal to the maximal satisficing cost. The set $S_{\mathcal{D}}(x)$ is convex and must not be empty.

The following section presents a method to select a particular solution in $S_{\mathcal{D}}(x)$, that leads to a situational altruism. In the method, the controllers try to reach a Pareto-optimal solution represented by the analytic center of the satisficing set. This results in an optimal altruism (in the sense that it leads to a Pareto-optimal solution) with an interesting characteristic: the controllers will be more altruistic to the less satisfied controller.

In the Appendix A we show another method where the controllers try to minimize their own selfish costs $V_m(x_m, \mathbf{u})$ offering a minimum of altruism to the other controllers.

4.2 SATISFICING MPC

Although satisficing, a solution in $S_{\mathcal{D}}(x)$ is normally suboptimal because it may not be Pareto-optimal. In the following we propose a controller that finds a particular satisficing solution in $S_{\mathcal{D}}(x)$ that is Pareto-optimal and has an interesting characteristic: the controllers will be more altruistic to the less satisfied controller.

It will be shown that a Pareto solution is obtained if the following

convex optimization problem is solved:

$$P^{\text{SMPC}}: \quad \begin{cases} \min_{\mathbf{u}} \ \sum_{m \in \mathcal{M}} -\gamma_m \cdot \log(\gamma_m - V_m(x_m, \mathbf{u})) \\ \text{subject to} \ \ \mathbf{u} \in \mathcal{D}(x) \end{cases} \tag{4.2}$$

At each time step, the SMPC solves problem (4.2) in a distributed manner. Observe that the logarithms in (4.2) are defined only if $V_m(x_m, \mathbf{u}_m) < \gamma_m$ (strictly), for all $m \in \mathcal{M}$ so that, the controllers must be satisficing. The solution provides the control action $\kappa_m = u_m^\dagger(0)$, for all controller $\mathcal{C}_m$, $m \in \mathcal{M}$, which is applied in a receding horizon scheme. The rest of the optimal trajectory is discarded.

The optimal solution of (4.2) is the constrained analytic center of the satisficing set $S(x)$ (see Figure 9).

Remark 4.2. *Notice that the constrained analytic center of the satisficing set $S(x)$ may be not the analytic center of the constrained satisficing set $S_{\mathcal{D}}(x) = S(x) \cap \mathcal{D}(x)$.*

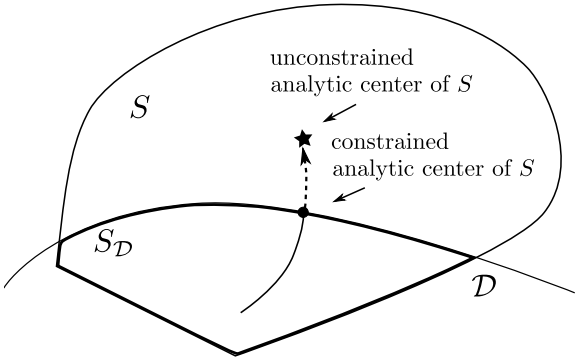

Figure 9 – Optimizing towards the analytic center of the jointly satisficing set $S(x)$. The solution is constrained by the constraints $\mathcal{D}(x)$.

In opposition to the solution obtained by minimizing the objective (3.6), the solution of Problem (4.2) induces a trade off that is not defined a priori and is variable with the current state, as shown in the following section.

4.2.1 Equivalence of the Analytic Center

In the following theorem the optimality properties of the constrained analytic center of the satisficing set, and hence of the optimal solution of problem P^{SMPC}, are studied.

Theorem 4.1. *The solution* $\mathbf{u}^\dagger = (\mathbf{u}_1^\dagger, \ldots, \mathbf{u}_M^\dagger)$ *of Problem* P^{SMPC}, *is Pareto optimal and equivalent to a solution* $\mathbf{u}^\star = (\mathbf{u}_1^\star, \ldots, \mathbf{u}_M^\star)$ *of problem*

$$V(x, \mathbf{u}^\star) = V(x) \triangleq \min \sum_{m=1}^{M} w_m(x_m) V_m(x_m, \mathbf{u}) \quad \text{subject to } \mathbf{u} \in \mathcal{D}(x) \tag{4.3}$$

where $w_m(x_m)$ *is defined as*

$$\begin{aligned} w_m(x_m) &= \gamma_m / (\gamma_m - V_m(x_m, \mathbf{u}^\dagger)) \\ &= \frac{\gamma_m}{\gamma_m - V_m(x_m)} \end{aligned} \tag{4.4}$$

and $V_m(x_m) = V_m(x_m, \mathbf{u}^\dagger)$.

Proof. Let us describe the convex set $\mathcal{D}(x)$ by

$$\mathcal{D}(x) = \{\mathbf{u} \mid h_i(x, \mathbf{u}) \leqslant 0,\ g_j(x, \mathbf{u}) = 0, \quad i = 1, \ldots, q,\ j = 1, \ldots, r\}$$

where h_i are convex functions and g_j are affine functions.

The Karush Kun Tucker (KKT) conditions (BOYD; VANDENBERGHE, 2004) will be applied to problem (4.2) and (4.3) to show their equivalence:

1) An optimal solution $\mathbf{u}^\star = (\mathbf{u}_1^\star, \ldots, \mathbf{u}_M^\star)$ to problem (4.3) satisfies the KKT conditions, that is, there exist Lagrange multipliers $\lambda_i^\star$ and $\nu_j^\star$ such that

$$\sum_{m=1}^{M} w_m(x_m) \nabla V_m(x_m, \mathbf{u}^\star) + \sum_{i=1}^{q} \lambda_i^\star \nabla h_i(x, \mathbf{u}^\star) + \sum_{j=1}^{r} \nu_j^\star \nabla g_j(x, \mathbf{u}^\star) = \mathbf{0}$$

for $i = 1, \ldots, q$ and $j = 1, \ldots, r$, and

$$\begin{aligned} h_i(x, \mathbf{u}^\star) &\leqslant 0, \\ g_j(x, \mathbf{u}^\star) &= 0, \\ \lambda_i^\star h_i(x, \mathbf{u}^\star) &= 0, \\ \lambda_i^\star &\geqslant 0 \end{aligned}$$

2) On the other hand, the solution $\mathbf{u}^\dagger = (\mathbf{u}_1^\dagger, \ldots, \mathbf{u}_M^\dagger)$ of problem (4.2) requires that there exist Lagrange multipliers $\lambda_i^\dagger$ and $\nu_j^\dagger$ such that

$$\sum_{m=1}^{M} \frac{\gamma_m}{\gamma_m - V_m(x_m, \mathbf{u}^\dagger)} \nabla V_m(x_m, \mathbf{u}^\dagger) + \sum_{i=1}^{q} \lambda_i^\dagger \nabla h_i(x, \mathbf{u}^\dagger) + \sum_{j=1}^{r} \nu_j^\dagger \nabla g_j(x, \mathbf{u}^\dagger) = \mathbf{0}$$

for $i = 1, \ldots, q$ and $j = 1, \ldots, r$, and

$$\begin{aligned} h_i(x, \mathbf{u}^\dagger) &\leqslant 0, \\ g_j(x, \mathbf{u}^\dagger) &= 0, \\ \lambda_i^\dagger h_i(x, \mathbf{u}^\dagger) &= 0, \\ \lambda_i^\dagger &\geqslant 0 \end{aligned}$$

Comparing 1) and 2), it follows that, when

$$w_m(x_m) = \frac{\gamma_m}{\gamma_m - V_m(x_m, \mathbf{u}^\dagger)}$$

the solution $\mathbf{u}^\dagger$ also fulfills the KKT conditions in 1) with $\mathbf{u}^\star = \mathbf{u}^\dagger$, $\lambda_i^\star = \lambda_i^\dagger$ and $\nu_j^\star = \nu_j^\dagger$. □

Theorem 4.1 proves that solving Problem (4.2) is equivalent to solving a centralized problem based on a global cost function with state dependent weights. Implicitly, the satisficing controllers share the global objective given by

$$V(x, \mathbf{u}) = \sum_{m=1}^{M} w_m(x_m) V_m(x_m, \mathbf{u}) \tag{4.5}$$

such that the trade off among the controllers are not fixed nor defined

by the designer.

Remark 4.3. *The weights $w_m(x_m)$ are known only after the solution of Problem P^{SMPC}, which implies that the equivalent centralized problem in Equation* (4.3) *just theoretical.*

According to the expression of $w_m(x_m)$ (Equation (4.4)), the farther a controller is from its maximal satisficing cost, more it is satisfied and less weight will be assigned to his local cost function. Observe that the equivalent weights $w_m(x_m)$ are normalized by γ_m:

$$w_m(x_m) = \frac{\gamma_m}{\gamma_m - V_m(x_m)} = \frac{1}{1 - \frac{V_m(x_m)}{\gamma_m}}$$

For example, being two controllers with their costs 10% from their satisficing costs

$$\gamma_1 = 100 \qquad \gamma_2 = 20$$
$$V_1(x_1) = 90 \qquad V_2(x_2) = 18$$

results in equal weights,

$$\frac{\gamma_1}{\gamma_1 - V_1(x_1)} = 10 \qquad \frac{\gamma_2}{\gamma_2 - V_2(x_2)} = 10$$

4.2.2 Optimal Altruism

According to Theorem 4.1, a solution of problem P^{SMPC} corresponds to a global objective function with a situational altruism given by:

$$\lambda_{m,j} = \frac{w_j}{w_m} = \frac{1 - {V_m(x_m)}/{\gamma_m}}{1 - {V_j(x_j)}/{\gamma_j}} > 0 \tag{4.6}$$

for all $m \in \mathcal{M}, j \in \mathcal{M}\backslash\{m\}$, which depends on the maximum dissatisfaction γ_m of each controller and on its cost function $V_m(x_m, \mathbf{u}^\dagger)$ which, in turn, depends on the current state and actions. Notice that a satisficing solution of problem P^{SMPC} implies a cost *strictly* below their maximum satisficing, $V_m(x_m) < \gamma_m$, because of the logarithm function in the problem's objective. Therefore the situational altruism given by Equation (4.6) is well defined and positive.

In every MPC cycle, problem P^{SMPC} is solved in a distributed manner, as shown in Section 4.2.3, resulting in a Pareto-optimal solution that is equivalent to an optimal altruism. Moreover, no one of

the controllers will be altruistic to the extent of being unsatisfied, that is, its cost will always be lower than or equal to its maximal satisficing cost. Also, from (4.6) it can be seen that the closer an affected controller is from its maximum cost, that is, if $V_j(x_j) \to \gamma_j$, less it is satisfied and more altruism will be deserved to it. On the contrary, if $V_m(x_m) \to \gamma_m$ then controller $\mathcal{C}_m$ will be less satisfied and will be less willing to be altruistic. This is a behavior difficult to obtain in classical approaches.

4.2.3 Distributed Solution

The constrained analytic center problem P^{SMPC} (4.2) is conceptually distributed in the sense that it is the combination of distributed costs of a network of subsystems with their own states and controls. Although this problem can be solved by a centralized solver, the concept is reinforced if the problem is iteratively solved by distributed controllers $\mathcal{C}_m$, each one with its own problem

$$P_m^{\text{SMPC}}(x, \mathbf{u}_{\neg m}) : \begin{cases} \min\limits_{\mathbf{u}_m} \sum\limits_{j \in \mathcal{M}} -\gamma_j \cdot \log(\gamma_j - V_j(x_j, \mathbf{u})) \\ \text{subject to} \;\; \mathbf{u}_m \in \mathcal{D}_m(x, \mathbf{u}_{\neg m}) \end{cases} \tag{4.7}$$

where $\mathbf{u}_{\neg m}$ denotes the action profile of all the other controllers but controller $\mathcal{C}_m$, and $\mathcal{D}_m(x, \mathbf{u}_{\neg m})$ is the projection of $\mathcal{D}(x)$ over the space spanned by $\mathbf{u}_m$ with $\mathbf{u}_{\neg m}$ fixed. The profiles $\mathbf{u}_{\neg m}$ are constants in Problem P_m^{SMPC} as well as the actual states.

Let us suppose that the coupled constraints $\mathcal{D}_m(x, \mathbf{u}_{\neg m})$ can be represented by convex functions $h_{m,i}$, $i = 1, \ldots, r_m$, such that

$$\mathcal{D}_m(x, \mathbf{u}_{\neg m}) = \{\mathbf{u}_m \mid h_{m,i}(\mathbf{u}_m; x, \mathbf{u}_{\neg m}) \leqslant 0, \;\; i = 1, \ldots, r_m\}$$

where the parameters of $h_{m,i}(\mathbf{u}_m; x, \mathbf{u}_{\neg m})$ are explicitly shown after the semicolon mark. Under this assumption, the set of problems $\{P_m^{\text{SMPC}}\}_{m=1}^{M}$ can be solved in two phases, phase I and phase II described below.

4.2.3.1 Phase I

Phase I is used to calculate a feasible solution to be used in phase II, and also to allow the controllers to negotiate in case of $S(x) \cap \mathcal{D}(x) = \varnothing$. Usually, phase I will be used just once due to the recursive feasibility

of the SMPC with guarantee of stability.

The phase I of controller $\mathcal{C}_m$ consists in solving:

$$
\begin{aligned}
PI_m : \min_{(\mathbf{u}_m, \tilde{\mathbf{s}}_m)} \; & f_m^{\mathrm{I}} = \sum_{j \in \mathcal{M}} s_j + \sum_{i=1}^{r_m} s_{m,i} \\
\text{s.t.} : \; & V_j(x_j, \mathbf{u}) < \gamma_j + s_j, \;\; \forall j \in \mathcal{M} \\
& h_{m,i}(\mathbf{u}_m; x, \mathbf{u}_{\neg m}) < s_{m,i}, \;\; i = 1, \ldots, r_m \\
& s_j \geqslant 0, \quad \forall j \in \mathcal{M} \\
& s_{m,i} \geqslant 0, \quad i = 1, \ldots, r_m
\end{aligned}
\tag{4.8}
$$

where $\tilde{\mathbf{s}}_m = (s_j, s_{m,i} \mid \forall j \in \mathcal{M}, i = 1, \ldots, r_m)$. Observe that phase I is always feasible because $\tilde{\mathbf{s}}_m \geqslant 0$ can be set as large as necessary. Let $f_m^{\mathrm{I}\star}$ be an optimal objective to Problem PI_m. If $f_m^{\mathrm{I}\star} = 0$ for all $m \in \mathcal{M}$, then the interior of the satisficing set is nonempty. If $f_m^{\mathrm{I}\star} > 0$ for any m, then there does not exist a simultaneously satisficing solution for all the controllers, in which case $\{m \in \mathcal{M} : s_m > 0\}$ is the subset of controllers that cannot be satisfied.

This phase is normally left implicit in the distributed literature since it is considered that the first feasible solution may be obtained measuring the system when in steady-state. In the satisficing approach, phase I becomes important because the system may be not satisficing at first, and also because there exists the possibility of negotiation as explained below.

4.2.3.2 Negotiation

From the definition of the satisficing set and cost functions, it can be seen that a smaller sensitivity α_m and/or greater dissatisfaction γ_m lead to larger satisficing sets. So, when the controllers can not be simultaneously satisfied, they have to negotiate by adjusting their sensitivity to cost or their maximal level of dissatisfaction.

In general, the negotiation protocol is context-dependent. An example of negotiation is found in Section 6.1 where some controllers agree in degrade their performance based in a predefined rule. Other protocols may be devised. A possibility is, once feasibility is declared impossible by the algorithm of phase I, to increase the maximal satisficing costs γ_m of all controllers by the value of their slack variables s_m. Another possibility is to define a hierarchy in order to decide which controller negotiates first.

Observe that, if closed-loop stability is guaranteed (see Chapter 5) then recursive feasibility is also guaranteed, so that an initially feasible solution will be always feasible and no further negotiation will be necessary, unless the control loses robustness due to measurements and model errors.

4.2.3.3 Phase II

Phase II solves, starting from the solution found in phase I, the set $\{PII_m\}_{m=1}^{M}$ of problems given by:

$$\begin{aligned} PII_m : \min_{\mathbf{u}_m} \; & f_m^{\mathrm{II}} = \sum_{j \in \mathcal{M}} -\gamma_j \cdot \log(\gamma_j - V_j(x_j, \mathbf{u})) \\ \text{s. t.: } & h_{m,i}(\mathbf{u}_m; x, \mathbf{u}_{\neg m}) \leqslant 0, \; i = 1, \ldots, r_m \end{aligned} \tag{4.9}$$

where functions f_m^{II}, a slightly modified log barrier (BOYD; VANDENBERGHE, 2004), force the solution in the direction of the analytic center of the satisficing set $S(x)$.

4.2.3.4 Algorithm

Let phase I and phase II be the class of problems such that the functions f_m^{I}, f_m^{II} and $h_{m,i}$ are convex and twice continuously differentiable and the problems are strictly feasible. In (CAMPONOGARA; SCHERER, 2011, Section IV), it is developed a distributed interior point algorithm designed to handle this kind of problems with coupled constraints.

Camponogara e Scherer (2011): For a given current state, neighboring conditions and initial feasible solution $\hat{\mathbf{u}} = (\hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_M)$, the controllers solve in an inner loop a series of problems

$$P_m(\epsilon^{(p)}, x, \mathbf{u}_{\neg m}) : \; \min_{\mathbf{u}_m} \theta_m(\mathbf{u}_m; \epsilon^{(p)}, x, \mathbf{u}_{\neg m}) = f_m + \epsilon^{(p)} \phi_m$$

where f_m is f_m^{I} or f_m^{II} if in phase I or phase II, respectively. The constraints are included in the objective by the barrier function

$$\phi_m(\mathbf{u}_m; x, \mathbf{u}_{\neg m}) = \sum_{i=1}^{r} -\log(-h_{m,i}(\mathbf{u}_m; x, \mathbf{u}_{\neg m}))$$

The parameter $\epsilon^{(p)} > 0$ is decreased at every outer iteration p until convergence, according to Algorithm 1.

Algorithm 1: Outer loop: interior-point method for solving P^{SMPC}

input: a strictly feasible $\mathbf{u}^{\text{s}}$, initial $\epsilon^{(0)}$, decrease rate $\mu < 1$, number of constraints r, and tolerance $\tau > 0$
initialize: $p := -1$;
repeat
  $p := p + 1$;
  centering step: obtain $\mathbf{u}^{(p)} = (\mathbf{u}_1, \ldots, \mathbf{u}_M)$ by distributively solving a series of problems $\{P_m(\epsilon^{(p)}, x, \mathbf{u}_{\neg m})\}_{m \in \mathcal{M}}$ starting from $\mathbf{u}^{\text{s}}$ (see Algorithm 2);
  if $r\epsilon^{(p)} > \tau$ **then**
    $\mathbf{u}^{\text{s}} := \mathbf{u}^{(p)}$;
    $\epsilon^{(p+1)} := \mu\epsilon^{(p)}$;
until $r\epsilon^{(p)} \leqslant \tau$;
output: $\mathbf{u}^{(p)}$

The outer loop forces the global solution $\mathbf{u}^{(p)}$ to follow a central path as shown in Figure 8 (Page 45) and eventually converge with arbitrary precision to the constrained optimal for an $\epsilon^{(p)} \to 0$ sufficiently small. After convergence, the controllers apply the first action $\kappa_m = u_m(0)$ of their plans $\mathbf{u}_m = \mathbf{u}_m^{(p)}$ and use it as the initial solution for the next iteration.

The problems $P_m(\epsilon^{(p)}, x, \mathbf{u}_{\neg m})$, for all $m \in \mathcal{M}$, are solved in a serial/parallel scheme whereby the controllers are divided in groups $K_i \subseteq \mathcal{M}$ such that $\bigcup_{i=1}^{q} K_i = \mathcal{M}$ and such that each group is composed by non-neighboring (non-coupled) controllers. All couplings are between groups. If the system is fully coupled, then there will be one controller per group. The problems are solved following the sequence of groups $\{K_1, \ldots, K_q\}$ that repeats until convergence.

Algorithm 2: Inner loop: distributed solution of the centering step

input: a strictly feasible $\mathbf{u}$, current state x, parameter $\epsilon^{(p)}$, and tolerance $\eta > 0$

initialize: $l := 0$; $i := 1$; $\mathbf{u}^{(0)} := \mathbf{u}$;

while $\|(\nabla\theta_1, \ldots, \nabla\theta_M)\| > \eta$ **do**

  for *each* $m \in K_i$ *in parallel* **do**

    use an available solver to obtain the solution $\mathbf{u}_m^{(l+1)}$ of $P_m(\epsilon^{(p)}, x, \mathbf{u}_{\neg m})$

  for *each* $m \notin K_i$ *in parallel* **do**

    $\mathbf{u}_m^{(l+1)} := \mathbf{u}_m^{(l)}$;

  $l := l + 1$, $i := (l \mod q) + 1$;

output: $\mathbf{u}^{(l)}$

5 STABILITY

The evolution of the subsystems, function of the actual state $x = (x_1, \ldots, x_M)$ and inputs $u = (u_1, \ldots, u_M)$, may be represented by the equation

$$x^+ = f(x, u)$$

where $f(x, u)$ groups the subsystems' dynamic equations (2.5) described on page 33, and x^+ is the successor state. The subsystems are submitted to a closed loop control, such that $u = \kappa(x)$ and $\kappa = (\kappa_1, \ldots, \kappa_M)$ is a vector with the satisficing MPC outputs. Observe that the control action is function of the current state:

$$\kappa = g(x)$$

so that the closed loop evolution of the system is given by

$$x^+ = f(x, g(x)) = F(x) \tag{5.1}$$

We want to show that the evolution (5.1) under satisficing control is asymptotically stable.

Definition 5.1. *(Stability) The origin, solution of the system* $x(k+1) = F(x(k))$*, is stable if for all* $\epsilon > 0$ *there exists* $\delta = \delta(\epsilon)$ *such that*

$$\forall x(0): \ |x(0)| \leqslant \delta \Rightarrow |x(k)| \leqslant \epsilon, \ \forall k$$

Stability of the origin means that any solution that starts near the origin will stay near it for all times.

Definition 5.2. *(Asymptotic stability) The origin, solution of the system* $x(k+1) = F(x(k))$*, is asymptotically stable if*

(stability:) *it is stable,*

(convergence:) *and, there exists a constant* $\delta > 0$ *such that*

$$\forall x(0): \ |x(0)| \leqslant \delta \Rightarrow x(k) \to 0 \ \textit{when} \ k \to \infty$$

With asymptotic stability, the solution will move closer to the origin as time elapses.

Asymptotic stability may be characterized by the existence of a *strict Lyapunov function* defined below.

Definition 5.3. *A definite positive function $V : \mathbb{R}^n \mapsto \mathbb{R}_+$ is a strict Lyapunov function in $\mathbb{X}_S \subset \mathbb{R}^n$ if there exist class $\mathcal{K}$ functions*[1] *$\sigma_1(x)$, $\sigma_2(x)$ and $\sigma_3(x)$ such that*

$$V(x) \geqslant \sigma_1(|x|), \ \forall x \in \mathbb{X}_S$$
$$V(x) \leqslant \sigma_2(|x|), \ \forall x \in \Omega \subseteq \mathbb{X}_S$$

and

$$\Delta V(x) \leqslant -\sigma_3(|x|)$$

where $\Delta V(x) = V(F(x)) - V(x)$, for all x in $\mathbb{X}_S$ and Ω is a subset of $\mathbb{X}_S$ with the origin in its interior.

5.1 STABILITY OF THE SMPC

Let us consider the following SMPC formulation,

$$\widetilde{P}^{\text{SMPC}}: \quad \begin{cases} \min_{\mathbf{u}} \ \sum_{m\in\mathcal{M}} -\gamma_m \cdot \log(\gamma_m - V_m(x_m, \mathbf{u})) \\ \text{subject to} \ \ \mathbf{u} \in \widetilde{\mathcal{D}}(x) \end{cases} \tag{5.2}$$

where, to impose asymptotic stability, the cost $V_m(x_m, \mathbf{u})$ is modified to include a positive definite terminal cost $V_{f,m}(x_m)$ such that

$$V_m(x_m, \mathbf{u}) = \sum_{k=0}^{N-1} \ell_m(x_m(k), u_m(k)) + V_{f,m}(x_m(N)) \tag{5.3}$$

and the set $\widetilde{\mathcal{D}}(x)$ is given by:

$$\widetilde{\mathcal{D}}(x) = \{\mathbf{u} \mid \mathbf{u} \in \mathcal{U}, \ \mathbf{x} \in \mathcal{X}, \ \text{stability constraints C.1 and C.2}\}$$

which includes the following stability conditions

$$x(N) \in \Omega \tag{C.1}$$
$$V_m(x_m, \mathbf{u}) \leqslant \rho_m, \ \forall m \in \mathcal{M} \tag{C.2}$$

where, by doing

$$\rho_m^+ = V_m(x_m) \tag{5.4}$$

[1] A scalar function $\sigma(x)$, $x \geqslant 0$, belongs to class $\mathcal{K}$ if it is continuous, strictly increasing and $\sigma(0) = 0$. A scalar function $\sigma(x)$, $x \geqslant 0$, belongs to class $\mathcal{K}_\infty$ if $\sigma(x) \in \mathcal{K}$ and $\lim_{x\to\infty} \sigma(x) = \infty$.

a contraction $V_m(x^+) \leqslant V_m(x_m)$ is imposed to the optimal local costs, that is, the optimal local costs do not increase at each next time. Condition C.1 is a terminal condition, and Ω is a terminal set.

As will be clear later, the maximal satisficing costs γ_m are required to be constant:

$$\gamma_m^+ = \gamma_m, \text{ constant} \tag{5.5}$$

In the following, the optimal value $V(x)$ of the SMPC ($\tilde{P}^{\text{SMPC}}$), evaluated in the optimal solution $\mathbf{u}^\dagger$, will be characterized as a strict Lyapunov function. First, let us define invariance and assume the following:

Definition 5.4. *(Admissible positive control invariant set) A set $\Omega \subseteq \mathbb{X}$ is admissible positive control invariant if, for all $x(k) \in \Omega$, there exist $u(k) \in \mathbb{U}$, such that $x(k+1) = f(x(k), u(k)) \in \Omega$ for all $k \geqslant 0$.*

Assumption 5.1. *The set $\Omega \subseteq \mathbb{X}$ is positive control invariant and admissible under a jointly control $\vartheta = (\upsilon_1, \ldots, \upsilon_m, \ldots, \upsilon_M) \in \mathbb{U}$ and the terminal costs $V_{f,m}(x_m)$ are definite positive with*

$$V_{f,m}(x_m^+) \leqslant V_{f,m}(x_m) - \ell_m(x_m, \upsilon_m)$$

for all $x = (x_1, \ldots, x_M) \in \Omega$ and all $m \in \mathcal{M}$.

Assumption 5.1 establishes the invariance of the set Ω and it also establishes that the terminal cost of all controllers are control Lyapunov functions (CLFs), simultaneously decreasing in Ω. A common choice of terminal cost and terminal constraint that satisfies Assumption 5.1 is $V_{f,m}(x_m) = 0$ and $\Omega = \{0\}$, for all $m \in \mathcal{M}$, such that $\vartheta = 0$. Another possibility is presented by Maestre et al. (2011) that shows a method to calculate a matrix K_m, a positive definite matrix P_m and a set Ω_m such that $\upsilon_m = K_m x_m$, $V_{f,m}(x_m) = x_m' P_m x_m$, and $\Omega = \Omega_1 \times \ldots \times \Omega_M \subseteq \mathbb{X}$ respect Assumption 5.1.

Let us also define the next time warm-start $\hat{\mathbf{u}}^\dagger$ (MAYNE et al., 2000) based in the last known optimal solution $\mathbf{u}^\dagger = (\mathbf{u}_1^\dagger, \ldots, \mathbf{u}_M^\dagger)$, according to the following rule:

$$\hat{\mathbf{u}}^\dagger = (\hat{\mathbf{u}}_1^\dagger, \ldots, \hat{\mathbf{u}}_m^\dagger, \ldots, \hat{\mathbf{u}}_M^\dagger)$$

$$\hat{\mathbf{u}}_m^\dagger = (u_m^\dagger(1), \ldots, u_m^\dagger(N-1), \upsilon_m),$$

and υ_m satisfies Assumption 5.1. The warm-start is used to derive useful results:

Lemma 5.1. *(Local convergence of the warm-start) The warm-start causes the local costs to decrease in a rate given by*

$$V_m(x_m^+, \hat{\mathbf{u}}^\dagger) - V_m(x_m) \leqslant -\ell_m(x_m, \kappa_m) \tag{5.6}$$

where $\kappa_m = u_m^\dagger(0)$ *is the control action of controller* $\mathcal{C}_m$*.*

Proof. The increment of the local costs are given by

$$V_m(x_m^+, \hat{\mathbf{u}}^\dagger) - V_m(x_m) = -\ell_m(x_m, \kappa_m) + T$$

where the term T, after some eliminations, is equal to

$$T = \ell_m(x_m(N), \upsilon_m) + V_{f,m}(x_m(N+1)) - V_{f,m}(x_m(N))$$

which is less than or equal to zero due to Assumption 5.1. It results that

$$V_m(x_m^+, \hat{\mathbf{u}}^\dagger) - V_m(x_m) \leqslant -\ell_m(x_m, \kappa_m)$$

□

Lemma 5.2. *(Recursive feasibility of the warm-start) The next time warm-start*

$$\hat{\mathbf{u}}^\dagger = (\hat{\mathbf{u}}_1^\dagger, \ldots, \hat{\mathbf{u}}_M^\dagger)$$

is feasible.

Proof. By construction, we have that $\hat{\mathbf{u}}^\dagger \in \mathcal{U}$. We also have that the resulting trajectory $\mathbf{x}^+ = (x(2), \ldots, x(N+1))$ has its first elements $(x(2), \ldots, x(N))$ feasible with $x(N) \in \Omega$. Because Assumption 5.1 establishes that Ω is invariant under action $\upsilon = (\upsilon_1, \ldots, \upsilon_M)$, the tail of all the trajectories remains in Ω, that is, $x(N+1) \in \Omega \subseteq \mathbb{X}$. Then, we have $\hat{\mathbf{u}}^\dagger \in \mathcal{U}$, $\mathbf{x}^+ \in \mathcal{X}$, $x(N+1) \in \Omega$. Because of Lemma 5.1 and condition (5.5), the warm-start is also satisficing since $V_m(x_m^+, \hat{\mathbf{u}}^\dagger) \leqslant V_m(x_m) - \ell(x_m, \kappa_m) \leqslant \gamma_m^+$. Altogether, the warm-start $\hat{\mathbf{u}}^\dagger \in S(x^+) \cap \widetilde{\mathcal{D}}(x^+)$ is feasible. □

We want to prove stability by proving that the cost function $V(x)$ is a Lyapunov function. First, let us show that:

Lemma 5.3. *The cost function* $V(x)$ *is bounded below and above by* $\mathcal{K}$*-functions.*

Proof. Substituting $w_m(x_m)$ in the expression of $V(x)$, we obtain

$$V(x) = \sum_{m=1}^{M} \frac{\gamma_m \cdot V_m(x_m)}{\gamma_m - V_m(x_m)}$$

The bounds of $V(x)$ will be derived considering that

a) $\theta(s) = \frac{\gamma s}{\gamma - s}$ defined in $[0, \gamma)$ is a $\mathcal{K}$-function and $\theta(s) \geqslant s$

b) $\beta(|x_1|) + \beta(|x_2|) + ... + \beta(|x_M|)$
$\geqslant \beta((|x_1| + |x_2| + ... + |x_M|)/2^M)$
$= \sigma_1(|x_1| + |x_2| + ... + |x_M|) \geqslant \sigma_1(|x|)$

c) $\theta \circ \beta_f(|x_m|) = \sigma(|x_m|)$

d) $\sum_m \sigma(|x_m|) \leqslant M\sigma(|x|) = \sigma_2(|x|)$, since $|x_m| \leqslant |x|$

where θ, β, β_f, σ, σ_1 and σ_2 are $\mathcal{K}$-functions.

Based on properties a) and b), and knowing that $V_m(x_m)$ is greater than or equal to $x_m' Q_m x_m$, we obtain

$$V(x) \geqslant \sum_m x_m' Q_m x_m \geqslant q \sum_m |x_m|^2 \geqslant \sigma_1(|x|)$$

for all $x \in \mathbb{X}_S \triangleq \{x = (x_1, \ldots, x_M) \mid \gamma_m - V_m(x_m) > 0, m = 1, \ldots, M\}$. It can be shown that $V_m(x_m) \leqslant V_{f,m}(x_m)$ when $x \in \Omega$ (see Proposition 5.5) and so, in virtue of c), d) and $V_f(x_m) \leqslant \beta_f(|x_m|)$, we have that

$$V(x) \leqslant \sum_m \theta(V_f(x_m)) \leqslant \sum_m \theta(\beta_f(|x_m|) = \sum_m \sigma(|x_m|) \leqslant \sigma_2(|x|)$$

for all $x \in \Omega$.

In conclusion,

$$V(x) \geqslant \sigma_1(|x|), \ \forall x \in \mathbb{X}_S \tag{5.7a}$$
$$V(x) \leqslant \sigma_2(|x|), \ \forall x \in \Omega \tag{5.7b}$$

□

The main result then follows:

Theorem 5.4. *The optimal satisficing MPC ($\widetilde{P}^{\mathrm{SMPC}}$) with contraction $\rho_m^+ = V_m(x_m)$ is asymptotically stabilizing if γ_m is constant, for all m.*

Proof. Considering that initially $V_m(x_m) \leqslant \gamma_m$, then, because $V_m(x_m^+) \leqslant V_m(x_m)$ (condition C.2) and $\gamma_m^+ = \gamma_m$, we have that $V_m(x_m^+) \leqslant \gamma_m^+$ and the SMPC is always feasible.

The cost increment is given by:

$$\begin{aligned}\Delta V(x) &= V(x^+) - V(x) \\ &= \sum_{m=1}^{M} w_m(x_m^+) V_m(x_m^+) - \sum_{m=1}^{M} w_m(x_m) V_m(x_m)\end{aligned}$$

From optimality we have $V(x^+) \leqslant V(x^+, \hat{\mathbf{u}}^+)$ and, then

$$\Delta V(x) \leqslant \sum_{m=1}^{M} w_m(x_m^+) V_m(x_m^+, \hat{\mathbf{u}}^+) - \sum_{m=1}^{M} w_m(x_m) V_m(x_m)$$

If

$$w_m(x_m^+) \leqslant w_m(x_m)$$

then

$$\Delta V(x) \leqslant \sum_{m=1}^{M} w_m(x_m)[V_m(x_m^+, \hat{\mathbf{u}}^+) - V_m(x_m)] \tag{5.8}$$

that, according to Lemma 5.1, results in

$$\Delta V(x) \leqslant -\sum_{m=1}^{M} w_m(x_m) \ell_m(x_m, \kappa_m)$$

In other words, it is sufficient for cost decreasing that $w_m(x_m)$ is not increasing for all $m \in \mathcal{M}$. Indeed, from $V_m(x_m^+) \leqslant V_m(x_m)$ (condition C.2) and $\gamma_m^+ = \gamma_m$ we have that

$$\gamma_m^+ - V_m(x_m^+) \geqslant \gamma_m - V_m(x_m)$$

that, from the definition of $w_m(x_m)$, results in $w_m(x_m^+) \leqslant w_m(x_m)$.

The negative increment of the optimal cost together with Lemma 5.3 characterize the optimal cost $V(x)$ as a control Lyapunov function and prove that the dynamic system is asymptotically stabilized by the optimal solution of the distributed SMPC. □

Remark 5.1. *In Theorem 5.4, the maximal satisficing cost γ_m was required to be constant. For the majority of applications, this limitation seems reasonable.*

5.1.1 Distributed Solution

The distributed solution shown in Section 4.2.3 shall be modified to include the warm-start and the stabilizing constraints.

The stabilizing distributed problems are

$$\widetilde{P}_m^{\mathrm{SMPC}}(x, \mathbf{u}_{\neg m}) : \begin{cases} \min\limits_{\mathbf{u}_m} \sum\limits_{j \in \mathcal{M}} -\gamma_j \cdot \log(\gamma_j - V_j(x_j, \mathbf{u})) \\ \text{subject to} \;\; \mathbf{u}_m \in \widetilde{\mathcal{D}}_m(x, \mathbf{u}_{\neg m}) \end{cases}$$

where $\mathbf{u}_{\neg m}$ denotes the action profile of all the other controllers but controller $\mathcal{C}_m$, and $\widetilde{\mathcal{D}}_m(x, \mathbf{u}_{\neg m})$ is the projection of $\widetilde{\mathcal{D}}(x)$ over the space spanned by $\mathbf{u}_m$ with $\mathbf{u}_{\neg m}$ fixed. The profiles $\mathbf{u}_{\neg m}$ are constants in Problem $\widetilde{P}_m^{\mathrm{SMPC}}$ as well as the actual states.

Algorithm 1, described in Section 4.2.3, shall be modified to use the warm-start $\hat{\mathbf{u}}^+ = \hat{\mathbf{u}}^\dagger$ as an initial solution for the next iteration as shown in Algorithm 3.

Algorithm 3: Outer loop modified for using the warm-start

input: a strictly feasible $\mathbf{u}^{\mathrm{s}}$, initial $\epsilon^{(0)}$, decrease rate $\mu < 1$, number of constraints r, and tolerance $\tau > 0$
initialize: $p := -1$;
repeat
  $p := p + 1$;
  centering step: obtain $\mathbf{u}^{(p)} = (\mathbf{u}_1, \ldots, \mathbf{u}_M)$ by distributively solving a series of problems $\{P_m(\epsilon^{(p)}, x, \mathbf{u}_{\neg m})\}_{m \in \mathcal{M}}$ starting from $\mathbf{u}^{\mathrm{s}}$ (see Algorithm 2);
  if $r\epsilon^{(p)} > \tau$ **then**
    $\mathbf{u}^{\mathrm{s}} := \hat{\mathbf{u}}^+$, obtained from $\mathbf{u}^{(p)}$;
    $\epsilon^{(p+1)} := \mu\epsilon^{(p)}$;
until $r\epsilon^{(p)} \leqslant \tau$;
output: $\mathbf{u}^{(p)}$

Remark 5.2. *The recursive feasibility of the SMPC (Lemma 5.2) has an interesting consequence: when the first solution is feasible (by negotiation or not), all other solutions will be feasible. Then, no further negotiation will be necessary, unless the control loses robustness due to measurements and model errors.*

5.2 WHEN ANY SATISFICING SOLUTION IS STABILIZING

In the following it will be shown a stabilizing formulation that guarantee stability based in a feasibility problem, without a specified objective. By proving that any feasible solution of Problem (5.9) below is asymptotically stabilizing, then it is established that any satisficing scheme is stabilizing.

The stabilizing satisficing problem is defined as

$$\overline{P}^{\mathrm{S}}(x): \quad \begin{cases} \text{find any} & \mathbf{u} \in \overline{\mathcal{D}}(x) \\ \text{such that:} & V_m(x_m, \mathbf{u}) \leqslant \gamma_m, \ \forall m \in \mathcal{M} \end{cases} \tag{5.9}$$

where $V_m(x_m, \mathbf{u})$ is given by Equation (5.3) and the set $\overline{\mathcal{D}}(x)$ is given by:

$$\overline{\mathcal{D}}(x) = \{\mathbf{u} \mid \mathbf{u} \in \mathcal{U}, \ \mathbf{x} \in \mathcal{X}, \text{ stability constraints C'.1, C'.2 and C'.3}\}$$

with the following stabilizing constraints added to the basic formulation:

$$x(N) \in \Omega \tag{C'.1}$$
$$V_m(x_m, \mathbf{u}) \leqslant \rho_m, \ \forall m \in \mathcal{M} \tag{C'.2}$$
$$V_m(x_m, \mathbf{u}) \leqslant V_{f,m}(x_m), \ \forall x \in \Omega, \ \forall m \in \mathcal{M} \tag{C'.3}$$

The maximal satisficing cost γ_m is a design parameter that must agree with the law

$$\gamma_m^+ \geqslant \gamma_m - \ell(x_m, u_m) \tag{5.10}$$

One strategy is, for example, to design γ_m fixed, that is, $\gamma_m^+ = \gamma_m$ for all m.

The formulation includes a contractive law, in our case chosen to be, among other possibilities[2],

$$\rho_m^+ = V_m(x_m^+, \hat{\mathbf{u}}^+) \tag{5.11}$$

where the next time warm-start $\hat{\mathbf{u}}^+$ is based in the last known feasible

[2]Other contractive laws may be used. For example, we may have

$$\rho_m^+ = \rho_m - \mu_m \cdot \ell_m(x_m, \kappa_m), \ \mu_m \in (0, 1]$$

such that the evolution of $V_m(x_m, \mathbf{u})$ is bounded above by a decreasing value.

solution $\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_M)$, according to the following rule:

$$\widehat{\mathbf{u}}^+ = (\widehat{\mathbf{u}}_1^+, \ldots, \widehat{\mathbf{u}}_m^+, \ldots, \widehat{\mathbf{u}}_M^+)$$

$$\widehat{\mathbf{u}}_m^+ = (u_m(1), \ldots, u_m(N-1), \upsilon_m),$$

and υ_m satisfies Assumption 5.1.

We want to prove, under Assumption 5.1 and constraints C'.1, C'.2 and C'.3, that the control action $\kappa = u(0)$ obtained from *any* constrained satisficing profile

$$\mathbf{u} \in S_{\overline{\mathcal{D}}}(x) = \{\mathbf{u} \mid V_m(x_m, \mathbf{u}) \leqslant \gamma_m,\ \forall m \in \mathcal{M}\} \cap \overline{\mathcal{D}}(x),$$

solution of Problem (5.9), asymptotically stabilizes the corresponding system.

Problem (5.9) is a feasibility problem, without a specified objective. This problem is the basis for other schemes where an objective function is included. By proving that any feasible solution of Problem (5.9) is asymptotically stabilizing, then it is established that any satisficing scheme is stabilizing, including the satisficing MPC described in Section 4.2 or, for example, the scheme described in Appendix A.

Due to the lack of a global objective function, the global cost $\overline{V}(x, \mathbf{u})$, defined below, will be our candidate Lyapunov function:

$$\begin{aligned} \overline{V}(x, \mathbf{u}) &= \sum_{m=1}^{M} V_m(x_m, \mathbf{u}) \\ &= \sum_{k=0}^{N-1} \overline{\ell}(x(k), u(k)) + \overline{V}_f(x(N)) \end{aligned} \tag{5.12}$$

where

$$\overline{V}_f(x) = \sum_{m=1}^{M} V_{f,m}(x_m)$$

$$\overline{\ell}(x, u) = \sum_{m=1}^{M} \ell_m(x_m, u_m)$$

5.2.1 Stabilizing Conditions

Condition C'.1 establishes Ω as a terminal constraint, and it is fundamental to guarantee that the MPC will be always feasible as

shown by Mayne et al. (2000) and Limón (2004).

Condition C'.2 is used to impose convergence by associating ρ_m to a contractive law. The law (5.11) establishes that the cost in the next state must be better than the cost obtained with the warm-start, based in the last known solution.

Condition C'.3 is used to guarantee the stability of the origin. It says that when $V_{f,m}(x_m) \to 0$ then $V_m(x_m, \mathbf{u}) \to 0$. In other words, when $|x_m| \to 0$ for all m, then $|\mathbf{u}| \to 0$.

This condition is always fulfilled by any action ϑ that respects Assumption 5.1 as it can be seen following (PANNOCCHIA et al., 2011, Proposition 9), repeated here:

Proposition 5.5. *Condition C'.3 is always fulfilled by any action ϑ that respects Assumption 5.1, for all $x = (x_1, \ldots, x_M) \in \Omega$.*

Proof. Pannocchia et al. (2011): Consider any $x \in \Omega$, define $x(0) = x$ and choose any $\vartheta(0) = (\upsilon_1(0), \ldots, \upsilon_M(0))$ satisfying Assumption 5.1. We thus obtain $V_{f,m}(x_m(1)) + \ell_m(x_m(0), \upsilon_m(0)) \leqslant V_{f,m}(x_m(0))$. Because $x(1) \in \Omega$, we can choose $\vartheta(1)$ satisfying Assumption 5.1 to obtain $V_{f,m}(x_m(2)) + \ell_m(x_m(1), \upsilon_m(1)) + \ell_m(x_m(0), \upsilon_m(0)) \leqslant V_{f,m}(x_m(1)) + \ell_m(x_m(0), \upsilon_m(0)) \leqslant V_{f,m}(x_m(0))$. Continuing in this fashion for $k = 2, 3, \ldots, N-1$, and defining

$$\mathbf{v}_m = (\upsilon_m(0), \upsilon_m(1), \ldots, \upsilon_m(N-1))$$

and

$$\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_M)$$

we obtain $V_m(x_m, \mathbf{v}) \leqslant V_{f,m}(x_m)$. □

An alternative condition is to use the modified condition C".3 below:

$$V_m(x_m, \mathbf{u}) \leqslant V_m(x_m, \mathbf{v}), \ \forall x \in \Omega \qquad \text{(C".3)}$$

for all m, where $x = (x_1, \ldots, x_M) \in \Omega$ and $\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_M)$ is defined in Proposition 5.5. Together with C'.2, this condition results that the solution $\mathbf{u}$, when $x \in \Omega$, is such that

$$V_m(x_m, \mathbf{u}) \leqslant \min(V_m(x_m, \mathbf{v}), \rho_m)$$

If condition C'.3 was changed to C".3, it means that the satisficing controllers will not perform less than the performance obtained by the terminal profile $\mathbf{v}$, in Ω (LIMóN et al., 2006).

5.2.2 Stability Proof

To prove stability, it must be shown that

- the satisficing problem $\overline{P}^{\mathrm{S}}$ (5.9) is always feasible (recursive feasibility);
- that the states converge to the origin when $\mathbf{u} \in S_{\overline{\mathcal{D}}}(x)$ and $\kappa = u(0)$ is the control action applied by the SMPC (convergence);
- and that the origin is stable when $\mathbf{u} \in S_{\overline{\mathcal{D}}}(x)$ (stability).

From the warm-start it can be derived some useful results:

Lemma 5.6. *The local costs $V_m(x_m, \mathbf{u})$ are decreasing in a rate given by*

$$V_m(x_m^+, \mathbf{u}^+) - V_m(x_m, \mathbf{u}) \leqslant -\ell_m(x_m, \kappa_m) \tag{5.13}$$

where $\kappa_m = u_m(0)$ is the control action of controller $\mathcal{C}_m$.

Proof. The increment of the local costs are given by

$$\begin{aligned} \Delta V_m(x_m, \mathbf{u}) &= V_m(x_m^+, \mathbf{u}^+) - V_m(x_m, \mathbf{u}) \\ &\leqslant V_m(x_m^+, \hat{\mathbf{u}}^+) - V_m(x_m, \mathbf{u}) \\ &= -\ell_m(x_m, \kappa_m) + T \end{aligned} \tag{5.14}$$

where the inequality came from constraint C'.2 and contraction law (5.11). The term T, after some eliminations, is equal to

$$T = \ell_m(x_m(N), \upsilon_m) + V_{f,m}(x_m(N+1)) - V_{f,m}(x_m(N))$$

which is less than or equal to zero due to Assumption 5.1 and constraint C'.1. It results that

$$V_m(x_m^+, \mathbf{u}^+) - V_m(x_m, \mathbf{u}) \leqslant -\ell_m(x_m, \kappa_m)$$

□

Corollary 5.7. *The global cost $\overline{V}(x, \mathbf{u})$ is decreasing in a rate given by*

$$\overline{V}(x^+, \mathbf{u}^+) - \overline{V}(x, \mathbf{u}) \leqslant -\overline{\ell}(x, \kappa) \tag{5.15}$$

where $\kappa = (\kappa_1, \ldots, \kappa_M)$ is the control action of controllers $\mathcal{C}_m$, $m \in \mathcal{M}$.

Proof. The result comes directly from Lemma 5.6. □

Lemma 5.8. *(Recursive feasibility of the warm-start) The next time warm-start*

$$\hat{\mathbf{u}}^+ = (\hat{\mathbf{u}}_1^+, \ldots, \hat{\mathbf{u}}_M^+)$$

is feasible for Problem $\overline{P}^{\mathrm{S}}(x)$.

Proof. By construction, we have that $\hat{\mathbf{u}}^+ \in \mathcal{U}$. We also have that the resulting trajectory $\mathbf{x}^+ = (x(2), \ldots, x(N+1))$ has its first elements $(x(2), \ldots, x(N))$ feasible with $x(N) \in \Omega$, due to condition C'.1. Because Assumption 5.1 establishes that Ω is invariant under action $\upsilon = (\upsilon_1, \ldots, \upsilon_M)$, the tail of all the trajectories remains in Ω, that is, $x(N+1) \in \Omega \subseteq \mathbb{X}$. Then, we have $\hat{\mathbf{u}}^+ \in \mathcal{U}$, $\mathbf{x}^+ \in \mathcal{X}$, $x(N+1) \in \Omega$. Condition C'.3 is fulfilled because of Proposition 5.5. Condition C'.2 is fulfilled when $\mathbf{u}^+ = \hat{\mathbf{u}}^+$. Then, $\hat{\mathbf{u}}^+ \in \overline{\mathcal{D}}(x^+)$. From (5.14) and condition (5.10), the warm-start is also satisficing since $V_m(x_m^+, \hat{\mathbf{u}}^+) \leqslant V_m(x_m, \mathbf{u}) - \ell(x_m, \kappa_m) \leqslant \gamma_m^+$. Altogether, the warm-start $\hat{\mathbf{u}}^+ \in S(x^+) \cap \overline{\mathcal{D}}(x^+)$ is feasible. □

The main result follows below.

Theorem 5.9. *Any feasible solution of the satisficing problem* $\overline{P}^{\mathrm{S}}$ (5.9) *is asymptotically stabilizing.*

Proof. The problem $\overline{P}^{\mathrm{S}}$ is well defined since it is always feasible according to Lemma 5.8. It remains to prove convergence and stability.

Because $\overline{V}(x, \mathbf{u})$ is positive definite and strictly decreasing (Corollary 5.7), it results that $\overline{V}(x, \mathbf{u}) \to \overline{V}(0, \mathbf{0})$ and the convergence of x to the origin is proved.

Eventually, the trajectory of the states enters the set Ω where stability is proven in the following. Because $x'Qx \leqslant \overline{V}(x, \mathbf{u}) \leqslant \overline{V}_f(x)$ in Ω, there exist $\mathcal{K}_\infty$-functions $\sigma_1(x)$ and $\sigma_2(x)$ such that

$$\sigma_1(|x|) \leqslant \overline{V}(x, \mathbf{u}) \leqslant \overline{V}_f(x) \leqslant \sigma_2(|x|), \ \forall x \in \Omega \tag{5.16}$$

Let us take $\delta = \sigma_2^{-1}(\sigma_1(\epsilon))$. Then, for all $|x(0)| \leqslant \delta$ in Ω, we have

$$\begin{aligned} \overline{V}(x(0), \mathbf{u}) &\leqslant \overline{V}_f(x(0)) \leqslant \sigma_2(|x(0)|) \\ &\leqslant \sigma_2(\delta) = \sigma_2(\sigma_2^{-1}(\sigma_1(\epsilon))) \\ &= \sigma_1(\epsilon) \end{aligned}$$

and, from convergence and (5.16),

$$\begin{aligned}&\sigma_1(|x(k)|) \leqslant \overline{V}(x(k), \mathbf{u}) \leqslant \overline{V}(x(0), \mathbf{u}) \leqslant \sigma_1(\epsilon)\\ &\Rightarrow \sigma_1(|x(k)|) \leqslant \sigma_1(\epsilon)\\ &\Rightarrow |x(k)| \leqslant \epsilon\end{aligned}$$

and stability is proven, according to Definition 5.1.

Then, because convergent and stabilizing, any feasible solution of the satisficing problem (5.9) is asymptotically stabilizing. □

It was shown that any satisficing problem that includes the stability constraints is stabilizing.

6 EXAMPLES

The following three examples extracted from the literature on distributed control are used to test the concepts and methods developed in this thesis.

The experimental analysis aims to assess the performance of the distributed satisficing MPC by comparing it to a centralized MPC. The formulation with guarantee of stability developed in Section 5.1 is applied only in the third example, which is an unstable system. The first two examples are stable systems and no stability conditions where imposed.

6.1 EXAMPLE 1: URBAN TRAFFIC NETWORK

In this section the SMPC is applied to an urban traffic network problem, as done in (LIMA; CAMPONOGARA, 2012). The urban traffic network problem is to decide times of green to each traffic light that control the access of vehicles to a specific junction.

The simple network illustrated in Figure 10 represents an urban street, with a link e and two junctions A and B. An expression in discrete time to represent the dynamics of link e is given by (GAZIS; POTTS, 1963):

$$x_e(k+1) = x_e(k) + \Delta T[q_e(k) + d_e(k) - p_e(k) - s_e(k)], \qquad (6.1)$$

where x_e represents the number of vehicles in link e, the value q_e is the flow coming in and p_e the flow leaving link e during one period, k is a discrete time index and ΔT is the discretization time (sample time). The values d_e and s_e represent the input and output disturbances (parking), respectively.

Figure 11 shows an example of network with 8 junctions where link e is characterized by the two junctions involved, the one that discharge its queues and the other affected by this discharge. The queue in the link connecting junction i to junction m, for example, is represented by $x_{m,i}$ and controlled by a traffic light which time of green is given by $u_{m,i}$. The boundary of our system was called $\varnothing$ so $x_{m,\varnothing}$ is the queue entering junction m from outside the system.

The system in Figure 11 can be readily represented as a directed graph $\mathcal{G} = \{\mathcal{M}, \mathcal{E}\}$, where nodes in $\mathcal{M}$ are junctions and links in $\mathcal{E}$ are

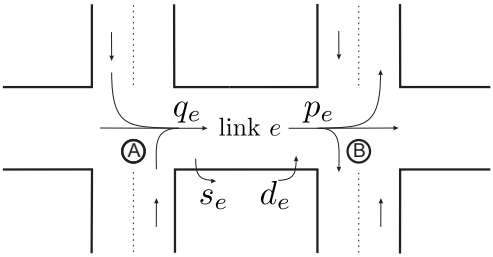

Figure 10 – Example of an urban street.

arcs connecting nodes as in Figure 12. The interaction between nodes can be generalized assuming a generic node m and a set of input nodes $I(m) = \{i_1, ..., i_{I_m}\}$ and output nodes $O(m) = \{j_1, ..., j_{O_m}\}$ [1] as in Figure 13. For example, node 3 has as input nodes the set $I(3) = \{1, 4\}$ and as output nodes the set $O(3) = \{2\}$. We classify the input nodes in internal and external nodes. Internal nodes are the ones under control, whereas external nodes are not controlled and represent the arrival of vehicles willing to get into the network. For example, $I(7) = I_{\mathrm{I}}(7) \cup I_{\mathrm{E}}(7)$ where $I_{\mathrm{I}}(7) = \{6\}$ is internal and $I_{\mathrm{E}}(7) = \{\varnothing 1\}$ is external.

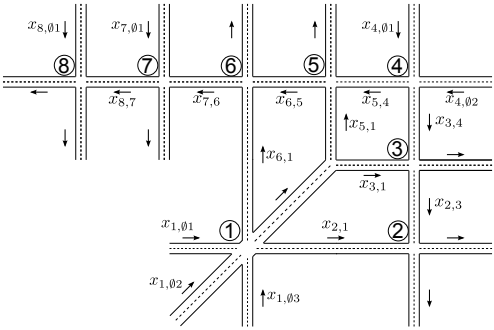

Figure 11 – Example of a traffic network. The queue $x_{i,j}$ is the queue from junction j to junction i.

The mathematical model chosen to describe the dynamics of the vehicle queues is based in Equation (6.1) and known as *store-and-forward* (GAZIS; POTTS, 1963). There, the evolution of queues depends on the initial queues, on physical characteristics of the network and on

[1] The cardinality of each set $I(m)$ and $O(m)$ depends on m, but this fact is left implicit to simplify notation.

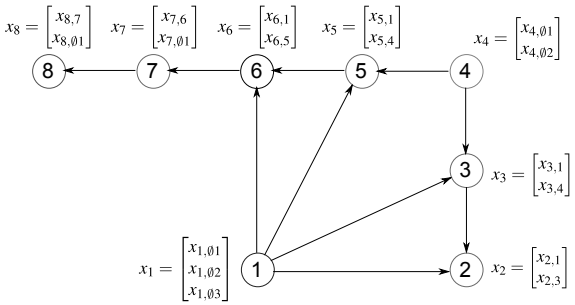

Figure 12 – Graph for the traffic network's example.

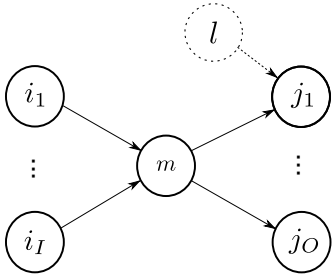

Figure 13 – Illustration of input and output nodes of a node m.

the time of green of each traffic lights. Not considering the disturbances for simplicity, the dynamic model that represents any junction m is given by:

$$\begin{cases} x_m(t+1) = A_m x_m(k) + B_{m,m} u_m(k) + \sum_{i \in I(m)} B_{m,i} u_i(k) \\ y_m(k) = x_m(k) \end{cases} \tag{6.2}$$

where vector $x_m(k) = (x_{m,i_1}(k), ..., x_{m,i_{I_m}}(k))$ has the queues of junction m influenced by the green time signals $u_i(k) = (u_{i,i_1}(k), .., u_{i,i_{I_i}}(k))$, and vector $u_m(k) = (u_{m,i_1}(k), .., u_{m,i_{I_m}}(k))$ consists of the green times of the traffic lights that compose junction m at instant k.

In this model, matrix A_m is the identity, matrix $B_{m,m}$ expresses the discharge of queues x_m as a function of green times u_m, and matrices $B_{m,i}$, $i \in I(m)$, represent how queues x_m build up as queues x_i are emptied by u_i green times. Matrices $B_{m,i}$, $i \in I(m) \cup \{m\}$, are functions of the physical characteristics of the traffic network. For example,

consider node 3 for which matrices are:

$$B_{3,3} = \Delta T_3 \begin{bmatrix} -\frac{S_{3,1}}{C_3} & 0 \\ 0 & -\frac{S_{3,4}}{C_3} \end{bmatrix}$$

$$B_{3,1} = \Delta T_3 \begin{bmatrix} \rho_{3,1,\varnothing 1} \cdot \frac{S_{1,\varnothing 1}}{C_1} & \rho_{3,1,\varnothing 2} \cdot \frac{S_{1,\varnothing 2}}{C_1} & \rho_{3,1,\varnothing 3} \cdot \frac{S_{1,\varnothing 3}}{C_1} \\ 0 & 0 & 0 \end{bmatrix}$$

$$B_{3,4} = \Delta T_3 \begin{bmatrix} 0 & 0 \\ \rho_{3,4,\varnothing 1} \cdot \frac{S_{4,\varnothing 1}}{C_4} & \rho_{3,4,\varnothing 2} \cdot \frac{S_{4,\varnothing 2}}{C_4} \end{bmatrix}$$

where ΔT_3 is the sample time (in seconds), $S_{i,j}$ is the saturation flow of link $x_{i,j}$ (in vehicles per second), $\rho_{m,i,j}$ is the rate at which vehicles from link $x_{i,j}$ enter node m, and C_i (in seconds) is the cycle time of junction i as explained below. Notice that the entries in $B_{3,3}$ are negative, indicating queue discharge as a function of green time signals u_3. The network physical parameters, saturation S in vehicles per minute and conversion rate ρ in percentage, are presented in Table 1. In Table 1 it is also informed the capacity $\text{cap}_{i,j}$ of the link $x_{i,j}$ that is the maximum number of vehicles that the link can support. The capacity is assessed based on the dimensions of the link.

The concept of cycle time is illustrated in Figure 14. Each cycle is composed by stages meaning a particular traffic lights configuration. In the example of Figure 14, after stage 3, stage 1 repeats starting another cycle. From one stage to another there is a lost time added to avoid interference between stages. The sum of all green times plus lost times in a junction gives the cycle time for that junction.

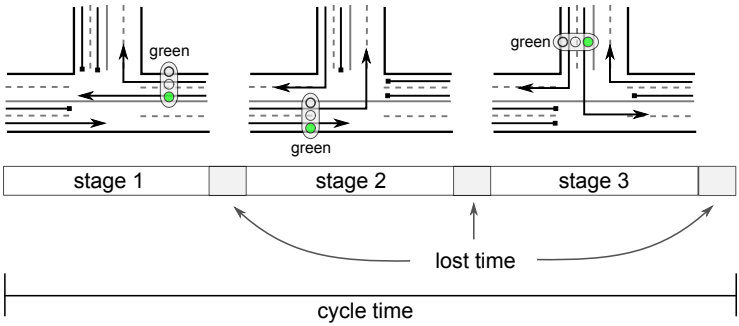

Figure 14 – Illustration of the cycle time.

Three constraints are imposed to the junctions:

Table 1 – Physical parameters

Link$_{i,j}$	cap$_{i,j}$ (veh)	$S_{i,j}$ (veh/min)	**conversion rate** $\rho_{m,i,j}$ (%)								
			$m =$ $\varnothing$	1	2	3	4	5	6	7	8
$x_{1,\varnothing 1}$	18	5			25	7		8	60		
$x_{1,\varnothing 2}$	120	30			70	7		8	15		
$x_{1,\varnothing 3}$	60	15			25	20		20	35		
$x_{2,1}$	120	20									
$x_{2,3}$	150	25									
$x_{3,1}$	30	5	60		40						
$x_{3,4}$	150	25	20		80						
$x_{4,\varnothing 1}$	180	30				70		30			
$x_{4,\varnothing 2}$	90	15				70		30			
$x_{5,1}$	30	5	40						60		
$x_{5,4}$	42	7	10						90		
$x_{6,1}$	54	9	10							90	
$x_{6,5}$	42	7	10							90	
$x_{7,6}$	42	7	10								90
$x_{7,\varnothing 1}$	60	10	50								50
$x_{8,7}$	42	7	100								
$x_{8,\varnothing 1}$	30	5	100								

Constraint 1 The sum of the green times $u_{m,i}$ and lost time $l_{m,i}$ must add up to the cycle time C_m of the junction m,

$$\sum_{i \in I(m)} (u_{m,i} + l_{m,i}) = C_m, \quad \forall m \in \mathcal{M}$$

or, in a compact form

$$\mathbf{1}' \cdot u_m \leqslant c_m$$

with $\mathbf{1}$ being the vector of ones of adequate size and c_m a constant that incorporates the lost times.

Constraint 2 Green times can not be negative,

$$u_m \geqslant 0, \quad \forall m \in \mathcal{M}$$

Constraint 3 States are always nonnegative,

$$x_m \geqslant 0, \quad \forall m \in \mathcal{M}$$

This constraint imposes couplings between inputs, but it does not add more arcs to the graph of the network.

For the urban traffic application, the controllers' cost will be defined as

$$V_m(x_m, \mathbf{u}) = \sum_{k=0}^{N-1} x_m(k+1)' Q_m x_m(k+1) + \alpha_m \cdot u_m(k)' R_m u_m(k),$$

where

$$R_m = \mathbf{0}$$

meaning that green times should not be penalized, and

$$Q_m = \text{diag}(1/\text{cap}_{m,i} : i \in I(m))$$

is diagonal matrix with its elements equal to the inverse of the capacity of each link that approaches junction m, as proposed in (DIAKAKI et al., 2002).

The satisficing controllers solve the constrained analytic center of the set of problems $\{P_m^{\text{SMPC}}\}_{m \in \mathcal{M}}$ presented in Equation (4.7), and a centralized controller solves $P^{\text{C}} : \min_{\mathbf{u}} \sum_{m=1}^{M} w_m V_m(x_m, \mathbf{u})$ with $w_m = 1$ fixed for all the controllers, while respecting the constraints.

6.1.1 Maximal Satisficing Costs

The maximal satisficing cost is define as

$$\gamma_m = N(x_m^{\text{s}'} Q_m x_m^{\text{s}}) \tag{6.3}$$

for all m, where *the satisficing queues*

$$x_m^{\text{s}} = (x_{m,i}^{\text{s}},\ \forall i \in I(m))$$

is a vector with the maximal but still satisfactory queues of junction m, specified by the user.

The value of the satisficing queues used to calculate γ_m will limit the average number of vehicles in node m as shown in Figure 15.

The idea is to maintain the average number of vehicles below the maximal satisficing queues.

Remark 6.1. *Observe that the maximal satisficing cost γ_m is defined based on physical values: prediction horizon N, capacity* $\text{cap}_{m,i}$ *of the*

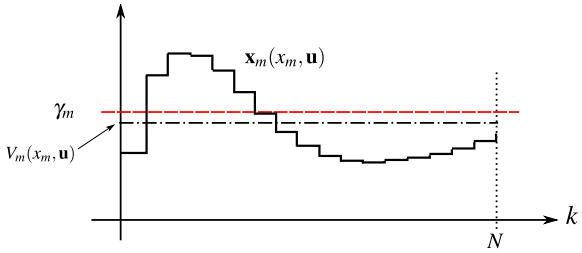

Figure 15 – Because $V_m(x_m, \mathbf{u}) \leqslant \gamma_m$, the satisficing queues used to calculate γ_m will limit the average number of vehicles $\mathbf{x}_m$ in node m.

links and maximal satisficing queues x_m^{s}.

6.1.2 Experimental Setup

There is one controller to each junction responsible for signaling that junction. The controllers objective is to adjust the time of green of the traffic lights that compose their junctions in order to maintain their queues and the queues of the affected controllers below their satisficing queues. The controllers were divided in three groups of non-neighboring controllers, $K_1 = \{1, 8\}$, $K_2 = \{2, 4, 6\}$ and $K_3 = \{3, 5, 7\}$, represented in Figure 16.

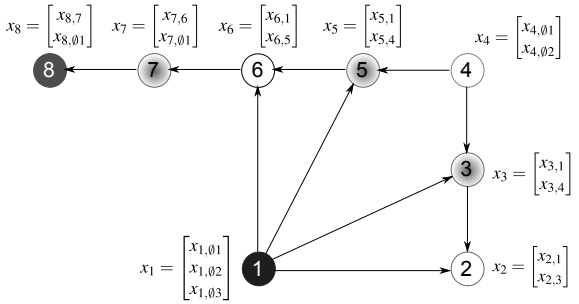

Figure 16 – Three groups of non-neighboring controllers, $K_1 = \{1, 8\}$, $K_2 = \{2, 4, 6\}$ and $K_3 = \{3, 5, 7\}$.

There are four parameters that must be defined by the user, with the advantage that all of them have a physical meaning. They are:

P.1) The cycle time C_m. The cycle time was chosen equal to 2 minutes

(120 seconds).

P.2) The prediction horizon N. The prediction horizon is equal to 5 sample times, for all controllers.

P.3) The lost times (not considered in this simulation).

P.4) The satisficing queues x_m^{s}: an average queue which is acceptable in each link of junction m. In this simulation, the maximal satisficing queues were defined as two times the maximal discharge obtained by the nominal green times

$$u_1^{\mathrm{nom}} = (40, 40, 40),\ u_{2..8}^{\mathrm{nom}} = (60, 60)$$

that is,

$$x_m^{\mathrm{s}} = -2B_{m,m} \cdot u_m^{\mathrm{nom}}$$

for all $m \in \mathcal{M}$. The result is in Table 2.

The remaining parameters were set according to the following rules:

R.1) $\Delta T_m = C_m$: the sample time ΔT_m was made equal to the cycle time for all controllers.

R.2) Matrix $R_m = 0$: in our simulation traffic signaling does not incur any cost. Green times should not be penalized.

R.3) Matrix $Q_m = \mathrm{diag}(1/\mathrm{cap}_{m,i} : i \in I(m))$: the matrix Q_m was set diagonal with its elements equal to the inverse of the capacity of each link that approaches junction m, as proposed in (DIAKAKI et al., 2002).

6.1.3 Negotiation Rule

With the original definition of γ_m (see Equation (6.3)), an excessive number of vehicles coming from outside can make impossible for the system to maintain all the queues below the maximal satisficing queues.

To avoid unfeasibility, a policy was designed to maintain the excess of vehicles outside the system. This policy, known as a "gating effect" (DIAKAKI et al., 2002), was implemented by modifying the original definition of γ_m by the rule R.4 below.

R.4) $\gamma_m = N(x_m^{s0'} Q_m x_m^{s0})$, where

$$x_m^{s0} = (x_{m,i}^{s0}, \forall i \in I(m))$$
$$x_{m,i}^{s0} = \max(x_{m,i}^{s}, x_{m,i}), \quad \forall i \in I_E(m)$$
$$x_{m,i}^{s0} = x_{m,i}^{s}, \quad \forall i \in I_I(m)$$

and $I(m) = I_I(m) \cup I_E(m)$ is the internal and external input nodes of node m.

This rule makes nodes $1, 4, 7$ and 8 to tolerate external queues greater than the satisficing, tolerating at least their actual number of vehicles, whatever it is. Observe that because nodes $2, 3, 5$ and 6 have only internal input nodes, their maximal satisficing queues are not modified by rule R.4.

6.1.4 Experimental Analysis

The analysis of the satisficing controllers was made against a centralized one in which weights were set equal to $w_m = \frac{1}{8}$ for all $m \in \mathcal{M}$. Notice that the tuning of the centralized controller is based on an ad hoc definition of weights.

This simulation considers the initial and satisficing queues given in Table 2, as well as a constant arrival of vehicles in node 1 and 4 in vehicles per cycle respectively.

Table 2 – Initial and satisfactory average queues

Controller	**Queues**		
	arrivals	initial, $x_m(0)$	satisficing, x_m^s
$\mathcal{C}_1$:	(5, 15, 10)	(10, 50, 20)	(6, 40, 20)
$\mathcal{C}_2$:		(60, 20)	(40, 50)
$\mathcal{C}_3$:		(5, 35)	(10, 50)
$\mathcal{C}_4$:	(15, 15)	(120, 30)	(60, 30)
$\mathcal{C}_5$:		(10, 20)	(10, 14)
$\mathcal{C}_6$:		(9, 20)	(18, 14)
$\mathcal{C}_7$:	(0,0)	(15, 20)	(14, 20)
$\mathcal{C}_8$:	(0,0)	(17, 9)	(14, 10)

Figures 17 to 24 show the evolution in 10 cycles (20 minutes) of the vehicle queues, the calculated green times, and the cost of the satis-

ficing and centralized controllers, respectively. The maximal satisficing costs, in dash-dot line, are defined only for the satisficing controllers but are repeated in the centralized case for comparison. The bars show the vector components stacked from below. For example, in Figure 17, the components of the vector $x_1 = (x_{1,\varnothing 1}, x_{1,\varnothing 2}, x_{1,\varnothing 3})$ are in black, gray and white respectively.

We can see in Figures 17 and 20 that the satisficing controllers $\mathcal{C}_1$ and $\mathcal{C}_4$ use less time of green than the centralized controller uses and, consequently, they accumulate more queues. This is because controllers $\mathcal{C}_1$ and $\mathcal{C}_4$ have a compromise with the satisfaction of the internal controllers. The negotiation policy leads controllers $\mathcal{C}_1$ and $\mathcal{C}_4$ to adjust their required levels of satisfaction (dash-dot line) to permit higher costs (solid lines) and to accommodate more queues. Remember that the negotiation policy maintains the specification of the internal nodes and degrades only the nodes receiving vehicles from outside the system. This compromise does not emerge easily in the centralized control due to its ad hoc nature.

In Figures 21 and 22 we see that the discharge made by the centralized controller overcharges node 5 and node 6 making then unable to maintain the sum of the corresponding queues below the maximal satisfactory (horizontal line) even with the maximal of green (120 seconds). The costs of nodes 5 and 6 are very above the maximal level specified for controllers $\mathcal{C}_5$ and $\mathcal{C}_6$ (horizontal dash-dot line) indicating their dissatisfaction in the centralized case. On the other hand, the satisficing controllers are able to maintain their performances near the specified.

Figure 25 shows the trajectories of each equivalent weights $w_m(x_m)$ of the SMPC, variable with time. In this figure, the weights are normalized so that $\sum_{m=1}^{M} w_m(x_m) = 1$. Figure 25 shows that controllers $\mathcal{C}_5$ and $\mathcal{C}_6$ gain more importance, while $\mathcal{C}_1$ and $\mathcal{C}_4$ have a lower importance.

6.1.5 Simulation Under Model Error

The satisficing controllers also seem to present a better behavior in the presence of model error. Figures 26 and 27 show the behavior of junction 5 and 6 when the rates of flow coming from junction 4 are greater than what is expected by the nominal model. In this case, instead of 30%, the flow from junction 4 to junction 5 is 70% of the junction total flow. It can be seen that the satisficing controllers maintain the queues in a satisfactory level while the queues build up

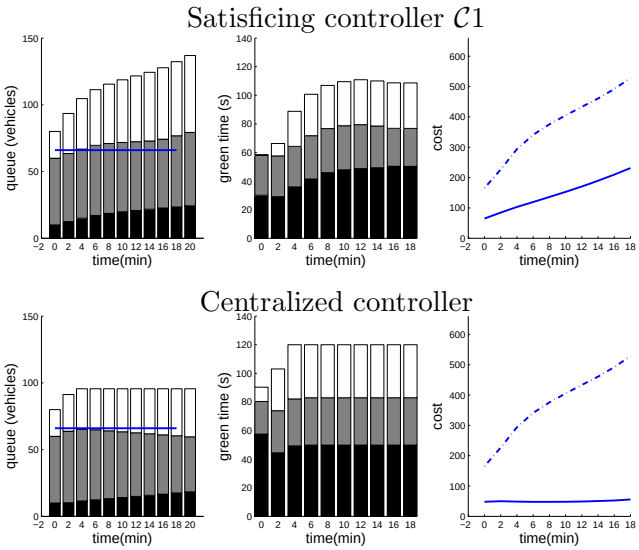

Figure 17 – Queues, control signals and costs in junction 1.

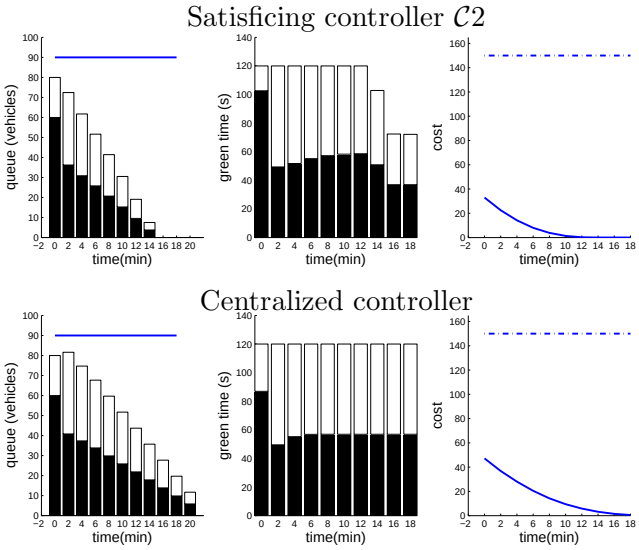

Figure 18 – Queues, control signals and costs in junction 2.

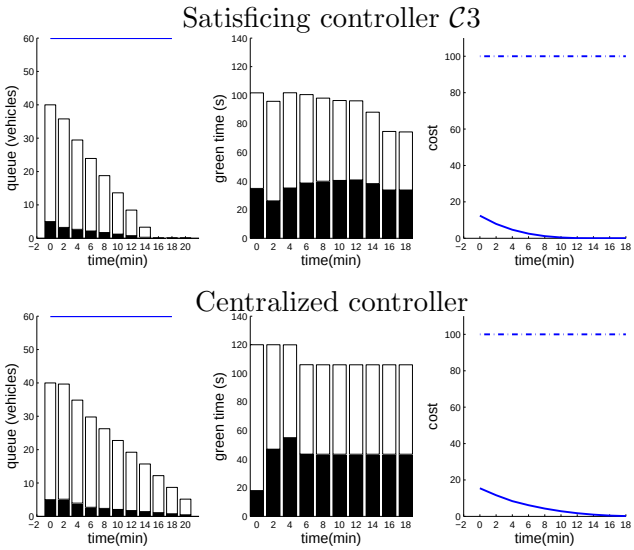

Figure 19 – Queues, control signals and costs in junction 3.

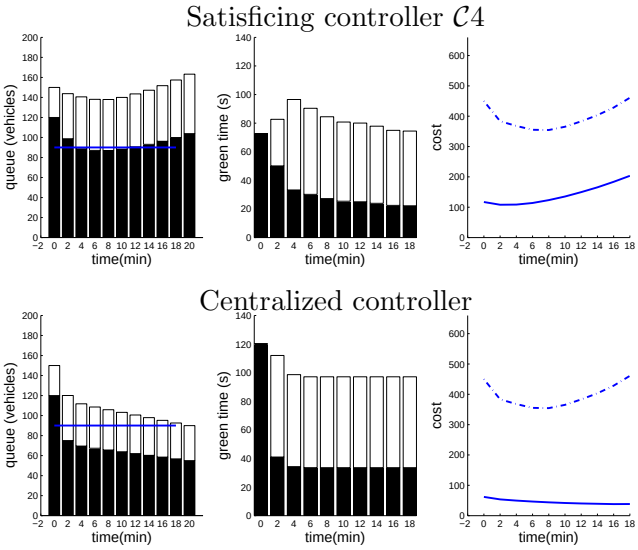

Figure 20 – Queues, control signals and costs in junction 4.

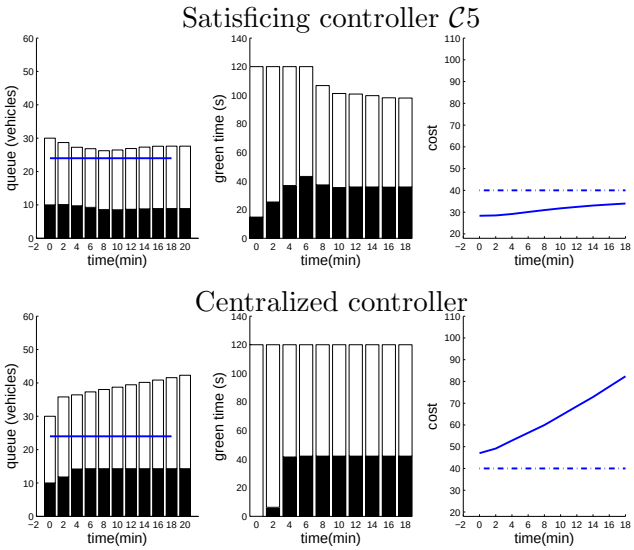

Figure 21 – Queues, control signals and costs in junction 5.

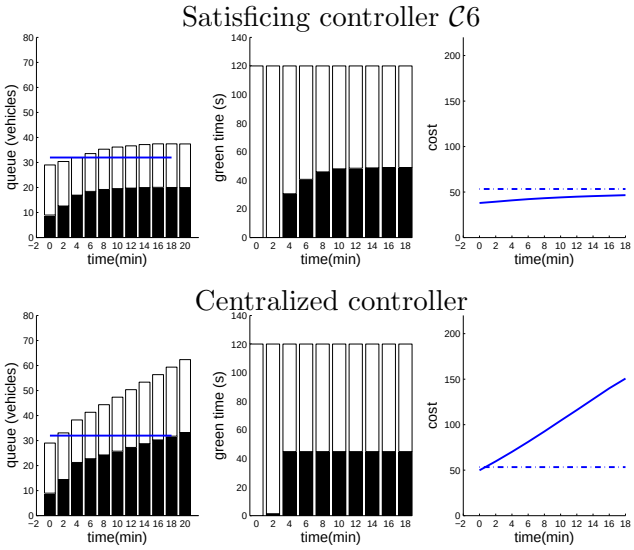

Figure 22 – Queues, control signals and costs in junction 6.

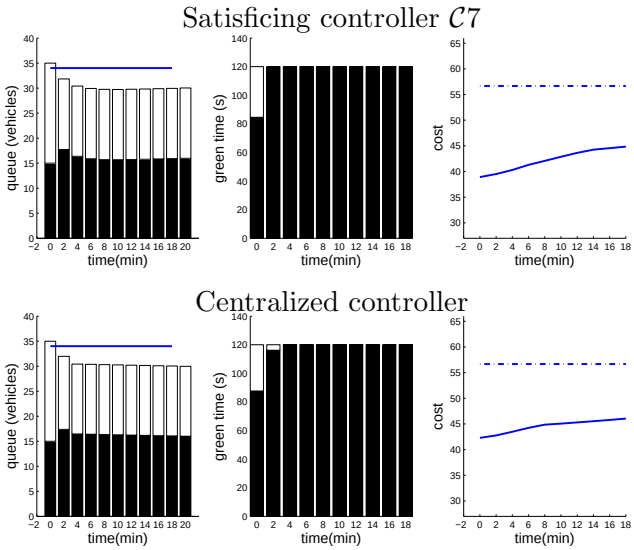

Figure 23 – Queues, control signals and costs in junction 7.

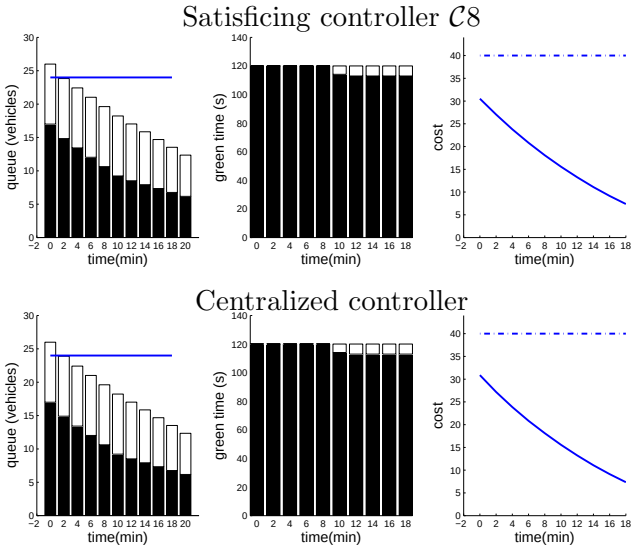

Figure 24 – Queues, control signals and costs in junction 8.

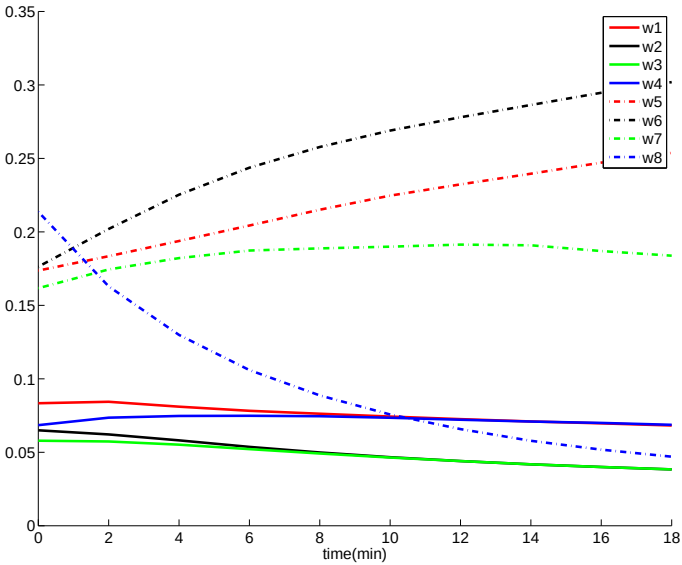

Figure 25 – Normalized, equivalent weights.

Table 3 – Mean time to solution

	time in seconds		
	phase I	phase II	total
serial satisficing $\{P_m^{\text{SMPC}}\}_{m\in\mathcal{M}}$	32	32	64
serial/paralell satisficing $\{P_m^{\text{SMPC}}\}_{m\in\mathcal{M}}$	12	12	24
centralized satisficing P^{SMPC}			8
centralized optimization P^{C}			0.8

even more under centralized control.

6.1.6 Numerical Analysis

The problems were simulated in `Matlab` and solved using `CVX` (GRANT; BOYD, 2011), a package for specifying and solving convex programs.

Table 3 shows the mean time to produce a solution at each cycle, considering the 10 cycles of the simulation. The mean time for convergence of the distributed satisficing controllers if iterating in series is shown in the first row. The second row considers the possibility of parallelism within groups[2]. The third row is the average time for solving the satisficing MPC by a centralized algorithm. The fourth row brings the mean time for solving the classical centralized optimization problem.

It can be seen that distributively solving the satisfacing problem takes more time to converge than in the centralized case, although some parallelism can greatly improve the time to convergence. Possibly, for large and weakly interconnected problems, more controllers can iterate in parallel in few groups and approach the time for convergence of the centralized controller. Nevertheless, the time spent by the satisficing controllers to iteratively produce an optimal solution was, in the worst case, less than the cycle time. The time to solution can be further reduced if we are only interested in a good enough solution. In such case, phase II is not necessary.

[2]Calculated multiplying the number of groups that in our case is three by the average time for convergence of a controller, that is the time for convergence of the group itself.

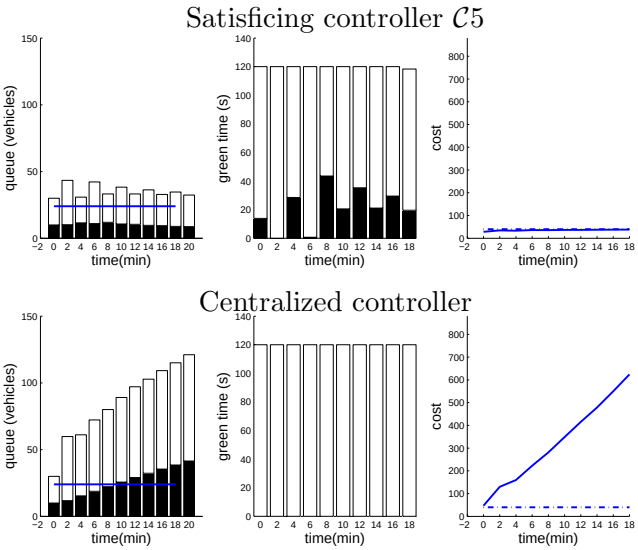

Figure 26 – Junction 5 under model error.

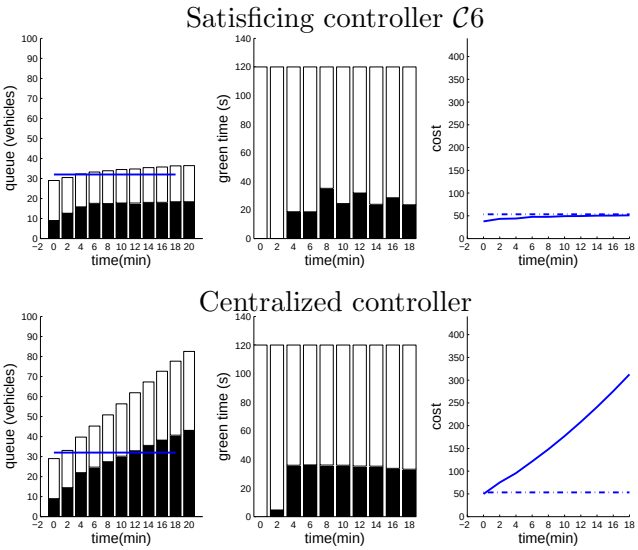

Figure 27 – Junction 6 under model error.

6.2 EXAMPLE 2: TWO REACTORS AND ONE SEPARATOR PROCESS

The example represented in Figure 28, extracted from (STEWART et al., 2010; VENKAT et al., 2006) and (LIU et al., 2009), consists of two continuously stirred tank reactors (CSTRs) and a flash tank separator.

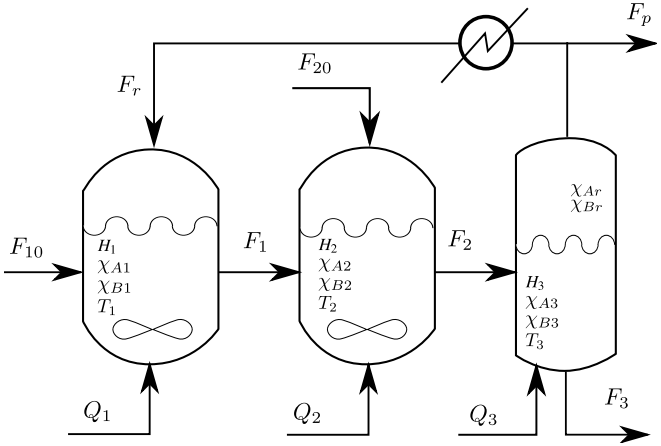

Figure 28 – Two reactors and one separator process.

Reactors 1 and 2 are fed with fresh reactant A by streams F_{10} and F_{20}, respectively. The reactant A is then converted to the desired product B and to an undesired side-product C. The effluent of reactor 2 feeds the separator tank. In the separator, the bottom product is removed and the overhead vapor is condensed and recycled to Reactor 1. Part of the overhead vapor is purged before being recycled.

The manipulated variables are the amount of heat $\mathcal{Q}_1$, $\mathcal{Q}_2$ and $\mathcal{Q}_3$ supplied to tanks 1, 2 and 3, respectively, and the feed and flesh flow F_{10} and F_{20}, as well as the recycle F_r. Flows F_1, F_2 and F_3 are consequence of the levels of liquid in the respective tank. The controlled variables, on the other hand, are concentrations χ_{A1}, χ_{B1}, χ_{A2}, χ_{B2} and χ_{A3}, χ_{B3}, as well as temperatures T_1, T_2 and T_3, and tank levels H_1, H_2 and H_3.

These variables follow the dynamic nonlinear equations (6.4).

$$
\begin{aligned}
f_1 &: \frac{H_1}{dt} = \frac{1}{\rho A_1}(F_{10} + F_r - F_1) \\
f_2 &: \frac{d\chi_{A1}}{dt} = \frac{1}{\rho A_1 H_1}(F_{10}\chi_{A0} + F_r\chi_{Ar} - F_1\chi_{A1}) - k_{A1}\chi_{A1} \\
f_3 &: \frac{d\chi_{B1}}{dt} = \frac{1}{\rho A_1 H_1}(F_r\chi_{Br} - F_1\chi_{B1}) - k_{A1}\chi_{A1} + k_{B1}\chi_{B1} \\
f_4 &: \frac{dT_1}{dt} = \frac{1}{\rho A_1 H_1}(F_{10}T_0 + F_rT_r - F_1T_1) + \\
&\quad - \frac{1}{C_p}(k_{A1}\chi_{A1}\Delta H_A + k_{B1}\chi_{B1}\Delta H_B) + \frac{\mathcal{Q}_1}{\rho A_1 C_p H_1} \\
f_5 &: \frac{H_2}{dt} = \frac{1}{\rho A_2}(F_{20} + F_1 - F_2) \\
f_6 &: \frac{d\chi_{A2}}{dt} = \frac{1}{\rho A_2 H_2}(F_{20}\chi_{A0} + F_1\chi_{A1} - F_2\chi_{A2}) - k_{A2}\chi_{A2} \\
f_7 &: \frac{d\chi_{B2}}{dt} = \frac{1}{\rho A_2 H_2}(F_1\chi_{B1} - F_2\chi_{B2}) - k_{A2}\chi_{A2} + k_{B2}\chi_{B2} \\
f_8 &: \frac{dT_2}{dt} = \frac{1}{\rho A_2 H_2}(F_{20}T_0 + F_1T_1 - F_2T_2) + \\
&\quad - \frac{1}{C_p}(k_{A2}\chi_{A2}\Delta H_A + k_{B2}\chi_{B2}\Delta H_B) + \frac{\mathcal{Q}_2}{\rho A_2 C_p H_2} \\
f_9 &: \frac{H_3}{dt} = \frac{1}{\rho A_3}(F_2 - F_p - F_r - F_3) \\
f_{10} &: \frac{d\chi_{A3}}{dt} = \frac{1}{\rho A_3 H_3}(F_2\chi_{A2} - (F_p + F_r)\chi_{Ar} - F_3\chi_{A3}) \\
f_{11} &: \frac{d\chi_{B3}}{dt} = \frac{1}{\rho A_3 H_3}(F_2\chi_{B2} - (F_p + F_r)\chi_{Br} - F_3\chi_{B3}) \\
f_{12} &: \frac{dT_3}{dt} = \frac{1}{\rho A_3 H_3}(F_2T_2 - (F_p + F_r)T_r - F_3T_3) + \frac{\mathcal{Q}_3}{\rho A_3 C_p H_3}
\end{aligned}
\tag{6.4}
$$

where, for all i=1,. . . ,3,

$$F_i = k_{fi}H_i \tag{6.5}$$

$$k_{Ai} = k_A \exp(-\frac{E_A}{RT_i}) \tag{6.6}$$

$$k_{Bi} = k_B \exp(-\frac{E_B}{RT_i}) \tag{6.7}$$

and the purge flow is

$$F_p = 0.01F_r \tag{6.8}$$

The concentrations of the side-product C and recycle concentrations are given by equations (6.9) and (6.10), respectively.

$$\begin{aligned} \chi_{C1} &= 1 - \chi_{A1} - \chi_{B1} \\ \chi_{C2} &= 1 - \chi_{A2} - \chi_{B2} \\ \chi_{C3} &= 1 - \chi_{A3} - \chi_{B3} \end{aligned} \tag{6.9}$$

$$\begin{aligned} \chi_{Ar} &= \frac{\alpha_A \chi_{A3}}{\alpha_A \chi_{A3} + \alpha_B \chi_{B3} + \alpha_C \chi_{C3}} \\ \chi_{Br} &= \frac{\alpha_B \chi_{B3}}{\alpha_A \chi_{A3} + \alpha_B \chi_{B3} + \alpha_C \chi_{C3}} \\ \chi_{Cr} &= \frac{\alpha_C \chi_{C3}}{\alpha_A \chi_{A3} + \alpha_B \chi_{B3} + \alpha_C \chi_{C3}} \end{aligned} \tag{6.10}$$

The remaining parameters of this problem are presented in Table 4, and the constraints in Table 5.

A_1, A_2	area of tanks 1 and 2	3.0	m^2
A_3	area of tank 2	1	m^2
E_A	activation energy of reaction A	-831.4	$kJ/kmol$
E_B	activation energy of reaction B	-1247.1	$kJ/kmol$
k_A	constant for reaction A→B	0.02	$1/s$
k_A	constant for reaction B→C	0.018	$1/s$
ΔH_1	Heat of reaction A→B	-40	kJ/kg
ΔH_2	Heat of reaction B→C	-50	kJ/kg
α_A	relative volatility of A	3.5	
α_B	relative volatility of B	1.1	
α_C	relative volatility of C	0.5	
C_p	heat capacity	25	$kJ/kg \cdot K$
R	gas constant	8.314	$kJ/kmol \cdot K$
ρ	solution density	0.15	kg/m^3
T_0	feed flow temperature	313	K
k_{fi}	flow coefficient, $i = 1, 2, 3$	2.5	kg/m s
χ_{A0}	concentration of A in the feed flow	1	

Table 4 – Parameters of the problem.

Observe that system (6.4) is coupled by the states, and not only

variable	lower bound	upper bound
F_{10}	0	10
$\mathcal{Q}_1$	0	50
F_{20}	0	10
$\mathcal{Q}_2$	0	50
F_r	0	75
$\mathcal{Q}_3$	0	50

Table 5 – System constraints.

by the inputs. The system must be linearized and transformed into an equivalent system coupled *only by the inputs*. This process is described below.

1. The system is linearized by calculating the Jacobian of the dynamic equations with respect to states

$$z = (H_1, \chi_{A1}, \chi_{B1}, T_1, H_2, \chi_{A2}, \chi_{B2}, T_2, H_3, \chi_{A3}, \chi_{B3}, T_3)$$

and inputs

$$v = (F_{10}, \mathcal{Q}_1, F_{20}, \mathcal{Q}_2, F_r, \mathcal{Q}_3)$$

around the steady-state operation point (z^*, v^*) given by Table 6. It results in matrices

$$A^{\mathrm{c}} = J_z(F(z,v))|_{(z^*,v^*)}$$

$$B^{\mathrm{c}} = J_v(F(z,v))|_{(z^*,v^*)}$$

where $F(z,v) = (f_1, \ldots, f_{12})$ is the vector of equations (6.4) and $J_z()|_{(z^*,v^*)}$ and $J_v()|_{(z^*,v^*)}$ are the Jacobians with respect to z and v, respectively, valued in (z^*, v^*) (command `jacobian` in Matlab).

In its discrete form, the linearized system is

$$\begin{aligned} \Delta z^+ &= A^{\mathrm{d}} \Delta z + B^{\mathrm{d}} u \\ y &= C^{\mathrm{d}} \Delta z \end{aligned} \tag{6.11}$$

with

$$\begin{aligned}\Delta z = (&H_1 - H_1^*,\ \chi_{A1} - \chi_{A1}^*,\ \chi_{B1} - \chi_{B1}^*,\ T_1 - T_1^*,\\ &H_2 - H_2^*,\ \chi_{A2} - \chi_{A2}^*,\ \chi_{B2} - \chi_{B2}^*,\ T_2 - T_2^*,\\ &H_3 - H_3^*,\ \chi_{A3} - \chi_{A3}^*,\ \chi_{B3} - \chi_{B3}^*,\ T_3 - T_3^*)\end{aligned}$$

and

$$u = (\ F_{10} - F_{10}^*,\ \mathcal{Q}_1 - \mathcal{Q}_1^*,\ F_{20} - F_{20}^*,\ \mathcal{Q}_2 - \mathcal{Q}_2^*,\ F_r - F_r^*,\ \mathcal{Q}_3 - \mathcal{Q}_3^*\ )$$

with $C^{\mathrm{d}} = I$ being the identity, so that $y = \Delta z$. Matrices A^{d}, B^{d} and C^{d} where obtained using the Matlab command `c2d` with sample time $\Delta T = 0.1s$.

$H_1^* = 29.8$	$H_2^* = 30$	$H_3^* = 3.27$	m
$\chi_{A1}^* = 0.542$	$\chi_{A2}^* = 0.503$	$\chi_{A3}^* = 0.238$	
$\chi_{B1}^* = 0.393$	$\chi_{B2}^* = 0.421$	$\chi_{B3}^* = 0.570$	
$T_1^* = 315$	$T_2^* = 315$	$T_3^* = 315$	K
$\mathcal{Q}_1^* = 10$	$\mathcal{Q}_2^* = 10$	$\mathcal{Q}_3^* = 10$	kJ/s
$F_{10}^* = 8.33$	$F_{20}^* = 0.5$	$F_r^* = 66.2$	kg/s

Table 6 – Steady-state operation point

2. To separate system (6.11) into subsystems, let us allocate one controller to each tank so that there will be $M = 3$ controllers and $\mathcal{M} = \{1, 2, 3\}$ subsystems for which the controlled variables are

$$\begin{aligned}y_m &= (y_{m,1},\ y_{m,2},\ y_{m,3},\ y_{m,4})\\ &= (H_m - H_m^*,\ \chi_{Am} - \chi_{Am}^*,\ \chi_{Bm} - \chi_{Bm}^*,\ T_m - T_m^*)\end{aligned}$$

for $m = 1, 2, 3$, and which manipulated variables are

$$\begin{aligned}u_1 &= (u_{1,1},\ u_{1,2})\\ &= (F_{10} - F_{10}^*,\ \mathcal{Q}_1 - \mathcal{Q}_1^*)\\ u_2 &= (u_{2,1},\ u_{2,2})\\ &= (F_{20} - F_{20}^*,\ \mathcal{Q}_2 - \mathcal{Q}_2^*)\\ u_3 &= (u_{3,1},\ u_{3,2})\\ &= (F_r - F_r^*,\ \mathcal{Q}_3 - \mathcal{Q}_3^*)\end{aligned}$$

To model the subsystems in the form

$$x_m^+ = A_m x_m + \sum_{i=1}^{3} B_{m,i} u_i$$
$$y_m = C_m x_m$$

for $m = 1, \ldots, 3$, suitable to the theory presented in this thesis, one must obtain all sub-models $S_{m,i}$ from the triple $(A^{\mathrm{d}}, B_i^{\mathrm{d}}, C_m^{\mathrm{d}})$ and associate then as in Figure 4.

The resulting subsystems are then transformed to their minimal form eliminating uncontrolled and unobserved modes (command `minreal` in Matlab).

This example shows how dynamic equations coupled by states can be transformed into dynamic equations coupled only by the inputs. The corresponding graph is fully coupled and shown in Figure 29. The states obtained by this procedure lose their physical meaning.

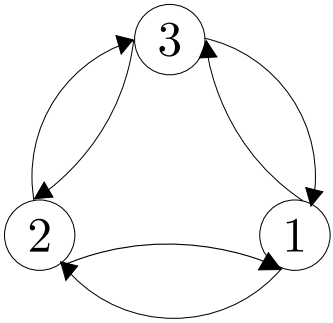

Figure 29 – Graph for the two reactors and one separator process. The system is fully coupled through the inputs.

The controllers' cost was defined as

$$V_m(x_m, \mathbf{u}) = \sum_{k=0}^{N-1} x_m(k+1)' Q_m x_m(k+1) + u_m(k)' R_m u_m(k),$$

where $N = 5$,

$$R_m = 0.01 \cdot I$$

and

$$Q_m = C_m' Q_{ym} C_m + 0.001 \cdot I$$

with

$$Q_{ym} = \text{diag}(1, 0, 0, 0.1), \text{ for } m = 1, 2$$

$$Q_{ym} = \text{diag}(1, 0, 10^3, 0), \text{ for } m = 3$$

The satisficing controllers solve the constrained analytic center of the set of problems $\{P_m^{\text{SMPC}}\}_{m\in\mathcal{M}}$ presented in Equation (4.7), and a centralized controller solves $P^{\text{C}} : \min_{\mathbf{u}} \sum_{m=1}^{M} w_m V_m(x_m, \mathbf{u})$ with $w_m = \frac{1}{3}$ fixed for all the controllers, while respecting the constraints.

6.2.1 Maximal Satisficing Costs

The maximal satisficing cost γ_m was adjusted considering that a control system is satisfactory, for example, if, after a disturbance, it allows a deviation from the steady-state of 20% in the reactors and 5% in the separator, repeated over the prediction horizon. The idea is to allow the reactors to absorb more variations and maintain the separator more steady to guarantee the quality of the final product. Then

$$\gamma_m = N \cdot {y_m^{\text{s}}}' Q_{ym} y_m^{\text{s}}$$

where the maximal satisficing deviation y_m^{s} is

$$y_1^{\text{s}} = 0.20 \cdot z_1^*, \qquad y_2^{\text{s}} = 0.20 \cdot z_2^*, \qquad y_3^{\text{s}} = 0.05 \cdot z_3^*$$

It results, approximately, in the following satisficing costs:

$$\gamma_1 = 2200$$

$$\gamma_2 = 2200$$

$$\gamma_3 = 5$$

6.2.2 Simulation Results

It was simulated an initial condition where a disturbance caused the level of reactor 2 to be 40% greater than the steady-state. The SMPC is compared with an optimal MPC were the the three costs are made equally important with $w_1 = w_2 = w_3 = \frac{1}{3}$. In the SMPC, on the other hand, the equivalent weights $w_m(x_m)$ of the resulting closed loop system are variable with time, as can be seen in Figure 30. In this figure, the weights are normalized so that $\sum_{m=1}^{M} w_m(x_m) = 1$.

Observe that, although the maximal satisficing costs γ_m are fixed, the equivalent weights $w_m(x_m)$ are variable and converge to $\frac{1}{3}$ when the controllers' costs approach zero. Observe also that the most severe limit of the satisficing cost of controller $\mathcal{C}_3$ results in the greatest equivalent weight.

The results are shown in Figures 31 to 33 and summarized in Table 7 that shows the response's cost of each controller in the simulation time interval $T_{\text{sim}} = 100$, for a sample time of $0.1s$, given by:

$$J_m = \sum_{t=0}^{T_{\text{sim}}-1} x_m(t+1)'Q_m x_m(t+1) + u_m(t)'R_m u_m(t)$$

for $m = 1, \ldots, 3$, where t is the discretized time of simulation.

Notice that, the satisficing controller $\mathcal{C}_2$ is the only one that has a wost performance if compared with the optimal MPC, mainly because of a higher deviation of temperature shown in Figure 32. As expected, the SMPC controller $\mathcal{C}_3$ performs much better due to the higher importance devoted to it. The surprise is the good performance of the SMPC controller $\mathcal{C}_1$ that has probably being benefited by the higher stability in the separator.

Table 7 – Costs of the MPC (classical) and SMPC

Controllers	J_1	J_2	J_3	Total
MPC	85	616	51	752
SMPC	31	713	12	756
SMPC %	-64 %	+16 %	-76 %	+0.5 %

Table 7 also shows that the satisficing controllers have reduced the cost of $\mathcal{C}_1$ in 64 % and $\mathcal{C}_3$ in 76 % with a increasing in the cost of $\mathcal{C}_2$ in only 16 %.

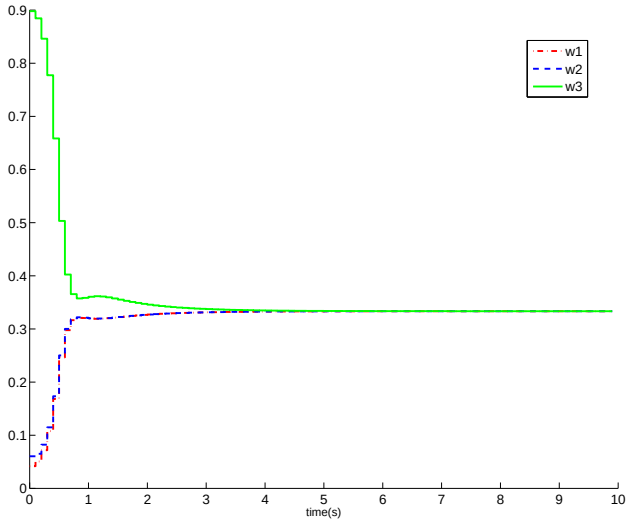

Figure 30 – Normalized equivalent weights of the SMPCs.

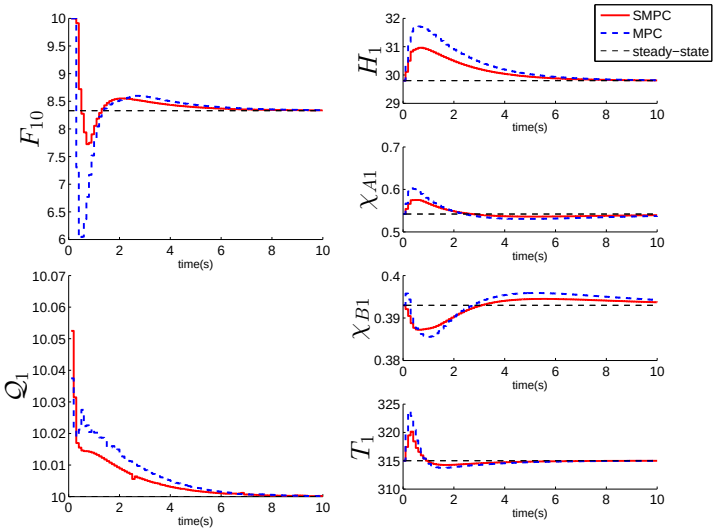

Figure 31 – Variables of subsystem 1

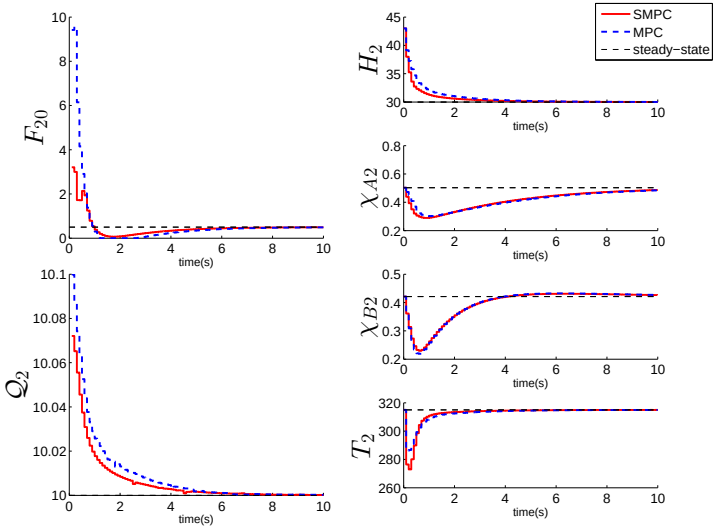

Figure 32 – Variables of subsystem 2

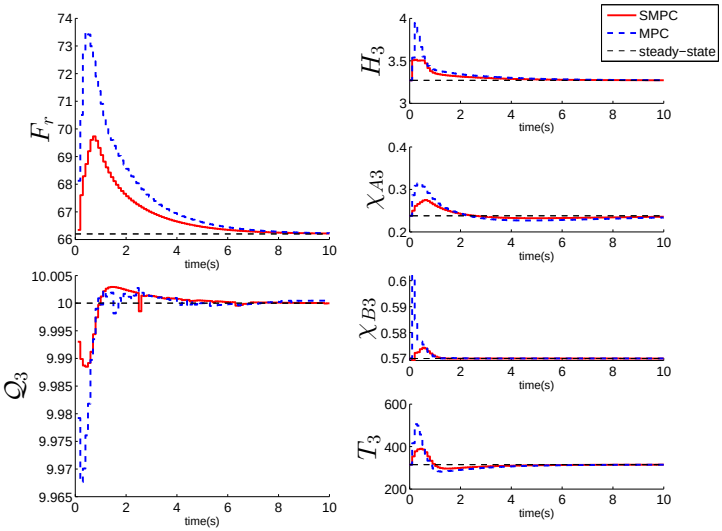

Figure 33 – Variables of subsystem 3

6.3 EXAMPLE 3: THREE UNSTABLE SUBSYSTEMS

The following system, described in (VENKAT et al., 2006), consists of three unstable subsystems $S_1 = (u_1, y_1)$, $S_2 = (u_2, y_2)$ and $S_3 = (u_3, y_3)$, represented by transfer functions, such that

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} G_{1,1} & 0 & G_{1,3} \\ G_{2,1} & G_{2,2} & 0 \\ 0 & G_{3,2} & G_{3,3} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix},$$

which nominal models are

$$G_{1,1} = \begin{bmatrix} \frac{s-0.75}{(s+10)(s-0.01)} & \frac{0.5}{(s+11)(s+2.5)} \\ \frac{0.32}{(s+6.5)(s+5.85)} & \frac{1}{(s+3.75)(s+4.5)} \end{bmatrix}$$

$$G_{1,3} = \begin{bmatrix} \frac{s-5.5}{(s+2.5)(s+3.2)} \\ \frac{0.3}{(s+11)(s+27)} \end{bmatrix}$$

$$G_{2,1} = \begin{bmatrix} \frac{s-0.3}{(s+6.9)(s+3.1)} & \frac{0.31}{(s+41)(s+34)} \\ \frac{-0.19}{(s+16)(s+5)} & \frac{0.67(s-1)}{(s+12)(s+7)} \end{bmatrix}$$

$$G_{2,2} = \begin{bmatrix} \frac{s-0.5}{(s+20)(s+25)} & \frac{0.6}{(s+14)(s+15)} \\ \frac{-0.33}{(s+3.0)(s+3.1)} & \frac{s-1.5}{(s+20.2)(s-0.05)} \end{bmatrix}$$

$$G_{3,2} = \begin{bmatrix} \frac{0.9}{(s+17)(s+10.8)} & \frac{-0.45}{(s+26)(s+5.75)} \end{bmatrix}$$

$$G_{3,3} = \begin{bmatrix} \frac{s-3}{(s+12)(s-0.01)} \end{bmatrix}$$

Observe that each subsystem has one unstable pole. The corresponding inputs and outputs are

$$u_1 = (u_{1,1}, u_{1,2}),\ u_2 = (u_{2,1}, u_{2,2}),\ u_3 = (u_{3,1}),$$

$$y_1 = (y_{1,1}, y_{1,2}),\ y_2 = (y_{2,1}, y_{2,2}),\ y_3 = (y_{3,1})$$

and the inputs are subject to the uncoupled constraints

$$\|u_{1,1}\| \leqslant 1,\ \|u_{1,2}\| \leqslant 0.15,$$

$$\|u_{2,1}\| \leqslant 1.5,\ \|u_{2,2}\| \leqslant 0.2$$

$$\|u_{3,1}\| \leqslant 0.75$$

The subsystems must be transformed in their discrete minimal state representation

$$x_m^+ = A_m x_m + \sum_{i=1}^{3} B_{m,i} u_i$$
$$y_m = C_m x_m$$

for $m = 1, \ldots, 3$, where matrices A_m, $B_{m,i}$ and C_m where obtained with sample time $\Delta T = 1s$. This is done transforming each transfer function $G_{m,i}$ in its discrete state space form (commands `ss` and `c2d` in Matlab) and combining them as in Figure 4. The resulting subsystems are then transformed to their minimal form eliminating uncontrolled and unobserved modes (command `minreal` in Matlab).

Observe from the graph shown in Figure 34 that the subsystems are not fully connected. Matrices $B_{1,2}, B_{2,3}$ and $B_{3,1}$ are zero and there are no further couplings induced by constraints.

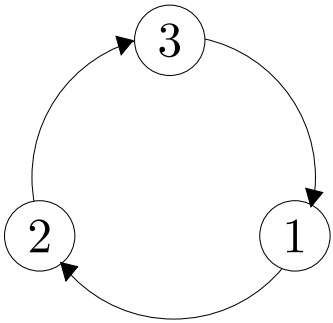

Figure 34 – Graph for the three unstable subsystems. Subsystem 2 is influenced by subsystem 1, and so on.

6.3.1 Controllers Setup

There is one controller to each subsystem m which manipulates u_m in order to control y_m. The controllers' cost was defined as

$$V_m(x_m, \mathbf{u}) = \sum_{k=0}^{N-1} x_m(k+1)' Q_m x_m(k+1) + u_m(k)' R_m u_m(k),$$

where $N = 25$,

$$R_m = I$$

and

$$Q_m = C_m' Q_{ym} C_m + 10^{-6} \cdot I$$

with

$$Q_{ym} = 25 \cdot I, \text{ for } m = 1, 2$$

$$Q_{ym} = 1, \text{ for } m = 3$$

and I is an identity matrix of proper dimension.

Assumption 5.1 is fulfilled by employing a terminal cost $V_{f,m} = 0$ and a terminal state constraint $\Omega = \{0\}$, that forces the states to the origin at the end of the prediction horizon and is invariant under control $\vartheta = 0$. Condition C.2 is that of Equation (5.4).

The maximal satisficing costs were set equal to

$$\gamma_1 = \gamma_2 = \gamma_3 = 10$$

6.3.2 Simulation Results

The system was simulated starting from the initially perturbed outputs $y_{1,1} = -1$ and $y_{3,1} = 1$. The results are compared with a optimal MPC were the the three costs are made equally important with $w_1 = w_2 = w_3 = \frac{1}{3}$.

Figure 35 shows that, in the SMPC, the equivalent weights change with time. Initially, controller $\mathcal{C}_3$ has the highest importance, followed by $\mathcal{C}_1$ and $\mathcal{C}_2$, but they soon converge to the same importance $w_m(x_m) = \frac{1}{3}$ when the controllers' costs approach zero. If we see the evolution of the local costs in Figure 37, the small importance of controller $\mathcal{C}_2$ is justified by its low cost compared to its maximal satisficing.

The result is summarized in Table 8 that shows the response's cost of each controller in the simulation time interval $T_{\text{sim}} = 40$, for a sample time of $1s$, given by:

$$J_m = \sum_{t=0}^{T_{\text{sim}}-1} x_m(t+1)' Q_m x_m(t+1) + u_m(t)' R_m u_m(t)$$

for $m = 1, \ldots, 3$, where t is the discretized time of simulation. We can see that both controller $\mathcal{C}_1$ and controller $\mathcal{C}_2$ contributed to make controller $\mathcal{C}_3$ more satisfied. To understand Table 8, we must consider

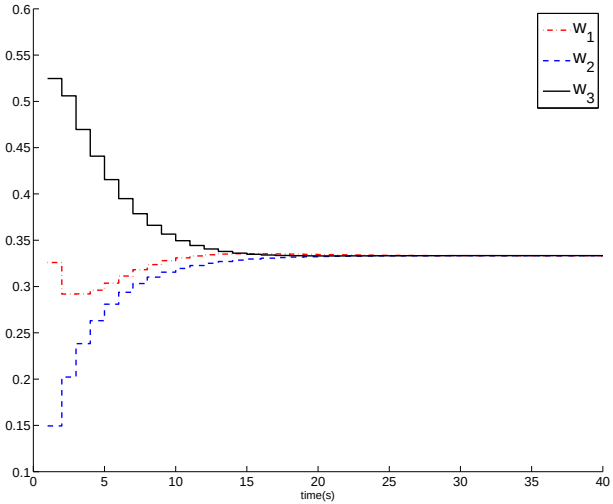

Figure 35 – Normalized equivalent weights of the SMPC.

the structure of the network (Figure 34), the interaction gains (make $s = 0$ in the transfer functions)

$$Gain(G_{2,1}) = \begin{bmatrix} -14 & 0.2 \\ 2 & -8 \end{bmatrix} \times 10^{-3},$$
$$Gain(G_{3,2}) = \begin{bmatrix} 5 & -3 \end{bmatrix} \times 10^{-3},$$
$$Gain(G_{1,3}) = \begin{bmatrix} -70 \\ 1 \end{bmatrix} \times 10^{-3}$$

as well as the set of resultant altruisms given in Figure 36. Observe that $\mathcal{C}_2$ is highly altruistic to $\mathcal{C}_3$ and, allied to the relatively small gain between them, results that controller $\mathcal{C}_2$ almost does not load controller $\mathcal{C}_3$. On the other hand, $\mathcal{C}_3$ does load controller $\mathcal{C}_1$ because of the low altruism and the high gain between them. It results that controller $\mathcal{C}_3$ has a better result. Considering controller $\mathcal{C}_1$, it receive a high load from controller $\mathcal{C}_3$ but its effect on controller $\mathcal{C}_2$ is limited because of the not so high gain between them.

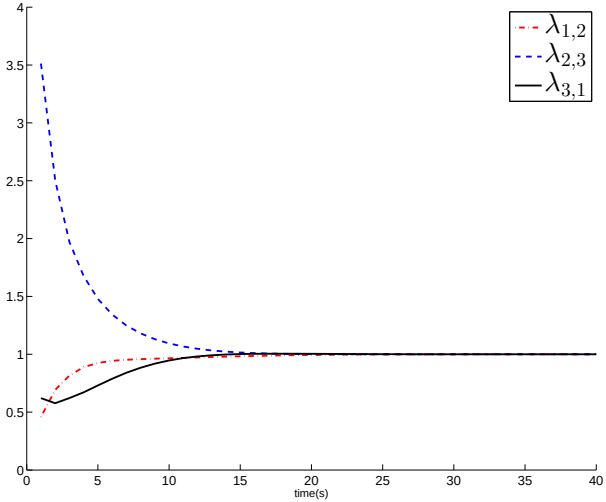

Figure 36 – Situational altruism.

Table 8 – Costs of the MPC (classical) and SMPC

Controllers	J_1	J_2	J_3	Total
MPC	3.76	0.0083	8.39	12.16
SMPC	5.17	0.0098	7.29	12.47
SMPC %	+37.5 %	+17.5 %	-13.1 %	+2.6 %

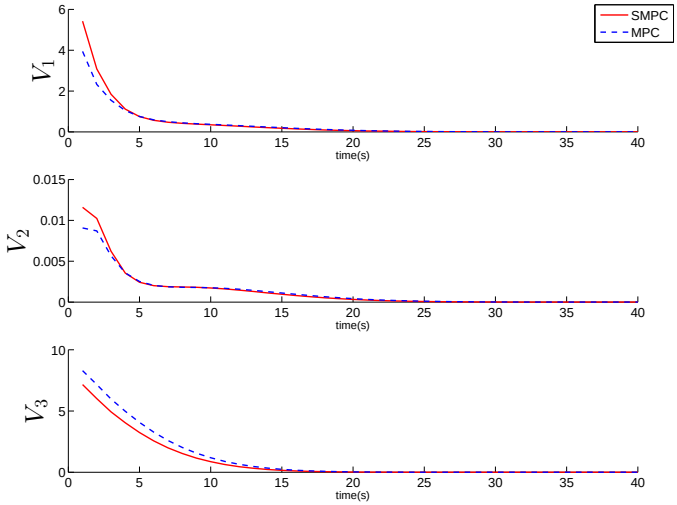

Figure 37 – Evolution of the predicted local costs $V_m(x, \mathbf{u})$.

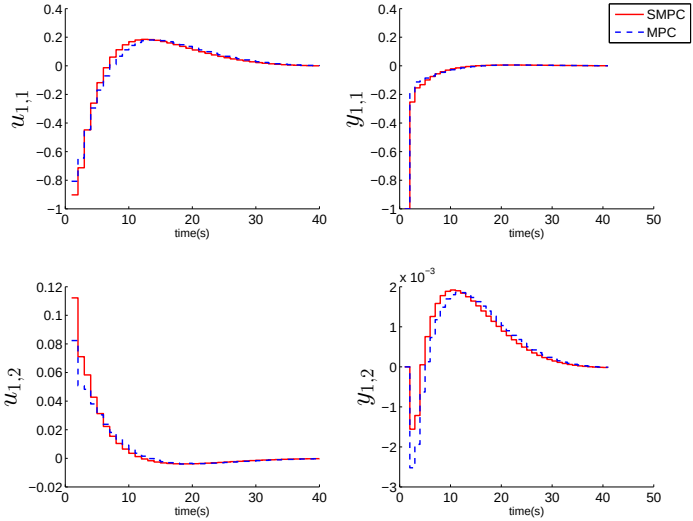

Figure 38 – Variables of subsystem 1

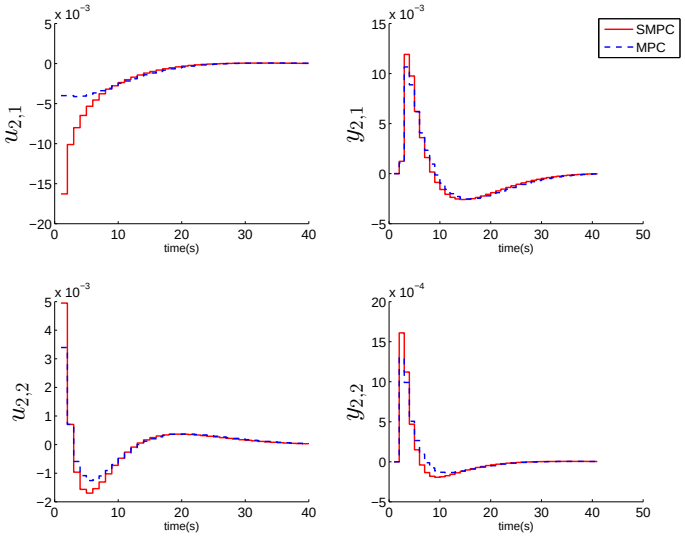

Figure 39 – Variables of subsystem 2

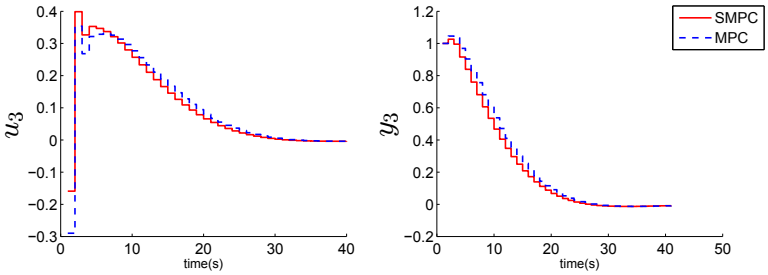

Figure 40 – Variables of subsystem 3

7 CONCLUDING REMARKS

The definition of maximal satisficing costs allows the satisficing controllers to coordinate themselves to a solution that belongs to the so called satisficing set. In the satisficing set, any solution is satisfactory and sufficient (satisficing = satisfy + suffice) for all controllers. In particular, the solution obtained by the SMPC is the constrained analytic center of the satisficing set that, besides satisficing, is also Pareto optimal.

To obtain a Pareto-optimal solution, the classical cooperative MPC implements a *categorical* altruism imposed by a fixed global cost shared by all the controllers. Instead, this thesis proposes a *situational* altruism where a global cost, not imposed nor fixed, emerges from the local costs and from the specification of the maximal satisficing costs γ_m. The value of γ_m constitutes a *local* requirement, differently of the weights w_m that are meaningful only in respect to others. In some cases, γ_m may be associated to physical parameters as is the case in the example of Section 6.1, or it may be associated to a minimal local performance as in Section 6.2. The solution of the SMPC, besides Pareto-optimal, gives more importance to the controllers with a worst performance at the moment. Situational altruism permits a more balanced division of resources, avoiding the exploitation of one controller by the others. These characteristics of adaptiveness and equilibrium allied to a negotiation mechanism to deal with infeasibility render the distributed satisficing MPC a good alternative to the classical MPC.

The SMPC is also stabilizing when endowed with stabilizing constraints, as can be seen in the example of Section 6.3.

The presented distributed interior-point algorithm avoids fixed points and reaches an optimal solution even under coupled constraints. Nevertheless, a more elaborate mechanism of negotiation should be developed and the algorithm should be optimized in order to be faster.

It is also worth to notice that, observing the results in the example of Section 6.1, it seems that the satisficing MPC is more robust to model mismatch. This is an important characteristic that should be studied in future works.

7.1 SMPC AND MULTI-AGENT SYSTEMS

Next, a parallel between MPC controllers and intelligent agents will be done to conclude that the theory presented here approximates control systems and multi-agent systems.

According to Jennings e Wooldridge (1998) or Wooldridge (1999), an intelligent agent is a computational system situated in an environment and capable of taking autonomous and *flexible* actions pursuing an objective. For flexibility it means reactivity, pro-activeness and social ability, which we summarize, somewhat arbitrarily, as adaptation and social ability. An adaptable agent reacts and changes appropriately to changes in the environment.

An *MPC agent* (LIMA et al., 2011) would be an agent with characteristics both from a MPC controller and from an intelligent agent. Observe in Table 9 that the more flexible is a MPC controller, the more it will approach the definition of intelligent agents. One MPC agent would be defined, then, as a MPC controller that incorporates the flexibility proper to intelligent agents.

Table 9 – Comparison between MPC controllers and MPC agents

Distributed MPC Controller	Intelligent Agent

• a computational system
• that interact with an environment
• capable of taking actions (control actions)
  – autonomous
  – model based
• to achieve an objective
	• a computational system
	• that interact with an environment
	• capable of taking actions
	  – autonomous
	  – *flexible*: result of social ability and adaptation
	• to achieve an objective

Also there is a similarity between the MPC controllers and belief-

desire-intention (BDI) agents (RAO; GEORGEFF, 1995). Note that it is possible to establish a direct relationship between the three main components of a MPC controller and the three functional blocks of a BDI agent. This relationship is:

1. Belief $\Longleftrightarrow$ Model

2. Desire $\Longleftrightarrow$ Objective function

3. Intention $\Longleftrightarrow$ Decision

Given the similarities between the MPC controllers and (intelligent) BDI agents, the evolution of the traditional MPC controllers could be towards the fusion of these concepts and techniques. This merging would result in what one could call MPC agents.

The satisficing MPC is a step towards a MPC agent. In special it is in harmony with the following sociality axioms defined by (STIRLING; FROST, 2007):

- Conditioning: "Members of a society (of controllers, in our case) may condition their preferences on the preferences of other members"

  In the SMPC, the control action of one controller is a function of its cost and also of the cost of other controllers, with costs being the expression of local preferences.

- Endogeny: "If preference orderings exist for an autonomous society, they must be determined by interactions among its members"

  Pareto optimality is a reasonable notion of group rationality. In the SMPC, a Pareto-optimal solution is not imposed from an imposed global cost but it emerges from local specifications.

- Coherence: "No members of a society must be categorically required to subjugate their own welfare to the society in all situations in order to benefit the society"

  The SMPC implements the situational altruism where more altruism will be deserved to the controller which needs most, but none of the controllers will be altruistic to the extent of being unsatisfied. If its cost approaches its maximal satisficing cost, the controller becomes less and less altruistic.

7.2 SUGGESTION FOR FURTHER RESEARCH

It follows a list of suggestions for future research:

- As the example in Section 6.1 suggests, the SMPC seems to be more robust to model mismatches. Robustness is an important issue that deserves a careful study.
- The controller must be integrated with a state observer.
- The theory should be extended to deal with setpoint changes.
- The theory should be extended to non-linear models.
- The theory should be extended to non-cooperative applications.
- The SMPC needs significant algorithm improvements, including more elaborate mechanisms of negotiation.
- The theory presented here would allow asynchronous communication as well as the controllers would work with different sample times.
- The similarities between MPC controllers and intelligent agents suggests possibilities of interaction between the communities of control and multi-agent systems. From this interaction it could arise solutions in areas like software infrastructure, interaction with the human operator, system diagnostics and fault isolation, coordination and communication in distributed applications and integration with other systems like production schedule.

APPENDIX A – Minimal Altruism

In this appendix it is shown that a minimum of altruism is necessary to maintain the controllers satisficing. It is also shown that this minimum altruism is associated with Lagrange multipliers (LIMA; CAMPONOGARA, 2011).

To this end, it is analyzed the altruism that must be added to selfish controllers to maintain all controllers satisfied. Selfish controllers are those that try to minimize their own selfish costs. It is considered that each controller solves the following problem over their own variables $\mathbf{u}_m$ given the states x and the variables of the other controllers $\mathbf{u}_{\neg m}$:

$$P_m^{\mathrm{MA}}(x, \mathbf{u}_{\neg m}): \quad \begin{cases} \min_{\mathbf{u}_m} & V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) \\ \text{subject to} & V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m}) \leqslant \gamma_i, \ \forall i \in \mathcal{M} \\ & \mathbf{u}_m \in \mathcal{D}_m(x, \mathbf{u}_{\neg m}) \end{cases} \quad \text{(A.1)}$$

where $\mathcal{D}_m$ is the problem's domain defined in (3.2) on page 39, and the influence of $\mathbf{u}_m$ on the costs was made explicit by separating it from the parameters: $V_m(x_m, \mathbf{u}) = V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m})$ and $V_i(x_i, \mathbf{u}) = V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m})$.

The solution obtained with this method is a Nash solution because the controllers minimize their own selfish costs. While at first glance it may appear that the controllers stand a better chance to obtain a higher selfish performance, there may exist a Pareto solution where all controllers perform better. Besides non Pareto-optimal, the convergence of the set of problems A.1 depends on the existence of a Nash equilibrium. Fixed point theorems can be used to derive conditions for the existence of the Nash equilibrium (see Kakutani's theorem in (AUBIN, 2003), for example).

Our intention here is not the method itself, but rather to analyze the minimum altruism resulting from the satisficing constraints $V_i(x_i, \mathbf{u}) \leqslant \gamma_i$, for all i, and associate it with Lagrange multipliers. Let us, then, focus on the problem only with satisficing constraints:

$$\widetilde{P}_m^{\mathrm{MA}}(x, \mathbf{u}_{\neg m}): \quad \begin{cases} \min_{\mathbf{u}_m} & V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) \\ \text{subject to} & V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m}) \leqslant \gamma_i, \ \forall i \in \mathcal{M} \end{cases} \quad \text{(A.2)}$$

which solution $\mathbf{u}_m^\star$ is satisficing,

$$\mathbf{u}_m^\star \in S_m(x, \mathbf{u}_{\neg m}) \triangleq \{\mathbf{u}_m \mid \mathbf{u} \in S(x)\}$$

and

$$S(x) \triangleq \{\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_M) \mid V_m(x_m, \mathbf{u}) \leqslant \gamma_m,\ \forall m \in \mathcal{M}\}$$

The solution of $\widetilde{P}_m^{\mathrm{MA}}$ produces the value

$$p_m^\star = V_m(\mathbf{u}_m^\star | x_m, \mathbf{u}_{\neg m}).$$

A.1 DUAL PROBLEM

The dual problem associated to the primal problem (A.2) above is obtained from the *Lagrangian* function defined as

$$L_m(\mathbf{u}_m, \Lambda_m) = V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) + \sum_{i=1}^{M} \lambda_{m,i}(V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m}) - \gamma_i)$$

where $\lambda_{m,i} \geqslant 0$, for all $i = 1, \ldots, M$, are the *Lagrange multipliers* associated to each satisficing constraint and $\Lambda_m = (\lambda_{m,1}, \ldots, \lambda_{m,M})$.

Observe that, because $V_i(x_i, \mathbf{u}) - \gamma_i$ is negative or zero in a satisficing solution, the Lagrangian is a lower bound to the cost $V_m(x_m, \mathbf{u})$ of controller $\mathcal{C}_m$. In special, the *dual function*

$$l_m(\Lambda_m) = \min_{\mathbf{u}_m \in S_m} L_m(\mathbf{u}_m, \Lambda_m)$$

is a lower bound to the value $p_m^\star$ of the primal problem (A.2), that is:

$$l_m(\Lambda_m) \leqslant p_m^\star$$

for any value of $\Lambda_m \geqslant 0$. The best estimate for the lower bound of $p_m^\star$ is, then:

$$l_m(\Lambda_m^\star) = \max_{\Lambda_m \geqslant 0} l_m(\Lambda_m). \tag{A.3}$$

Problem (A.3) is the dual of the primal problem (A.2) and its value is $d_m^\star$:

$$\begin{aligned} d_m^\star &= l_m(\Lambda_m^\star) \\ &= \max_{\Lambda_m \geqslant 0} \left[\min_{\mathbf{u}_m \in S_m} L_m(\mathbf{u}_m, \Lambda_m) \right]. \end{aligned}$$

Two important properties hold when *strong duality* (BOYD; VAN-

DENBERGHE, 2004) is verified:

$$p_m^\star = d_m^\star$$

and

$$\max_{\Lambda_m \geqslant 0} \left[\min_{\mathbf{u}_m \in S_m} L_m(\mathbf{u}_m, \Lambda_m) \right] = \min_{\mathbf{u}_m \in S_m} \left[\max_{\Lambda_m \geqslant 0} L_m(\mathbf{u}_m, \Lambda_m) \right]$$

so that

$$p_m^\star = \min_{\mathbf{u}_m \in S_m} \left[L_m(\mathbf{u}_m, \Lambda_m^\star) \right] \tag{A.4}$$

Assumption A.1. *Strong duality holds. The Slater Condition (BOYD; VANDENBERGHE, 2004) says that, if the primal problem* (A.2) *is convex and strictly feasible, then strong duality holds.*

From (A.4) we have that

$$\begin{aligned}
p_m^\star &= \min_{\mathbf{u}_m \in S_m} \left[L_m(\mathbf{u}_m, \Lambda_m^\star) \right] \\
&= \min_{\mathbf{u}_m \in S_m} \left[V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) + \sum_{i=1}^{M} \lambda_{m,i}^\star (V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m}) - \gamma_i) \right] \\
&= \min_{\mathbf{u}_m \in S_m} \left[V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) + \sum_{i=1}^{M} \lambda_{m,i}^\star V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m}) - \sum_{i=1}^{M} \gamma_i \lambda_{m,i}^\star \right] \\
&= \min_{\mathbf{u}_m \in S_m} \left[V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) + \sum_{i=1}^{M} \lambda_{m,i}^\star V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m}) \right]
\end{aligned}$$

because $\sum_{i=1}^{M} \gamma_i \lambda_{m,i}^\star$ is constant. Therefore, we can conclude that the primal problem (A.2) is equivalent to minimize the combination of the costs, weighted by $\lambda_{m,i}^\star$. As before, the parameter $\lambda_{m,i}^\star$ is the altruism from controller $\mathcal{C}_m$ to controller $\mathcal{C}_i$.

Next, the expression (A.4) is used to characterize the altruism $\lambda_{m,i}^\star$ as the minimal altruism necessary to maintain the controllers satisfied.

A.2 MINIMAL ALTRUISM

Notice from Equation (A.4) that

$$\begin{aligned} p_m^\star &= \min_{\mathbf{u}_m \in S_m} \left[L_m(\mathbf{u}_m, \Lambda_m^\star) \right] \\ &= \min_{\mathbf{u}_m \in S_m} \left[V_m(\mathbf{u}_m | x_m, \mathbf{u}_{\neg m}) + \sum_{i=1}^{M} \lambda_{m,i}^\star (V_i(\mathbf{u}_m | x_i, \mathbf{u}_{\neg m}) - \gamma_i) \right] \\ &\leqslant V_m(\mathbf{u}_m^\star | x_m, \mathbf{u}_{\neg m}) + \sum_{i=1}^{M} \lambda_{m,i}^\star (V_i(\mathbf{u}_m^\star | x_i, \mathbf{u}_{\neg m}) - \gamma_i) \\ &\leqslant V_m(\mathbf{u}_m^\star | x_m, \mathbf{u}_{\neg m}) = p_m^\star \end{aligned} \tag{A.5}$$

where the last inequality is from the fact that $\lambda_{m,i}^\star \geqslant 0$ and $V_i(\mathbf{u}_m^\star | x_i, \mathbf{u}_{\neg m}) - \gamma_i \leqslant 0$ so that the summation is negative or zero. In Equation (A.5), we conclude that the inequalities hold with equalities, what implies the condition known as complementary slackness (BOYD; VANDENBERGHE, 2004):

$$\sum_{i=1}^{M} \lambda_{m,i}^\star (V_i(\mathbf{u}_m^\star | x_i, \mathbf{u}_{\neg m}) - \gamma_i) = 0$$

Then,

- if $V_i(\mathbf{u}_m^\star | x_i, \mathbf{u}_{\neg m}) < \gamma_i$, controller $\mathcal{C}_i$ will be satisfied and controller $\mathcal{C}_m$ will select $\lambda_{m,i}^\star$ to be zero, that is, $\mathcal{C}_i$ will not deserve any altruism from $\mathcal{C}_m$;
- on the other hand, if $V_i(x_i, \mathbf{u}_m^\star | \mathbf{u}_{\neg m}) = \gamma_i$, controller $\mathcal{C}_i$ will be in the limit of satisfaction requiring some altruism from controller $\mathcal{C}_m$ to guarantee its satisfaction.

The controller $\mathcal{C}_m$ will be altruistic to the minimum necessary for the satisfaction of controller $\mathcal{C}_i$.

REFERENCES

AUBIN, J.-P. *Optima and Equilibria: An Introduction to Nonlinear Analysis*. 2nd. ed. [S.l.]: Springer-Verlag, 2003.

BASAR, T.; OLSDER, G. J. *Dynamic Noncooperative Game Theory*. 2nd. ed. [S.l.]: SIAM, 1999. (Classics In Applied Mathematics).

BOYD, S.; VANDENBERGHE, L. *Convex Optimization*. [S.l.]: Cambridge University Press, 2004.

CAMACHO, E. F.; BORDONS, C. *Model Predictive Control*. 2nd. ed. [S.l.]: Springer-Verlag, 2004.

CAMPONOGARA, E.; LIMA, M. L. de. Distributed optimization for MPC of linear networks with uncertain dynamics. *IEEE Transactions on Automatic Control*, v. 57, n. 3, p. 804–809, March 2012.

CAMPONOGARA, E.; OLIVEIRA, L. B. de. Distributed optimization for model predictive control of linear-dynamic networks. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, v. 26, n. 6, p. 1331–1338, 2009.

CAMPONOGARA, E.; SCHERER, H. F. Distributed optimization for model predictive control of linear dynamic networks with control-input and output constraints. *IEEE Transactions on Automation Science and Engineering*, v. 8, n. 1, p. 233–242, 2011.

CHRISTOFIDES, P. D.; SCATTOLINI, R.; PEÑA, D. M. de la; LIU, J. Distributed model predictive control: A tutorial review and future research directions. *Computers & Chemical Engineering*, v. 51, p. 21–41, 2013.

CUI, H.; JACOBSEN, E. W. Performance limitation in descentralized control. *Journal of Process Control*, v. 12, p. 485–494, 2002.

DIAKAKI, C.; PAPAGEORGIOU, M.; ABOUDOLAS, K. A multivariable regulator approach to traffic-responsive network-wide signal control. *Control Engineering Practice*, v. 10, p. 183–195, 2002.

GAMBIER, A. MPC and PID control based on multi-objective optimization. In: *American Control Conference*. [S.l.: s.n.], 2008.

GAZIS, D. C.; POTTS, R. B. The oversaturated intersection. In: *Proceedings of the Second International Symposium on Traffic Theory*. Londres, Inglaterra: [s.n.], 1963. p. 221–237.

GOODRICH, M. A.; STIRLING, W. C.; FROST, R. L. A theory of satisficing decisions and control. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, v. 28, n. 6, p. 763–779, 1998.

GRANT, M.; BOYD, S. *CVX: Matlab Software for Disciplined Convex Programming, version 1.21*. abr. 2011.

JENNINGS, N.; WOOLDRIDGE, M. Agent technology: Foundations, applications, and markets. In: _____. [S.l.]: Springer-Verlag, 1998. cap. Applications of Intelligent Agents, p. 3–28.

LIMA, M. L. de; CAMPONOGARA, E. A framework for adaptive tuning of distributed model predictive controllers by Lagrange multipliers. In: *IEEE Multi-Conference on Systems and Control*. [S.l.: s.n.], 2011. p. 1516–1523.

LIMA, M. L. de; CAMPONOGARA, E. Urban traffic network control by distributed satisficing agents. In: *Proceedings of the 7th Workshop on Agents in Traffic and Transportation*. [S.l.: s.n.], 2012.

LIMA, M. L. de; CAMPONOGARA, E.; MARRUEDO, D. L.; PEñA, D. M. de la. Distributed satisficing MPC. *IEEE Transactions on Control Systems Technology*, –. Unpublished manuscript.

LIMA, M. L. de; HUBNER, J. F.; CAMPONOGARA, E. Sobre agentes e controladores preditivos. In: *V Workshop-Escola de Sistemas de Agentes, seus Ambientes e aplicações*. Curitiba, Paraná: IEEE Computer Society, 2011. p. 195–198.

LIMóN, D. *Control predictivo de sistemas no lineales con restricciones: estabilidad y robustez*. Phd. Thesis — Depto. Ingeniería de Sistemas y Automática, Universidad de Sevilla, 2004.

LIMóN, D.; ALAMO, T.; SALAS, F.; CAMACHO, E. F. On the stability of constrained MPC without terminal constraint. *IEEE Transaction on Automatic Control*, v. 51, p. 832–836, May 2006.

LIU, J.; PEñA, D. M. de la; CHRISTOFIDES, P. D. Distributed model predictive control of nonlinear process systems. *AIChE Journal*, v. 55, p. 1171–1184, 2009.

LUC, D. T. Pareto optimality, game theory and equilibria. In: _____. [S.l.]: Springer, 2008. cap. Pareto Optimality, p. 481–514.

MAESTRE, J. M.; PEñA, D. M. de la; CAMACHO, E. F. Distributed model predictive control based on a cooperative game. *Optimal Control Applications and Methods*, v. 32, p. 153–176, 2011.

MAYNE, D. Q.; RAWLINGS, J. B.; RAO, C. V.; SCOKAERT, P. O. M. Constrained model predictive control: stability and optimality. *Automatica*, v. 36, n. 6, p. 789–814, jun. 2000.

NIKOLAOU, M. Model predictive controllers: A critical synthesis of theory and industrial needs. *Advances in Chemical Engineering*, v. 26, p. 131–204, 2001.

PANNOCCHIA, G.; RAWLINGS, J. B.; WRIGHT, S. J. Conditions under which suboptimal nonlinear MPC is inherently robust. *Systems & Control Letters*, v. 60, n. 9, p. 747–755, 2011.

PAPPALARDO, M. Pareto optimality, game theory and equilibria. In: _____. [S.l.]: Springer, 2008. cap. Multiobjetive Optimization: A Brief Overview, p. 517–528.

QIN, S. J.; BADGWELL, T. A. A survey of industrial model predictive control technology. *Control Engineering Practice*, v. 11, p. 733–764, 2003.

RAO, A. S.; GEORGEFF, M. P. BDI agents: From theory to practice. In: *Proceedings of the first international conference on multi-agent systems (ICMAS-95)*. [S.l.: s.n.], 1995. p. 312–319.

SCATTOLINI, R. Architectures for distributed and hierarchical model predictive control - a review. *Journal of Process Control*, v. 19, p. 723–731, 2009.

SEBORG, D. E. A perspective on advanced strategies for process control. *Modeling, Identification and Control*, v. 15, n. 3, p. 179–189, 1994.

SEBORG, D. E. A perspective on advanced strategies for process control (revisited). In: *5th European Control Conference ECC99*. [S.l.: s.n.], 1999.

SIMON, H. A. A behavioral model of rational choice. *The Quarterly Journal of Economics*, v. 69, n. 1, p. 99–118, 1955.

STEWART, B. T.; VENKAT, A. N.; RAWLINGS, J. B.; WRIGHT, S. J.; PANNOCCHIA, G. Cooperative distributed model predictive control. *Systems & Control Letters*, v. 59, n. 8, p. 460–469, July 2010.

STIRLING, W. C. *Satisficing Games and Decision Making: with Application to Engineering and Computer Science*. [S.l.]: Cambridge University Press, 2003.

STIRLING, W. C.; FROST, R. L. Reconciling group and individual preferences of multi-agent systems. *IEEE International Conference on Networking, Sensing and Control*, p. 477–482, 2007.

VENKAT, A. N.; RAWLINGS, J. B.; WRIGHT, S. J. Stability and optimality of distributed model predictive control. *44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05.*, p. 6680 – 6685, December 2005.

VENKAT, A. N.; RAWLINGS, J. B.; WRIGHT, S. J. *Stability and Optimality of Distributed, Linear Model Predictive Control, Part I: State feedback*. [S.l.], November 2006.

WOOLDRIDGE, M. Multiagent systems, a modern approach to distributed artificial intelligence. In: _____. [S.l.]: The MIT Press, 1999. cap. 1 - Intelligent Agents, p. 27–78.