# Chapter 25
# Distributed MPC Based on a Team Game

**J. M. Maestre, F. J. Muros, F. Fele, D. Muñoz de la Peña and E. F. Camacho**

**Abstract** In this chapter we present a distributed scheme based on a team game for the particular case in which the system is controlled by two agents. The main features of the proposed scheme are the limited amount of global information that the agents share and the low communication burden that it requires. For this reason, this scheme is a good candidate to be implemented in systems with reduced capabilities, for example in wireless sensor and actuator networks.

## 25.1 Introduction

In this chapter, a distributed model predictive control (DMPC) scheme based on a team game, originally proposed in [8] and formalized in [7], is presented. In this scheme two different agents communicate in order to find a solution to the problem of controlling two constrained linear systems coupled through the inputs; see Fig. 25.1. The assumptions about the amount of global information that the agents have are very restrictive in this scheme. In particular, we assume that each agent only has local model and state information. The only global information that an agent has is how the neighbor's input affects him. Notice that although this assumption can be

J. M. Maestre (✉) · F. J. Muros · F. Fele · D. Muñoz de la Peña · E. F. Camacho
Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Seville, Spain
e-mail: pepemaestre@cartuja.us.es

F. J. Muros
e-mail: franmuros@gmail.com

F. Fele
e-mail: filiberto@cartuja.us.es

D. Muñoz de la Peña
e-mail: davidmps@cartuja.us.es

E. F. Camacho
e-mail: eduardo@cartuja.us.es

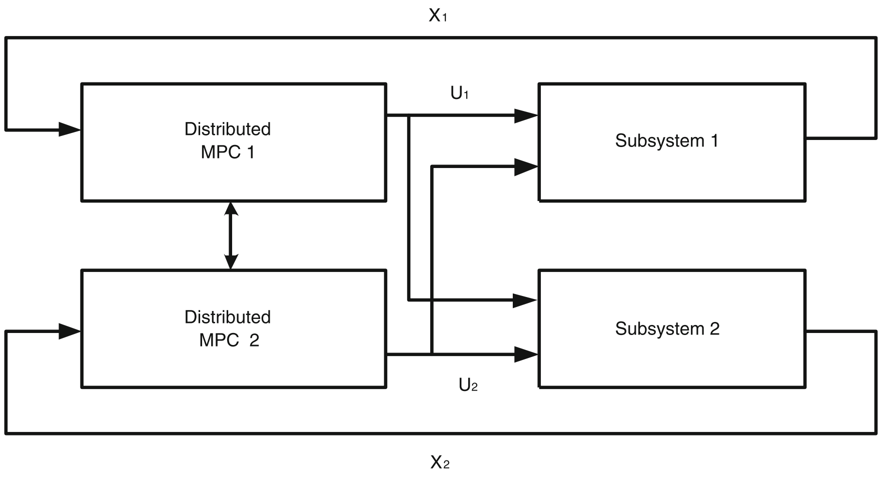


**Fig. 25.1** Proposed distributed MPC scheme for two agents

restrictive, it is realistic in the sense that global information about the models and the objectives is not always available. We also assume that the available communication capabilities only allow two communications at each time interval. For this reason, the coordination between the agents is based on a team game that is built in two communication cycles.

## 25.2 Statement of the Problem and Main Assumptions

Consider the class of distributed linear systems composed by a set of $\mathcal{N} = \{1, 2\}$ interconnected subsystems coupled by the inputs whose dynamics can be described mathematically as:

$$\begin{aligned} \mathbf{x}_1(k+1) &= \mathbf{A}_1\mathbf{x}_1(k) + \mathbf{B}_{12}\mathbf{u}_1(k) + \mathbf{B}_{12}\mathbf{u}_2(k), \\ \mathbf{x}_2(k+1) &= \mathbf{A}_2\mathbf{x}_2(k) + \mathbf{B}_{21}\mathbf{u}_1(k) + \mathbf{B}_{22}\mathbf{u}_2(k), \end{aligned} \tag{25.1}$$

where $\mathbf{x}_i(k) \in \mathbb{R}^{n_{x_i}}$, $i \in \mathcal{N}$ are the states of subsystem $i$, and $\mathbf{u}_i(k) \in \mathbb{R}^{n_{u_i}}$, $i \in \mathcal{N}$ are their respective inputs. For simplicity we will show the definitions and equations for agent $i$. In the remaining of the chapter, the neighboring agent will be denoted as $ni$; for example: $\mathbf{u}_{ni} = \mathbf{u}_2$ if $i = 1$ and $\mathbf{u}_{ni} = \mathbf{u}_1$ if $i = 2$.

States and inputs are constrained into two independent sets defined by a set of linear inequalities

$$\mathbf{x}_i(k) \in \mathcal{X}_i, \quad \mathbf{u}_i(k) \in \mathcal{U}_i, \quad i = 1, 2. \tag{25.2}$$

The agent responsible of subsystem $i$ has complete knowledge of its local model and state $\mathbf{x}_i(k)$, and is able to manipulate the control action $\mathbf{u}_i(k)$. Hence, no agent has full model or state information and communication is required in order to obtain a cooperative solution.

Without loss of generality, we assume that the control objective is to regulate the system to the origin while satisfying the constraints. To this end, we define the following performance index for each agent, which depends on the future evolution of its state and input along a prediction horizon of length $N_\text{p}$:

$$J_i\left(\mathbf{x}_i(k), \mathbf{u}_i(k:k+N_\text{p}-1), \mathbf{u}_{ni}(k:k+N_\text{p}-1)\right) = \sum_{k=0}^{N_\text{p}-1} L_i\left(\mathbf{x}_i(k), \mathbf{u}_i(k)\right) + F_i\left(\mathbf{x}_i(N_\text{p})\right), \quad (25.3)$$

where $L_i(\cdot)$ and $F_i(\cdot)$ with $i \in \mathcal{N}$ are the stage cost and the terminal cost functions respectively, defined as:

$$L_i(\mathbf{x}, \mathbf{u}) = \mathbf{x}_i^\text{T}\mathbf{Q}_i\mathbf{x}_i + \mathbf{u}_i^\text{T}\mathbf{R}_i\mathbf{u}_i, \quad (25.4)$$

$$F_i(\mathbf{x}) = \mathbf{x}_i^\text{T}\mathbf{P}_i\mathbf{x}_i, \quad (25.5)$$

with $\mathbf{Q}_i, \mathbf{P}_i > 0$, $\mathbf{R}_i \geq 0$, and the vector $\mathbf{u}_i(k:k+N_\text{p}-1)$ stands for the future input sequence of agent $i$:

$$\mathbf{u}_i(k:k+N_\text{p}-1) = \left[\mathbf{u}_i(k)^\text{T}, \mathbf{u}_i(k+1)^\text{T}, \ldots, \mathbf{u}_i(k+N_\text{p}-1)^\text{T}\right]^\text{T}, \quad i \in \mathcal{N}. \quad (25.6)$$

The agents solve a sequence of optimization problems during each sampling interval. These problems are built using local information and assuming a fixed input trajectory for its neighbor. In particular, an agent assumes that its neighbor will behave as in the previous agreed input trajectory, which we will denote with the superscript d, i.e., $\mathbf{u}_i^\text{d}(k-1:k+N_\text{p}-2)$ is the input trajectory that agent $i$ agreed to implement at time $k-1$ for the time interval $[k-1, k+N_\text{p}-2]$.

At this point it is important to remark that the trajectory that agent $i$ assumes for its neighbor must cover the time interval $[k, k+N_\text{p}-1]$. As the last agreed trajectory only provides a trajectory between $[k-1, k+N_\text{p}-2]$ and the first component of that trajectory is already implemented in the system, it is necessary to calculate a new component for the time $k+N_\text{p}-1$. If stability is not an issue in the system, it is possible to complete the sequence simply by adding a 0 vector of the proper size or copying the value of the last known element of the trajectory. In this chapter we assume that each agent updates his decided input trajectory using a feedback gain $K_i$ and the $N_p$-1 steps ahead prediction of the state assuming. These feedbacks allow us to define the shifted input sequence of agent $i$ at time $k$ as:

$$\mathbf{u}_i^\text{s}(k:k+N_\text{p}-1) = \left[\mathbf{u}_i^\text{d}(k)^\text{T}, \mathbf{u}_i^\text{d}(k+1)^\text{T}, \ldots, \mathbf{u}_i^\text{d}(k+N_\text{p}-2), (\mathbf{K}_i\mathbf{x}_i(k+N_\text{p}-1))^\text{T}\right]^\text{T}, \quad (25.7)$$

where $\mathbf{x}_i(k + N_\text{p} - 1)$ is the state of the subsystem $i(N_\text{p} - 1)$-steps ahead in the future, which is obtained applying the agreed input trajectories.

## 25.3 Description of the Approach

Algorithm 25.1 describes the DMPC scheme proposed in this chapter.

From a game theory point of view, at each time step both agents are playing a team game. This game can be synthesized in strategic form by the three by three matrix of Table 25.1. At each time step, the controllers decide among three different options. The shifted optimal input trajectory $\mathbf{u}_i^\text{s}(\cdot)$ keeps applying the latest optimal trajectory, so it can be seen like a stable decision. The selfish option $\mathbf{u}_i^*(\cdot)$ provides the best improvement in $J_i$ if the rest of the systems manipulated variables stay unchanged. The altruist option $\mathbf{u}_i^\text{w}(\cdot)$ provides the best improvement for the neighbor agent cost function $J_{ni}$ in case it applies its selfish option; i.e., the agent $i$ sacrifices its own welfare in order to improve the overall performance. The cells of the matrix contain the sum of the cost functions of both agents for a particular choice of future inputs, resulting nine possibilities. At each time step, the first option that provides the lowest global cost is chosen. Note that both agents share this information, so they both choose the same option. Some important facts must be remarked about the algorithm:

- The computational burden of the proposed algorithm is in general lower than the one corresponding to the centralized scheme. In particular, the quadratic programming problems solved have less optimization variables.
- Agents operate in parallel in steps 2 and 3, which speeds up the calculation of the input trajectories.
- The minimun number of communication steps that are necessary to obtain a cooperative control scheme is two: in the first one each agent broadcasts its proposals to its neighbors and during the second one feedback about them is received.
- The proposed scheme is cooperative from a game theory point of view because each agent chooses the solution that minimizes a value that depends on the cost of both subsystems. If there were no cooperation between the agents, the solution

**Table 25.1** Cost function table used for the decision making

| | $\mathbf{u}_2^\text{s}(\kappa)$ | $\mathbf{u}_2^*(\kappa)$ | $\mathbf{u}_2^\text{w}(\kappa)$ |
|---|---|---|---|
| $\mathbf{u}_1^\text{s}(\kappa)$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^\text{s}(\kappa), \mathbf{u}_2^\text{s}(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^\text{s}(\kappa), \mathbf{u}_1^\text{s}(\kappa))$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^\text{s}(\kappa), \mathbf{u}_2^*(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^*(\kappa), \mathbf{u}_1^\text{s}(\kappa))$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^\text{s}(\kappa), \mathbf{u}_2^\text{w}(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^\text{w}(\kappa), \mathbf{u}_1^\text{s}(\kappa))$ |
| $\mathbf{u}_1^*(\kappa)$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^*(\kappa), \mathbf{u}_2^\text{s}(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^\text{s}(\kappa), \mathbf{u}_1^*(\kappa))$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^*(\kappa), \mathbf{u}_2^*(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^*(\kappa), \mathbf{u}_1^*(\kappa))$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^*(\kappa), \mathbf{u}_2^\text{w}(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^\text{w}(\kappa), \mathbf{u}_1^*(\kappa))$ |
| $\mathbf{u}_1^\text{w}(\kappa)$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^\text{w}(\kappa), \mathbf{u}_2^\text{s}(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^\text{s}(\kappa), \mathbf{u}_1^\text{w}(\kappa))$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^\text{w}(\kappa), \mathbf{u}_2^*(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^*(\kappa), \mathbf{u}_1^\text{w}(\kappa))$ | $J_1(\mathbf{x}_1(k), \mathbf{u}_1^\text{w}(\kappa), \mathbf{u}_2^\text{w}(\kappa)) + J_2(\mathbf{x}_2(k), \mathbf{u}_2^\text{w}(\kappa), \mathbf{u}_1^\text{w}(\kappa))$ |

$\kappa$ stands for the time interval $[k : k + N_\text{p} - 1]$

**Algorithm 25.1** DMPC based on a team game

```
1: At time step $k$, each agent $i$ measures its state $\mathbf{x}_i(k)$.
2: Each agent $i$ minimizes $J_i$ assuming that the neighbor applies the shifted control sequence
   calculated during the previous time step and solves the following optimization problem, $i \in \mathcal{N}$
   $$\begin{aligned}&\mathbf{u}_i^*(k:k+N_\mathrm{p}-1) = \\ &\arg\min_{\mathbf{u}_i(k:k+N_\mathrm{p}-1)} J_i\left(\mathbf{x}_i(k), \mathbf{u}_i(k:k+N_\mathrm{p}-1), \mathbf{u}_{ni}^\mathrm{s}(k:k+N_\mathrm{p}-1)\right)\\ &\text{s.t. } \mathbf{x}_i(k+1) = \mathbf{A}_i\mathbf{x}_i(k) + \mathbf{B}_{ii}\mathbf{u}_i(k) + \mathbf{B}_{i,ni}\mathbf{u}_{ni}(k),\\ &\mathbf{x}_i(l) \in \mathcal{X}_i, \quad l = k+1,\ldots,k+N_\mathrm{p},\\ &\mathbf{u}_i(l) \in \mathcal{U}_i, \quad l = k,\ldots,k+N_\mathrm{p}-1,\\ &\mathbf{x}_i(k+N_\mathrm{p}) \in \Omega_i.\end{aligned} \tag{25.8}$$
   The sets $\Omega_i$, $i \in \mathcal{N}$ define the terminal region constraints necessary to prove closed-loop
   practical stability following a terminal region/terminal cost approach. If stability is not an issue
   these constraints can be removed.
3: Each agent $i$ minimizes $J_i$ optimizing the neighbor input sequence $\mathbf{u}_{ni}(\cdot)$ while applying to its
   own subsystem the input trajectory $\mathbf{u}_i^*(\cdot)$ computed in step 2. This means for agent $i$ solving the
   following optimization problem, $i \in \mathcal{N}$
   $$\begin{aligned}&\mathbf{u}_{ni}^\mathrm{w}(k:k+N_\mathrm{p}-1) = \\ &\arg\min_{\mathbf{u}_{ni}(k:k+N_\mathrm{p}-1)} J_i\left(\mathbf{x}_i(k), \mathbf{u}_i^*(k:k+N_\mathrm{p}-1), \mathbf{u}_{ni}(k:k+N_\mathrm{p}-1)\right)\\ &\text{s.t. } \mathbf{x}_i(k+1) = \mathbf{A}_i\mathbf{x}_i(k) + \mathbf{B}_{ii}\mathbf{u}_1(k) + \mathbf{B}_{i,ni}\mathbf{u}_{ni}(k),\\ &\mathbf{x}_i(l) \in \mathcal{X}_i, \quad l = k+1,\ldots,N_\mathrm{p},\\ &\mathbf{u}_{ni}(l) \in \mathcal{U}_{ni}, \quad l = k,\ldots,k+N_\mathrm{p}-1,\\ &\mathbf{x}_i(k+N_\mathrm{p}) \in \Omega_i.\end{aligned} \tag{25.9}$$
   The solution of this optimization problem, $\mathbf{u}_{ni}^\mathrm{w}(k:k+N_\mathrm{p}-1)$, is a proposal for the neighbor (a
   wished behaviour), i.e., an input trajectory that the neighbor $ni$ can implement to minimize the
   cost of agent $i$. Again, the constraint involving $\Omega_i$ can be removed if stability is not an issue.
4: Both agents communicate, sending both $\mathbf{u}_i^*(k:k+N_\mathrm{p}-1)$ and $\mathbf{u}_{ni}^\mathrm{w}(k:k+N_\mathrm{p}-1)$ to the other
   agent, and receiving $\mathbf{u}_{ni}^*(k:k+N_\mathrm{p}-1)$ and $\mathbf{u}_i^\mathrm{w}(k:k+N_\mathrm{p}-1)$.
5: Each agent evaluates its local cost function $J_i$ for each of the nine possible combinations of the
   input trajectories obtained: $\mathbf{u}_i \in \left\{\mathbf{u}_i^\mathrm{s}(\cdot), \mathbf{u}_i^\mathrm{w}(\cdot), \mathbf{u}_i^*(\cdot)\right\}$, $i \in \mathcal{N}$ (see Table 25.1).
6: Both agents communicate and share the information of the value of local cost function for each
   possible combination of input trajectories. In this step, both agents receive enough information
   to take a cooperative decision. In addition, this communication cycle can be used as well to
   transmit the information regarding the extra component that is necessary to complete the shifted
   input trajectory vector.
7: Each agent applies the input trajectory that minimizes $J = J_1 + J_2$. Because both agents have
   access to the same information after the second communication cycle, both agents choose the
   same optimal input trajectories, i.e., $\mathbf{u}_i^\mathrm{d}(\cdot)$, $i \in \mathcal{N}$.
8: The first input of each optimal sequence is applied and the procedure is repeated at the next time
   step.
```

attained would converge after several iterations towards the Nash equilibrium of the multi-objective optimization problem defined by the cost functions of the agents.

- Although the outcome chosen by the algorithm is a Pareto optimal of the game that both agents are playing, in general it is not a Pareto optimal of the multi-objective optimization problem defined by the cost functions $J_1$ and $J_2$.

- The proposed scheme can be extended to deal with a greater number of agents, i.e., $|\mathcal{N}| = M$. However, the complexity of building the corresponding team game matrix grows exponentially with the number of agents. In order to reduce the complexity, the structure of the system may be exploited taking into account that an input may no affect all the outputs. Also, in general not all the possible cooperation options are employed with the same frequency, so it is possible to reduce further the complexity by not taking into account the less frequent options. In [7], the reader can find a modified distributed scheme which deal with this topic in depth.

## 25.4 Theoretical Results Availability

In this section we introduce the main theoretical propierties of the proposed DMPC scheme. Please, notice that many theoretical details have been omitted for the sake of clarity. The interested reader is recommended to see [5] for a more rigorous treatment of the topics discussed in this section.

### *25.4.1 Stability Properties*

Controlling a system between two independient agents may lead to an unstable closed-loop system. The resulting closed-loop system is a multiprocess system and studying the stability of this class of systems is in general a difficult task. Following a terminal region/terminal constraint approach, in [5] we provided sufficient conditions that guarantee practical stability of the closed-loop system as well as an optimization based procedure to design the controller so that these conditions are satisfied.

In [5] it is proved that if there exist linear feedbacks $\mathbf{u}_i = \mathbf{K}_i\mathbf{x}_i$, terminal cost functions defined by matrices $\mathbf{P}_i$, and regions $\Omega_i$ that satisfy the following conditions for all $i \in \mathcal{N}$, then the system in closed-loop with the proposed controller is ultimately bounded in a region that contains the origin:

$$F_i\left((\mathbf{A}_i + \mathbf{B}_{ii}\mathbf{K}_i)\,\mathbf{x}_i + \mathbf{B}_{i,ni}\mathbf{K}_{ni}\mathbf{x}_{ni}\right) - F_i(\mathbf{x}_i) + L_i\,(\mathbf{x}_i, \mathbf{K}_i\mathbf{x}_i) - d_i \leq 0, \forall \mathbf{x}_{ni} \in \Omega_{ni}, \tag{25.10}$$

$$\mathbf{x}_i \in \Omega_i \rightarrow (\mathbf{A}_i + \mathbf{B}_{ii}\mathbf{K}_i)\,\mathbf{x}_i + \mathbf{B}_{i,ni}\mathbf{K}_{ni}\mathbf{x}_{ni} \in \Omega_i,\ \forall \mathbf{x}_{ni} \in \Omega_{ni}, \tag{25.11}$$

$$\mathbf{K}_i\mathbf{x}_i \in \mathcal{U}_i,\ \forall \mathbf{x}_i \in \Omega_i, \tag{25.12}$$

$$\Omega_i \in \mathcal{X}_i. \tag{25.13}$$

If the aforementioned conditions are satisfied, then it is possible to prove that if $\mathbf{x}_i(0)$ and $\mathbf{u}_i^s(0)$ for all $i \in \mathcal{N}$ are given such that the optimization problem (25.8) is feasible, then the proposed algorithm is feasible for all time steps $k \geq 0$ and system (25.1) in closed-loop with the proposed DMPC controller is ultimately bounded in a region that contains the origin in its interior. Likewise, it is possible to guarantee that the closed-loop system is ultimately bounded in a closed region that contains the origin. Moreover, it is possible to prove that the proposed controller provides asympthotic stability if (25.10) is modified. Specific details about this topic can be found in [5, 7].

## *25.4.2 Design Procedure*

In [5, 7] an optimization based procedure is given to find, for $i \in \mathcal{N}$, local controllers $\mathbf{K}_i$, matrices $\mathbf{P}_i$ and regions $\Omega_i$ such that (25.10)–(25.13) holds for a given system. The procedure determines first matrices $\mathbf{K}_i$ and $\mathbf{P}_i$ such that (25.10)–(25.13) hold for any given sets $\Omega_i$ solving a linear matrix inequality (LMI) optimization problem. Once the local feedbacks $\mathbf{K}_i$ are fixed, the invariant sets $\Omega_i$ are obtained.

The linear feedbacks must guarantee that each system is stable and must provide a certain degree of robustness with respect to the neighbor control input. To certain degree, each local controller assumes that the neighbor input is a bounded disturbance that has to be rejected. This allows us to use well known tools from control of linear uncertain systems in order to determine a local controller such that a given degree of robustness is guaranteed. Constraint (25.10) can be transformed into an LMI and solved using standard techniques. In particular, the following LMI provides a mean for the calculation of $\mathbf{K}_i$ and $\mathbf{P}_i$:

$$\begin{bmatrix} \gamma_i I & 0 & B_{i,ni}^{\mathrm{T}} & 0 & 0 \\ * & \mathbf{W}_i & \mathbf{W}_i \mathbf{A}_{ii}^{\mathrm{T}} + \mathbf{Y}_i^{\mathrm{T}} \mathbf{B}_{ii}^{\mathrm{T}} & \mathbf{W}_i \mathbf{Q}_i^{\frac{1}{2}} & \mathbf{Y}_i^{\mathrm{T}} \mathbf{R}_i^{\frac{1}{2}} \\ * & * & \mathbf{W}_i & 0 & 0 \\ * & * & * & I & 0 \\ * & * & * & * & I \end{bmatrix} > 0. \tag{25.14}$$

where $\mathbf{P}_i = \mathbf{W}_i^{-1}, \mathbf{K}_i = \mathbf{Y}_i \mathbf{W}_i^{-1}$, and $\gamma_i$ is a constant related with the size of the maximum admissible disturbance for system $i$: $d_i = \gamma_i \max_{\mathbf{x} \in \Omega_{ni}} (\mathbf{K}_{ni} \mathbf{x})^{\mathrm{T}} \mathbf{K}_{ni} \mathbf{x}$. Notice that the size of the disturbance, given by $d_i$, depends at the same time on the size of the sets $\Omega_i$. Likewise, note that the conditions presented assume that each input $\mathbf{u}_i$ only depends on the state $\mathbf{x}_i$. This structure can be generalized for local controllers that take into account the full state space.

Once the local controllers and the terminal cost functions are fixed, it is necessary to find sets $\Omega_i$ such that (25.10)–(25.13) hold. In general this is a difficult problem because each of the sets depends on the other. The size of the terminal region for agent $i$ is determined by the magnitude of the disturbances induced by its neighbor agent $ni$ and viceversa. The main idea behind the calculation of the invariant sets of both agents is a scalation of their respective input constraints sets by a factor $\lambda_i \leq 1$ with the goal of finding a fair trade off between the disturbances induced to the neighbor and the manipulating capabilities of each agent. There exist several methods to find a set $\Omega_i$ that satisfies all the local constraints $\mathcal{X}_i$ and $\lambda_i \mathcal{U}_i$ while coping with the disturbances induced by the neighbor, which depend on $\lambda_{ni}\mathcal{U}_{ni}$; e.g.: in [12] a procedure to find an approximation of the minimal robust positive invariant is given, and in [11] a similar class of invariant systems was studied within the polytopic games framework.

At this point, it is important to stress that the set $\Omega_1 \times \Omega_2$ is an invariant set for the system in closed loop with the linear feedbacks $\mathbf{K}_1, \mathbf{K}_2$; however, the opposite does not hold necessarily, i.e., not all invariant sets satisfy the stability conditions. Note that each set is defined in their corresponding subspace $\mathbf{x}_i$. We denote these sets jointly invariant sets. Note as well that there may exist an infinite number of possible values of $\lambda_{ji}$ such that these sets exist. In order to choose one, we propose to solve an optimization problem to maximizes the feasibility region of the distributed MPC controller. In [11] it was proved that the feasibility region of this problem is convex. In [9] we prove that the jointly invariant sets $\Omega_i$ are polyhedra defined by a set of inequalities whose right hand side can be expressed as an affine combination of the constants $\gamma_{ij}$. Using this result, the optimization problem can be cast into a convex optimization problem if the objective function is chosen appropriately, for instance, if the criterium to compare the invariant sets is the radium of a Chebyshev ball inside the invariant region.

Once matrices $\mathbf{K}_i$, $\mathbf{P}_i$, and the sets $\Omega_i$, $i \in \mathcal{N}$ are determined, constants $d_i$, $i \in \mathcal{N}$ can be calculated in order to obtain an estimation of the set in which the closed-loop system is ultimately bounded.

## 25.5 Applications of the DMPC Scheme

The algorithm that we propose in this chapter has been tested with simulated and real systems. The applications and the results that are described in this section are explained in detail in [2, 5–8, 10].

### *25.5.1 Supply Chain Problem*

In [5, 6], the proposed controller is applied to a reduced version of the MIT beer game and compared with other distributed control schemes. The MIT beer game is based on the concept of a supply chain, i.e., the set of structures and processes used by an organization to provide a service or a good to a costumer. The original MIT beer game is composed of four agents: retailer, wholesaler, distributor and factory. In our case, a reduced version of the problem with only two agents is considered: the retailer and the supplier, see Fig. 25.2. Notice that there is no loss of generality since the structure of the game is regular: there is a cascade of firms, each maintaining and controlling its stock.

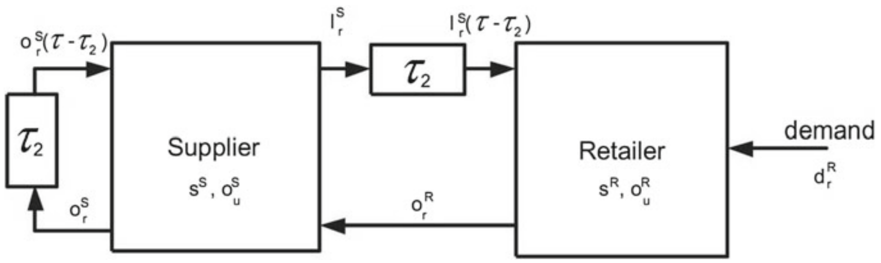


**Fig. 25.2** Reduced beer game

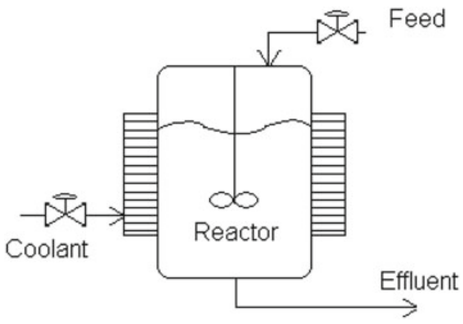
The control objective is to regulate the number of stocked beers to a given reference and the manipulated variable is the number of beers that are ordered at each node. The retailer must satisfy the demand of beers and orders new beers to the supplier, which at the same time asks the factory for beers.

The performance comparison between the different schemes considered is based on simulations that are carried out in four different scenarios. Each scenario is defined by a different initial state, a different retailer demand, and a different demand forecast. In general, the proposed algorithm provides a performance in the same order of magnitude than the centralized MPC which, as expected, obtains the best results. The simulations show that the proposed distributed scheme outperforms a noncooperative iterative DMPC scheme with a greater number of communication cycles.

## 25.5.2 *Continuosly Stirred Tank Reactor*

In [8, 10] the proposed scheme is applied to a system described by a transfer function: a continuosly stirred tank reactor (CSTR) (taken from [3]). The manipulated variables are respectively the flow rate and the flow of coolant in the jacket. The controlled variables are respectively the effluent concentration and the reactor temperature,

**Fig. 25.3** Continuously stirred tank reactor (CSTR)



see Fig. 25.3. The control objective is to track a given constant reference from a random initial state. Our results show that the proposed scheme is able to stabilize the closed-loop system while other decentralized schemes fail. On top of that, the rise time and the convergence rate of the overall cost function are determined as a function of the average number of data losses in the communication channel. Hence, this system is used to test the robustness of the algorithm against data losses. Notice that as the reliability of the communication channel decreases, so does the amount of information shared by the agents, and the controller tends to operate in a decentralized manner. In particular, when there are data losses, the agents do not receive $\mathbf{u}_i^{\mathrm{w}}$, which is needed to build the global cost table (Table 25.1). In this case, each agent must decide whether to keep applying the last optimal input trajectory $\mathbf{u}_i^{\mathrm{s}}$, or behave selfishly and try to minimize its local cost function choosing $\mathbf{u}_i^*$. In

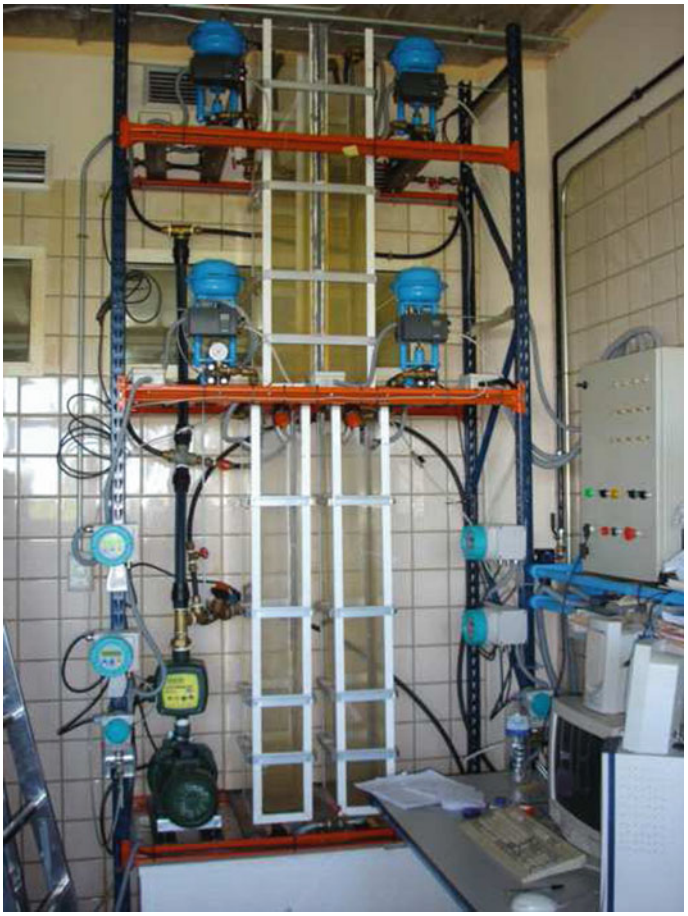


**Fig. 25.4** The four tank process: the real plant diagram

order to test the robustness of the proposed approach on the worst possible case, we assume that when communication errors occur each controller applies $\mathbf{u}_i^*$. A set of simulations with different average values of data losses can be found in [10]. The simulation results show that the algorithm is able to stabilize the closed-loop system whenever the number of data losses is lower than the 50 %.

### *25.5.3 The Four Tank Process*

In [2, 5] the proposed scheme and other DMPC policies are compared in an real benchmark: a four tank process (see Fig. 25.4), which is one of the benchmarks of the european project HD-MPC. The physical plant is located in facilities of the University of Seville and is described in [1]. It is an educational plant designed to test control techniques with real industrial instrumentation and control devices. The plant is a hydraulic process of four inteconnected tanks inspired by the educational quadruple tank process proposed by Johansson [4]. The main characteristic of this process is that is a simple multivariable system with highly coupled nonlinear dynamics that can exhibit transmission zeros dynamics. The four tank process has proven to be a very interesting system for control education and research.

The objective of the benchmark is to test and compare centralized, decentralized and distributed predictive controllers under similar operation conditions. To this end an experiment is defined in [2, 5], in which the controllers must regulate the levels of tanks 1 and 2 to follow a set of reference changes by manipulating the inlet flows based of the measured levels of the four tanks.

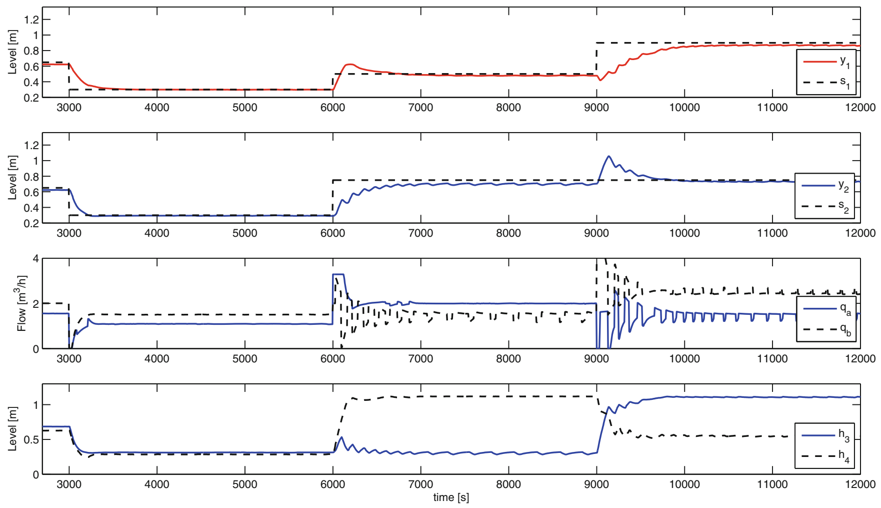


**Fig. 25.5** Experimental results on the four tank process

Our results with this plant show a performance index relatively close to the centralized MPC and, moreover, our DMPC scheme outperforms other more complex schemes that have higher communication burdens. Nevertheless our results also show a possible implementation issue for other applications: the resulting input trajectories are not smooth (see Fig. 25.5), which is natural since the controller chooses among nine different modes of operation. Depending on the application, this switching may not be acceptable. Figure 25.5 also shows the cooperative nature of the proposed scheme in the sense that the inputs switch constantly between the two possible trajectories that suit best with the local objectives of each agent.

## 25.6 Conclusions

We have presented a DMPC scheme based on game theory for a class of systems controlled by two agents. The proposed controller only needs two communication steps in order to provide a cooperative solution for the centralized optimization problem. Each agent solves an optimization problem that depends only on its local model and state information. After sharing information about the cost of the control actions considered, the agents choose the solution that provides the best global performance among a set of possible suboptimal choices (the options are suboptimal because each agent has an incomplete view of the system and they propose the best solutions from a local perspective).

The proposed algorithm guarantees feasibility and stability of the closed-loop system if the feedback laws are designed according to the proposed procedure. On top of that, our results show a good behavior of the control scheme, which is specially remarkable when its low communication and informational requirements are taken into account. Likewise, the robustness of the proposed scheme against failures in the communication channel has been tested in simulation as well.

Finally, it is worthwhile to mention that although it is possible to extend the proposed scheme to control problems with more than 2 agents, the size of the team game that must be built grows exponentially with the number of agents. Therefore it is necessary to reduce the number of options that are proposed and evaluated by the agents such as it is done in [9], where an evolution of this scheme is presented

**Acknowledgments** Financial support from the HYCON2 EU-project from the ICT-FP7 and MEC-Spain, DPI2008-05818, and F.P.I. grants is gratefully acknowledged.

## References

1. I. Alvarado, D. Limón, W.M. Garca, T. Álamo, E.F. Camacho, An educational plant based on the quadruple-tank process, in *Preprints of the 7th IFAC Symposium on Advances in Control Education*, Madrid, Spain, June 2006
2. I. Alvarado, D. Limón, D. Muñoz de la Peña, J.M. Maestre, F. Valencia, H. Scheu, R.R. Negenborn, M.A. Ridao, B. De Schutter, J. Espinosa, W. Marquardt, A comparative analysis

of distributed MPC techniques applied to the HD-MPC four tank benchmark. J. Process Control **21**(5), 800–815 (June 2011)
3. E.F. Camacho, C. Bordóns, *Model Predictive Control in the Process Industry*, 2nd edn. (Springer, London, England, 2004)
4. K.H. Johansson, The quadruple-tank process: a multivariable laboratory process with an adjustable zero. IEEE Trans. Control Syst. Technol. **8**(3), 456–465 (May 2000)
5. J.M. Maestre, *Distributed Model Predictive Control Based on Game Theory*, PhD thesis, Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Seville, Spain, Nov 2010
6. J.M. Maestre, D. Muñoz de la Peña, E.F. Camacho, Distributed MPC: a supply chain case study, in *Proceedings of the joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Dec 2009
7. J.M. Maestre, D. Muñoz de la Peña, E.F. Camacho, Distributed model predictive control based on a cooperative game. Optimal Control Appl. Methods **32**(2), 153–176, Mar/Apr 2011
8. J.M. Maestre, D. Muñoz de la Peña, E.F. Camacho, A distributed MPC scheme with low communication requirements, in *Proceedings of the American Control Conference*, pp. 2797–2802, June 2009
9. J.M. Maestre, D. Muñoz de la Peña, E.F. Camacho, T. Álamo, Distributed model predictive control based on agent negotiation. J. Process Control **21**(5), 685–697 (June 2011)
10. F.J. Muros, J.M. Maestre, E.F. Camacho, *Estudio de robustez frente a retardos y pérdida de datos de una estrategia dmpc basada en pocos ciclos de comunicación* (In Actas de las XXIX Jornadas de Automática, Tarragona, Spain, Sept, 2008)
11. S. Rakovic, E. De Santis, 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*, pp. 7686–7691, Seville, Spain, Dec 2005
12. S. Rakovic, E. Kerrigan, K. Kouramas, D. Mayne, Robust MPC for nonlinear discrete-time systems. IEEE Trans. Automat. Control **50**, 406–410 (2005)