A software platform for coalitional control: a water network case study

J. M. Maestre†

Abstract— **During the last years, the evolution of distributed control techniques has led to the development of schemes where the communication burden is adapted dynamically. When the overall system is far from the control goal, the communication burden is augmented to improve the system performance. Likewise, when the system is close to its goal and there is no need of coordination, the distributed control scheme tends to behave as a decentralized one. As a result of this policy, local controllers are divided into dynamical groups or coalitions where the communication is essential to ensure the cooperation. In this paper, the basic concepts behind coalitonal control are formalized. In addition, a class programmed in MATLAB is provided in order to ease the implementation of this kind of control schemes. Finally, different examples of the use of the class are provided with special attention to water networks, which are a classical example in the context of distributed control applications and have potential to become a benchmark for the assessment of this novel type of control schemes.**

I. INTRODUCTION

The concepts and the challenges of non centralized control schemes have been known for a long time. For example, [1], which was published in the late 70's presents a survey of decentralized control techniques. Nevertheless, the important advances in information and communication technologies (ICT) during the 90's – very particularly the advances in wireless communication technologies – renewed the interest in this kind of control schemes. In particular, during the last decade a great effort has been put into the development of distributed model predictive control (DMPC) techniques, mainly because the DMPC framework can take the most out of the ICT infrastructure available.

A basic MPC controller uses a mathematical model to predict the plant behaviour during a certain horizon. If a control goal is provided, an optimization problem can be built based on the model and the best possible sequence of control actions can be calculated. On top of that, it is possible to consider explicitly constraints on the system variables, delays, and other complex issues when solving the optimization problem. Due to this, MPC has become a widely used control technique in the industry [2].

In DMPC, the idea is to have several MPCs governing locally different parts of the plant in order to satisfy local control objectives. Hence, none of the controllers has complete system information although the local controllers or *agents* can communicate with each other in order to improve the overall performance. Applications of DMPC comprise irrigation canals [3], transportation networks [4], or supply chains [5], [6], among many others.

†J. M. Maestre is with the Department of Systems and Automation Engineering, University of Seville, Spain pepemaestre@us.es

Literally, dozens of schemes have been proposed during the last years [7], [8], [9]. In this work, we are specially interested in the way the local controllers communicate to improve the system performance. Traditionally, two big categories have been considered from the point of view of the communication. If the local controllers do not communicate at all, i.e., the mutual influence is modeled as a mere disturbance, the term decentralized is used. Properly speaking, we only speak of distributed controllers if we assume they can communicate in order to coordinate their control tasks. More recently, the evolution of the field has led to the development of control schemes in which the local controllers adopt a decentralized attitude when the coupling between the control tasks is low and a distributed approach when it is high. In other words, the communication burden is adapted to the coupling between the control tasks. As a result of this, the local controllers are separated dynamically into cooperative groups or *coalitions*. Some schemes that can be classified as coalitional controllers are: [10], where the set of active constraints is used to modify the sets of cooperative agents; [11], where the coupling structure of the plant is exploited to divide it into hierarchicaly coupled clusters; [12], where a hierarchical MPC based on time-variant structures is proposed for the control of a large-scale system; and [13], [14], [15], where only the links with a meaningful contribution to the overall system performance are enabled.

The main contribution of this work is to present, to the best of our knowledge, the first serious formalization attempt of the concepts behind coalitional control. Moreover, a class for MATLAB has been programmed to ease the implementation of coalitional control strategies. In addition, examples of use of the class are given. It is also remarkable that one of the examples provided includes function for the representation of water tanks using only MATLAB's built-in functions. In this way, users can create graphical interfaces for their water networks, which are great examples for the assessment of distributed and coalitional control strategies. Notice that, despite the meaningful development of the theory, there is still a lot of work to be done regarding the assessment of distributed control techniques. Actually, there are few works to the best of our knowledge that really make a serious comparison between DMPC techniques. For example, in [16], a four plant tank benchmark is used to test some schemes. Another work is [17], which also presents a comparison between DMPC schemes using power management as a commom benchmark. Probably, one of the reasons of this reduced amount of work in the comparison of schemes is the lack of public software

like the one presented here [1].

The rest of the paper is organized as follows. In Section 2, a formalization of the basic concepts of coalitonal control is given. Section 3 details the implementation of a Matlab class for coalitional systems and provides the reader with hints about how to use it. Section 4 presents two simulation examples that are provided together with the class. Finally, conclusions and comments about the future work are presented in Section 5.

II. Coalitional Control

In this section, the basic concepts behind coalitional control are introduced in a systematic manner. To the best of our knowledge, this is the first attempt to formalize the framework in which this type of control strategies is based.

In the first place, let us assume a system that can be partitioned into $\mathcal{N} = \{1, 2, \ldots, N\}$ of subsystems whose discrete dynamics are, in general:

$$\begin{aligned} x_i(k+1) &= f_i(x_i(k), u_i(k), d_i(k)) \\ d_i(k) &= g_i(x_{\mathcal{N}-\{i\}}(k), u_{\mathcal{N}-\{i\}}(k)), \end{aligned} \tag{1}$$

where $x_i \in \mathbb{R}^{q_i}$ and $u_i \in \mathbb{R}^{r_i}$ with $i = 1, \ldots, n$, are the states and inputs of each subsystem respectively. The variable d_i is the influence of the neighbors' states and inputs in the update of x_i. Notice that, with a slight abuse of notation, the subindex $\mathcal{N} - \{i\}$ has been used to denote all the elements of $\mathcal{N}$ but i. Notice that f_i and g_i are in general nonlinear functions.

In this context, each subsystem $i \in \mathcal{N}$ is governed by a local controller that has partial information regarding the overall system. In particular, it is assumed that the local controller i has access only to the local state x_i and decides the value of its corresponding input u_i. The set of controllers can communicate through a network whose physical topology is described by a graph $(\mathcal{N}, \mathcal{L})$, $\mathcal{L}$ is the set of links or edges $\mathcal{L} \subseteq \mathcal{L}^{\mathcal{N}} = \{\{i, j\} | \{i, j\} \subseteq \mathcal{N}, i \neq j\}$, i.e., each real communication links that connects two local controllers is a member of $\mathcal{L}$.

The key idea of coalitional control is that the network infrastructure can be managed to adapt the communication burden to the control necessities. For this reason, it is assumed that each communication link $l \in \mathcal{L}$ can be either enabled or disabled. Moreover, it is assumed that each enabled link l involves a certain stage cost $c_l(k)$. Notice that the role played by this cost is crutial in this context. Otherwise, there would be no incentive to disable links and full communication would be used at each time step. However, this may not be possible due to several reasons: the overall control problem may be too big to be solved during the time step or the use of links might be restricted for some reasons (e.g.: energy consumption restrictions in wireless networks). As result of this, there is a set of active links at time step k, which is defined as the *network mode* or topology $\Lambda \subseteq \mathcal{L}$.

[1]The software will be published as soon as possible in the following web page: www.distributedmpc.net.

Given a network topology Λ, it is important to note that the set $\mathcal{N}$ is partitioned into disjoint groups or coalition of controllers, which are, properly speaking, cooperation or communication components. Note that two local controllers are in the same communication component iff they are connected [18]. The set of communication components is denoted by $\mathcal{N}/\Lambda$; notice that $\bigcup_{C \in \mathcal{N}/\Lambda} C = \mathcal{N}$.

Inside a communication component, the local controllers cooperate to choose the value of their input variables $u_C = (u_i)_{i \in C}$ and behave collectively as a one system with dynamics given by

$$\begin{aligned} x_C(k+1) &= f_C(x_C(k), u_C(k), d_C(k)) \\ d_C(k) &= g_C(x_{\mathcal{N}-\{C\}}(k), u_{\mathcal{N}-\{C\}}(k)) \end{aligned} \tag{2}$$

where $x_C = (x_i)_{i \in C}$ and $u_C = (u_i)_{i \in C}$ are respectively the aggregate of the states and inputs of the subsystems in C. If full communication is enabled, there is only one communication component C and the overall system dynamics can be written taking $C = \mathcal{N}$, i.e.,

$$x_{\mathcal{N}}(k+1) = f_{\mathcal{N}}(x_{\mathcal{N}}(k), u_{\mathcal{N}}(k)). \tag{3}$$

Notice that there is no coupling term $d_{\mathcal{N}}$ because all the mutual interaction effects are considered inside the global model.

In general, it is possible to assume that the control objective is to minimize a cost that depends on the state, input trajectories and the communication links employed. The stage cost of each local controller is defined as follows

$$\ell_i(k) = J_i(x_i(k), u_i(k), \Lambda_i(k))$$

where $\Lambda_i(k)$ is the number of links that directly connect agent i to other agents. Analogously, the control goal of a communication component C is to minimize

$$\ell_C(k) = J_C(x_C(k), u_C(k), \Lambda_C(k)).$$

Likewise, it is possible to define the same concept at systemwide level:

$$\ell_{\mathcal{N}}(k) = J_{\mathcal{N}}(x_{\mathcal{N}}(k), u_{\mathcal{N}}(k), \Lambda_{\mathcal{N}}(k)).$$

From a centralized point of view, the overall problem is to minimize

$$\min_{x_{\mathcal{N}}(k), u_{\mathcal{N}}(k), \Lambda(k)} \sum_{k=0}^{\infty} \ell_{\mathcal{N}}(k) \tag{4}$$

subject to (3) and $\Lambda(k) \subseteq \mathcal{L}$. As it can be seen, problem (4) is a mixed integer optimization problem that belongs to the class of NP-complete problems. In particular, the continuous variables are the system variables and the boolean ones are related with the status of the links. In order to find a distributed control scheme that can deal with this problem, it is also necessary to take into account the problem structure. There are different sources of coupling besides the link that connect the controllers, such as the system dynamics, the objective function, the system variables, or the constraints, and they have a direct impact in the way the problem can be distributed.

Notice that problem (4) also suffers from the classical combinatorial explosion of distributed problems, which limit the feasibility of exhaustive search approaches. For example, if there are $|\mathcal{L}|$ different links, there are $2^{|\mathcal{L}|}$ different network topologies. It is clear that a large number of links leads to an untractable problem even if a non-finite horizon is considered. Nevertheless, this number of network topologies corresponds to the worst case; some of them may not make sense in a particular application and therefore the number of options is likely going to be reduced.

Another alternative to reduce the number of options is to pay attention to the coalitions of controllers. For example, let us assume that the maximum size a coalition can have is M. If the number of agents is N, i.e., $|\mathcal{N}| = N$, then the number of combinations is:

$$\binom{N}{M} = \frac{N!}{M!(N-M)!},$$

which again can be large for a large scale systems.

Nevertheless, and despite the challenge of problem (4), it is possible to formulate workarounds to calculate at least a suboptimal solution in the context of dynamic programming and MPC, just like it is done in [11], [14], [13], [15], [12].

In any case, the ultimate goal of coalitional control is to solve (4) in a real peer-to-peer fashion, so that each pair of local controllers connected by a link decide whether to enable or disable it. In this work, the same problem setting employed in [14], [13] is followed: linear objective function and dynamics, coupling through the system variables, quadratic cost function, and a cost due to the network considered simply as an additive term in the cost function.

III. NETv0, a MATLAB class for coalitional control

Dealing with coalitional control scheme presents important challenges such as the combinatorial explosion, which is a classic issue in the context of distributed systems, or resetting the controller properties according to the new network topology. In order to relieve the coding part, a basic class has been programmed in Matlab. In addition, the source code of the class is provided as well, so that it can be enhanced by other members of the control community. Likewise, the fact of providing a class has an additional educational value since the object oriented programming properties of matlab are too often forgot.

The class NETv0 has been developed to support the description of a system from the overall point of view. In particular, the following set of properties are included in this version of the class:

- System description properties: the class includes the matrices x_c, A_c and B_c to support the following overall state space model:

$$x_c(k+1) = A_c x_c(k) + B_c u_c. \tag{5}$$

- Controller description properties: the class includes the matrices Q_c and R_c to support the following overall stage cost:

$$J(k) = x_c^{\mathrm{T}}(k) Q_c x_c + u_c^{\mathrm{T}} R_c u_c. \tag{6}$$

  In addition, the class provided contains the matrices K and P, which are respectively an overall feedback control law and a matrix that is the basis of the Lyapunov function $V(k) = x_c(k)^{\mathrm{T}} P x_c(k)$. As it will be shown later, K and P can be calculated automatically by the class.
- Agent description properties: the class has a table of structures named $Agents$, which collects important information regarding the subsystems that result from the overall system partition. Each agent is identified by a unique number and needs information regarding the overall input and state indices that are present on the local subsystem. Likewise, the class calculates automatically other useful information such as the links connected to the agent.
- Network description properties: the class includes a table of structures named $Links$, which collects essential information regarding the network structure. In particular, each link is identified by an unique number and the source and destination agents must be provided. Likewise, the status of the link – enabled or disabled – is stored.
- Coalitional properties: the class includes a table of structures named $Coalitions$, which is generated automatically. This list contains information about what agents are connected either directly or indirectly through the communication network. This information is used, for example, to determine the structure of the feedback gain K, which must respect the communication constraints that result from the current network topology.
- Historical data properties: the class includes the matrices $Xhist$, $Uhist$, and $cost$, and stores automatically the system evolution.

Likewise, the class NETv0 provides the user with the following methods:

- Agent description methods: a function is provided to describe an agent ($defineagent$).
- Network description methods: functions are provided to describe the agents connected by a link ($connect$), and to enable ($enable$), disable ($disable$), and switch the state of a given link ($switchlink$).
- Coalitional methods: the function $calculatecoalitions$ updates the list of coalitions according with the current state of the network.
- Controller methods: the function $LMIsolve$ is provided so that the linear matrix inequality (LMI) for coalitional control proposed in [13] is solved for the current network topology. In this way, the matrices K and P are calculated while respecting the communication constraints imposed by the network. In particular, the following optimization problem is solved to find the

matrices: constraints are satisfied

$$\begin{aligned}&\max tr(W)\\&\text{s.t.}\\&\begin{bmatrix} W & WA_c^{\mathrm{T}}+Y^{\mathrm{T}}B_c^{\mathrm{T}} & WQ_c^{1/2} & Y^{\mathrm{T}}R_c^{1/2}\\ A_cW+B_cY & W & 0 & 0\\ Q_c^{1/2}W & 0 & I & 0\\ R_c^{1/2}Y & 0 & 0 & I\end{bmatrix} > 0, \end{aligned} \tag{7a}$$

$$W_{ij}=0, Y_{ij}=0 \quad \text{such that } i\in C, j\notin C \tag{7b}$$

with $P=W^{-1}$ and $K=YW^{-1}$. In [13], it is proved that the matrix K calculated according to the previous optimization problem stabilizes the closed-loop system. Likewise, the matrix P provides a bound on the cost-to-go of the system. In addition, the function $implement$ is provided so that the overall state of the system can be updated with the overall control action given.

- Representing options: the class includes methods to represent the connectivity matrix ($plotcon$) and to plot historical data ($plot$).

IV. Examples

In this section, two simple examples of the use of the class are given. The first one is an academic example and the second one consists of a set of interconnected water tanks. In both cases, the code shows how the system is introduced into an object that belongs to the class NETv0. Once the overall system is introduced, it is easy to modify the network topology and to calculate a feedback that respects the constraints imposed by the network topology. In this way, it is possible to calculate the feedback that corresponds to each network mode and apply the following two-layer control scheme from an overall perspective:

Let $D\in\mathbb{N}^+$ a number of time instants. At each time step k,

1) If k is a multiple of D, the local controllers share their state in order to calculate the network mode Λ that minimizes the sum of the cost-to-go of the overall system and the communication costs. Otherwise, each agent shares its state only to those agents connected to it.
2) Each agent uses information received to update its control action.

A. Academic Example

The first example was presented in [13] and is shown in Figure 1. It consists of four subsystems coupled by the inputs, which are represented by boxes and arabic numbers ($\mathcal{N}=\{1,2,3,4\}$), and four links, represented by arrows and roman numbers ($\mathcal{L}=\{I,II,III,IV\}$). As there are four links, there are 16 possible network topologies. The matrices that define the subsystem dynamics are the following:

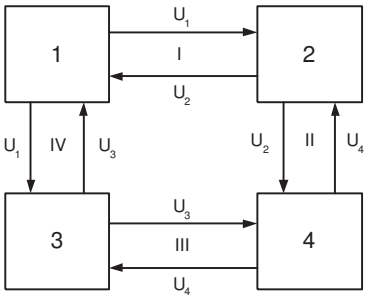

Fig. 1. Four subsystems coupled by the inputs [13].

$$\begin{aligned}A_{11}&=\begin{bmatrix}1 & 0.8\\ 0 & 0.7\end{bmatrix} A_{22}=\begin{bmatrix}1 & 0.6\\ 0 & 0.7\end{bmatrix}\\ A_{33}&=\begin{bmatrix}1 & 0.9\\ 0 & 0.8\end{bmatrix} A_{44}=\begin{bmatrix}1 & 0.8\\ 0 & 0.5\end{bmatrix}\\ A_{ij}&=\begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}\ i\neq j\\ B_{ii}&=\begin{bmatrix}0\\ 1\end{bmatrix}\quad B_{ij}=\begin{bmatrix}0\\ 0.15\end{bmatrix}\quad i\neq j\end{aligned} \tag{8}$$

where $x_i\in\mathbb{R}^2$ and $u_i\in\mathbb{R}$ are respectively the states and the input of each subsystem $i\in\mathcal{N}$. The stage costs ℓ_i of all the subsystems are defined by matrices $Q_i=\mathrm{diag}(1,1),\ R_i=1$ with $i\in\mathcal{N}$. Notice that the system model can be written according to that of the class, which is presented in (5).

As it can be seen in the code, the use of the class provided simplifies enormously the computation of the feedback matrices that are used for each network topology. Once the controllers are calculated, it is possible to tune the cost associated to the use of a given link and see how it affects the evolution of the system state variables. A deeper analysis of this example can be found in [13].

B. Water network

The second example consists of a set of interconnected water tanks. For simplicity, it has been assumed that it is possible to manipulate the flow of the pipes that connect the tanks. This allows to present very simple dynamics:

$$x_i(k+1)=x_i(k)+T_s\frac{1}{A_i}\sum_{j\in N_i}u_{ij} \tag{9}$$

where x_i is the level of the water stored in tank i and A_i is its surface, T_s is the time step length , u_{ij} is the flow through the pipe connecting tanks i and j, and N_i is the set of tanks connected to tank i. For simplicity, Q_c and R_c are unit matrices of the proper size.

Once more, the tuning of the communication costs leads to different switching in the network modes and different evolutions of the system variables.

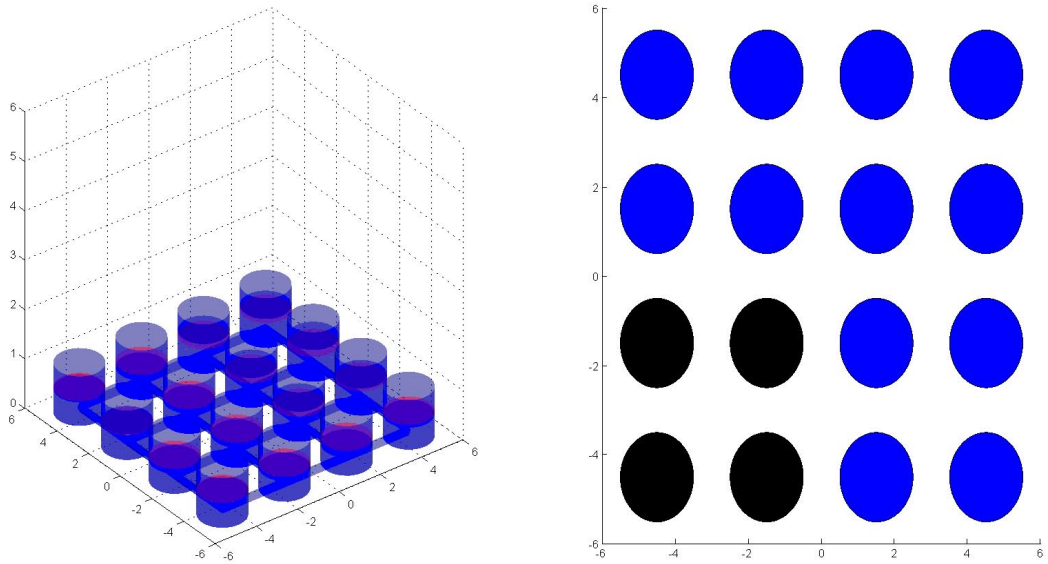

Fig. 3. Coalition of controllers.

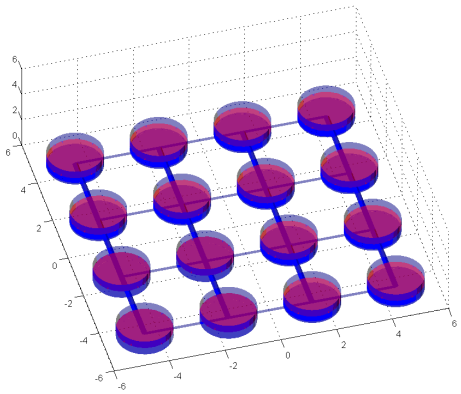

Fig. 2. Sixteen subsystems coupled water tanks.

In order to improve the educational contribution of the example, a graphical user interface has been developed in Matlab, so that the evolution of the system is represented as shown in Figure 2. In this figure, the water reference level is depicted as a red circle in the tank and the current water level is depicted as a blue one. Notice that the user can change the plotting options in order to adjust the perspective.

In Figure 3 it is possible to see another graphical outcome of this example. In particular, the association of the controllers in coalitions in shown using different colors. More specifically, blue is used for those controllers that are working in a decentralized fashion and black has been used to represent a coalition of four controllers that cooperated during the time instant in which the screenshot was taken.

Finally, it is important to remark that two functions – $DrawTank$ and $DrawPipe$ – are provided with the software, so that the user can easily build new simulation scenarios composed of interconnected water tanks.

V. Conclusions and Future Work

In this paper, a general formulation of coalitional control has been presented. In addition, a MATLAB class, which is free to download, has been introduced in order to ease the implementation of this type of control architecture. Likewise, different examples are provided as well to simplify the simulation and assessment of the schemes.

Given the novelty of this type of control strategies, there are a lot of topics to research in the future. Probably, the most interesting one is the development of bottom-up coalitional control schemes because so far most of the schemes in the literature present a hierarchical structure that decides the network topology. Ideally, a true peer-to-peer architecture should be the ultimate goal. Likewise, classical research topics such as stability or performance bounds have to be addressed as well.

From a software perspective, the class provided has to be enhanced into several directions. In the first place, the class constitutes a good base for an extension towards the framework of MPC and, consequently, this should be the first step. A second step could be to take into account explicitly the way in which systems interact with each other in order to include a method that calculates the distributed invariant set of the coalitional control scheme according to [19], [6]. This is an interesting problem, since it is not clear what behaviour should be expected from the agents outside a given coalition. On top of that, more complex network architectures

can be computed as well. For example, a hierarchy between the different local controllers could be calculated in order to test hierarchical-distributed control strategies. Finally, it would be interesting to consider as well different types of timing such as event-driven or asynchronous communication.

REFERENCES

[1] N. Sandell, P. Varaiya, M. Athans, and M. Safonov, "Survey of decentralized control methods for large scale systems," *IEEE Transactions on Automatic Control*, vol. 23, no. 2, pp. 108–128, 1978.

[2] E. F. Camacho and C. Bordons, *Model Predictive Control in the Process Industry. Second Edition.* London, England: Springer-Verlag, 2004.

[3] R. R. Negenborn, P. J. Van Overloop, T. Keviczky, and B. De Schutter, "Distributed model predictive control of irrigation canals," *Networks and Heterogeneus Media*, vol. 4, pp. 359–380, 2009.

[4] R. R. Negenborn, B. De Schutter, and H. Hellendoorn, "Multi-agent model predictive control for transportation networks: Serial versus parallel schemes," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 3, pp. 353–366, April 2008.

[5] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho, "Distributed MPC: a supply chain case study," in *Proceedings of Conference on Decision and Control. Accepted for publication*, 2009.

[6] J. M. Maestre, D. Muñoz de la Peña, E. F. Camacho, and T. Alamo, "Distributed model predictive control based on agent negotiation," *Journal of Process Control*, vol. 21, no. 5, 2011.

[7] R. Scattolini, "Architectures for distributed and hierarchical model predictive control - a review," *Journal of Process Control*, vol. 19, pp. 723–731, 2009.

[8] P. D. Christofides, R. Scattolini, D. Muñoz de la Peña, and J. Liu, "Distributed model predictive control: A tutorial review and future research directions," *Computers & Chemical Engineering*, vol. 51, no. 0, pp. 21 – 41, 2013.

[9] J. M. Maestre and R. R. Negenborn, Eds., *Distributed Model Predictive Control Made Easy*, ser. Intelligent Systems, Control and Automation: Science and Engineering. Springer, 2014, vol. 69.

[10] P. Trodden and A. G. Richards, "Adaptive cooperation in robust distributed model predictive control," in *Proceedings of the* 24^{th} *IEEE International Symposium on Intelligent Control*, St. Petersburg, Russia, July 2009, pp. 896–901.

[11] M. Jilg and O. Stursberg, "Optimized distributed control and topology design for hierarchically interconnected systems," in *Proceedings of the 2013 European Control Conference*, Zurich, Switzerland, 2013, pp. 4340–4346.

[12] A. Nuñez, C. Ocampo-Martínez, B. De Schutter, F. Valencia, J. López, and J. Espinosa, "A multiobjective-based switching topology for hierarchical model predictive control applied to a hydro-power valley," in *2013 IFAC International Conference on Intelligent Control and Automation Science*, Chengdu, China, 2013.

[13] J. M. Maestre, D. Muñoz de la Peña, A. Jiménez Losada, E. Algaba, and E. F. Camacho, "A coalitional control scheme with applications to cooperative game theory," *Optimal Control Applications and Methods*, September 2013, DOI: 10.1002/oca.2090.

[14] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho, "An application of cooperative game theory to distributed control," in *Submitted to the 18th IFAC World Congress*, 2010.

[15] F. Fele, J. Maestre, F. Muros, and E. Camacho, "Coalitional control: An irrigation canal case study," in *2013 10th IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 2013, pp. 759–764.

[16] I. Alvarado, D. Limon, D. Muñoz de la Peña, J. M. Maestre, F. Valencia, H. Scheu, R. R. Negenborn, M. A. Ridao, B. De Schutter, J. Espinosa, and W. Marquardt, "A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark," *Journal of Process Control*, vol. 21, no. 5, pp. 800–815, 2011.

[17] R. M. Hermans, A. Jokic, M. Lazar, A. Alessio, P. P. van den Bosch, I. A. Hiskens, and A. Bemporad, "Assessment of non-centralised model predictive control techniques for electrical power networks," *International Journal of Control*, vol. 85, no. 8, pp. 1162–1177, 2012.

[18] M. Slikker and A. Van den Nouweland, *Social and Economics Networks in Cooperative Game Theory*. Kluwer Academic Publishers, 2001.

[19] S. V. Rakovic, E. De Santis, and P. Caravani, "Invariant equilibria of polytopic games via optimized robust control invariance," in *Proceedings of the 44th IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, December 2005, pp. 7686–7691.