Efficient Cooperative Distributed MPC using Partial Enumeration ⋆

Gabriele Pannocchia * **Stephen J. Wright** ** **Brett T. Stewart** ***
James B. Rawlings ***

* *Dip. Ing. Chim., Chim. Ind. e Sc. Mat. (DICCISM), Univ. of Pisa, Pisa, Italy (e-mail: g.pannocchia@ing.unipi.it)*
** *Computer Science Dept., Univ. of Wisconsin, Madison, WI, USA (e-mail: swright@cs.wisc.edu)*
*** *Chemical and Biological Engineering Dept., Univ. of Wisconsin, Madison, WI, USA (btstewart@wisc.edu, rawlings@engr.wisc.edu)*

Abstract: We discuss in this paper a novel and efficient implementation of distributed Model Predictive Control (MPC) systems for large-scale systems. The method is based on Partial Enumeration (PE), an approach that allows to compute the (sub)optimal solution of the Quadratic Program associated to the MPC problem by using a solution table that stores only a few most recently optimal active sets. This method is applied to the each local MPC system with significant improvements in terms of computational efficiency, and the original PE algorithm is modified to guarantee robust stability of the overall closed-loop system. We also discuss how input constraints that involve different units, e.g. on the summation of common utility consumption, can be appropriately handled. We illustrate the benefits of proposed method by means a simulated example comprising three units.

Keywords: Distributed MPC, Partial Enumeration, Explicit MPC, QP, Plant-wide Control

1. INTRODUCTION AND MOTIVATIONS

Model predictive control (MPC) is the most successful advanced control technique applied in the process industries (Qin and Badgwell, 2003), which nowadays tend to implement MPC systems in more and more plant units. Since units are often interconnected, it is clear that in some extent different MPCs may interfere, and depending on the steady-state and dynamic coupling of the units, these interactions may limit the overall achievable performance. From a pure theoretical point of view, the desire for optimality should push practitioners to implement a smaller number of (larger) MPC systems that encompass several units. From a practical point of view, however, the use of larger number of (smaller) MPC units may be preferred due to increased flexibility of the overall plant-wide control system, e.g. when one unit requires maintenance. Furthermore, depending on the size of the overall plant, a global centralized MPC system may simply be too large and too demanding in terms of computational resources. For these reasons, researchers are investigating so-called distributed MPC strategies, which aim to interconnect different MPC units with a minimal overhead structure and without increasing the complexity of the online problem solved by each MPC unit (Venkat et al., 2007; Dunbar, 2007; Rawlings and Stewart, 2008; Aske et al., 2008).

In the design of distributed MPC systems, several different "flavors" can be considered. The first one is the fully decentralized structure: each MPC unit optimizes its own objective function and no information regarding the computed input is exchanged among the MPC units. The second one is the so-called "non-cooperative" distributed MPC: the different units exchange their optimal input sequence, i.e. each MPC unit considers the other unit's planned input sequences in its optimal control problem. Both these approaches have no proven stability properties in closed-loop. In decentralized MPC the potential for instability comes first of all by the inherent model error induced by neglecting the interactions between different units. Furthermore in both decentralized and "non-cooperative" MPC structures, instability may arise because the different MPC systems optimize over different and competing objectives. When the closed-loop system is stable, "non-cooperative" MPC leads to a so-called Nash equilibrium point, which may be arbitrarily far away from the centralized optimum, also known as Pareto equilibrium point.

These issues are extensively discussed by Venkat et al. (2006a, 2007), who proposed the so-called "cooperative" distributed MPC architecture. In this distributed MPC system, each local controller optimizes, over its inputs, a common (overall) objective function and shares the computed optimal input sequence with all other controllers. As discussed by Venkat et al. (2006a,b), this scheme guarantees nominal stability, constraint satisfaction, and convergence towards the optimal centralized MPC solution, provided that no constraint involves coupling of inputs from different units.

In a recent paper (Pannocchia et al., 2007), we proposed for large-scale centralized MPC systems a novel online solution method called Partial Enumeration (PE) that allows fast evaluation of (a sub-) optimal solution of the MPC problem. Such method shares some ideas with Explicit MPC (Bemporad et al., 2002; Alessio and Bemporad, 2008; Baotic et al., 2008), which however is applicable only to small dimensional systems. In this paper, we investigate the use of PE for the solution of the local MPC problems with the aim of increasing the size and

⋆ This work was supported by National Science Foundation (Grant CTS-0456694)

complexity of problems that can be addressed efficiently by distributed MPC systems. A second objective of the present paper is to address the issue of coupled input constraints, which may limit the achievable performance of distributed MPC systems (Rawlings and Stewart, 2008).

2. COOPERATIVE MODEL PREDICTIVE CONTROL

2.1 Overall system, input constraints and local subsystems

We consider an overall time-invariant system (plant) in the discrete-time form:

$$x^+ = Ax + Bu, \qquad y = Cx\,, \tag{1}$$

in which $x \in \mathbb{R}^n$ and $x^+ \in \mathbb{R}^n$ are the state at a given time and at the successive time, respectively; $u \in \mathbb{R}^m$ is the input and $y \in \mathbb{R}^p$ is the output. Inputs are assumed to be constrained:

$$Du \le d\,, \tag{2}$$

in which the $d \in \mathbb{R}^q$ has non-negative components.

We assume that the plant is divided into $\mathcal{M}$ sub-systems (units), and each unit i has p_i outputs which are affected, in general, by *all* plant inputs. The objective is to design for each unit a Model Predictive Controller (MPC) that optimizes over a subset of inputs denoted with $u_i \in \mathbb{R}^{m_i}$, $i = 1, \ldots, \mathcal{M}$. The complementary input vector is denoted by $\bar{u}_i \in \mathbb{R}^{m-m_i}$. The subvectors u_i are *not* assumed to be disjoint. We define the selection matrices $T_i \in \mathbb{R}^{m_i \times m}$ and $\bar{T}_i \in \mathbb{R}^{(m-m_i)\times m}$ to be row submatrices of the identity, such that

$$u_i = T_i u, \qquad \bar{u}_i = \bar{T}_i u,$$

and thus

$$u = T_i' u_i + \bar{T}_i' \bar{u}_i\,.$$

Then, the dynamic evolution of each unit $i = 1, \ldots, \mathcal{M}$ can then described in the following form:

$$x_i^+ = A_i x_i + B_i u_i + \bar{B}_i \bar{u}_i, \qquad y_i = C_i x_i\,,$$

in which we distinguish the contribution of the inputs that belong to the $i-$th unit from the contribution of the other inputs. The subset of constraints in (2) that involve only u_i can be written as

$$D_i u_i \le d_i\,, \tag{3}$$

with D_i equal to the non-zero rows of (DT_i') and with d_i the corresponding elements of d. Similarly, the subset of constraints in (2) that involve $\bar{u}_i$ can be written as $\bar{D}_i \bar{u}_i \le \bar{d}_i$ with $\bar{D}_i$ equal to the non-zero rows of $(D\bar{T}_i')$ and with $\bar{d}_i$ the corresponding elements of d.

We consider the following assumptions.

Assumption 1. (Properties of subsystems). For each subsystem $i = 1, \ldots, \mathcal{M}$, the pair (A_i, C_i) is detectable, the pair (A_i, B_i) is stabilizable, and the inequality (3) represents *all* and *only* the constraints that involve elements of input vector u_i. The system from the input $\bar{u}_i$ to y_i is stable.

Remark 2. (Shared inputs). Notice that Assumption 1 admits the possibility that some inputs belong to more than one subsystem. It does require that all constraints involving any element of u_i can be written as constraints that do not involve elements of $\bar{u}_i$. Furthermore it assumes that inputs not belonging to unit i, $\bar{u}_i$, do not excite any unstable mode of A_i.

To clarify this point we present the following example.

Example 3. (Coupled constraints). Consider an overall system with four inputs and the following input constraint matrix and right-hand-side vector:

$$D = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ \hline 0 & 1 & 1 & 0 \end{array}\right], \qquad d = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \hline 1 \end{bmatrix}.$$

The first eight rows define upper and lower bound on each input whereas the last row defines an upper bound on the sum of second and third input. Suppose that we want to design two MPCs, one of which optimizes over (u_1, u_2) whereas the other one optimizes over (u_3, u_4). Since the last constraint involve both u_2 and u_3 that, in principle, belong to different units, in order to satisfy Assumption 1, we need to include u_3 in the set of inputs for Unit 1 and u_2 in the set of inputs of Unit 2. Thus, for Unit 1 we will consider (u_1, u_2, u_3) as inputs, and for Unit 2, we will consider (u_2, u_3, u_4) as inputs.

2.2 Centralized MPC problem

To simplify the notation and given the time invariance of the system, we consider that the current decision time to be $k = 0$. Let input, state and output targets be given, and satisfy:

$$x^s = Ax^s + Bu^s, \qquad y^s = Cx^s\,.$$

Notice that such targets can be either computed by a plant-wide steady-state optimizer or as the combination of $\mathcal{M}$ local steady-state optimizers. For convenience of notation we define:

$$w = x - x^s, \qquad v = u - u^s\,.$$

We consider a finite-horizon sequence of deviation inputs $\mathbf{v} = (v(0), v(1), \ldots, v(N-1))$ and define the overall cost:

$$V(w(0), \mathbf{v}) = \frac{1}{2}\sum_{k=0}^{N-1} w(k)'Qw(k) + v(k)'Rv(k) + \frac{1}{2}w(N)'Pw(N), \qquad \text{s.t. } w^+ = Aw + Bv\,,$$

Before defining the centralized MPC optimal problem, we make the following assumptions.

Assumption 4. (Properties of overall system). The matrices Q and R are positive definite. The matrix P is the given by $P = S_s'\Pi S_s$ with Π solution to the Lyapunov equation:

$$\Pi = A_s'\Pi A_s + S_s'QS_s\,,$$

where (A_s, S_s) come from the real Schur decomposition of A:

$$A = [S_s \;\; S_u]\begin{bmatrix} A_s & A_{su} \\ 0 & A_u \end{bmatrix}\begin{bmatrix} S_s' \\ S_u' \end{bmatrix}\,,$$

and A_s contains all stable eigenvalues of A.

The centralized MPC controller solves the following problem:

$$\mathbb{P}: \qquad \min_{\mathbf{v}} V(w(0), \mathbf{v}) \qquad \text{s.t.}$$
$$Dv \le d - Du^s, \quad S_u'w(N) = 0\,. \tag{4}$$

Remark 5. The constraint $Dv \le d - Du^s$ is equivalent to $Du \le d$. The terminal constraint $S_u'w(N) = 0$ is present only if the system is open-loop unstable (or integrating) and is needed to zero the unstable modes at the end of the horizon N. Furthermore, the cost function term $\frac{1}{2}w(N)'Pw(N)$ represents the infinite horizon cost-to-go when $v(k) = 0$ for $k \ge N$.

2.3 Distributed cooperative MPC subproblems

Let $\bar{\mathbf{v}}_i$ be a known sequence (in deviation variables) of the inputs that do not belong to Unit i, and define the control problem solved by the $i-$th MPC controller as follows:

$$\mathbb{P}_i: \quad \min_{\mathbf{v}} V(w(0),\mathbf{v}) \qquad \text{s.t.}$$
$$Dv \le d - Du^s, \quad S_u' w(N) = 0, \quad \bar{\mathbf{T}}_i \mathbf{v} = \bar{\mathbf{v}}_i\,, \quad (5)$$

in which $\bar{\mathbf{T}}_i \in \mathbb{R}^{(m-m_i)N \times mN}$ is the block diagonal matrix formed with $\bar{T}_i$, $i = 1,\dots,\mathcal{M}$. Similarly, later we use $\mathbf{T}_i \in \mathbb{R}^{m_i N \times mN}$ to denote the block diagonal matrix formed with blocks equal to T_i. We denote with $\tilde{\mathbf{v}}_i$ the solution to (5).

Remark 6. The last equality constraint enforces the inputs that do not belong to Unit i to be equal to the known value $\bar{\mathbf{v}}_i$.

The problem $\mathbb{P}_i$ (5) contains a large number of decision variables that are fixed, namely, all inputs of the other units. We can eliminate these inputs and reformulate this problem as follows. Let the deviation input sequence $\mathbf{v}$ be expressed as

$$\mathbf{v} = \mathbf{T}_i' \mathbf{v}_i + \bar{\mathbf{T}}_i' \bar{\mathbf{v}}_i\,, \quad (6)$$

in which $\mathbf{v}_i = \mathbf{T}_i \mathbf{v}$ is the sequence of inputs that belong to Unit i, and $\bar{\mathbf{v}}_i = \bar{\mathbf{T}}_i \mathbf{v}$ is the sequence of complementary inputs. We can now write the local control problem as:

$$\mathbb{P}_i: \quad \min_{\mathbf{v}_i} V(w(0),\mathbf{v}) \qquad \text{s.t. (6) and}$$
$$D_i v_i \le d_i - D_i u_i^s, \quad S_u' w(N) = 0\,. \quad (7)$$

Remark 7. We note that in (7) we consider only constraints for the inputs of Unit i, and constraints for the other inputs are assumed to be satisfied, i.e. $\bar{D}_i \bar{v}_i \le \bar{d}_i - \bar{D}_i \bar{u}_i^s$. Moreover, the terminal state constraint may contain equations that are not affected by $\mathbf{v}_i$, and such constraints can be eliminated.

2.4 Algorithm and properties

In distributed MPC, each local MPC unit optimizes and communicates its solution with other MPC units, forming a convex combination of the all $\mathcal{M}$ unit solutions to obtain an overall solution. If decision time permits, this procedure is repeated iteratively until convergence or until a specified maximum number of iterations is reached. The distributed MPC algorithm is initiated with an overall input sequence computed at the previous decision time, as follows:

$$\mathbf{v}^0 = (u^*(1) - u^s, \dots, u^*(N-1) - u^s, 0)\,, \quad (8)$$

in which we emphasize that the terms $u^*(\cdot)$ are the components of the (sub)optimal sequence computed at the previous decision time, whereas u^s is the input target at the current decision time.

Remark 8. Such initial sequence is feasible with respect to the input constraint $Dv \le d - Du^s$, and it is also feasible for the terminal constraint $S_u' w(N) = 0$ if it exists, provided the target has not changed from the previous decision time.

We now describe the distributed cooperative MPC algorithm.

Algorithm 1. (Distributed Cooperative MPC). Data: current target (u^s, x^s), deviation state $w(0) = x - x^s$, an overall initial sequence $\mathbf{v}^0$ as in (8). Relative tolerance parameter ρ, maximum number of iterations $l_{\max}$.

(1) (Local MPC problems) Set $l = 1$ and for each MPC unit i repeat the following steps:
   (a) Define $\bar{\mathbf{v}}_i = \bar{\mathbf{T}}_i \mathbf{v}^{l-1}$, solve problem $\mathbb{P}_i$. Let $\mathbf{v}_i$ be the optimal solution to $\mathbb{P}_i$.
   (b) Construct the "complete" solution obtained by Unit i: $\tilde{\mathbf{v}}_i = \mathbf{T}_i' \mathbf{v}_i + \bar{\mathbf{T}}_i' \bar{\mathbf{v}}_i$.

(2) (Convex Step) Define the "overall" solution as combination of the local solutions $\mathbf{v}^l = \sum_{i=1}^{\mathcal{M}} \lambda_i \tilde{\mathbf{v}}_i$, with $\lambda_i > 0$ and $\sum_{i=1}^{\mathcal{M}} \lambda_i = 1$.

(3) (Convergence Test) If $\frac{\|\mathbf{v}^l - \mathbf{v}^{l-1}\|}{1+\|\mathbf{v}^{l-1}\|} < \rho$ or $l = l_{\max}$, set $\mathbf{v}^* = \mathbf{v}^l$ and stop. Otherwise, increase $l \leftarrow l+1$ and go to 1.

It is possible to show that such cooperative MPC algorithm converges to the optimal centralized solution in the limit of infinite iterations. Furthermore, we can establish closed-loop stability for any finite number of iterations l.

3. PARTIAL ENUMERATION

3.1 Introduction

Both the centralized problem $\mathbb{P}$ and each problem $\mathbb{P}_i$ can be written as convex Quadratic Programs, and for small to medium scale systems, the solution can be computed efficiently using either Active Set Method (ASM) or Interior Point Method (IPM) solvers (Rao et al., 1998; Bartlett et al., 2002; Milman and Davison, 2003). However, as the system dimension increases, online solvers cannot provide a solution within an acceptable decision time. In order to compute a (suboptimal) solution for large-scale systems that are currently out of the range of QP solvers, we recently proposed an approach called Partial Enumeration (Pannocchia et al., 2007). In Partial Enumeration (PE) we use a solution table that stores a (small) number of optimal active sets and the associated piecewise linear solution (Bemporad et al., 2002). This approach was applied to large-scale centralized MPC problems in (Pannocchia et al., 2007) with average speed-up factors of 80-200 times compared to conventional QP solvers, and with small closed-loop suboptimality. We review PE here and make appropriate modifications for applying it to the distributed MPC problem $\mathbb{P}_i$.

3.2 PE algorithm and properties

We first consider the centralized MPC problem $\mathbb{P}$ and write it as a parametric QP as follows:

$$\min_{\mathbf{v}} \frac{1}{2}\mathbf{v}'\mathbf{H}\mathbf{v} + \mathbf{v}'\mathbf{G}w(0) + \frac{1}{2}w(0)'\mathbf{P}w(0) \qquad \text{s.t.} \quad (9a)$$
$$\mathbf{D}\mathbf{v} + \mathbf{C}u^s \le \mathbf{d}, \quad \mathbf{E}\mathbf{v} + \mathbf{F}w(0) = 0. \quad (9b)$$

Note that $z = [w(0)', \ u^{s\prime}]'$ is the parameter that changes at each decision time point, while all other terms are constant and omitted in the sake of space.

Given a point $\mathbf{v}^*$, we denote by $(\mathbf{D}_a, \mathbf{C}_a, \mathbf{d}_a)$ the stacked rows of $(\mathbf{D}, \mathbf{C}, \mathbf{d})$ such that $\mathbf{D}_a\mathbf{v}^* + \mathbf{C}_a u^s = \mathbf{d}_a$ (i.e. the active constraints). We also denote with $(\bar{\mathbf{D}}_a, \bar{\mathbf{C}}_a, \bar{\mathbf{d}}_a)$ the complementary stacked rows, i.e. such that $\bar{\mathbf{D}}_a\mathbf{v}^* + \bar{\mathbf{C}}_a u^s < \bar{\mathbf{d}}_a$ (i.e. the inactive constraints). Next, we define:

$$\mathcal{G} = [\mathbf{G} \ 0]\,, \quad \mathcal{A} = \begin{bmatrix}\mathbf{D}_a \\ \mathbf{E}\end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix}0 & \mathbf{C}_a \\ \mathbf{F} & 0\end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix}\mathbf{d}_a \\ 0\end{bmatrix}.$$

In order for $\mathbf{v}^*$ to be optimal for (9), the following first-order optimality KKT conditions must hold:

$$\mathbf{H}\mathbf{v}^* + \mathcal{G}z + \mathcal{A}'\lambda^* = 0, \quad (10a)$$
$$\mathcal{A}\mathbf{v}^* + \mathcal{B}z = \mathbf{b}, \quad (10b)$$
$$\lambda_j^* \ge 0, \qquad j \in \{\text{indices of active inequalities}\}, \quad (10c)$$
$$\bar{\mathbf{D}}_a\mathbf{v}^* + [0 \ \bar{\mathbf{C}}_i]z \le \bar{\mathbf{d}}_a. \quad (10d)$$

We now solve the system (10) to derive $\mathbf{v}^*$ as a linear function of the parameter z and we derive the conditions on z for which the considered active set is optimal. To this aim, several

approaches can be followed, and in this paper we use the so-called Null-Space method.

Let $\mathcal{Z}$ be a full rank matrix such $\mathcal{A}\mathcal{Z} = 0$, and consider the point $\mathbf{v}_0 = \mathcal{A}^+(\mathbf{b} - \mathcal{B}z)$, which $\mathcal{A}^+$ is the pseudo-inverse of $\mathcal{A}$. We can express any point that is feasible for (10b) as $\mathbf{v} = \mathbf{v}_0 + \mathcal{Z}p$ and thus rewrite (10a) as follows:

$$\mathbf{H}\mathcal{Z}p + \mathbf{H}\mathbf{v}_0 + \mathcal{G}z + \mathcal{A}'\lambda^* = 0\,.$$

Next, we multiply (on the left) by $\mathcal{Z}'$ (to eliminate the term $\mathcal{Z}'\mathcal{A}'\lambda^*$) and solve for p to obtain

$$\begin{aligned} p &= \left(\mathcal{H}^{-1}(\mathcal{Z}'\mathbf{H}\mathcal{A}^+\mathcal{B} - \mathcal{Z}'\mathcal{G})\right) z - \mathcal{H}^{-1}(\mathcal{Z}'\mathbf{H}\mathcal{A}^+)\mathbf{b} \\ &= \mathbf{\Gamma}_p z + \gamma_p\,, \end{aligned}$$

with $\mathcal{H} = \mathcal{Z}'\mathbf{H}\mathcal{Z}$. Finally, we compute $\mathbf{v}^*$ as follows:

$$\begin{aligned} \mathbf{v}^* &= \mathbf{v}_0 + \mathcal{Z}p = \mathcal{A}^+(\mathbf{b} - \mathcal{B}z) + \mathcal{Z}(\mathbf{\Gamma}_p z + \gamma_p) \\ &= \mathbf{\Gamma} z + \gamma\,. \end{aligned} \tag{11}$$

Now (10a) can be solved for λ^* as follows:

$$\lambda^* = -(\mathcal{A}')^+(\mathbf{H}\mathbf{v}^* + \mathcal{G}z) = -(\mathcal{A}')^+(\mathbf{H}\mathbf{\Gamma} + \mathcal{G})z - (\mathcal{A}')^+\mathbf{H}\gamma.$$

Finally, we write the Primal and Dual inequalities (10c) and (10d) as follows:

$$\begin{bmatrix} \bar{\mathbf{D}}_a\mathbf{\Gamma} + [0\ \bar{\mathbf{C}}_a] \\ [I\ 0]\,(\mathcal{A}')^+(\mathbf{H}\mathbf{\Gamma} + \mathcal{G}) \end{bmatrix} z \le \begin{bmatrix} \bar{\mathbf{d}}_a - \bar{\mathbf{D}}_a\gamma \\ -[I\ 0]\,(\mathcal{A}')^+\mathbf{H}\gamma \end{bmatrix},$$

or more concisely as

$$\begin{bmatrix} \mathbf{\Psi}_P \\ \mathbf{\Psi}_D \end{bmatrix} z \le \begin{bmatrix} \psi_P \\ \psi_D \end{bmatrix}. \tag{12}$$

Furthermore, by inserting the solution (11) into the objective function of (9), we can write the optimal cost for the current active set as: $V^*(z) = \frac{1}{2}z'V_2z + V_1z + V_0$.

In Partial Enumeration we store $(\mathbf{\Psi}_P, \mathbf{\Psi}_D, \mathbf{\Gamma})$, (ψ_P, ψ_D, γ), and also (V_0, V_1, V_2), for a fixed number of active sets that were optimal in the most recent decision time points. Online we scan the table to check if, for the given parameter z, optimality conditions (12) are satisfied, and in such case compute the optimal solution from (11). However, given the fact that not all possible optimal active sets are stored, it is possible that no entry in the table is optimal. In such cases it is necessary to compute a suboptimal solution for closed-loop control. Nonetheless, a QP solver is called afterwards to compute the optimal solution $\mathbf{v}^*$, and thus derive the matrices/vectors $(\mathbf{\Psi}_P, \mathbf{\Psi}_D, \mathbf{\Gamma})$, (ψ_P, ψ_D, γ), (V_0, V_1, V_2) for the corresponding optimal active sets. Whenever this table entry becomes available, it is inserted into the table. When the table exceeds its maximum size (defined by the user), we delete the entry that was optimal least recently. Thus, the table size is fixed and hence the table lookup process is fast, but the table entries are updated to keep track of new operating conditions for the plant.

In order to compute a suboptimal input sequence when the table does include the optimal active set for the current parameter z several options can be considered. It is important to ensure that the given suboptimal solution guarantees, at least, nominal closed-loop stability, and this can be obtained if we ensure a cost decrease from the previous decision time point. Here, we propose a procedure that allows us to prove robust stability of the closed-loop under PE MPC. The procedure requires two points, the first one which needs to be feasible and its computation is discussed later in Algorithm 2. The second point, instead, is a particular minimizer of (9a) subject to the equality constraint (if present) and all the input inequalities that are active at the target point. More specifically, given the input target u^s, let $(\bar{\mathbf{D}}, \bar{\mathbf{d}})$ denote the subset of rows of $(\mathbf{D}, \mathbf{d})$ such that $\bar{\mathbf{C}}u^s = \bar{\mathbf{d}}$. We define $\hat{\mathbf{v}}$ as the solution to:

$$\min_{\mathbf{v}} V(w(0), \mathbf{v}) \text{ s.t. } \bar{\mathbf{D}}\mathbf{v} = 0,\ \mathbf{E}\mathbf{v} + \mathbf{F}w(0) = 0\,. \tag{13}$$

We can show that $\hat{\mathbf{v}} = \hat{\mathbf{\Gamma}}(u^s)w(0)$, where the dependence of the matrix $\hat{\mathbf{\Gamma}}$ on u^s comes from the fact that u^s defines $(\bar{D}, \bar{d})$.

In the following, we denote by $\mathbf{v}^0 = (u^*(1) - u^s, \ldots, u^*(N-1) - u^s, 0)$ the previous shifted optimal sequence, where the inputs $(u(1)^*, \ldots, u^*(N-1))$ were computed at the previous decision time, while u^s is the current input target.

Algorithm 2. (Partial Enumeration). Data: table with M entries, each comprising the terms $(\mathbf{\Psi}_P, \mathbf{\Psi}_D, \mathbf{\Gamma})$, (ψ_P, ψ_D, γ), (V_0, V_1, V_2); current parameter $z = [w(0)'\ u^{s\prime}]'$; candidate sequence $\mathbf{v}^0$ and its cost V^0 if feasible (otherwise $V^0 = \infty$); maximum table size $M_{\max}$. Output: Input sequence $\mathbf{v}^*$ and updated table. Set $j = 0$, $\tilde{V} = V^0$, $\tilde{\mathbf{v}} = \mathbf{v}^0$.

(1) (Table scanning.) Set $j \leftarrow j+1$. If $j > M$ and $\tilde{\mathbf{v}}$ is feasible go to 4. If $j > M$ and $\tilde{\mathbf{v}}$ is infeasible go to 3. Otherwise, perform the following steps for the $j-$th entry:
   (a) If $\mathbf{\Psi}_P z \le \psi_P$ does not hold, go to 1. Otherwise,
   (b) If $\mathbf{\Psi}_D z \le \psi_D$ holds go to 2. Otherwise,
   (c) Compute the cost V. If $V < \tilde{V}$, set $\tilde{\mathbf{v}} = \mathbf{\Gamma}_u(\mathbf{d}_a - \mathbf{C}_a u^s) + \mathbf{\Gamma}_w w(0)$. Go to 1.

(2) (Optimal solution found.) Compute the optimal solution $\mathbf{v}^*$. Inject the optimal input. Put this entry in first position of the table. Stop.

(3) (Feasibility recovery; arrive at this step only if $\tilde{\mathbf{v}}$ is not feasible.) Solve the LP

$$\begin{aligned} &\min_{\mathbf{q},\mathbf{s}} \mathbf{1}'(\mathbf{q} + \mathbf{s}) \qquad \text{s.t. } \mathbf{D}(\mathbf{q} - \mathbf{s}) \le \mathbf{r}_1, \\ &\mathbf{E}(\mathbf{q} - \mathbf{s}) = \mathbf{r}_2, \quad \mathbf{q} \ge 0, \quad \mathbf{s} \ge 0 \end{aligned}$$

   where $\mathbf{r}_1 = \mathbf{d} - \mathbf{C}u^s - \mathbf{D}\tilde{\mathbf{v}}$, $\mathbf{r}_2 = -\mathbf{F}w(0) - \mathbf{E}\tilde{\mathbf{v}}$, and $\mathbf{1}$ is the vector of ones. Redefine $\tilde{\mathbf{v}} \leftarrow \tilde{\mathbf{v}} + \mathbf{q} - \mathbf{s}$ and compute its cost $\tilde{V}$.

(4) (Solution improvement; $\tilde{\mathbf{v}}$ is feasible at this point.) Evaluate $\hat{\mathbf{v}}$, and compute the largest $t \in [0, 1]$ such that $\mathbf{D}(\hat{\mathbf{v}} - \tilde{\mathbf{v}})t \le \mathbf{d} - \mathbf{C}u^s - \mathbf{D}\tilde{\mathbf{v}}$. Set $\mathbf{v}^* = \tilde{\mathbf{v}}(1-t) + t\hat{\mathbf{v}}$.

(5) (Table update, performed in parallel.) Solve the QP (9), and find the terms $(\mathbf{\Psi}_P, \mathbf{\Psi}_D, \mathbf{\Gamma})$, (ψ_P, ψ_D, γ), (V_0, V_1, V_2) for the optimal active set. Insert this entry in first position of the table, set $M \leftarrow M+1$. If $M = M_{\max}+1$, delete the entry that was optimal least recently, and set $M = M_{\max}$.

Remark 9. The "feasibility recovery" step 3 is required *only if* the system is open-loop unstable *and* either the target changed from the previous decision time or a disturbance occurred. In the nominal case without target change, such step is not performed because $\mathbf{v}^0$ is always feasible. Step 3 is the only "expensive" computation in Algorithm 2 and is justified by closed-loop stability reasons of an open-loop unstable plant. For input bound constraints (i.e., $u_{\min} \le u \le u_{\max}$) further simplifications that allow increased speedup and lower sub optimality can be considered.

It can be shown that PE MPC is nominally stabilizing and robustly stabilizing for sufficiently small measurement noise and additive disturbances (Pannocchia et al., 2009).

3.3 Application of Partial Enumeration to cooperative MPC

Each $\mathbb{P}_i$ in (7) can be written as the following parametric QP:

Table 1. Outputs and inputs of the three units, according to two design schemes: Design A (existing), Design B (optimal).

	Outputs	Inputs	
		Design A	Design B
Unit 1	(y_1, y_2, y_3)	(u_1, u_2, u_3)	$(u_1, u_2, u_3, u_4, u_8)$
Unit 2	(y_4, y_5, y_6)	(u_4, u_5, u_6)	$(u_3, u_4, u_5, u_6, u_8)$
Unit 3	(y_7, y_8)	(u_7, u_8)	(u_3, u_4, u_7, u_8)

$$\min_{\mathbf{v}_i} \frac{1}{2}\mathbf{v}_i'\mathbf{H}_i\mathbf{v}_i + \mathbf{v}_i'\mathcal{G}_i z_i + \frac{1}{2}z_i\mathbf{P}_i z_i \qquad \text{s.t.} \tag{14a}$$

$$\mathbf{D}_i\mathbf{v}_i + \mathbf{C}_i z_i \leq \mathbf{d}, \quad \mathbf{E}_i\mathbf{v}_i + \mathbf{F}_i z_i = 0\,, \tag{14b}$$

in which $z_i = [z',\ \bar{\mathbf{v}}_i']'$ is the parameter augmented with the sequence of inputs that do not belong to Unit i, and $\mathbf{H}_i = \mathbf{T}_i\mathbf{H}\mathbf{T}_i'$, $\mathcal{G}_i = \begin{bmatrix}\mathcal{G} & \mathbf{T}_i\mathbf{H}\bar{\mathbf{T}}_i'\end{bmatrix}$, $\mathbf{P}_i = \begin{bmatrix}\mathbf{P} & 0 & \mathcal{G}'\bar{\mathbf{T}}_i' \\ 0 & 0 & \\ \bar{\mathbf{T}}_i\mathcal{G} & & \bar{\mathbf{T}}_i\mathbf{H}\bar{\mathbf{T}}_i'\end{bmatrix}$, $\mathbf{D}_i = \mathbf{D}\mathbf{T}_i'$, $\mathbf{C}_i = \begin{bmatrix}0 & \mathbf{C} & \mathbf{D}\bar{\mathbf{T}}_i'\end{bmatrix}$, $\mathbf{E}_i = \mathbf{E}\mathbf{T}_i'$, $\mathbf{F}_i = \begin{bmatrix}\mathbf{F} & 0 & \mathbf{E}\bar{\mathbf{T}}_i'\end{bmatrix}$. Notice that several rows of $\mathbf{D}_i$ and $\mathbf{E}_i$ are zero and can be deleted along with the corresponding rows of $\mathbf{C}_i$ and $\mathbf{F}_i$.

We notice that the QP (14) is in the same form of (9), with the main difference that the parameter z is augmented with the known sequence of inputs not belonging to Unit i. Given this increase in dimensionality, a full explicit MPC is impractical even for small systems. On the other hand, PE Algorithm 2 can be readily applied to solve (14). Since PE does not guarantee that each $\mathbb{P}_i$ is solved exactly, no convergence to the optimal centralized solution can be proved. Nonetheless, we can show closed-loop nominal stability and robust stability for sufficiently small disturbances.

4. APPLICATION EXAMPLE

4.1 Overall system and units definition

As an example, we consider a stable system with 8 inputs, 8 outputs and 48 states, whose details are omitted in the sake of space. Each input of the system is constrained in $[-1, 1]$, and the following coupled constraint holds:

$$[0\ 0\ 0\ 1\ 1\ 0\ 0\ 1]\,u \leq 1\,. \tag{15}$$

In the MPC design we use: $Q = I$, $R = I$, and $N = 30$.

We consider that this overall plant is divided in three units. Outputs and inputs of each unit are reported in Table 1, where we emphasize two different design schemes. In Design A, which can be regarded as the existing scheme for this plant, no inputs belong to more than one unit at a time. However, because of the coupled constraint (15), such input partition scheme does not satisfy Assumption 1. Therefore, for such scheme convergence to the optimal centralized solution cannot be guaranteed. For this reason, we consider an alternative input partition scheme (Design B) in which the inputs (u_3, u_4, u_8) belong to all three units.

4.2 Effect of coupled constraints

First of all we investigate about the different convergence properties for the two distributed MPC architectures. We consider that at decision time 10, the input target changes from 0 to $u^s = (0, 0, 0.5, 0.2, 0, 0, 0, 0.3)'$, thus making the coupled constraint active. We report in Figure 1 the closed-loop response of $u_3 + u_4 + u_8$ obtained by three controllers: CMPC is the centralized controller, DMPC-A is the distributed control structure with $l = 1$ iteration based on Design A, DMPC-B is the distributed control structure with $l = 1$ iteration based on Design B. For this study, we solve the optimal control problems exactly, i.e. we do not use Partial Enumeration. We report in Table 2, the suboptimality of DMPC-A and DMPC-B as the number of iterations l increases, defined by the index:

$$S_I = 100\frac{V_{CL} - V_{CL}^*}{V_{CL}}\,,$$

in which V_{CL} is the closed-loop cost for the considered (distributed) controller and V_{CL}^* is the closed-loop cost for the optimal centralized controller. As expected DMPC-B handles the coupled constraint much better than DMPC-A, and as the number of iterations increases, DMPC-B converges to the optimal centralized MPC solution, whereas the suboptimality index for DMPC-A does not go to zero.

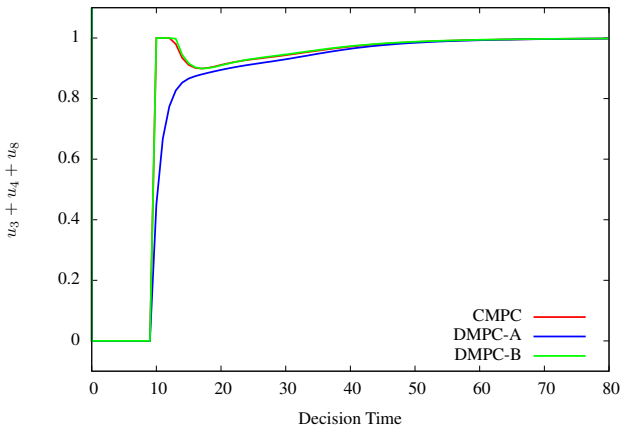

Fig. 1. Effect of coupled input constraints: closed-loop response of $u_3 + u_4 + u_8$ for centralized MPC (CMPC), distributed cooperative MPC based on Design A (DMPC-A), distributed cooperative MPC based on Design B (DMPC-B). Both DMPC-A and DMPC-B make $l = 1$ iteration.

Table 2. Suboptimality of DMPC-A and DMPC-B for different number of iterations l.

Dec. MPC	S_I			
	$l = 1$	$l = 5$	$l = 10$	$l = 50$
DMPC-A	22.9	0.885	0.682	0.673
DMPC-B	2.29	$4.86 \cdot 10^{-2}$	$5.43 \cdot 10^{-4}$	$1.53 \cdot 10^{-8}$

4.3 Comparison of PE-based and QP-based distributed MPC

We now present the results for several decentralized controllers that solve the local MPC problems $\mathbb{P}_i$ either via PE or via an exact (active set) QP solver. We are interested in assessing the suboptimality of each scheme, as well as the computational efficiency quantified by the two indices [1] :

- Average Speed Factor: $A_{SF} = \frac{T_{\text{aver}}^*}{T_{\text{aver}}}$ where T_{aver}^* is the average CPU time required to solve the centralized problem $\mathbb{P}$ via QP solver, and T_{aver} is the average CPU time required to compute the solution using Algorithm 1 (either via PE or via exact QP solver).
- Worst Case Speed Factor: $W_{SF} = \frac{T_{\max}^*}{T_{\max}}$, where $T_{\max}^*$ and $T_{\max}$ are the maximum CPU times for the centralized (QP based) problem $\mathbb{P}$ and for the distributed Algorithm 1 (PE or QP based), respectively.

[1] All computations are performed using GNU Octave on a Pentium-M (1.86 GHz, 1 GB RAM) running Linux.

Table 3. Comparison of suboptimality and computational efficiency for several DMPC-B, based on PE or exact QP solver

	QP based			PE based		
Iter.	S_I	A_{SF}	W_{SF}	S_I	A_{SF}	W_{SF}
$l = 1$	14.7	5.24	12.1	14.8	93.5	285
$l = 5$	0.604	1.25	4.66	0.612	22.0	66.7
$l = 10$	0.0408	0.797	2.37	0.0490	13.9	34.4

We consider a closed-loop simulation of 5000 decision time points, in the presence of random output noise, affecting the state estimate and the target at each decision time, and 14 large target changes. When PE is used, each MPC unit deploys an initially empty table of maximum dimension $M_{\max} = 10$. The results are summarized in Table 3. We can observe, first of all, that as number of iterations l increases, DMPC-B converges to the centralized optimum performance, as indicated by the negligible suboptimality index S_I. Next, we can see that for a given number of iterations l, the suboptimality index obtained by solving the local problems with the QP solver is essentially equal to that obtained with the PE solver. However, the computational requirements using QP and PE solvers are remarkably different. If we compare the distributed controllers using the same number of iterations, DMPC-B based on local PE solvers can compute the solution 17–18 times faster (on average), 14–24 times faster (worst case) than the corresponding DMPC-B based on local QP solvers. In practice, since the time allowed for computation of local solutions and iterations among the distributed controllers may be regarded as fixed, the goal of using local PE solvers is that we can allow more iterations and thus (almost) achieve the centralized optimal performance. If compared with the centralized MPC, most of the computational benefits of using DMPC-B based on PE solvers are achieved with a limited number of iterations, e.g. $l = 5$, which allows one to obtain a suboptimality less than 1% with an average speedup factor of 22 and a worst case speedup factor of 67.

A final remark can be made regarding the possible (apparent) overlap of applicability and scope of Partial Enumeration and distributed MPC, i.e. as alternative means for solving MPC problems in large-scale systems. We want to stress that the main motivation for distributed MPC is organizational rather than computational and, in fact, if the number of iterations l is increased the distributed MPC architecture (based on QP solvers) may be even more time consuming than a centralized MPC architecture (notice the average "speedup" factor less than 1 for DMPC-B based on QP with $l = 10$ iterations). Therefore, distributed MPC should not be considered as a possible competitor of Partial Enumeration centralized MPC which, on the other hand, is motivated by computational issues.

5. CONCLUSIONS

We proposed in this paper an efficient implementation for distributed cooperative Model Predictive Control. The approach is based on Partial Enumeration, that solves the Quadratic Program associated to the MPC problem by means of a small solution table, which includes the most recently optimal active sets. If the optimal solution is not found in the table, a quick suboptimal solution is computed for closed-loop control. In parallel, the optimal active set is evaluated and inserted into the table, possibly deleting the least recently optimal active set. In this way the size of the table is kept small, thus limiting the required time for scanning it. We applied such approach for the solution of "local" MPC problems that are solved in each unit of a distributed MPC system. We also revised the cooperative distributed MPC architecture to optimally handle the case of coupled input constraints. Finally, we presented a simulation example of an 8 input 8 output plant comprising three units in which we achieved relevant speedup factors and negligible suboptimality compared to QP-based MPC.

REFERENCES

Alessio, A. and Bemporad, A. (2008). A survey on explicit model predictive control. In *Proceedings of the International Workshop on Assessment and Future Directions of NMPC*. Pavia, Italy.

Aske, E.M.B., Strand, S., and Skogestad, S. (2008). Coordinator mpc for maximizing plant throughput. *Comp. Chem. Eng.*, 32, 195–204.

Baotic, M., Borrelli, F., Bemporad, A., and Morari, M. (2008). Efficient on-line computation of constrained optimal control. *SIAM J. Control Optim.*, 47, 2470–2489.

Bartlett, R.A., Biegler, L.T., Backstrom, J., and Gopal, V. (2002). Quadratic programming algorithms for large-scale model predictive control. *Journal of Process Control*, 12(7), 775–795.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E.N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38, 3–20.

Dunbar, W.B. (2007). Distributed receding horizon control of dynamically coupled nonlinear systems. *IEEE Trans. Auto. Contr.*, 52, 1249–1263.

Milman, R. and Davison, E.J. (2003). Fast computation of the quadratic programming subproblem in model predictive control. In *Proceedings of the American Control Conference*, 4723–4729.

Pannocchia, G., Rawlings, J.B., and Wright, S.J. (2007). Fast, large-scale model predictive control by partial enumeration. *Automatica*, 43(5), 852–860.

Pannocchia, G., Wright, S.J., and Rawlings, J.B. (2009). On the robust stability of partial enumeration model predictive control. *In preparation*.

Qin, S.J. and Badgwell, T.A. (2003). A survey of industrial model predictive control technology. *Contr. Eng. Pract.*, 11, 733–764.

Rao, C., Wright, S.J., and Rawlings, J.B. (1998). Application of interior-point methods to model predictive control. *J. Optim. Theory Applic.*, 99, 723–757.

Rawlings, J.B. and Stewart, B.T. (2008). Coordinating multiple optimization-based controllers: New opportunities and challanges. *J. Proc. Contr.*, 18, 839–845.

Venkat, A.N., Rawlings, J.B., and Wright, S.J. (2006a). Stability and optimality of distributed, linear MPC. Part 1: state feedback. Technical Report 2006–03, TWMCC, Department of Chemical and Biological Engineering, University of Wisconsin–Madison (Available at http://jbrwww.che.wisc.edu/tech-reports.html).

Venkat, A.N., Rawlings, J.B., and Wright, S.J. (2006b). Stability and optimality of distributed, linear MPC. Part 2: output feedback. Technical Report 2006–04, TWMCC, Department of Chemical and Biological Engineering, University of Wisconsin–Madison (Available at http://jbrwww.che.wisc.edu/tech-reports.html).

Venkat, A.N., Rawlings, J.B., and Wright, S.J. (2007). Distributed model predictive control of large-scale systems. In *Assessment and Future Directions of Nonlinear Model Predictive Control*, 591–605. Springer.