Delft University of Technology

Delft Center for Systems and Control

Technical report 09-025

A distributed version of Han's method for DMPC using local communications only*

D. Doan, T. Keviczky, I. Necoara, M. Diehl, and B. De Schutter

If you want to cite this report, please use the following reference instead:

D. Doan, T. Keviczky, I. Necoara, M. Diehl, and B. De Schutter, "A distributed version of Han's method for DMPC using local communications only," *Control Engineering and Applied Informatics,* Special Issue on Distributed Control in Networked Systems, vol. 11, no. 3, pp. 6–15, 2009.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: `https://www.dcsc.tudelft.nl`

*This report can also be downloaded via `https://pub.deschutter.info/abs/09_025.html`

A distributed version of Han's method for DMPC using local communications only

Dang Doan * **Tamás Keviczky** * **Ion Necoara** **
Moritz Diehl *** **Bart De Schutter** *

* *Delft University of Technology, Delft, The Netherlands (e-mail: {m.d.doan,t.keviczky}@tudelft.nl, b@deschutter.info)*
** *Automatic Control and Systems Engineering Department, Politehnica University of Bucharest, Bucharest, Romania (e-mail: i.necoara@ics.pub.ro)*
*** *K.U.Leuven, Heverlee, Belgium (e-mail: moritz.diehl@esat.kuleuven.be)*

Abstract: The study of Distributed Model Predictive Control (DMPC) for dynamically coupled linear systems has so far typically focused on situations where coupling constraints between subsystems are absent. In order to address the presence of convex coupling constraints, we present a distributed version of Han's parallel algorithm for a class of convex programs. The distributed algorithm relies on local iterative updates only, instead of system-wide information exchange as in Han's parallel algorithm. The new algorithm then provides the basis for a distributed MPC method that is applicable to sparsely coupled linear dynamical systems with coupled linear constraints. Convergence to the global optimum, recursive feasibility, and stability are established using only local communications between the subsystems.

Keywords: distributed optimization, model predictive control, large-scale systems, dual decomposition, decentralized and cooperative control

1. INTRODUCTION

Model predictive control (MPC) is the most successful advanced control technology implemented in industry due to its ability to handle complex systems with hard input and state constraints (Maciejowski, 2002; Mayne et al., 2000; García et al., 1989). The essence of MPC is to determine a control profile that optimizes a cost criterion over a prediction window and then to apply this control profile until new process measurements become available. Then the whole procedure is repeated and feedback is incorporated by using the measurements to update the optimization problem for the next step.

For control of large-scale networked systems, *centralized* MPC is considered impractical, inflexible, and unsuitable due to information exchange requirements and computational aspects. The subsystems in the network may belong to different authorities that prevent sending all necessary information to one processing center. Moreover, the optimization problem yielded by centralized MPC can be excessively large for real-time computation. In order to deal with these limitations, *distributed model predictive control* (DMPC) has been proposed for control of such large-scale systems, by decomposing the overall system into small subsystems (Jia and Krogh, 2001; Camponogara et al., 2002; Rawlings and Stewart, 2008). The subsystems employ distinct MPC controllers that only solve local optimization problems, use local information from neighboring subsystems, and collaborate to achieve globally attractive solutions.

Approaches to DMPC design differ from each other in the problem setup. For systems with decoupled dynamics Dunbar and Murray (2006) proposed a DMPC scheme focusing on multiple vehicles with coupled cost functions, and utilizing predicted trajectories of the neighbors in each subsystem's optimization. A DMPC scheme with a sufficient stability test for dynamically decoupled systems was proposed by Keviczky et al. (2006), in which each subsystem optimizes also over the behaviors of its neighbors. Richards and How (2007) proposed a robust DMPC method for decoupled systems with coupled constraints, based on constraint tightening and a serial solution approach. For systems with coupled dynamics and decoupled constraints Venkat et al. (2008) proposed a distributed MPC scheme, based on a Jacobi algorithm that deals with the primal problem, using a convex combination of new and old solutions. Other research related to the DMPC field is reported by Jia and Krogh (2002); Du et al. (2001); Li et al. (2005); Camponogara and Talukdar (2007); Mercangoz and Doyle III (2007); Alessio and Bemporad (2007, 2008); Necoara et al. (2008). A recent survey on DMPC is (Scattolini, 2009).

The DMPC algorithm proposed in this paper is able to handle linear time-invariant dynamics with linear dynamical couplings, and the presence of coupled linear constraints. Each local controller will only need to communicate with its direct neighbors (which will be a very limited number, depending on the sparsity of the network) to exchange predictions, which are iteratively updated by the local controllers. The algorithm can be implemented using

only local communications. In order to simplify our exposition, the proposed algorithm relies on an MPC framework with a zero terminal point constraint for achieving global feasibility and stability. However, the proposed algorithm can also be extended to other MPC framework in which the optimization problems have convex cost functions and subject to convex constraints, provided that the local stabilizing terminal controllers are available.

This paper is organized as follows. The problem setup is described in Section 2. A specific system class composed of a finite series interconnection of coupled oscillators is described in Section 3 in order to help illustrate the main aspects of our approach. In Section 4, the centralized MPC problem is formulated and the underlying optimization problem is stated. This problem can be solved using a parallel computing scheme based on Han's method, which is summarized in Section 5. The paper's main contribution is then presented in the form of a distributed algorithm exploiting the structure of the optimization problem for local communications, followed by the proof of its equivalence to Han's algorithm in Section 6. As a consequence of this equivalence, the proposed DMPC scheme using this distributed optimization procedure achieves the global optimum upon convergence and thus inherits feasibility and stability properties from its centralized MPC counterpart. The simulation results in Section 7 illustrate the properties of the DMPC scheme for the example setup with the coupled oscillators. Section 8 concludes the paper and indicates some directions for future research.

2. PROBLEM DESCRIPTION

2.1 Subsystems and their neighborhood

Consider a plant consisting of M subsystems. The dynamics of each subsystem are assumed to be influenced directly by only a *small number* of other subsystems. Moreover, each subsystem i is assumed to have local linear coupled constraints involving only variables from a small number of other subsystems.

Based on the couplings, we define the *neighborhood of subsystem* i, $\mathcal{N}^i$, as the set of indices of subsystem i and the subsystems that have either a direct dynamical or linear constraint coupling with subsystem i. In Figure 1, we demonstrate this with a map where each node stands for one subsystem, the dotted links show constraint couplings and the solid links represent dynamical couplings.

In order to benefit from an increased computational speed when using a distributed algorithm, the couplings in dynamics and constraints between subsystems are assumed to be sparse, or equivalently, the size of each neighborhood $\mathcal{N}^i$ is relatively small in comparison with the total number of subsystems M.

2.2 Coupled subsystem model

We assume that each subsystem can be represented by a discrete-time, linear time-invariant model of the form:

$$x^i_{k+1} = \sum_{j\in\mathcal{N}^i} A^{ij}x^j_k + B^{ij}u^j_k, \tag{1}$$

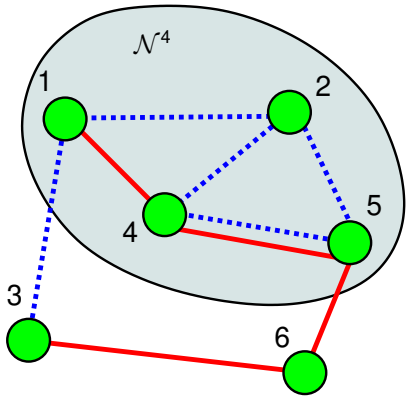

Fig. 1. A map showing the constraint couplings (dotted links) and dynamical couplings (solid links) between subsystems. In this example, $\mathcal{N}^4 = \{4, 1, 2, 5\}$.

where $x^i_k \in \mathbb{R}^{n^i}$ and $u^i_k \in \mathbb{R}^{m^i}$ are the states and control inputs of the i-th subsystem at time step k, respectively.

2.3 Linear coupled constraints

Each subsystem i is assumed to have local linear coupled constraints involving only variables within its neighborhood $\mathcal{N}^i$. Within one prediction period, all constraints that subsystem i is involved in can be written in the following form

$$\sum_{j\in\mathcal{N}^i}\sum_{k=0}^{N-1} D^{ij}_k x^j_k + E^{ij}_k u^j_k = c_{\text{eq}} \tag{2}$$

$$\sum_{j\in\mathcal{N}^i}\sum_{k=0}^{N-1} \bar{D}^{ij}_k x^j_k + \bar{E}^{ij}_k u^j_k \le \bar{c}_{\text{ineq}} \tag{3}$$

in which N is prediction horizon, c_{eq} and $\bar{c}_{\text{ineq}}$ are column vectors, and $D^{ij}_k, E^{ij}_k, \bar{D}^{ij}_k$, and $\bar{E}^{ij}_k$ are matrices with appropriate dimensions.

3. COUPLED OSCILLATORS SETUP FOR ILLUSTRATIVE PURPOSES

Although our approach is developed for systems as general as (1) with sparse dynamical and constraint couplings among the subsystems, we will illustrate the method on an example system composed of a finite series interconnection of coupled oscillators.

Let us now briefly describe this specific problem setup involving coupled oscillators. The system consists of M oscillators that can move only along the vertical axis, and that are coupled by springs that connect each oscillator with its two closest neighbors. An exogenous vertical force will be used as the control input for each oscillator. The setup is shown in Figure 2.

Each oscillator is considered as one subsystem. Let the superscript i denote the index of the oscillators. The continuous-time dynamics equation of oscillator i is then defined as

$$ma^i_t = k_1 p^i_t - f_{\text{s}} v^i_t + k_2(p^{i-1}_t - p^i_t) + k_2(p^{i+1}_t - p^i_t) + F^i_t, \tag{4}$$

where p^i_t, v^i_t, and a^i_t denote the position, velocity, and acceleration of oscillator i at time t, respectively. The

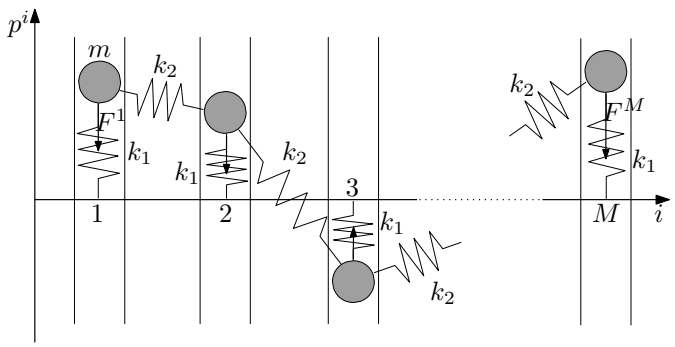

Fig. 2. Setup with coupled oscillators

control force exerted at oscillator i is F_t^i. The system parameters are k_1, k_2, m, and f_s, representing respectively the stiffness of the vertical spring at each oscillator, the stiffness of the springs that connect the oscillators, the mass of each oscillator, and the friction coefficient for movement of the oscillators.

The positions of the subsystems are required to satisfy the coupled constraints:

$$\left| p_t^i - \frac{p_t^{i-1} + p_t^{i+1}}{2} \right| \leq 1, \quad i = 2, ..., M-1, \forall t \tag{5}$$

which means that each oscillator must not deviate too far from the middle of its two closest neighbors.

The control objective is to regulate the system from a non-zero initial state (which is assumed to satisfy the constraints) to the equilibrium.

Based on dynamical couplings and constraint couplings, the neighborhood of each subsystem inside the chain is defined to contain itself and its two closest neighbors: $\mathcal{N}^i = \{i-1, i, i+1\}, i = 2, \ldots, M-1$, while for the two ends $\mathcal{N}^1 = \{1, 2\}$ and $\mathcal{N}^M = \{M, M-1\}$. We define the state vector as $x_t^i = [p_t^i, v_t^i]^T$, and the input as $u_t^i = F_t^i$. The discretized local dynamics with a proper sampling time is represented by the following equations:

$$x_{k+1}^i = A^{ii} x_k^i + A^{i,i-1} x_k^{i-1} + A^{i,i+1} x_k^{i+1} + B^{ii} u_k^i \tag{6}$$

in which $A^{ij} \in \mathbb{R}^{2\times 2}, \forall i, j$ and $B^{ii} \in \mathbb{R}^{2\times 1}, \forall i$.

4. CENTRALIZED MPC PROBLEM

We will formulate the centralized MPC problem for systems of the form (1) using a *terminal point constraint* approach that imposes that all states are steered to 0 at the end of the prediction horizon. Under the conditions that a feasible solution of the centralized MPC problem exists, and that the point with zero states and inputs is in the relative interior of the constraint set, this MPC scheme ensures feasibility and stability, as shown by Mayne et al. (2000) and Keerthi and Gilbert (1988). The algorithm proposed in this paper will also work with any other centralized MPC approach that does not require *terminal point constraint*, given assumption that the subsystems have local stabilizing terminal controllers. We will further assume without loss of generality that the initial time is zero.

4.1 Choice of decision variable

The optimization variable of the centralized MPC problem is constructed as a stacked vector of predicted subsystem control inputs *and* states over the prediction horizon:

$$\mathbf{x} = \Big[\left(u_0^1\right)^T, \ldots, \left(u_0^M\right)^T, \ldots, \left(u_{N-1}^1\right)^T, \ldots, \left(u_{N-1}^M\right)^T, \left(x_1^1\right)^T, \ldots, \left(x_1^M\right)^T, \ldots, \left(x_N^1\right)^T, \ldots, \left(x_N^M\right)^T \Big]^T \tag{7}$$

Recall that n^i and m^i denote the numbers of states and inputs of subsystem i. The number of optimization variables for the centralized problem is thus:

$$n_{\mathbf{x}} = N \sum_{i=1}^{M} m^i + N \sum_{i=1}^{M} n^i \tag{8}$$

4.2 Cost function

The cost function of the centralized MPC problem is assumed to be decoupled and convex quadratic:

$$J = \sum_{i=1}^{M} \sum_{k=0}^{N-1} \left(\left(u_k^i\right)^T R_i u_k^i + \left(x_{k+1}^i\right)^T Q_i x_{k+1}^i \right) \tag{9}$$

with positive definite weights $R_i, Q_i, \forall i$.

It can be rewritten using the decision variable $\mathbf{x}$ as

$$J = \mathbf{x}^T H \mathbf{x} \tag{10}$$

in which the Hessian H is a block-diagonal, positive definite matrix

$$H = \begin{bmatrix} R & 0 \\ 0 & Q \end{bmatrix} \tag{11}$$

where R and Q are block-diagonal, positive definite weights and are built from R_i and Q_i as follows:

$$R = \operatorname{diag}(\underbrace{\tilde{R}, \ldots, \tilde{R}}_{N \text{ times}}) \quad \text{with } \tilde{R} = \operatorname{diag}(R_1, \ldots, R_M)$$
$$Q = \operatorname{diag}(\underbrace{\tilde{Q}, \ldots, \tilde{Q}}_{N \text{ times}}) \quad \text{with } \tilde{Q} = \operatorname{diag}(Q_1, \ldots, Q_M)$$

Remark 4.1 The positive definiteness assumption on Q_i and R_i and the choice (7) of centralized variable without eliminating state variables will help to compute the inverse of the Hessian easily, by allowing simple inversion of each block on the diagonal of the Hessian.

4.3 Problem formulation

The centralized MPC problem is defined as:

$$\min_{\mathbf{x}} \quad \mathbf{x}^T H \mathbf{x} \tag{12}$$
s.t.
$$x_{k+1}^i = \sum_{j \in \mathcal{N}^i} A^{ij} x_k^j + B^{ij} u_k^j,$$
$$i = 1, \ldots, M, \quad k = 0, \ldots, N-1 \tag{13}$$
$$x_N^i = 0, \quad i = 1, \ldots, M \tag{14}$$
$$\sum_{j \in \mathcal{N}^i} \sum_{k=0}^{N-1} D_k^{ij} x_k^j + E_k^{ij} u_k^j = c_{\text{eq}}, \quad i = 1, \ldots, M \tag{15}$$
$$\sum_{j \in \mathcal{N}^i} \sum_{k=0}^{N-1} \bar{D}_k^{ij} x_k^j + \bar{E}_k^{ij} u_k^j \leq \bar{c}_{\text{ineq}}, \quad i = 1, \ldots, M \tag{16}$$

Remark 4.2 For the coupled oscillators example, the centralized optimization problem at each sampling step has $3MN$ degrees of freedom, subject to $2M(N+1)$ scalar linear equality constraints and $2MN + 2(M-2)(N-1)$ scalar linear inequality constraints. Each equality or inequality constraint involves only a few variables, with at most 8 variables appearing in each of them.

4.4 Centralized optimization problem

We can rewrite the problem (12)–(16) in a compact form as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^T H \mathbf{x} \\ \text{s.t.} \quad & a_l^T \mathbf{x} = b_l, \quad l = 1, \ldots, n_{\text{eq}} \\ & a_l^T \mathbf{x} \leq b_l, \quad l = n_{\text{eq}}+1, \ldots, s \end{aligned} \tag{17}$$

with $s = n_{\text{eq}} + n_{\text{ineq}}$.

Remark 4.3 Note that all b_l are scalars and all a_l are vectors. For the coupled oscillators example, we have $n_{\text{eq}} = 2M(N+1)$, $n_{\text{ineq}} = 4MN - 2M - 4N + 4$, and $b_l = 0, l = 1, \ldots, n_{\text{eq}}$. Due to sparse couplings between subsystems, a_l has very few non-zero elements. Each parameter a_l contains at most 8 non-zero elements (out of $3MN$ elements). Recall that H is block-diagonal with positive definite blocks of size $n^i \times n^i$, where for the coupled oscillators example all $n^i = 2$.

In the following sections, Han's parallel algorithm for a class of convex programs will be described and a distributed version exploiting the problem structure will be developed and applied to this optimization problem.

5. HAN'S ALGORITHM

First, we summarize the main elements of Han's method (Han and Lou, 1988) for a class of convex programs, followed by a simplified version for the case of *definite quadratic programming*. The main idea of Han's algorithm is to solve the dual problem of the centralized optimization using parallel computations in an iterative scheme.

5.1 Han's algorithm for general convex problems

The class of optimization problems tackled by Han's algorithm is the following:

$$\begin{aligned} \min_{\mathbf{x}} \quad & q(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in C \triangleq C_1 \cap \cdots \cap C_s \end{aligned} \tag{18}$$

where $C_1, \cdots, C_s$ are closed convex sets and $C \neq \varnothing$, and where $q(\mathbf{x})$ is uniformly convex and differentiable on $\mathbb{R}^{n_{\mathbf{x}}}$. The function $q(\mathbf{x})$ is uniformly convex if there is a constant $\rho > 0$ such that for any $\mathbf{x}_1,\ \mathbf{x}_2 \in \mathbb{R}^{n_{\mathbf{x}}}$ and for any $\lambda \in (0,1)$

$$q(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \lambda q(\mathbf{x}_1) + (1-\lambda) q(\mathbf{x}_2) - \rho\lambda(1-\lambda)\|\mathbf{x}_1 - \mathbf{x}_2\|^2$$

Algorithm 1. **Han's method for convex programs**
The algorithm is an iterative procedure. We use p as iteration counter of the algorithm. We use a superscript (p) to denote the values of variables computed at iteration p.

Let α be a sufficiently large number [1] and define $\mathbf{y}^{(0)} = \mathbf{y}_1^{(0)} = \cdots = \mathbf{y}_s^{(0)} = 0$, with $\mathbf{y}^{(0)}, \mathbf{y}_l^{(0)} \in \mathbb{R}^{n_{\mathbf{x}}}, l = 1, \ldots, s$, and $\mathbf{x}^{(0)} = \nabla q^* \left(\mathbf{y}^{(0)}\right)$ with q^* being the conjugate function [2] of q. For $p = 1, 2, \ldots,$ we perform the following computations:

1) For $l = 1, \ldots, s$, find $\mathbf{z}_l^{(p)}$ that solves
$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{1}{2}\|\mathbf{z} + \alpha \mathbf{y}_l^{(p-1)} - \mathbf{x}^{(p-1)}\|_2^2 \\ \text{s.t.} \quad & \mathbf{z} \in C_l \end{aligned} \tag{19}$$
2) Assign
$$\mathbf{y}_l^{(p)} = \mathbf{y}_l^{(p-1)} + (1/\alpha)\left(\mathbf{z}_l^{(p)} - \mathbf{x}^{(p-1)}\right) \tag{20}$$
3) Set $\mathbf{y}^{(p)} = \mathbf{y}_1^{(p)} + \cdots + \mathbf{y}_s^{(p)}$
4) Compute
$$\mathbf{x}^{(p)} = \nabla q^* \left(\mathbf{y}^{(p)}\right) \tag{21}$$

Remark 5.1 Han's method essentially solves the dual problem of (18), so that $\mathbf{y}^{(p)}$ converges to the solution of the dual problem:

$$\min_{\mathbf{y}} q^*(\mathbf{y}) - \delta^*(\mathbf{y}|C) \tag{22}$$

in which $\delta(\mathbf{x}|C)$ is the indicator function, which is 0 if $\mathbf{x} \in C$ and ∞ otherwise. The conjugate function of $\delta(\mathbf{x}|C)$ is $\delta^*(\mathbf{y}|C) = \sup_{\mathbf{x} \in C} \mathbf{y}^T \mathbf{x}$. Due to Fenchel's duality theorem (Rockafellar, 1970), a value $\mathbf{y}^*$ is an optimizer of (22) if and only if $\mathbf{x}^* = \nabla q^* (\mathbf{y}^*)$ is the solution of (17).

Remark 5.2 The optimization (19) is to find the projection of $\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}$ on the set C_l, its dual problem is

$$\min_{\mathbf{y}} (\alpha/2)\|\mathbf{y}_l^{(p-1)} - \mathbf{y}\|_2 + {\mathbf{x}^{(p-1)}}^T \mathbf{y} - \delta^*(\mathbf{y}|C_l) \tag{23}$$

The update formula (20) gives $\mathbf{y}_l^{(p)} = \nabla \phi \left(\mathbf{z}_l^{(p)}\right)$, where $\phi(\mathbf{z})$ is the cost function of (19). Fenchel's duality also guarantees that $\mathbf{y}_l^{(p)} = \nabla \phi(\mathbf{z}_l^{(p)})$ is the solution of (23), since $\mathbf{z}_l^{(p)}$ is the solution of (19).

---

[1] α is a design parameter that has to be sufficiently large. With $\alpha \geq s/\rho$ Han's method will converge (Han and Lou, 1988). For positive definite QPs we can choose ρ as one half of the smallest eigenvalue of the Hessian matrix. A smaller α leads to a faster convergence rate, but an α that is too small could lead to convergence problems as will be illustrated in Section 7.

[2] The conjugate function of a function $q(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^{n_{\mathbf{x}}}$ is defined by: $q^*(\mathbf{y}) = \sup_{\mathbf{x} \in \mathbb{R}^{n_{\mathbf{x}}}} \left(\mathbf{y}^T \mathbf{x} - q(\mathbf{x})\right)$. The conjugate function q^* is always convex (Boyd and Vandenberghe, 2004).

The uniform convexity of q is used to derive a contraction inequality:

$$(\mathbf{x}^{(p)} - \mathbf{x}^{(p-1)})^T(\mathbf{y}^{(p)} - \mathbf{y}^{(p-1)}) \leq \frac{s}{2\rho}\|\mathbf{y}_l^{(p)} - \mathbf{y}_l^{(p-1)}\|_2^2 \tag{24}$$

Convergence of $\|\mathbf{y}^{(p)} - \mathbf{y}^{(p-1)}\|_2 \to 0$ and $\|\mathbf{x}^{(p)} - \mathbf{x}^{(p-1)}\|_2 \to 0$ as $p \to \infty$ can be shown using (24) and the implicit meaning of (23).

Han and Lou (1988) also showed that their algorithm converges to the global optimum if $q(\mathbf{x})$ is uniformly convex and differentiable on $R^{n\times}$ and $C_1, \cdots, C_s$ are closed convex sets and $C \triangleq C_1 \cap \cdots \cap C_s \neq \varnothing$.

5.2 Han's algorithm for definite quadratic programs

In case the optimization problem has a positive definite cost function and linear constraints as in (17), the optimization problem (19) and derivative (21) have analytical solutions, and then Han's method becomes simpler. In the following we revise how the analytical solutions of (19) and (21) can be obtained when applying Algorithm 1 to problem (17).

Remark 5.3 The result of simplifying Han's method in this section is slightly different from the original one described in Han and Lou (1988), to correct the minor mistakes we found in that paper.

As in (17), each constraint $\mathbf{x} \in C_l$ is implicitly expressed by a scalar linear equality or inequality constraint. So (19) takes one of following two forms:

$$\begin{aligned} \min \quad & \frac{1}{2}\|\mathbf{z} + \alpha\mathbf{y}_l^{(p-1)} - \mathbf{x}^{(p-1)}\|_2^2 \\ \text{s.t.} \quad & a_l^T\mathbf{z} = b_l \end{aligned} \tag{25}$$

or

$$\begin{aligned} \min \quad & \frac{1}{2}\|\mathbf{z} + \alpha\mathbf{y}_l^{(p-1)} - \mathbf{x}^{(p-1)}\|_2^2 \\ \text{s.t.} \quad & a_l^T\mathbf{z} \leq b_l \end{aligned} \tag{26}$$

First consider (26):

- If $a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)}\right) \leq b_l$, then $\mathbf{z}_l^{(p)} = \mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)}$ is the solution of (26). Substituting this $\mathbf{z}_l^{(p)}$ into (20), leads to the following update of $\mathbf{y}_l^{(p)}$:

$$\begin{aligned} & \mathbf{y}_l^{(p)} = \mathbf{y}_l^{(p-1)} + (1/\alpha)\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)} - \mathbf{x}^{(p-1)}\right) \\ & \Rightarrow \mathbf{y}_l^{(p)} = 0 \end{aligned} \tag{27}$$

- If $a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)}\right) > b_l$, then the constraint is active. The optimization problem (26) is to find the point in the half-space $a_l^T\mathbf{z} \leq b_l$ that minimizes its distance to the point $\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)}$ (which is outside that half-space). The solution is the projection of the point $\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)}$ on the hyperplane $a_l^T\mathbf{z} = b_l$, which is given by the following formula:

$$\mathbf{z}_l^{(p)} = \mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)} - \frac{a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l\right) - b_l}{a_l^Ta_l}a_l \tag{28}$$

Substituting this $\mathbf{z}_l^{(p)}$ into (20), leads to:

$$\begin{aligned} \mathbf{y}_l^{(p)} &= \mathbf{y}_l^{(p-1)} + \\ & \frac{1}{\alpha}\left(-\alpha\mathbf{y}_l^{(p-1)} - \frac{a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)}\right) - b_l}{a_l^Ta_l}a_l\right) \\ &= -\frac{a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}_l^{(p-1)}\right) - b_l}{\alpha a_l^Ta_l}a_l \end{aligned} \tag{29}$$

Then defining $\gamma_l^{(p)} = a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}^{(p-1)}\right) - b_l$ yields

$$\mathbf{y}_l^{(p)} = -\frac{\gamma_l^{(p)}}{\alpha a_l^Ta_l}a_l \tag{30}$$

If we define $\gamma_l^{(p)} = \max\{a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}^{(p-1)}\right) - b_l, 0\}$, then we can use the update formula (30) for both cases.

Similarly, for the minimization under equality constraint (25), we define $\gamma_l^{(p)} = a_l^T\left(\mathbf{x}^{(p-1)} - \alpha\mathbf{y}^{(p-1)}\right) - b_l$ and the update formula (30) gives the result of (20).

Now consider step 4) of Algorithm 1. As shown by Boyd and Vandenberghe (2004), the function $q(\mathbf{x}) = \mathbf{x}^TH\mathbf{x}$ with H being a positive definite matrix, is strongly convex and has the conjugate function:

$$q^*(\mathbf{y}) = \frac{1}{2}\mathbf{y}^TH^{-1}\mathbf{y} \tag{31}$$

$$\Rightarrow \nabla q^*(\mathbf{y}) = H^{-1}\mathbf{y} \tag{32}$$

Consequently, in Han's algorithm for the *definite quadratic program* (17), it is not necessary to compute $\mathbf{z}^{(p)}$, and $\mathbf{y}^{(p)}$ can be eliminated using (30), which leads to the following simplified algorithm:

Algorithm 2. **Han's method for definite quadratic programs**
The optimization problem to be considered is (17). As discussed in Han and Lou (1988), we choose $\alpha = s/\rho$, where $s = n_{\text{eq}} + n_{\text{ineq}}$ is the number of constraints and ρ is one half of the smallest eigenvalue of H.

For each $l = 1, \ldots, s$, compute

$$c_l = \frac{-1}{\alpha a_l^Ta_l}H^{-1}a_l \tag{33}$$

Initialize $\gamma_1^{(0)} = \cdots = \gamma_s^{(0)} = 0$ and $\mathbf{x}^{(0)} = 0$. For $p = 1, 2, \ldots$, we perform the following computations:

1) For each l corresponding to an equality constraint ($l = 1, \ldots, n_{\text{eq}}$), compute $\gamma_l^{(p)} = a_l^T\mathbf{x}^{(p-1)} + \gamma_l^{(p-1)} - b_l$.
   For each l corresponding to an inequality constraint ($l = n_{\text{eq}} + 1, \ldots, s$), compute $\gamma_l^{(p)} = \max\{a_l^T\mathbf{x}^{(p-1)} + \gamma_l^{(p-1)} - b_l, 0\}$;
2) Set

$$\mathbf{x}^{(p)} = \sum_{l=1}^{s}\gamma_l^{(p)}c_l \tag{34}$$

Note that Han's method splits up the computation into s parallel subproblems, where s is the number of constraints. However, although Algorithm 2 is simpler than the original form in Algorithm 1, it still requires a *global update scheme*

and the parallel problems still operate with the full-sized decision vector. Implementing the scheme in a DMPC system, where the goal is to reduce the size of local computations and to rely on local communication between subsystems only, is not straightforward. In the following section, we will exploit the structure of the problem (17), resulting in a distributed algorithm that does not require global communications.

6. DISTRIBUTED ALGORITHM FOR THE CENTRALIZED MPC OPTIMIZATION PROBLEM

For the algorithm presented in this section, we use M local controllers attached to M subsystems. Each controller i then computes $\gamma_l^{(p)}$ with regard to a small set of constraints indexed by l. Subsequently, it performs a local update for its own variables, such that the parallel local update scheme will be equivalent to the global update scheme in Algorithm 2. Note that for the centralized MPC problem (12)–(16), H is block-diagonal, which will be used in this section.

6.1 Initialization of the algorithm

The parameter α is chosen as in Algorithm 2 and stored in the memory of all local controllers.

We also compute s invariant values c_l as in Algorithm 2:

$$c_l = \frac{-1}{\alpha a_l^T a_l} H^{-1} a_l, \quad l = 1, \ldots, s \tag{35}$$

in which each c_l corresponds to one constraint of (17). Since H^{-1} is block-diagonal, c_l is as sparse as the corresponding a_l. We can see that c_l can be computed locally by a local controller with *a priori* knowledge of the parameter a_l and the weighting blocks on the diagonal of H that correspond to the non-zero elements of a_l. Note that H^{-1} is computed easily by inverting each block of H, and has the same structure as H.

We assume that each local controller i knows its local dynamics, and the input and state weights of its neighbors in the cost function. Then each local controller i can compute the c_l associated with its dynamic equality constraints.

6.2 Assign responsibility of each local controller

Each local controller is in charge of updating the variables of its subsystem. Moreover, we also assign to each local controller the responsibility of updating *some* intermediate variables that relate to several equality or inequality constraints in which its subsystem's states or inputs appear. The control designer has to assign each of the s scalar constraints to one of the M local controllers [3] such that the following requirements are satisfied:

- Each constraint is taken care of by one and only one local controller (even for a coupled constraint, there will be only one controller that is responsible).
- A local controller can only be in charge of constraints that involve its own variables.

[3] Note that s is often much larger than M.

Let L_i denote the set of indices l that local controller i is in charge of. The first requirement above can be written compactly as

$$L_i \cap L_j = \emptyset, \forall i, j \tag{36}$$

$$L_1 \cup \cdots \cup L_M = \{1, \ldots, s\} \tag{37}$$

Note that this division is not unique and has to be created according to a procedure that is performed in the initialization phase.

Let us now demonstrate this task with the coupled oscillators example: controller i is in charge of $2(N+1)$ equality constraints (corresponding to local dynamics (13) for x_{k+1}^i and (14) for its terminal state x_N^i), and $2N$ inequality constraints for bounding its inputs. In addition, for $i = 2, \ldots, M-1$ there are $2(N-1)$ additional inequality constraints for the coupled constraints (16).

We also define $L_{\mathcal{N}^i}$ as the set of indices l corresponding to the constraints that are taken care of by subsystem i or by any neighbor of i:

$$L_{\mathcal{N}^i} = \bigcup_{j \in \mathcal{N}^i} L_j \tag{38}$$

If a local controller is in charge of the constraints indexed by ℓ, then it computes c_ℓ using (35) and exchanges these values with its neighbors. Then each local controller i stores $\{c_\ell\}_{\ell \in L_{\mathcal{N}^i}}$ in its memory throughout the optimization process.

6.3 Iterative procedure

The distributed algorithm consists of an iterative procedure running within each sampling interval. At each iteration, four steps are executed: two steps are communications between each local controller and its direct neighbors, and two are computation steps that are performed locally by controllers in parallel. Since feasibility is only guaranteed upon convergence of Han's algorithm, we assume that the sampling time used is large enough such that the algorithm can converge within one sampling interval.

For consistency of notation, in this algorithm p is also used to denote the iteration step. Values of variables obtained at iteration p are denoted with superscript (p).

Definition 6.1. (Index matrix of subsystems). In order to present the algorithm compactly, we introduce the *index matrix of subsystems*: each subsystem i has a square matrix $\mathfrak{I}^i \in R^{n_{\mathbf{x}} \times n_{\mathbf{x}}}$ that is diagonal, with an entry on the diagonal being 1 if it corresponds to the position of a variable of subsystem i in the vector $\mathbf{x}$, and 0 otherwise. In short, $\mathfrak{I}^i$ is a selection matrix such that the multiplication $\mathfrak{I}^i \mathbf{x}$ only retains the variables $u_0^i, \ldots, u_{N-1}^i, x_1^i, \ldots, x_N^i$ of subsystem i in its nonzero entries. We have the following relation:

$$\sum_{i=1}^{M} \mathfrak{I}^i = I \tag{39}$$

Definition 6.2. (Self image). We denote with $\mathbf{x}^{(p)|i} \in \mathbb{R}^{n_{\mathbf{x}}}$ the vector that has the same size as $\mathbf{x}$, containing $u_0^{i,(p)}, \ldots, u_{N-1}^{i,(p)}, x_1^{i,(p)}, \ldots, x_N^{i,(p)}$ (i.e. the values of i's variables computed at iteration p) at the right positions, and zeros for the other entries. This vector is called the *self*

image of $\mathbf{x}^{(p)}$ made by subsystem i. Using the *index matrix* notation, the relation between $\mathbf{x}^{(p)|i}$ and $\mathbf{x}^{(p)}$ is:

$$\mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}^{(p)} \tag{40}$$

Definition 6.3. (Neighborhood image). Extending the concept of *self image*, we denote with $\mathbf{x}^{(p)|\mathcal{N}^i}$ the *neighborhood image* of subsystem i made from $\mathbf{x}$. At step p of the iteration, subsystem i constructs $\mathbf{x}^{(p)|\mathcal{N}^i}$ by putting the values of its neighbors' variables and its own variables to the right positions, and filling in zeros for the remaining slots of $\mathbf{x}$. The *neighborhood image* $\mathbf{x}^{(p)|\mathcal{N}^i}$ satisfies the following relations:

$$\mathbf{x}^{(p)|\mathcal{N}^i} = \sum_{j \in \mathcal{N}^i} \mathbf{x}^{(p)|j} \tag{41}$$

$$\mathbf{x}^{(p)|\mathcal{N}^i} = \left(\sum_{j \in \mathcal{N}^i} \mathfrak{I}^j \right) \mathbf{x}^{(p)} \tag{42}$$

By definition, we also have the following relation between the *self image* and the *neighborhood image* made by the same subsystem:

$$\mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}^{(p)|\mathcal{N}^i} \tag{43}$$

Algorithm 3. **Distributed algorithm for the centralized MPC optimization problem**

Initialize with $p = 0$, $x_k^{i,(0)} = 0, u_k^{i,(0)} = 0, \forall i, k \neq 0$ (this means $\mathbf{x}^{(0)|i} = 0, \forall i$, and the centralized variable $\mathbf{x}^{(0)} = 0$), and $\gamma_l^{(0)} = 0, l = 1, \ldots, s$ (recall that s is the number of constraints of the centralized optimization problem).

Next, for $p = 1, 2, \ldots$, the following steps are executed:

1) **Communications to get the updated main variables**
   Each controller i communicates with its neighbors $j \in \mathcal{N}^i$ to get updated values of their variables, contained in $\mathbf{x}^{(p-1)|j}$. Vice versa, i also sends its updated variables in $\mathbf{x}^{(p-1)|i}$ to its neighbors as requested [4].
   After getting information from the neighbors, controller i constructs the *neighborhood image* $\mathbf{x}^{(p-1)|\mathcal{N}^i}$ using formula (41).
2) **Update intermediate variables γ_l in parallel**
   In this step, the local controllers update γ_l corresponding to each constraint l under their responsibility. More specifically, each local controller i updates γ_l for each $l \in L_i$ in the following manner:
   - If constraint l is an equality constraint ($l = 1, \ldots, n_{\text{eq}}$), then $\gamma_l^{(p)} = a_l^T \mathbf{x}^{(p-1)|\mathcal{N}^i} + \gamma_l^{(p-1)} - b_l$.
   - If constraint l is an inequality constraint ($l = n_{\text{eq}} + 1, \ldots, s$), then $\gamma_l^{(p)} = \max\{a_l^T \mathbf{x}^{(p-1)|\mathcal{N}^i} + \gamma_l^{(p-1)} - b_l, 0\}$.
3) **Communications to get the updated intermediate variables**
   Each local controller i communicates with its neighbors to get updated $\gamma_l^{(p)}$ values that the neighbors just computed in step 2).
4) **Update main variables in parallel**
   Local controller i uses all $\gamma_l^{(p)}$ values that it has (by communications and those computed by itself) to compute an *assumed neighborhood image* of $\mathbf{x}$:

$$\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}} = \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \tag{44}$$

   Note that $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$ has the same structure as $\mathbf{x}^{(p-1)|\mathcal{N}^i}$. However, it is not the exact update of the *neighborhood image*, instead $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$ is only an *assumed neighborhood image*. An interpretation will be given later (see Remark 6.4 below).
   Then controller i selects the values of its variables in $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$ to construct the new *self image*:

$$\mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}} \tag{45}$$

   which contains $u_0^{i,(p)}, \ldots, u_{N-1}^{i,(p)}, x_1^{i,(p)}, \ldots, x_N^{i,(p)}$.
   After updating their variables, each local controller checks the local termination criteria. When all local controllers have converged [5], the algorithm stops and the local control actions are implemented, otherwise the controllers proceed to step 1) to start a new iteration.

Remark 6.4: Interpretation of the assumed neighborhood image

At the end of step 4), each local controller i has an *assumed neighborhood image* $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$ of $\mathbf{x}$ that contains information within its interest (i.e., has non-zero values only corresponding to the variables within its neighborhood). However, controller i knows exactly only its own variables, while the variables of i's neighbors contained in $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$ are the assumption of controller i (since i does not know the interaction between its neighbors and their other neighbors, thus their updates will be different from what i assumes for them). Therefore, i only extracts $u_0^{i,(p)}, \ldots, u_{N-1}^{i,(p)}, x_1^{i,(p)}, \ldots, x_N^{i,(p)}$ from $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$ and throws away the other values. The real *neighborhood image* will be made in the next iteration after i receives updated values of its neighbors. In fact, for the actual implementation of the algorithm, we can combine (44) and (45) since we do not need to use $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$.

Remark 6.5 The equivalence between global and local update schemes will be shown later in Section 6.4 by proving that

$$\mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathbf{x}^{(p)|i} \tag{46}$$

where $\mathbf{x}^{(p)}$ is the centralized variable update resulting from the global update scheme in Algorithm 2, while $\mathbf{x}^{(p)|i}$ are the updates at the end of step 4) in Algorithm 3.

Remark 6.6 Note that in all steps of Algorithm 3, there is no explicit optimization performed by the local controllers. This does not mean that the local DMPC problems are missing. On the contrary, there are in fact

[4] Since $\mathbf{x}^{(p-1)|i}$ only has few non-zero elements, which are $u_0^{i,(p-1)}, \ldots, u_{N-1}^{i,(p-1)}, x_1^{i,(p-1)}, \ldots, x_N^{i,(p-1)}$, to save time in communications only these values need to be transmitted by subsystem i.

[5] Checking the termination criteria in a distributed fashion requires a dedicated logic scheme, the description of which is omitted for brevity.

local optimization problems. However, their solutions can be computed locally by an explicit formula. Indeed, due to the fact that the centralized MPC problem (12)–(16) is a definite quadratic program with linear constraints, local optimization problems have analytical solutions, as shown in Section 5.2.

6.4 Proof of equivalence to Han's algorithm using a global update scheme

In Algorithm 2, at step 2), the centralized variable $\mathbf{x}^{(p)}$ is updated via a global update scheme. In Algorithm 3, by the local update scheme we obtain $\mathbf{x}^{(p)|i}$ for $i = 1, \ldots, M$. The equivalence of these two algorithms is stated in the following proposition:

Proposition 1. Applying Algorithms 2 and 3 to the same problem (17) with the same parameter α, at any iteration p, the following properties hold:

a) $\gamma_l^{(p)}$ are the same in Algorithms 2 and 3, for all $l \in \{1, \ldots, s\}$.
b) $\mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathbf{x}^{(p)|i}$, in which $\mathbf{x}^{(p)}$ is calculated in Algorithm 2 while $\mathbf{x}^{(p)|i}, i = 1, \ldots, M$ are calculated in Algorithm 3.

As a consequence, Algorithm 2 and Algorithm 3 are equivalent.

Proof: The proposition will be proved by induction.

It is clear that properties a) and b) hold for $p = 0$.

Next, we will show that if the two Algorithms 2 and 3 are equivalent at iteration $p-1$, they will be equivalent at iteration p. By induction, this implies that they will be equivalent for all iterations.

Now consider iteration p, and assume that the properties a) and b) hold for all iterations before iteration p.

First, we prove property a). For any l and i such that $l \in L_i$, we have:

$$\begin{aligned} a_l^T \mathbf{x}^{(p-1)} &= a_l^T \sum_{j=1}^{M} \mathfrak{J}^j \mathbf{x}^{(p-1)|j} \qquad (47) \\ &= a_l^T \left(\sum_{j \in \mathcal{N}^i} \mathfrak{J}^j \mathbf{x}^{(p-1)|j} + \sum_{j \notin \mathcal{N}^i} \mathfrak{J}^j \mathbf{x}^{(p-1)|j} \right) \end{aligned}$$

Due to the definition of *neighborhood*, a subsystem outside $\mathcal{N}^i$ does not have any coupled constraints with subsystem i. Therefore, $a_l^T \sum_{j \notin \mathcal{N}^i} \mathfrak{J}^j \mathbf{x}^{(p-1)|j} = 0$, which leads to:

$$a_l^T \mathbf{x}^{(p-1)} = a_l^T \sum_{j \in \mathcal{N}^i} \mathfrak{J}^j \mathbf{x}^{(p-1)|j} = a_l^T \mathbf{x}^{(p-1)|\mathcal{N}^i} \qquad (48)$$

The second equality holds due to (41). Equation (48) guarantees that $\gamma_l^{(p)}$ computed at step 1) of Algorithm 2 and at step 2) of Algorithm 3 are the same.

Now we consider property b), where the main argument is the following: In order to calculate $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$, subsystem i uses all $\gamma_l^{(p)}$ and c_l that involve any variable of i; thus the updates of i's variables in $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$ are equal to the updates of i's variables made by the centralized scheme in $\mathbf{x}^{(p)}$ (in step 4) of Algorithm 2). The vector $\mathbf{x}^{(p)|i}$ only contains values of i's variables selected from $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$. Similarly, the updates made by each other subsystem for its variables are guaranteed to be the same as the results of the centralized update scheme. Making the sum of all $\mathbf{x}^{(p)|i}$ is similar to composing them into one vector, which leads to $\mathbf{x}^{(p)}$.

More specifically, we can express the formula of $\mathbf{x}^{(p)|i}$ computed in Algorithm 3 as

$$\begin{aligned} \mathbf{x}^{(p)|i} = \mathfrak{J}^i \mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i} &= \mathfrak{J}^i \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \\ \Rightarrow \sum_{i=1}^{M} \mathbf{x}^{(p)|i} &= \sum_{i=1}^{M} \mathfrak{J}^i \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \qquad (49) \end{aligned}$$

Note that in the following equations, $\mathbf{x}^{(p)}$ refers to the update of the decision variable computed by (34) in Algorithm 2, which we can express as

$$\mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathfrak{J}^i \mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathfrak{J}^i \sum_{l=1}^{s} \gamma_l^{(p)} c_l \qquad (50)$$

in which the first equality is due to the relation (39), the second equality is from (34).

Recall that c_l has the same structure as a_l, and if $l \notin L_{\mathcal{N}^i}$ then a_l and c_l do not have any non-zero values at the positions associated with variables of subsystem i. Therefore

$$\begin{aligned} \mathfrak{J}^i \sum_{l=1}^{s} \gamma_l^{(p)} c_l &= \mathfrak{J}^i \left(\sum_{l \notin L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l + \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \right) \\ &= \mathfrak{J}^i \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \qquad (51) \end{aligned}$$

The equality (51) shows that (50) and (49) are equivalent, thus proving the equality in property b):

$$\mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathbf{x}^{(p)|i} \qquad (52)$$

□

The equivalence of Algorithms 2 and 3 leads to the following result.

Corollary 2. Problem (17) can be solved using Algorithm 3, which has the same convergence property as Han's algorithm 2 for definite quadratic programs.

This allows us to implement a DMPC scheme using Algorithm 3 that does not need global communications: in the DMPC scheme, no computation using global variables is required; moreover, each local controller only needs to communicate with its direct neighbors and the information to exchange is the updates of their predicted variables.

6.5 Convergence, feasibility, and stability of the DMPC scheme

Convergence, feasibility and stability properties of the DMPC scheme using Algorithm 3 are established by the following propositions:

Proposition 3. Assume that Q_i and R_i are positive definite for $i = 1, \ldots, M$, and (12)–(16) has a feasible solution.

Then Algorithm 3 converges to the centralized solution of (12)–(16) at each sampling step.

Proof: In Han and Lou (1988) it is shown that Han's method is guaranteed to converge to the centralized solution of the convex quadratic program (17) under the conditions that $q(\mathbf{x})$ is uniformly convex and differentiable on $\mathbb{R}^{n\times}$ and (17) has a feasible solution. Due to the assumption on the positive definiteness of Q_i and R_i, and the assumption that (12)–(16) has a feasible solution, such conditions hold for the centralized MPC problem (12)–(16). Hence, Corollary 2 implies that the distributed scheme in Algorithm 3 converges to the centralized solution. □

Proposition 4. Assume that at every sampling step, Algorithm 3 converges. Then the DMPC scheme is recursively feasible and stable.

Proof: By letting Algorithm 3 converge at every sampling step, the centralized solution of (12)–(16) is obtained. Recursive feasibility and stability is guaranteed as a consequence of centralized MPC with terminal point constraint, as shown by Mayne et al. (2000) and Keerthi and Gilbert (1988). □

7. SIMULATION RESULTS

In this section, we present the results from simulating the proposed DMPC scheme on the coupled oscillators setup of Section 3.

From physical laws we have derived the continuous dynamics of the coupled oscillators system (see (4)). Then we have used the first-order Euler forward method to transform the continuous dynamics into discrete dynamics of the form (6). For a given sampling time T_s these discretized dynamics are represented by the following matrices:

$$A^{ij} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \forall j \notin \mathcal{N}^i$$

$$A^{i,i-1} = \begin{bmatrix} 0 & 0 \\ T_\mathrm{s}k_2 & 0 \end{bmatrix}, i = 2, ..., M$$

$$A^{ii} = \begin{bmatrix} 1 & T_\mathrm{s} \\ T_\mathrm{s}(k_1 - 2k_2) & 1 - T_\mathrm{s}f_\mathrm{s} \end{bmatrix}, i = 1, ..., M$$

$$A^{i,i+1} = \begin{bmatrix} 0 & 0 \\ T_\mathrm{s}k_2 & 0 \end{bmatrix}, i = 1, ..., M-1$$

$$B^{ij} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \forall j \neq i$$

$$B^{ii} = \begin{bmatrix} 0 \\ T_\mathrm{s} \end{bmatrix}, i = 1, ..., M$$

The following parameters were used in the simulation example:

$$k_1 = 0.4, \quad k_2 = 0.3$$
$$f_\mathrm{s} = 0.4, \quad T_\mathrm{s} = 0.5, \quad m = 1$$
$$M = 20, \quad N = 15$$
$$Q_i = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \quad R_i = 10$$

The physical limitation of inputs and bounds on states are given as

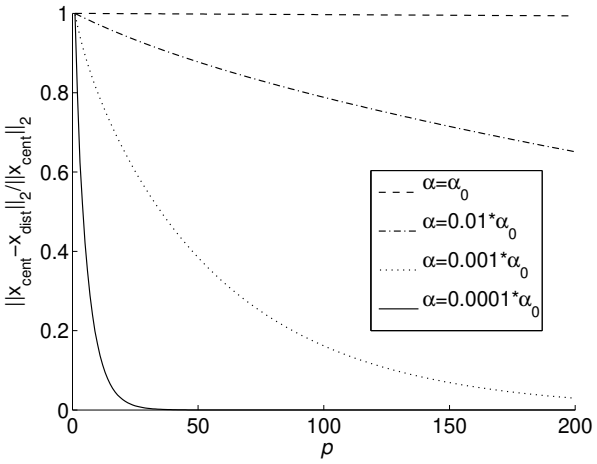

Fig. 3. Normalized norm of difference between the centralized and the distributed solutions versus the iteration step p for different values of α.

$$|F^i| \leq 100, \quad i = 1, \ldots, M$$
$$|p^i| \leq 2, \quad i = 1, \ldots, M$$
$$|v^i| \leq 2, \quad i = 1, \ldots, M$$

Moreover, the discrete-time version of the coupled constraints (5) should also be satisfied.

The control objective is to stabilize the system from a certain initial disturbed state. The DMPC scheme proposed in Section 6 is applied to this regulation problem.

In Figure 3, we show the evolution in the first sampling interval of the normalized 2-norm error between the solution of Algorithm 3 and the centralized optimum for problem (12)–(16) as a function of the iteration step p, for different values of α. Clearly, as more iterations are performed, the error reduces. Although in Han and Lou (1988) the recommended design parameter α is $\alpha_0 = \frac{s}{\rho}$, we have performed simulations with different values of α to show the influence of α on the convergence speed. We see that with the recommended $\alpha = \alpha_0$, the convergence speed is very low, and that when α is smaller, the algorithm converges faster. However, we cannot reduce α too much, there is a lower limit of α so that the algorithm still converges. In fact, we illustrate in Figure 4 that the algorithm diverges for $\alpha = 0.00001\alpha_0$.

Through our simulation experiments, we have observed that besides α, the cost function and constraint matrices also have a notable influence on the convergence speed of the algorithm. Studying and characterizing this influence in detail will be a topic for future research.

8. CONCLUSIONS

This paper has presented a distributed version of Han's method and proposed its use for distributed model predictive control of a class of linear time-invariant system with coupled dynamics and coupled linear constraints. The proposed approach makes use of local communications only between directly connected subsystems, which is especially beneficial in the case of sparse subsystem interconnection topologies. Global optimality is achieved, leading to feasibility and stability. We have illustrated the

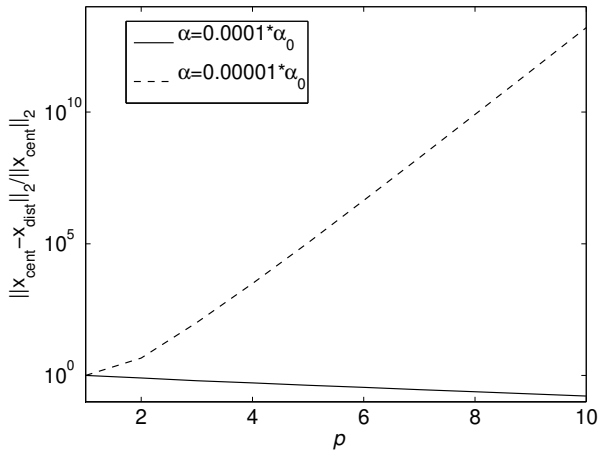

Fig. 4. Normalized norm of difference between the centralized and the distributed solutions versus the iteration step p. For $\alpha = 0.00001\alpha_0$ the algorithm diverges.

proposed optimization scheme for distributed MPC with a simulation example involving coupled oscillators.

Our future research includes a detailed convergence analysis to gain insight into and to characterize the influence of different parameters on the convergence speed, and to determine the conditions for a finite-iteration algorithm. We will perform an extensive comparison between the proposed algorithm and other dual decomposition methods that can be applied for DMPC. Other topics are finding efficient communication schemes for checking the termination criteria, and relaxing the terminal point constraint requirement by finding a method relying on only local communications to compute local terminal stabilizing controllers.

ACKNOWLEDGEMENTS

This research has been supported by the European 7th framework STREP project "Hierarchical and distributed model predictive control (HD-MPC)", contract number INFSO-ICT-223854.

REFERENCES

Alessio, A. and Bemporad, A. (2007). Decentralized model predictive control of constrained linear systems. In *European Control Conference*, 2813–2818. Kos, Greece.

Alessio, A. and Bemporad, A. (2008). Stability conditions for decentralized model predictive control under packet drop communication. In *American Control Conference*, 3577–3582. Seattle, WA.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, Cambridge, MA.

Camponogara, E., Jia, D., Krogh, B., and Talukdar, S. (2002). Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1), 44–52.

Camponogara, E. and Talukdar, S. (2007). Distributed model predictive control: Synchronous and asynchronous computation. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 37(5), 732–745.

Du, X., Xi, Y., and Li, S. (2001). Distributed model predictive control for large-scale systems. In *American Control Conference*, volume 4, 3142–3143. Arlington, VA.

Dunbar, W.B. and Murray, R.M. (2006). Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42, 549–558.

García, C.E., Prett, D.M., and Morari, M. (1989). Model predictive control: Theory and practice — A survey. *Automatica*, 25(3), 335–348.

Han, S.P. and Lou, G. (1988). A parallel algorithm for a class of convex programs. *SIAM Journal on Control and Optimization*, 26(2), 345–355.

Jia, D. and Krogh, B. (2002). Min-max feedback model predictive control for distributed control with communication. In *American Control Conference*, volume 6, 4507–4512. Anchorage, AK.

Jia, D. and Krogh, B. (2001). Distributed model predictive control. In *American Control Conference*, volume 4, 2767–2772. Arlington, VA.

Keerthi, S.S. and Gilbert, E.G. (1988). Optimal infinite-horizon feedback control laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations. *Journal of Optimization Theory and Applications*, 57(2), 265–293.

Keviczky, T., Borrelli, F., and Balas, G.J. (2006). Decentralized receding horizon control for large scale dynamically decoupled systems. *Automatica*, 42, 2105–2115.

Li, S., Zhang, Y., and Zhu, Q. (2005). Nash-optimization enhanced distributed model predictive control applied to the shell benchmark problem. *Information Sciences*, 170(2-4), 329–349.

Maciejowski, J.M. (2002). *Predictive Control with Constraints*. Prentice Hall, Harlow, England.

Mayne, D.Q., Rawlings, J.B., Rao, C.V., and Scokaert, P.O.M. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(7), 789–814.

Mercangoz, M. and Doyle III, F.J. (2007). Distributed model predictive control of an experimental four-tank system. *Journal of Process Control*, 17(3), 297–308.

Necoara, I., Doan, D., and Suykens, J. (2008). Application of the proximal center decomposition method to distributed model predictive control. In *IEEE Conference on Decision and Control*, 2900–2905. Cancun, Mexico.

Rawlings, J.B. and Stewart, B.T. (2008). Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9), 839–845.

Richards, A. and How, J. (2007). Robust distributed model predictive control. *International Journal of Control*, 80(9), 1517–1531.

Rockafellar, R.T. (1970). *Convex Analysis*. Princeton University Press, Princeton, NJ.

Scattolini, R. (2009). Architectures for distributed and hierarchical model predictive control - A review. *Journal of Process Control*, 19(5), 723–731.

Venkat, A., Hiskens, I., Rawlings, J., and Wright, S. (2008). Distributed MPC strategies with application to power system automatic generation control. *IEEE Transactions on Control Systems Technology*, 16(6), 1192–1206.