Available online at www.sciencedirect.com
ScienceDirect
Automatica 43 (2007) 852–860
automatica
www.elsevier.com/locate/automatica

Brief paper

Fast, large-scale model predictive control by partial enumeration☆

Gabriele Pannocchia[a], James B. Rawlings[b,*], Stephen J. Wright[c]

[a]Department of Chemical Engineering, Industrial Chemistry and Science of Materials, University of Pisa, Via Diotisalvi, 2, Pisa 56126, Italy
[b]Department of Chemical and Biological Engineering, University of Wisconsin, 1415 Engineering Drive, Madison, WI 53706, USA
[c]Computer Sciences Department, University of Wisconsin, 1210 West Dayton Street, Madison, WI 53706, USA

Received 26 July 2005; received in revised form 28 March 2006; accepted 27 October 2006
Available online 6 March 2007

Abstract

Partial enumeration (PE) is presented as a method for treating large, linear model predictive control applications that are out of reach with available MPC methods. PE uses both a table storage method and online optimization to achieve this goal. Versions of PE are shown to be closed-loop stable. PE is applied to an industrial example with more than 250 states, 32 inputs, and a 25-sample control horizon. The performance is less than 0.01% suboptimal, with average speedup factors in the range of 80–220, and worst-case speedups in the range of 4.9–39.2, compared to an existing MPC method. Small tables with only 25–200 entries were used to obtain this performance, while full enumeration is intractable for this example.

Keywords: Model predictive control; On-line optimization; Large-scale systems

1. Introduction

As reviewed by Mayne, Rawlings, Rao, and Scokaert (2000), most of the early industrial implementations of MPC traded the offline complexity of solving and storing the entire optimal feedback control law $u(x)$ for the online complexity of solving the open-loop optimal control problem given the particular x initial condition of interest at any given sample time. For linear models, the online problem is a quadratic program (QP), and efficient QP solvers allowed practitioners to tackle processes with small to moderate model dimension and control horizon (Bartlett, Biegler, Backstrom, & Gopal, 2002; Cannon, Kouvaritakis, & Rossiter, 2001; Chisci & Zappa, 1999; Lie, Diez, & Hauge, 2005; Rao, Wright, & Rawlings, 1998). Recently, researchers have developed interesting methods for solving and storing the closed-loop feedback law for linear, *constrained* models that work well for problems of low dimension (Bemporad, Morari, Dua, & Pistikopoulos, 2002; Seron, Goodwin, & De Doná, 2003). The information to be stored for looking up the optimal solution grows exponentially in the dimension of the process model and the control horizon, however. Practitioners face the following obstacle to further progress. Small problems are well addressed by both online methods and offline methods. But, as the problem size increases, eventually neither online nor offline MPC computational approaches can deliver the control decision in the available sample time. Practitioners already are starting to address problems of this size.

This paper addresses the class of large problems that cannot be treated with current methods by proposing the partial enumeration (PE) technique. PE eschews optimal control for something that may be slightly suboptimal, but which remains tractable for real-time implementation on large problems. The approach combines online optimization and storage methods to achieve this end. The following features of the control problem must be present for this approach to be effective: (i) The large magnitude deterministic disturbances and setpoint signals change reasonably slowly with time. (ii) The stochastic disturbances, which change quickly with time, have reasonably small magnitude. As the name "partial enumeration" suggests, the active constraint sets that appear with highest frequency are determined, given a reasonable collection of disturbances and

☆ This paper was not present at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Michael Henson under the direction of Editor Frank Allgöwer.

* Corresponding author.
E-mail address: rawlings@engr.wisc.edu (J.B. Rawlings).

0005-1098/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2006.10.019

setpoint changes. The optimal solution for these frequently occurring active constraint sets are computed offline and stored in a table. This table is searched online for the best control. During online operation the optimal solution is expected to be missing from the table at many sample times because the number of entries in the table is small compared to the number of possible optimal active sets. However, the table is adapted online to incorporate new active sets as the need for them arises. When the optimal solution is not found at a sample time, a simpler, suboptimal strategy is used for the current control, but the optimal solution is also found and inserted in the table. At most sample points, the solution is found in the table (giving an optimal control), but by not enforcing strict optimality at every sample, the control can be computed quickly even for large problems.

Under the category of methods that approximate the solution of the MPC QP, Kouvaritakis, Rossiter, and Schuurmans (2000) proposed a method for fast computation of a suboptimal input sequence by means of an inner ellipsoidal approximation to the polyhedral constraint set, and developed heuristic methods to search outside the ellipsoid for a less conservative solution (Kouvaritakis, Cannon, & Rossiter, 2002). A current limitation of this method is that it does not allow the origin of the system to be shifted without solving a large semidefinite program (SDP). This method is therefore not applicable to the problem considered here, because the system is augmented with an integrating disturbance model to achieve offset-free control (see Section 2.1). The estimate of the integrating disturbance shifts the origin of the system at *every* sample time. Rojas, Goodwin, Serón, and Feuer (2004) described a technique for approximately solving the MPC QP by performing an offline SVD of the Hessian of the objective function, and using it for an online change of coordinates. Basis vectors of the SVD are added until the next addition violates an input constraint. This approach assumes that the origin (steady state) is strictly inside the feasible region. This method is not well suited for the applications considered here because the steady state is often constrained, so the origin is usually on the boundary of the feasible region (see Section 4.2).

2. Model predictive control

2.1. Controller formulation

Consider a linear time-invariant system model, augmented with integrating states to remove offset (Pannocchia & Rawlings, 2003):

$$x_{k+1} = Ax_k + Bu_k + B_d d_k,$$

$$d_{k+1} = d_k,$$

$$y_k = Cx_k + C_d d_k, \tag{1}$$

in which $x_k \in \mathbb{R}^n$ is the state, $u_k \in \mathbb{R}^m$ is the input, $y_k \in \mathbb{R}^p$ is the measured output, $d_k \in \mathbb{R}^p$ is the additional integrating state, and the matrices A, B, C, B_d, C_d are fixed matrices with appropriate dimensions. At each sampling time, given the current measured output y_k, estimates of the augmented state are assumed available. Denote these estimates by $\hat{x}_{k|k}$ and $\hat{d}_{k|k}$, respectively, which can be computed e.g. by means of a steady-state Kalman filter (Pannocchia, Rawlings, & Wright, 2006). Denote by $\{\hat{x}_{k+j|k}\}$ and $\{\hat{d}_{k+j|k}\}$ (with $j > 0$) the corresponding predicted sequences obtained using the model (1) for a given input sequence $\{u_{k+j|k}\}$, and stipulate the following "hard" input constraints:

$$Du_k \leqslant d, \tag{2}$$

in which $D \in \mathbb{R}^{q\times m}$ and $d \in \mathbb{R}^q$ are specified by the user. Other types of constraints can be readily included but are omitted for simplicity of presentation.

Assuming that a linear transformation $z = H_z y$ (usually a subvector) of the measured output vector y have known setpoints $\bar{z}$, a target calculation module is used at each sampling time to compute the state and input steady state that drives the controlled variables to their setpoints while respecting the constraints (2). First, a solution of the following QP is attempted:

$$\min_{\bar{u}_k,\bar{x}_k} \frac{1}{2}(\bar{u}_k - \bar{u})' R_s (\bar{u}_k - \bar{u}) \tag{3a}$$

subject to:

$$\bar{x}_k = A\bar{x}_k + B\bar{u}_k + B_d\hat{d}_{k|k}, \tag{3b}$$

$$\bar{z} = H_z(C\bar{x}_k + C_d\hat{d}_{k|k}), \tag{3c}$$

$$D\bar{u}_k \leqslant d, \tag{3d}$$

in which R_s is a positive definite matrix, $\bar{u}$ represents the desired setpoint for the inputs, and $\bar{z}$ represents the desired output setpoints. If (3) is infeasible, the controlled variables z cannot be moved to the given setpoint vector $\bar{z}$ without offset for the current integrating state estimate. For this case, a second QP aimed at minimizing the offset (Muske & Rawlings, 1993; Pannocchia et al., 2006) is solved.

Having calculated the targets, the following MPC subproblem is solved at each decision timepoint. First, define deviation variables w_j and v_j as follows:

$$w_j = \hat{x}_{k+j|k} - \bar{x}_k, \quad v_j = u_{k+j|k} - \bar{u}_k. \tag{4}$$

The MPC optimization problem is then

$$\min_{\{w\}_{j=1}^N,\{v\}_{j=0}^{N-1}} \sum_{j=0}^{N-1} \frac{1}{2}\{w_j' Q w_j + v_j' R v_j\} + \frac{1}{2} w_N' P w_N \tag{5a}$$

subject to:

$$w_{j+1} = Aw_j + Bv_j, \quad j = 0,\ldots,N-1, \tag{5b}$$

$$Dv_j \leqslant d - D_s b_s, \quad j = 0,\ldots,N-1, \tag{5c}$$

in which D_s is defined as $D_s = [D\ \ 0]$, b_s is defined as $b_s = [\bar{u}_k'\ \ \bar{x}_k']'$, R is positive definite, the matrices Q and P are positive semidefinite, and $(A, Q^{1/2})$ is detectable.

Remark 1. In the nominal case, i.e. when the initial state is known and the actual plant satisfies (1) with $d_k = 0$ for all k,

it follows that $\hat{d}_{k|k} = 0$ for all k. Hence, the target b_s remains constant at each decision timepoint (unless the setpoint $\bar{z}$ is changed).

One can use (5b) to eliminate the state variables $w = [w_1' \ \cdots \ w_N']'$ from the formulation. Doing so produces the following strictly convex QP:

$$\min_v \frac{1}{2} v'\mathbf{H}v + \mathbf{g}'v \tag{6a}$$

subject to

$$\mathbf{A}v \geqslant \mathbf{b}, \tag{6b}$$

in which $v = [v_0' \ \cdots \ v_{N-1}']'$; $\mathbf{H}$ (positive definite) and $\mathbf{A}$ are constant matrices (Pannocchia et al., 2006). The vectors $\mathbf{g}$ and $\mathbf{b}$ depend on the parameters b_s and w_0 as follows:

$$\mathbf{g} = \mathbf{G}_w w_0, \tag{7a}$$

$$\mathbf{b} = \bar{\mathbf{b}} + \mathbf{B}_s b_s, \tag{7b}$$

in which $\bar{\mathbf{b}}$ is a constant vector while $\mathbf{G}_w$ and $\mathbf{B}_s$ are constant matrices (Pannocchia et al., 2006). Given the optimal solution v^*, (4) is used to recover the first input u_k, that is,

$$u_k = \bar{u}_k + v_0^*, \tag{8}$$

and u_k is injected into the plant.

2.2. Solving the MPC subproblem

Each MPC subproblem differs from the ones that precede and follow it only in the parameters b_s and w_0. The unique solution to (6) must satisfy the following optimality (KKT) conditions:

$$\mathbf{H}v^* + \mathbf{g} - \mathbf{A}_a'\lambda_a^* = 0, \tag{9a}$$

$$\mathbf{A}_a v^* = \mathbf{b}_a, \tag{9b}$$

$$\mathbf{A}_i v^* \geqslant \mathbf{b}_i, \tag{9c}$$

$$\lambda_a^* \geqslant 0, \tag{9d}$$

for some choice of active constraint indices a, whose complement i denotes the inactive constraint indices.

The options for solving this problem are of three basic types:

(i) Solution of the formulation (6) using active-set techniques for quadratic programming such as `qpsol`.
(ii) Solution of the (larger but more structured) problem (5) using an interior point quadratic programming solver.
(iii) The multi-parametric quadratic programming (mp-QP) approach.

Approach (i) has the advantage of compactness of the formulation; the elimination of the states can reduce the dimensions of the matrices considerably. However, $\mathbf{H}$ and $\mathbf{A}$ are in general not sparse, and since their dimension is proportional to N, the time to solve (6) is usually $O(N^3)$. Active-set methods such as `qpsol` essentially search for the correct active set a by making a sequence of guesses, adding and/or removing an element from a at each iteration. If the inequalities (9c) and (9d) are satisfied by the solution v^* and λ_a^* corresponding to the current a, the algorithm terminates with a solution. Otherwise, the violations of these conditions provide guidance as to which indices should be added to or dropped from a. Milman and Davison (2003a,b) describe an active-set approach based on (6) in which changes to a are made in "blocks," feasibility is not enforced at every iterate, and active-set changes are made preferentially in the early part of the time interval.

Details of approach (ii) are described in Rao et al. (1998). In this technique, the highly structured nature of problem (5) is exploited by using a stagewise ordering of the primal and dual variables. A banded Gaussian elimination algorithm is applied to the systems of linear equations that are solved to obtain the primal–dual step at each iteration.

In approach (iii) (Bemporad et al., 2002), the dependence of the problem on the parameters b_s and w_0 is used to express the solution explicitly in terms of these parameters, and to partition the space occupied by (b_s, w_0) into polyhedral regions within which these expressions are valid. Since the PE scheme is related to this approach, it is summarized here.

Eqs. (9) and (7) give

$$\begin{bmatrix} \mathbf{H} & -\mathbf{A}_a' \\ \mathbf{A}_a & 0 \end{bmatrix} \begin{bmatrix} v^* \\ \lambda_a^* \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{\mathbf{b}}_a \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{B}_{as} \end{bmatrix} b_s + \begin{bmatrix} -\mathbf{G}_w \\ 0 \end{bmatrix} w_0, \tag{10}$$

in which $\bar{\mathbf{b}}_a$ and $\mathbf{B}_{as}$ represent the row subvector/submatrix corresponding to the active set a. Assuming for the present that $\mathbf{A}_a$ has full row rank, it is clear that the solution of (10) depends linearly on w_0 and b_s:

$$v^* = \mathbf{K}_{as} b_s + \mathbf{K}_{aw} w_0 + \mathbf{c}_v, \tag{11a}$$

$$\lambda_a^* = \mathbf{L}_{as} b_s + \mathbf{L}_{aw} w_0 + \mathbf{c}_\lambda, \tag{11b}$$

in which $\mathbf{K}_{as}$, $\mathbf{K}_{aw}$, $\mathbf{L}_{as}$, $\mathbf{L}_{aw}$, $\mathbf{c}_v$, and $\mathbf{c}_\lambda$ are quantities that depend on the active set a and the problem data but not on the parameters b_s and w_0. The region of validity of the solutions in (11) is determined by the remaining optimality conditions (9c) and (9d). By substituting from (11), one can obtain explicit tests involving the parameters as follows:

$$\mathbf{L}_{as} b_s + \mathbf{L}_{aw} w_0 + \mathbf{c}_\lambda \geqslant 0, \tag{12a}$$

$$\mathbf{A}_i \mathbf{K}_{as} b_s + \mathbf{A}_i \mathbf{K}_{aw} w_0 + (\mathbf{A}_i \mathbf{c}_v - \mathbf{b}_i) \geqslant 0. \tag{12b}$$

These formulae motivate a scheme for solving the parametrized QPs that moves most of the computation "offline." In the offline part of the computation, the coefficient matrices and vectors in formulae (11) and (12) are calculated and stored for all sets a that are valid active sets for some choice of parameters (b_s, w_0). In the online computation, the active set a for which (12) holds is identified, for a given particular value of (b_s, w_0). Having found the appropriate a, the optimal v^* and λ_a^* are then computed from (11).

For the offline computation, Bemporad et al. (2002) describe a scheme for traversing the space occupied by (b_s, w_0) to determine all possible active sets a. Tondel, Johansen, and

Bemporad (2003) describe a more efficient scheme for finding all active sets a for which the set of (b_s, w_0) satisfying (12) is non-empty and has full dimension. They also show how degeneracies and redundancies can be removed from the description (12). Johansen and Grancharova (2003) construct a partition of the space occupied by (b_s, w_0) into hypercubes (with faces orthogonal to the principal axes) and devise a feedback law for each hypercube that is optimal to within a specified tolerance. The nature of the tree allows it to be traversed efficiently during the online computation. Given the large number of possible active sets (potentially exponential in the number of inequality constraints), the online part of the computation may be slow if many evaluations of the form (12) must be performed before the correct a is identified.

3. Partial enumeration

3.1. Introduction

From a purely theoretical viewpoint, the complete enumeration strategy described above is unappealing, as it replaces a polynomial-time method for solving each MPC subproblem (i.e. an interior-point method) by a method that is obviously not polynomial. Practically speaking, the offline computation—identification of all regions in the (b_s, w_0) space—may be tractable for SISO problems with relatively short control horizon N, but it quickly becomes intractable as the dimensions (m, n) and control horizon N grow.

The goal of the PE strategy is to make the online computations rapid, producing an input that is (close to) optimal, within the decision time allowed by the system. The goal is to expand the size and complexity of systems for which MPC may be viable, by restricting the possible active sets a that are evaluated in the online computation to those that have arisen most often at recent decision points. Essentially, the history of the system is used to improve the practical efficiency of the control computation. Observations on numerous large-size practical problems indicate that the parameters (b_s, w_0) encountered at the decision points during a "time window" fall within a relatively small number of regions defined by the inequalities (12) over all possible active sets a. Hence, it would appear that the offline computations performed by complete enumeration, which involve a comprehensive partitioning of the (b_s, w_0) space, are largely wasted.

The PE approach has the following key features:

(i) The matrices and vectors in (11a) and (12) are stored for only a small subset of possible active sets a in a table of fixed length.
(ii) The count of how frequently each active set a in the table was optimal during the last T decision points is stored. Given the current (b_s, w_0), the entries in the table are searched in order of decreasing frequency of correctness.
(iii) Suppose now that none of the active sets a in the table passes the tests (12) for the given (b_s, w_0). The following are some practical options to define the control input:
  (a) Solve a simplified subproblem (the same problem with a shorter control horizon).
  (b) Look for the "least suboptimal" a from the table among all feasible ones.
  (c) Fall back on a control decision computed at an earlier decision point.
(iv) Independently of the controller's calculation, the MPC problem (5) is solved to obtain the data for the (11a) and (12) corresponding to the optimal active set a and add it to the table. If the table thereby exceeds its maximum size, the entry that was correct least recently is deleted.
(v) The table is initialized by simulating the control procedure for a given number of decision stages (a "training period"), adding random disturbances in such a way as to force the system into regions of (b_s, w_0) space that are likely to be encountered during actual operation of the system.

The table should be large enough to fit the range of optimal active sets encountered during the current range of operation of the system, that is, to keep the number of "misses" at a reasonable level. When the system transitions to a new operating region, some of the current entries in the table may become irrelevant. There will be a spike in the proportion of misses. Once there has been sufficient turnover in the table, however, one can expect the fraction of misses to stabilize at a lower value.

3.2. Implementation

In order to give a proper description of how the PE MPC solver is implemented, consider the following two alternatives that are used to compute the control input when the optimal active set is not in the current table. The first one is the "restricted" or "reserve" control sequence:

$$v^{\mathrm{res}}=(v_0^{\mathrm{res}}, v_1^{\mathrm{res}}, \ldots, v_{N-2}^{\mathrm{res}}, v_{N-1}^{\mathrm{res}})=(v_1^*, v_2^*, \ldots, v_{N-1}^*, K w_N^*), \tag{13}$$

in which, in this definition, $\{v_j^*\}$ and w_N^* are the o ptimal inputs and terminal state computed at the previous decision timepoint. The second alternative is the "short control horizon" optimal control sequence defined as the optimal solution of (5) for a short horizon $\bar{N} < N$, chosen so that the time required to solve this problem is much shorter than that required to solve the problem with control horizon N. Denote this control sequence as

$$v^{\mathrm{sh}} = (v_0^{\mathrm{sh}}, \ldots, v_{\bar{N}-1}^{\mathrm{sh}}). \tag{14}$$

A formal definition of the proposed PE MPC algorithm can now be given.

Algorithm 1 (*Partial enumeration MPC*). *Data at time k*: *a table with* $L_k \leqslant L_{\max}$ *entries* (*each entry contains the matrices/vectors in the systems* (11a)–(12), *a time stamp of the last time the entry was optimal*, *a counter of how many times the entry was optimal*), "*restricted*" *sequence* v^{res} (*computed at time* $k-1$), *previous target* $b_s^{\mathrm{prev}} = [\bar{u}'_{k-1} \;\; \bar{x}'_{k-1}]'$, *current target* b_s *and deviation state* w_0.

Repeat the following steps:

1. *Search the table entries in decreasing order of optimality rate. If an entry satisfies* (12), *compute the optimal control sequence from* (11a), *define the current control input as in* (8), *and go to Step* 5. *Otherwise*,
2. *If the following condition holds*:

$$\delta = \frac{\|b_s - b_s^{\text{prev}}\|_2}{1 + \|b_s\|_2} \leqslant \delta_{\max}, \tag{15}$$

for a user-specified (*small*) *positive scalar* $\delta_{\max}$, *define the current control input as*

$$u_k = v_0^{\text{res}} + \bar{u}_{k-1}, \tag{16}$$

and go to Step 4. *Otherwise*,
3. *Solve the* "*short control horizon*" *MPC problem and define the current control input as*

$$u_k = v_0^{\text{sh}} + \bar{u}_k. \tag{17}$$

4. *Solve* (5) (*or* (6)) *and compute the matrices/vectors in* (11a)–(12) *for the optimal active set. Add the new entry* (*possibly deleting the oldest entry if the* $L_k = L_{\max}$).
5. *Update time stamp and frequency for the optimal entry. Define* v^{res} *for the next decision timepoint as in* (13).
6. *Increase* $k \leftarrow k + 1$ *and go to Step* 1.

Remark 2. In Step 4, for systems with a large number of states the solution of (6) is to be preferred, while for systems with moderate state dimension and large control horizon the solution of (5) is expected to be more convenient (Rao et al., 1998).

Remark 3. Notice that (16) and (13) imply that the current (fall-back) control input is defined as the corresponding optimal one computed at the previous decision timepoint, i.e. $u_k = u_{k|k-1}$.

If, for the current (b_s, w_0), the optimal active set is not found in the table, then Step 4 is executed: the optimal input sequence is calculated for this (b_s, w_0) and the corresponding information is added to the table, in time (if possible) for the next stage $k+1$. When these computations are not completed prior to the next sampling time, as may happen in practice, several modifications to the algorithm are needed. Since the reserve sequence v^{res} in Step 5 must be adjusted without knowing the results of Step 4, this sequence is left unchanged, except to shift it forward one interval, as in (13). When, at some future timepoint, the computations for the given pair (b_s, w_0) are completed, the corresponding information is added to the table. The variable v^{res} also is updated with the newly available optimal sequence (with the obsolete entries for the earlier stages removed) provided that the reserve sequence was not updated in the interim as a result of finding an optimal sequence in the table. Moreover, it is important to notice that, in (15), b_s^{prev} denotes the target around which the reserve sequence was computed, and $\bar{u}_{k-1}$ in (16) is the corresponding input target. Section 4 provides some experiments in which there are delays in performing the computations in Step 4.

3.3. Properties

The main theoretical properties of the proposed PE MPC algorithm are now presented. In particular, Theorem 4 states that the proposed PE MPC algorithm retains the nominal constrained closed-loop stability of the corresponding "optimal" MPC algorithm. To save space, the proofs are omitted here, but can be found in Pannocchia et al. (2006).

The penalty P in (5) is chosen as the optimal cost-to-go matrix of the following "unconstrained" linear quadratic infinite horizon problem:

$$\min_{w,v} \sum_{j=0}^{\infty} \frac{1}{2}\{w_j' Q w_j + v_j' R v_j\} \tag{18a}$$

subject to:

$$w_{j+1} = A w_j + B v_j, \quad j = 0, 1, 2, \ldots . \tag{18b}$$

With such a choice of P and given the corresponding stabilizing gain matrix

$$K = -(R + B'PB)^{-1} B'PA, \tag{19}$$

the unconstrained "deviation" input and state evolves as

$$v_j = K w_j, \quad w_{j+1} = A w_j + B v_j = A_K w_j, \quad j = 0, 1, 2, \ldots, \tag{20}$$

in which $A_K = A + BK$ is a strictly Hurwitz matrix, and the optimal objective value is $\frac{1}{2} w_0^{\mathrm{T}} P w_0$. Define O_∞ to be the set of target/initial state pairs such that the unconstrained solution (20) satisfies the constraints (5c) at all stages, i.e.

$$O_\infty = \{(b_s, w) \mid Dv_j \leqslant d - D_s b_s \text{ for all } v_j, w_j \text{ satisfying (20) with } w_0 = w\}. \tag{21}$$

The set O_∞ is said to have a *finite representation* if this infinite set of inequalities can be reduced to a finite set.

Remark 4. The formulae (20) can be used to eliminate the v_j and w_j from this definition and write O_∞ as an infinite system of inequalities involving only w and b_s, as follows:

$$O_\infty = \{(b_s, w) \mid D_w A_K^j w \leqslant d - D_s b_s \text{ for all } j = 1, 2, \ldots\}, \tag{22}$$

in which $D_w = DK$. Notice that O_∞ is non-empty since $(b_s, 0) \in O_\infty$ for any "feasible" target b_s, i.e. such that (3d) holds.

Now define the following "parametrized" sets:

$$W(b_s) = \{w \mid (b_s, w) \in O_\infty\}, \tag{23}$$

$$W_N(b_s) = \{w \mid w_j \text{ and } v_j, \text{ optimal solution to (5) with } w_0 = w, \text{ satisfy } w_N \in W(b_s)\}. \tag{24}$$

Notice that $W_N(b_s)$ is the set of initial states such that the optimal state trajectory enters $W(b_s)$ in at most N timesteps.

The first result defines conditions under which $W(b_s)$ has a non-empty interior and a finite representation.

Lemma 1. *Assume that b_s satisfies $D_s b_s < d$, that $\tilde{W}(b_s) = \{w | D_w w \leqslant d - D_s b_s\}$ is bounded, and that A_K has all eigenvalues inside the open unit circle. Then, $W(b_s)$ is non-empty with the origin in its interior and has a finite representation.*

The next result describes a key property of the sets $W_N(b_s)$ and the relationship between these sets and $W(b_s)$.

Lemma 2. *Suppose the assumptions of Lemma* 1 *hold and that P in* (5) *is chosen as the solution of the Riccati equation associated with* (18). *Then*

$$W(b_s) \subseteq W_1(b_s) \subseteq W_2(b_s) \subseteq \cdots. \tag{25}$$

Moreover, $W_N(b_s)$ is positively invariant for the system $\bar{w}_{j+1} = A\bar{w}_j + B\bar{v}_j$ with $\bar{v}_j$ defined as the first component of the control sequence solution of (5).

Conditions under which the initial deviation state w_0 lies in $W_N(b_s)$, for all N sufficiently large are now described.

Theorem 3. *Suppose the assumptions of Lemma* 1 *hold. Assume that for the current deviation state w_0 there are state and input infinite sequences $\{w_j\}$ and $\{v_j\}$ that satisfy* (5b) *and* (5c) (*in which these constraints hold for all j*) *and such that the corresponding objective function in* (18) *is finite. Then, there exists a finite integer N'* (*which depends on w_0*) *such that*

$$w_0 \in W_N(b_s) \quad \text{for all } N \geqslant N'. \tag{26}$$

Finally, a nominal stability property for Algorithm 1 is described.

Theorem 4. *The control input u_k computed from Algorithm* 1 *is feasible with respect to* (2) *for any current deviation state w_0 and target b_s. Moreover, under the assumptions of Lemma* 2, *the outlined procedure is closed-loop nominally stabilizing, that is, $(w_j, v_j) \to 0$ for any $w_0 \in W_N(b_s)$.*

4. Application examples

Two examples are examined to evaluate the proposed PE algorithm and compare it with a commercial active-set solver, `qpsol`. All simulations are performed using Octave on a 1.2 GHz PC, and time is measured with the function `cputime` (more details can be found in Pannocchia et al., 2006). PE solvers are implemented, for backup calculation of a suboptimal input, with $\delta_{\max} = 0.0001$ and short control horizon of $\bar{N} = 3$.

The following performance indices are considered.

- Optimality rate: $O_R := N_{opt}/N_{tot}$, in which N_{opt} is the number of decision timepoints in which the optimal solution was found in the table and N_{tot} is the overall number of decision timepoints.
- Suboptimality index: $S_I := |\Phi - \Phi^*|/\Phi^*$, in which Φ is the achieved closed-loop objective function and Φ^* is the optimal one.
- Average speed factor: $A_{SF} := T^*_{aver}/T_{aver}$, in which T^*_{aver} and T_{aver} are the average times required to compute the optimal and the (sub)optimal input.
- Worst-case speed factor: $W_{SF} := T^*_{max}/T_{max}$, in which T^*_{max} and T_{max} are the maximum times required to compute the optimal and the (sub)optimal input.

4.1. Example #1

The first example is a stable system with $m = 3$ inputs, $p = 2$ outputs and $n = 12$ states, in which the sampling time is 1 s and the normalized inputs must satisfy the constraints: $-\mathbf{1} \leqslant u_k \leqslant \mathbf{1}$. The MPC regulator is designed with the following parameters: $N = 100$, $Q = C'C$, $R = 0.01 I_m$, and thus the possible different active sets are $3^{200} = 2.656 \times 10^{95}$. Several PE solvers are compared: PE1, PE25, PE50, PE100 and PE200 use an active-set table with $N_e = 1$, 25, 50, 100 and 200 entries, respectively. As a comparison another solver is also considered: SH always computes and injects the optimal solution of (6) with a short control horizon of $\bar{N} = 3$.

The different solvers are compared in a 20,000 decision timepoints simulation featuring 20 random step disturbances acting on the states, 20 random setpoint changes and normally distributed output noise. Table 1 provides the performance indices achieved by each solver. From this table it is evident that the use of a short control horizon can cause severe performance degradation while PE allows a fast computation of the optimal solution most of the times, with a small overall performance degradation. Fig. 1 presents the cumulative frequency versus the number of entries scanned in the table by each PE solver. Fig. 2 describes the evolution of the active-set table for PE25 during the simulation, quantified by the following indices: $R_d(k, 0) = D(k, 0)/N_e$, $R_d(k, k-N_e) = D(k, k-N_e)/N_e$, in which $D(k, j)$ is the number of table entries at time k that are not in the table at time j, while N_e is the number of all entries in the table. Fig. 2 shows that the active-set table changes over time to adjust to new disturbances and operating conditions.

The effect of the delay in solving the MPC problem in Step 4 was investigated first by introducing an integer parameter t_{lag} which inflates the time required for solution of the full MPC problem. Specifically, if it is necessary to perform Step 4 at time k, the solution is assumed to become available for insertion in the table at time k_{avail} defined by $k_{avail} = k + t_{lag}$. Fig. 3 presents the results of this study for Example #1, in the case in which the table contains 25 entries. Note that, even for the small number of table entries, the suboptimality index S_I remains close to its optimal value of 0 for t_{lag} up to 50 and within a factor of one at $t_{lag} = 200$. For $t_{lag} = 5000$, the performance is as poor as the short-horizon backup strategy.

In a second x-axis at the top of Fig. 3, the effect of a computational delay is quantified differently. Consider the ratio of the time it takes to solve the full MPC problem to the sample time. A parameter α is defined, with which this ratio is inflated, so a solution computed at time k is available for insertion in

Table 1
Performance indices for the examples

	Example #1				Example #2			
	O_R	S_I	A_{SF}	W_{SF}	O_R	S_I	A_{SF}	W_{SF}
SH	–	6.083	523	240				
PE1	0.888	0.0580	302	230	0.654	6.9×10^{-5}	610	399
PE25	0.890	0.0537	225	46.0	0.752	6.8×10^{-5}	220	39.2
PE50	0.891	0.0537	181	20.9	0.762	6.8×10^{-5}	162	20.1
PE100	0.892	0.0537	135	17.8	0.770	6.8×10^{-5}	121	10.1
PE200	0.894	0.0537	89.6	14.4	0.786	6.8×10^{-5}	83.3	4.79

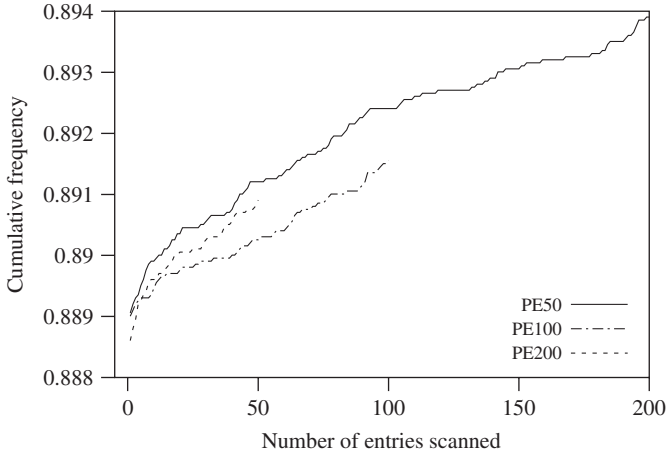


Fig. 1. Example # 1: cumulative frequency of finding an optimal entry versus number of entries scanned. Most of the benefit is achieved with small tables, and then optimality increases only slowly with table size.

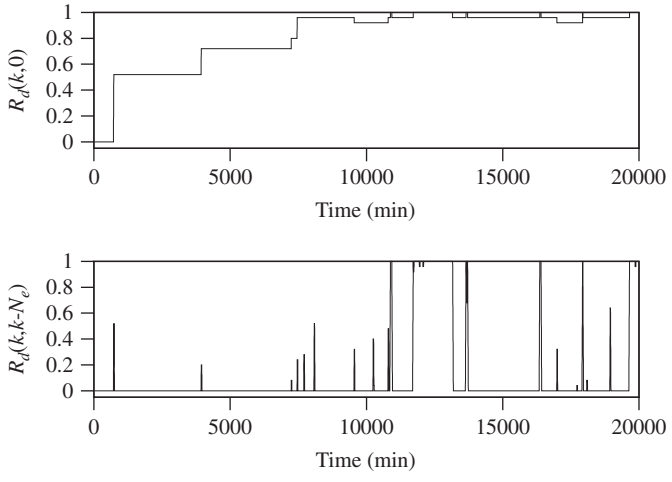


Fig. 2. Example #1: table relative differences for PE25. The top figure shows that the original table is completely replaced as plant operation evolves. The bottom figure shows that the table turns over quickly during some periods and remains fairly constant during other periods.

the table at time $k_{\text{avail}} = k + \lceil \alpha T_{QP}(k)/T_S \rceil$. As displayed in Fig. 3, the control performance loss is not significant until the full MPC calculation runs about 1000 times slower (or the sample time is 1000 times faster).

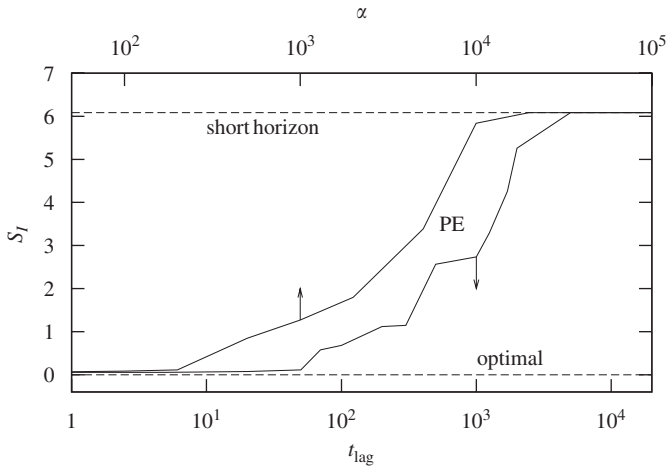


Fig. 3. Example #1: suboptimality versus computational delay. For a wide range of the computational delay, partial enumeration provides almost optimal performance.

4.2. Example # 2: crude distillation unit

The second example is a crude distillation unit model with $m = 32$ inputs, $p = 90$ outputs and $n = 252$ states. Inputs and outputs are normalized, and the inputs must satisfy bound constraints. Only four outputs, representing the quality of the crude distillation unit side products, have desired setpoints, and the MPC regulator is designed with the following parameters: $N = 25$, $Q = C'C$, $R = 20I_m$. The number of possible different active sets is $3^{800} = 4.977 \times 10^{381}$. Response of the "true" plant is simulated by adding, to the nominal model response, unmeasured random step disturbances on the crude composition, on the fuel gas quality and on the steam header pressure, and normally distributed output noise.

Five PE solvers are compared: PE1, PE25, PE50, PE100, PE200 use an active-set table of 1, 25, 50, 100 and 200 entries, respectively. The different solvers are compared on a 5 day (7201 decision timepoints) simulation with 10 random disturbances and five setpoint changes. Table 1 provides the performance indices obtained with each solver. Table 1 shows that the PE solvers provide low suboptimality with remarkable speed factors, especially when small tables are used. This large-scale example is particularly challenging since on

average `qpsol` takes about 26 s to compute the optimal input with a worst-case of about 53 s. In comparison, PE25 requires in the worst-case about 1.3 s, and is therefore definitely applicable for real-time implementation with sampling time of 1 min. Active constraints are the rule in this example. On average, about nine inputs (out of 32) are saturated at their (lower or upper) bound at each sample. The fully unconstrained solution (LQ regulator) is not feasible at *any* sample time in the simulation. The crude distillation unit operation under MPC control is always on the boundary of the feasible region, which is expected to be typical of large-scale, multi-variable plants.

5. Conclusions

Partial enumeration (PE) has been presented as a method for addressing large-scale, linear MPC applications that cannot be treated with current MPC methods. Depending on the problem size and sampling time, online QP methods may be too slow, and offline feedback storage methods cannot even store the full feedback control law. The PE method was shown to have reasonable nominal theoretical properties, such as closed-loop stability. But the important feature of the method is its ability to handle large applications. The industrial distillation large example presented demonstrates suboptimality of less than 0.01%, with average speedup factors in the range of 83–220, and worst-case speedups in the range of 4.8–39.2. Small tables with only 25–200 entries were used to obtain this performance. These small tables contain the optimal solution at least 75% of the time given an industrially relevant set of disturbances, noise, and setpoint changes.

Many extensions of these basic ideas are possible. If one has access to several processors, the storage table can be larger and hashed so that different CPUs search different parts of the table. Several CPUs also enable one to calculate more quickly the optimal controls missing from the current table. One can envision a host of strategies for populating the initial table depending on the user's background and experience, ranging from an empty table in which all table entries come from online operation, to more exhaustive simulation of all disturbance scenarios expected in normal plant operation. In this work two backup strategies are presented for use when the optimal control is not in the current table: a restriction of the previous, optimal trajectory, or a short control horizon solution. One can readily imagine other backup strategies that may prove useful in practice.

Of course, a complete solution for large-scale problems remains a challenge. The PE method pushes back the current boundary between what size application is and is not tractable using MPC.

Acknowledgments

The authors would like to acknowledge Dr. Thomas A. Badgwell of Aspen Technology, Inc. for providing the crude distillation unit model.

References

Bartlett, R. A., Biegler, L. T., Backstrom, J., & 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., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, *38*, 3–20.

Cannon, M., Kouvaritakis, B., & Rossiter, J. A. (2001). Efficient active set optimization in triple mode MPC. *IEEE Transactions on Automatic Control*, *46*(8), 1307–1312.

Chisci, L., & Zappa, G. (1999). Fast QP algorithms for predictive control. In *Proceedings of the 38th IEEE Conference on Decision and Control*. (vol. 5, pp. 4589–4594).

Johansen, T., & Grancharova, A. (2003). Approximate explicit constrained linear model predictive control via orthogonal search tree. *IEEE Transactions on Automatic Control*, *48*(5), 810–815.

Kouvaritakis, B., Cannon, M., & Rossiter, J. A. (2002). Who needs QP for linear MPC anyway. *Automatica*, *38*, 879–884.

Kouvaritakis, B., Rossiter, J. A., & Schuurmans, J. (2000). Efficient robust predictive control. *IEEE Transactions on Automatic Control*, *45*(8), 1545–1549.

Lie, B., Diez, M. D., & Hauge, T. A. (2005). A comparison of implementation strategies for MPC. *Modeling Identification and Control*, *26*(1), 39–50.

Mayne, D. Q., Rawlings, J. B., Rao, C. V., & Scokaert, P. O. M. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, *36*(6), 789–814 ⟨http://www.elsevier.com/locate/automatica⟩.

Milman, R., & Davison, E. J. (2003a). Evaluation of a new algorithm for model predictive control based on non-feasible search directions using premature termination. *Proceedings of the 42nd IEEE conference on decision and control* (pp. 2216–2221).

Milman, R., & Davison, E. J. (2003b). Fast computation of the quadratic programming subproblem in model predictive control. *Proceedings of the American control conference* (pp. 4723–4729).

Muske, K. R., & Rawlings, J. B. (1993). Model predictive control with linear models. *AIChE Journal*, *39*, 262–287.

Pannocchia, G., & Rawlings, J. B. (2003). Disturbance models for offset-free model predictive control. *AIChE Journal*, *49*, 426–437.

Pannocchia, G., Rawlings, J. B., & Wright, S. J. (2006). *The partial enumeration method for model predictive control*: *Algorithm and examples*. TWMCC-2006-01. Available at ⟨ http://jbrwww.che.wisc.edu/jbr-group/tech-reports/twmcc-2006-01.pdf⟩.

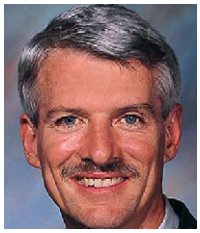
Rao, C. V., Wright, S. J., & Rawlings, J. B. (1998). Application of interior-point methods to model predictive control. *Journal of Optimization Theory and Applications*, *99*, 723–757.

Rojas, O., Goodwin, G. C., Serón, M. M., & Feuer, A. (2004). An SVD based strategy for receding horizon control of input constrained linear systems. *International Journal of Robust and Nonlinear Control*, *14*(13–14), 1207–1226.

Seron, M. M., Goodwin, G. C., & De Doná, J. A. (2003). Characterisation of receding horizon control for constrained linear systems. *Asian Journal of Control*, *5*(2), 271–286.

Tondel, P., Johansen, T. A., & Bemporad, A. (2003). An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*, *39*, 489–497.

Gabriele Pannocchia received the Master degree in Chemical Engineering in 1998 and the Ph.D. degree in Chemical Engineering in 2002, from the University of Pisa (Italy).
He held a Visiting Associate position at the University of Wisconsin, Department of Chemical Engineering, Madison (WI, USA) in 2000/2001, and a Post-Doctoral position at the Department of Chemical Engineering, University of Pisa.
Since 2006, he is an Assistant Professor at the University of Pisa. His research interests include model predictive control, systems theory, model identification, inferential control, process simulation and optimization, numerical optimization algorithms, biomedical applications of automatic control.



James B. Rawlings received the B.S. from the University of Texas in 1979 and the Ph.D. from the University of Wisconsin in 1985, both in Chemical Engineering. He spent one year at the University of Stuttgart as a NATO postdoctoral fellow and then joined the faculty at the University of Texas. He moved to the University of Wisconsin in 1995 and is currently the Paul A. Elfers Professor of Chemical and Biological Engineering and the co-director of the Texas–Wisconsin Modeling and Control Consortium (TWMCC).
His research interests are in the areas of chemical process modeling, monitoring and control, nonlinear model predictive control, moving horizon state estimation, particulate systems modeling, and crystal engineering.

Stephen J. Wright received the B.Sc. (Hons) and Ph.D. degrees from the University of Queensland, Australia, in 1981 and 1984, respectively. After holding positions at North Carolina State University, Argonne National Laboratory, and the University of Chicago, he joined the Computer Sciences Department at the University of Wisconsin-Madison as a professor in 2001. His research interests include theory, algorithms, and applications of computational optimization. Dr. Wright serves as editor-in-chief of Mathematical Programming, Series B, and is the chair-elect of the Mathematical Programming Society. He also serves on the board of trustees of the Society for Industrial and Applied Mathematics.