A parametric branch and bound approach to suboptimal explicit hybrid MPC

Daniel Axehill, Thomas Besselmann, Davide Martino Raimondo and Manfred Morari

Linköping University Post Print

N.B.: When citing this work, cite the original article.

Original Publication:
http://dx.doi.org/10.1016/j.automatica.2013.10.004
Copyright: Elsevier / International Federation of Automatic Control (IFAC)
http://www.ifac-control.org/

Postprint available at: Linköping University Electronic Press
http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-105420
Brief paper

A parametric branch and bound approach to suboptimal explicit hybrid MPC

Daniel Axehill$^{a,d,1}$, Thomas Besselmann$^b,d$, Davide Martino Raimondo$^c,d$, Manfred Morari

$^a$ Linköping University, Division of Automatic Control, 58183 Linköping, Sweden
$^b$ ABB Corporate Research, Automation and Control, 5405 Baden-Dättwil, Switzerland
$^c$ University of Pavia, Dipartimento di Ingegneria Industriale e dell’Informazione, 27100 Pavia, Italy
$^d$ ETH Zurich, Automatic Control Laboratory, 8092 Zurich, Switzerland

ARTICLE INFO

Article history:
Received 10 February 2012
Received in revised form
11 June 2013
Accepted 24 September 2013
Available online 1 November 2013

Keywords:
Suboptimal
Explicit MPC
Hybrid systems
Branch and bound
Stability

ABSTRACT

In this article we present a parametric branch and bound algorithm for computation of optimal and suboptimal solutions to parametric mixed-integer quadratic programs and parametric mixed-integer linear programs. The algorithm returns an optimal or suboptimal parametric solution with the level of suboptimality requested by the user. An interesting application of the proposed parametric branch and bound procedure is suboptimal explicit MPC for hybrid systems, where the introduced user-defined suboptimality tolerance reduces the storage requirements and the online computational effort, or even enables the computation of a suboptimal MPC controller in cases where the computation of the optimal MPC controller would be intractable. Moreover, stability of the system in closed loop with the suboptimal controller can be guaranteed a priori.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Many problems in control require the solution of mixed-integer programs which involve continuous and discrete optimization variables. One example is hybrid systems, where continuous and discrete dynamics interact. In this paper, we consider mixed logical dynamical (MLD) systems. The MLD system class is a common modeling class for hybrid phenomena, which, under some technical assumptions, is equivalent to many other hybrid system classes, Heemels, de Schutter, and Bemporad (2001).

A common technique to control hybrid systems is Model Predictive Control (MPC), Camacho and Bordons (2004). MPC is a flexible control methodology which can take hybrid phenomena as well as hard constraints on states and inputs into account. In MPC, optimization problems are solved repeatedly at each sampling instance, in what is called the receding horizon control strategy. The major drawback of MPC is the computational effort. In order to mitigate this drawback a technique based on parametric programming called Explicit MPC, has been developed, Bank, Gudat, Klatte, Kummer, and Tammer (1982) and Bemporad, Morari, Dua, and Pistikopoulos (2002). Parametric programming allows one to solve the optimal control problems not only for a single, but for a set of states, and thus to shift the computational effort from online to offline. Even though this potentially allows for fast online applications, the complexity of the solution that is sought for might make the process intractable already at the point of offline computation of the explicit solution. This situation motivated the development of a variety of different approximate explicit MPC schemes, see, e.g., Summers, Jones, Lygeros, and Morari (2011).

In the case of hybrid systems and a quadratic cost function, the computation of explicit MPC control laws boils down to solving a parametric mixed-integer quadratic program. Several solution strategies have been proposed, based on the solution of mixed-integer nonlinear programs (MINLP), Dua, Bozinis, and Pistikopoulos (2002), the enumeration of all switching sequences, Kvasnica, Grieder, Baotic, and Morari (2004), or dynamic programming, Borrelli (2003) and Baotic (2005).
In this article, the focus is on parametric mixed-integer quadratic problems. We introduce a parametric branch and bound procedure to search for suboptimal solutions with a guaranteed bound on the suboptimality. The proposed procedure does not require knowledge about the optimal solution. In the MPC framework this allows the user to enforce guarantees on stability and on the maximal absolute or relative performance loss. The motivation for the introduction of suboptimality is that it can lead to significant reductions of online as well as offline computational effort and a reduction of the complexity of the obtained solution. Even though the focus is on quadratic-cost MPC schemes, the presented algorithms can as well be applied to the (simpler) case of piecewise linear cost functions. Another interesting application of the algorithm presented can be found in Axehill and Morari (2010). A preliminary version of this work has been published in the conference paper (Axehill, Besselmann, Raimondo, & Morari, 2011).

2. Problem statement

We consider the class of discrete-time mixed logical dynamical (MLD) systems, which is described by the following relations:

\[ x_{k+1} = Ax_k + Bu_k + B_3 \delta_k + B_2 z_k, \]

subject to \( E \delta_k + E_2 z_k \leq E_i u_k + E_q x_k + e_3 \),

(1b)

with the state \( x_k = [x^T_{t,c} \; x^T_{u,b} \; x^T_{f,b}]^T \in \mathbb{R}^{n_c} \times \{0, 1\}^{n_b} \), and the input \( u_k = [u_c^T \; u_{b}^T \; u_{f}^T]^T \in \mathbb{R}^{n_c} \times \{0, 1\}^{n_b} \) comprising continuous and binary elements, \( Z_2 \in \mathbb{R}^{n_c} \) and \( \delta_3 \in \{0, 1\}^{n_b} \) denote continuous and binary auxiliary variables, and \( A, B, E \) and \( e_3 \) denote real matrices and a real vector of appropriate dimensions, respectively. Constraints on the state and the input of an MLD system such as \( x_k \in X, u_k \in U \) can be incorporated in (1b). It is assumed that both sets are compact and contain the origin in their interior. An MLD system is called completely well-posed if \( x_0 \) and \( z_0 \) are uniquely determined by (1b) for given \( x_0, u_0 \), implying that also \( x_{k+1} \) is uniquely determined.

Consider the following constrained finite-time optimal control (CFTOC) problem for MLD systems, for a given initial state \( x_0 \):

\[
\begin{align*}
\text{minimize} & \quad J(U_k; x_0) \\
\text{subject to} & \quad \text{MLD system dynamics in (1)}, \\
& \quad x_{k+N} \in X_T, \\
& \quad x_{k+N} \in X_T.
\end{align*}
\]

The set \( X_T \subseteq X, 0 \in X_T \), denotes a compact polyhedral terminal set, constraining the state \( X_{k+N} \) at the (finite) final time instant \( N \). In quadratic cost optimal control, the cost function is defined as a quadratic function of the states and inputs within the prediction horizon \( N \),

\[
J(U_k; x_0) = V_T(x_{k+N}) + \sum_{i=0}^{N-1} l(x_{k+i}, u_{k+i})
\]

with \( V_T(x_{k+N}) = x_{k+N}^T P x_{k+N} \) and \( l(x_{k+i}, u_{k+i}) = x_{k+i}^T Q x_{k+i} + u_{k+i}^T R u_{k+i} \), \( U_k = \{u_k, \ldots, u_{k+N-1}\} \) denotes the sequence of control actions, and the weight matrices \( Q, P \in \mathbb{R}^{n_c} \) and \( R \in \mathbb{R}^{n_b} \) are assumed to be positive (semi)definite.

Explicit MPC for MLD systems requires the parametric solution of the CFTOC problem (2) for a range of different states \( x_0 \), i.e., the solution of a multi-parametric mixed-integer quadratic program (mp-MIQP). In order to mitigate complexity issues within the explicit hybrid MPC framework, we are interested in suboptimal solutions to mp-MIQPs. The aim of this paper is a procedure to compute a suboptimal sequence of control laws \( \hat{U}_k(x_0) \) to the CFTOC problem (2), such that the resulting suboptimal cost \( \hat{J}(x_0) \) satisfies

\[
\hat{J}(x_0) - J^*(x_0) \leq \sigma(x_0)
\]

for all \( x_0 \) in \( \hat{X}_T \subseteq X \), where \( J^*(x_0) \) denotes the optimal cost, where the finite positive \( \sigma(x_0) : \mathbb{R}^n \to \mathbb{R}_+ \) denotes a user-defined suboptimality tolerance, and where \( \hat{X}_T \) denotes the set of states for which the problem in (2) is feasible. The resulting suboptimal receding horizon control law will be denoted by \( \hat{u}_{RHL}(x_0) \).

3. Multi-parametric programming

In this section, optimization problems that depend on a parameter \( \gamma \in \Gamma \subseteq \mathbb{R}^n \) will be discussed. Throughout the paper, the set \( \Gamma \) is assumed to be polyhedral. Both multi-parametric MIQP (mp-MIQP) problems and multi-parametric QP (mp-QP) problems will be considered in this work. The mp-MIQP problems considered are in the form

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^T H x + f^T x \\
\text{subject to} & \quad A_k x = b_k, \\
& \quad A_k x \leq b_k, \\
& \quad x_i \in \{0, 1\} \quad \forall i \in B, \\
& \quad \text{where } B \text{ is a set containing the indices to the binary components of } x.
\end{align*}
\]

Furthermore, relaxations of the mp-MIQP problem in (4), with some of the relaxed binary variables fixed, are of interest. These problems are in the form

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^T H x + f^T x \\
\text{subject to} & \quad (4b), (4c), \\
& \quad 0 \leq x_i \leq 1, \quad \forall i \in B, \\
& \quad x_i = 0, \quad \forall i \in B_0, \quad x_i = 1, \quad \forall i \in B_1, \\
& \quad \text{which is an mp-QP problem and } B_0 \cup B_1 \subseteq B.
\end{align*}
\]

Many contributions have been published in the area of mp-QP and mp-MIQP, see, e.g., Bomporad et al. (2002), Dua and Pistikopoulos (2000) and Dua et al. (2002). The following definitions are fundamental for what follows.

Definition 1. A function \( f(x) : X \to \mathbb{R}^n \), where \( X \subseteq \mathbb{R}^n \), is piecewise affine (PWA) if it is possible to partition \( X \) into regions \( \mathcal{R}_i \) such that \( f(x) = H_i x + k_i, x \in \mathcal{R}_i \).

Definition 2. A function \( f(x) : X \to \mathbb{R}^n \), where \( X \subseteq \mathbb{R}^n \) is a polyhedral set, is polyhedral piecewise affine (PPWA) if it is PWA and the regions \( \mathcal{R}_i \) are convex polyhedral regions.

Piecewise quadratic functions (PWQ) and polyhedral piecewise quadratic functions (PPWQ) are defined analogously. The most important properties of the mp-QP solution are summarized in Theorem 3.

Theorem 3 (Bank et al., 1982). Consider the mp-QP in (5), assume \( H \in \mathbb{R}^{n_c}, \) and \( \Gamma \) convex. Then the set of feasible parameters \( \Gamma^* \subseteq \Gamma \) is convex, the optimizer \( x^*(\gamma) \) is continuous and PPWA, and the optimal objective function (the value function) \( J^*(\gamma) \) is continuous, convex and PPWQ.

Similarly, the properties of the mp-MIQP solution are summarized as

Theorem 4. Consider the mp-MIQP in (4), assume \( H \in \mathbb{R}^{n_c}, \) and \( \Gamma \) convex. Then the set of feasible parameters \( \Gamma^* \subseteq \Gamma \) is not necessarily convex, the optimizer \( x^*(\gamma) \) is PWA, but not necessarily continuous, and the optimal objective function \( J^*(\gamma) \) is PWQ, but in general, neither convex nor continuous. An optimal solution to the mp-MIQP has an objective function given by

\[
J^*(\gamma) = \min_{p} J_p(\gamma),
\]

(6)
where each $f_p$ denotes the optimal objective function to an mp-QP, and thus has the properties described in Theorem 3.

Proof. See, e.g., Borrelli (2003). The expression in (6) follows directly from fixing the binary variables in (4) to each possible combination $p$ of binaries, computing the corresponding explicit solutions with value functions $J_p(y)$, and finally taking $J^*(y)$ as the point-wise minimizer. □

4. Suboptimal branch and bound

Branch and bound is a method that in many cases can solve an MIQP problem more efficiently than explicitly enumerating all possible combinations of integer variables by organizing the search space in the form of a binary search tree. For more information, see, e.g., Wolsey (1998). In this section, it is investigated how branch and bound can be used to compute suboptimal solutions to non-parametric MIQP-problems.

Definition 5. A $\sigma$-suboptimal solution is a solution to an optimization problem with a corresponding objective function value $J$ that satisfies $J - J^* \leq \sigma$ if $J^* < \infty$.

A formal algorithmic description of a branch and bound algorithm that computes an $(\epsilon + (1 + \rho)\xi + \rho J^*)$-suboptimal solution to an MIQP optimization problem $P$ is presented in Algorithm 1, where $P$ is an MIQP problem with feasible set $\delta$, $\delta$ is an MIQP problem with feasible set $\delta$ constructed from $P$ but additionally subject to a set of constraints locking a subset of the binary variables either to 0 or to 1, and $\delta$ is the QP relaxation of $P$ with feasible set $\delta$. Furthermore, $x$ and $J$ are the solution and objective function value, respectively, of the best known integer feasible solution so far, $x_0$ and $J_0$ are the $\xi$-suboptimal solution and objective function value to relaxation $P_0$, and LIST is a sort list implementing a priority queue. The ordering of the nodes (also called vertices) in LIST corresponds to the choice of the tree exploration strategy (e.g., depth first).

The constant $\epsilon \geq 0$ is an allowance parameter that controls the absolute suboptimality of the solution and the constant $\rho \geq 0$ is a relative allowance parameter that controls the relative suboptimality of the solution. The constant $\xi \geq 0$ bounds the suboptimality of the solutions to the relaxations $P_0$. In the optimal case it holds that $\epsilon = \rho = \xi = 0$. Even though $\epsilon$, $\rho$, and $\xi$ affect the suboptimality of the computed solution in a similar way, their origins are different. Typically, $\epsilon$ and $\rho$ are intentionally chosen greater than zero by the user in order to try to reduce the size of the tree for the price of suboptimality, while $\xi$ is often a result of an approximation scheme used to compute solutions to the relaxations. The effect of a non-zero $\epsilon$ is studied in more detail in Ibaraki, Muro, Murakami, and Hasegawa (1983). The effect of a non-zero $\epsilon$ appears to be less investigated in the literature. On termination, a solution $x$ is returned by Algorithm 1 with a corresponding objective function value $J$ and a quality guaranteed by Theorem 6.

Theorem 6. Assume $\epsilon\rho\xi \in \mathbb{R}_+$, $J^* \geq 0$ and that $\xi$-suboptimal solutions can be computed to the relaxations $P_\ell$. Furthermore, assume that $P$ is feasible. Then, when Algorithm 1 terminates it holds that $x$ is an $(\epsilon + (1 + \rho)\xi + \rho J^*)$-suboptimal solution to $P$.

Proof. Denote with $B$ the binary tree explored by Algorithm 1 and with $V(B)$ its vertex set. Each vertex in $V(B)$ represents a problem denoted $P_\ell$ in Algorithm 1. Denote the root vertex $r \in V(B)$ and a leaf containing an optimal solution to $P$ by $l \in V(B)$. Define $J(v)$ as the optimal objective function value to the relaxation at vertex $v$ denoted $P(v)$. Furthermore, define an approximate solution to $P(v)$ as $x(v)$ with objective function value $\hat{J}(v)$ that satisfies $\hat{J}(v) \leq J(v) + \xi$. Finally, let $\hat{J}(v)$ denote the objective function value of the incumbent at vertex $v$ before any update of it has been performed. Consider the unique $r$, $\ell$-trail $T(\ell) \subseteq B$ of length $n_\ell$. Along this trail, it holds by the construction of the tree that $J(v) \leq J^*, \forall v \in T(\ell)$. In particular it holds that $J(\ell) = J^*$. If Algorithm 1 does not explore all vertices $v \in T(\ell)$, it then either holds that $x(v) \in B$ (cut by line 12) or that $\epsilon + (1 + \rho)J(v) \geq \hat{J}(v)$ at a vertex $v \in T(\ell)$ (cut by line 14, or line 10 since infeasibility is represented by infinite cost). First, if $x(v) \in \delta$, then it holds that $J(v) \leq \hat{J}(v) + \xi \leq \xi + \xi = \xi$. Hence, after the update of the incumbent (lines 6–9), it holds that $J(v^+) - J^* \leq \xi$, where $v^+$ is the vertex explored after $v$. Second, if $\epsilon + (1 + \rho)J(v) \geq \hat{J}(v)$, it holds that $J(v) \leq \epsilon + (1 + \rho)\xi + (1 + \rho)\xi \Rightarrow J(v) - J^* \leq \epsilon + (1 + \rho)\xi + \rho J^* \Rightarrow \hat{J}(v^+) - J^* \leq \epsilon + (1 + \rho)\xi + \rho J^*$. Hence, $x(v^+)$ has been proven to be $(\epsilon + (1 + \rho)\xi + \rho J^*)$-suboptimal. Since the incumbent can only be improved during the execution of Algorithm 1 (follows from line 6), the desired result follows. □

5. mp-MIQP using branch and bound

In this section, an algorithm for computing the parametric solution to mp-MIQP problems using multi-parametric branch and bound (mp-BnB) will be presented. It extends ideas presented in Acevedo and Pistikopoulos (1997) for multi-parametric mixed-integer linear programs, to multi-parametric mixed-integer quadratic programs. Furthermore, compared to Acevedo and Pistikopoulos (1997) where only the optimal case is considered, this work considers a more general case where the user can intentionally relax the optimality requirement and specify absolute and relative suboptimality bounds a priori. Moreover, in this work the case when the mp-QP relaxations in the nodes are not solved to optimality is considered, which is motivated by the recent increased interest in approximate mp-QP (i.e., approximate explicit linear MPC).

Definition 7. A $\sigma(y)$-suboptimal solution, with $\sigma(y) : \mathbb{R}^n_y \rightarrow \mathbb{R}_+$ finite and non-negative, is a solution to a parametric optimization

Algorithm 1 Branch and bound for MIQP

\begin{itemize}
  \item [Input:] MIQP problem $P$, allowance constant $\epsilon$, relative allowance constant $\rho$, and a suboptimality bound $\xi$.
  \item [Output:] $\hat{x}$, $\hat{J}$
\end{itemize}

\begin{algorithm}
  1. $\hat{x} \leftarrow \infty$, $\hat{J} \leftarrow \epsilon$
  2. Add $P$ to LIST.
  3. While length(LIST) > 0 Do
     4. Pop $P_i$ from LIST.
     5. Compute an $\xi$-suboptimal solution to $P_i \Rightarrow \hat{x}_i, \hat{J}_i$.
     6. If $\hat{x}_i \neq \emptyset$ and $x_i \in \delta$ and $\hat{J}_i \leq \hat{J}$ then
      7. A better integer feasible solution has been found.
     8. $\hat{x} \leftarrow \hat{x}_i$, $\hat{J} \leftarrow \hat{J}_i$
     9. End if
    10. If $\delta = \emptyset$ then
      11. No feasible solution exists for $P_i$.
    12. Else if $x_i \in \delta$ then
      13. There exists no feasible solution of $P_i$ which is more than $\xi$ better than $x_i$.
    14. Else if $\epsilon + (1 + \rho)\xi \geq \hat{J}$ then
      15. There exists no feasible solution to $P_i$ which is more than $\epsilon + (1 + \rho)\xi + \rho J^*$ better than $\hat{x}$.
    16. Else
      17. Split $\delta_i$ into $\delta_0$ and $\delta_1$.
      18. Push $P_{0i}$ and $P_{1i}$ to LIST.
    19. End if
  20. End while
\end{algorithm}

D. Axehill et al. / Automatica 50 (2014) 240–246
Algorithm 2 Branch and bound for mp-MIQP

**Input:** mp-MIQP problem \( P(\gamma) \), allowance function \( \epsilon(\gamma) \), relative allowance constant \( \rho \), and a suboptimality bound function \( \xi(\gamma) \).

**Output:** \( \hat{x}(\gamma), J(\gamma) \), \( \hat{F} \)

1. \( \hat{J}(\gamma) = -\infty \), \( \forall \gamma \in \Gamma, \hat{x}(\gamma) \leftarrow \text{void} \), \( \forall \gamma \in \Gamma \)
2. \( \hat{F} \leftarrow \emptyset \)
3. Add \( P(\gamma) \) to LIST.
4. While \( \text{length(LIST)} > 0 \) do
   5. Pop \( P(\gamma) \) from LIST.
   6. Compute an \( \xi(\gamma) \)-suboptimal solution to \( P(\gamma) \) \( \Rightarrow \hat{J}(\gamma), \hat{x}(\gamma) \), and \( \hat{L}_\gamma \subseteq \Gamma_\gamma \).
   7. If \( \exists \gamma : x(\gamma) \in \hat{L}_\gamma \) then
      8. A potentially better integer feasible solution has been found for a subset of the parameter space.
      9. \( \hat{F}_1 = \{ \gamma \in \hat{L}_\gamma : J(\gamma) \leq \hat{J}(\gamma), x(\gamma) \in \hat{L}_\gamma \} \)
      10. \( \{ \gamma \in \hat{L}_\gamma : J(\gamma) \leq \hat{J}(\gamma), x(\gamma) \notin \hat{F}_1 \} \)
      11. \( \hat{x}(\gamma), \gamma, \gamma \notin \hat{F}_1 \)
   12. \( \hat{F} \leftarrow \hat{F} \cup \hat{F}_1 \)
   13. end if
   14. If \( \exists \gamma : x(\gamma) = \emptyset \), \( \forall \gamma \in \Gamma \) then
   15. No feasible solution exists to \( P(\gamma) \) for any \( \gamma \in \Gamma \).
   16. else if \( x(\gamma) \notin \hat{L}_\gamma, \forall \gamma \in \hat{L}_\gamma \) then
   17. There does not exist any \( \gamma \) for which there exists a solution to \( P(\gamma) \) which is more than \( \xi(\gamma) \) better than \( x(\gamma) \).
   18. else if \( x(\gamma) + (1 + \rho)\xi(\gamma) > J(\gamma), \forall \gamma \in \hat{L}_\gamma \) then
   19. No feasible solution exists to \( P(\gamma) \) which is more than \( \epsilon(\gamma) + (1 + \rho)\xi(\gamma) \) better than \( \hat{x}(\gamma) \).
   20. else
   21. Split \( \delta_\gamma(\gamma) \) into \( \delta_\gamma(\gamma) \) and \( \delta_\gamma(\gamma) \).
   22. Push \( P_0(\gamma) \) and \( P_1(\gamma) \) to LIST.
   23. end if
24. end while

where the parameter \( \gamma \) with a corresponding value function \( J(\gamma) \) for which it holds that \( J(\gamma) - J^*(\gamma) \leq \sigma(\gamma), \forall \gamma \in \Gamma^* \), where \( \Gamma^* \) is the set of parameters for which there exists a feasible solution.

In this work the mp-MIQP problem is solved using the branch and bound method, where mp-QP problems (relaxations) are solved in the nodes. This approach provides two benefits compared to explicit enumeration of all feasible binary sequences ("switching sequences"). First, there is a potential of handling larger problems with more binary variables, with the same motivation as for non-parametric branch and bound. Second, it provides an efficient framework for computation of suboptimal solutions. By relaxing the optimality requirement, many integer feasible solutions are possible to cut away during the branch and bound process, since they only marginally contribute to the optimal solution. In the parametric case, this means that the number of mp-QP problems to be solved offline typically can be significantly reduced. By introducing the possibility to use approximate mp-QP solutions for the relaxations in the nodes, the cost for these can be reduced, the comparison of the different bounds can potentially become computationally cheaper, and the final explicit solution might be simplified even further.

The mp-BnB is presented in detail in Algorithm 2, where the value functions are assumed to be infinite for those \( \gamma \) where the problem is infeasible.

The used variables are defined analogously as in Algorithm 1 apart from a possible parameter dependence. The sets \( \hat{F} \) and \( \hat{L}_\gamma \) denote the obtained set of feasible parameters for the best known integer feasible solution so far and the obtained set of feasible parameters for the relaxation in node \( i \), respectively. The user-defined allowance function \( \epsilon(\gamma) \geq 0 \) is a function that controls the absolute suboptimality of the solution. Similarly, the constant \( \rho \geq 0 \) is a relative allowance constant that controls the relative suboptimality of the solution. The function \( \xi(\gamma) \geq 0 \) should be chosen as a bound on the suboptimality of the mp-QP solutions to the relaxations. If a globally optimal parametric solution is sought for, then \( \epsilon(\gamma) = 0, \rho = 0, \xi(\gamma) = 0, \forall \gamma \). When Algorithm 2 terminates, the solution has the quality stated in Theorem 8.

**Theorem 8.** Assume that \( 0 \leq \epsilon(\gamma) < \infty, 0 \leq \xi(\gamma) < \infty, \forall \gamma \in \Gamma, 0 \leq \rho < \infty, \) and that \( \xi(\gamma) \)-suboptimal solutions to \( P(\gamma) \) can be computed. Then, when Algorithm 2 terminates it holds that \( \hat{x}(\gamma) \) is \( (\epsilon(\gamma) + (1 + \rho)\xi(\gamma) + \rho \xi^*(\gamma)) \)-suboptimal.

**Proof.** Can be performed analogously to the proof of Theorem 6 while considering the parametric cut conditions used in Algorithm 2.

Compared to Algorithm 1, Algorithm 2 involves some significantly more complicated operations. On line 6, this algorithm solves relaxations in the form in (5) parametrically instead of only for a single \( \gamma \). On line 9 and line 18, two PWQ functions are compared. An algorithm that solves the relatively advanced task to compare PWQ functions is presented in Section 6.

This section is concluded with a corollary describing the set of feasible parameters \( \hat{F} \) of the parametric solution returned by Algorithm 2.

**Corollary 9.** Denote the set of feasible parameters for a globally optimal solution \( \Gamma^* \subseteq \Gamma \). Assume the assumptions in Theorem 8 hold. Then \( \hat{F} = \Gamma^* \).

**Proof.** \( \hat{F} \supseteq \Gamma^* \) follows from Theorem 8. \( \hat{F} \subseteq \Gamma^* \) follows from the fact that the feasible set of parameters cannot be larger for a suboptimal solution than for an optimal solution.

6. Comparison of PWQ functions

In this section an algorithm to compare PWQ functions is presented, Besselmann (2010). This is the key to extending the result in Acedo and Pistikopoulos (1997) to the more complicated quadratic cost case. Note that also other comparison strategies are possible, e.g., the one in Alessio and Bemporad (2006) which is based on DC programming, Horst and Thoai (1999).

The best-so-far solution \( \hat{J}(\gamma) \) in Algorithm 2 is PWQ, but not necessarily PPWQ, making it difficult to store, to evaluate and to handle. However, \( J(\gamma) \) is composed of PWQ integer feasible solutions \( I_p(\gamma), p = 1, \ldots, n_p \). Instead of working with \( J(\gamma) \) directly, one can thus substitute \( J(\gamma) \) by \( \min_p I_p(\gamma) \). The following corollary ensures that the comparisons in lines 10 and 19 of Algorithm 2 consequently boil down to comparisons of PWQ functions.

**Corollary 10.** Let \( \hat{J}(\gamma) : \hat{F} \rightarrow R \) be the best-so-far solution from Algorithm 2 and let \( J(\gamma) : F \rightarrow R \) be an arbitrary function with \( \Gamma, \hat{F} \subseteq R^n \). For a given \( \gamma \in (\hat{F} \cap \Gamma) \) it holds \( J(\gamma) \leq \hat{J}(\gamma) \iff J(\gamma) \leq I_p(\gamma), \forall p \) and \( J(\gamma) \geq \hat{J}(\gamma) \iff \exists p : J(\gamma) \geq I_p(\gamma) \).

**Definition 11.** We call a region \( R \subseteq R^n \) dominated by a finite set of regions \( \{ R_i \}_{i=1}^n \subseteq R^n \), if \( \forall \gamma \in R, \exists i \in \{ 1, \ldots, n \} : \gamma \in R_i \land J(\gamma) \leq I_p(\gamma) \). A dominated region is also called redundant.

In order to save computation time and storage space, regions are treated as atomic units, i.e., regions are not split into subregions, only entire regions are marked as redundant and all comparisons are carried out region-by-region. A region-by-region version of the conditions in Corollary 10 is
Corollary 12. Let \( \hat{J}(\gamma) : \hat{F} \to \mathbb{R} \) be the best-so-far solution from Algorithm 2 and let \( J(\gamma) : \Gamma \to \mathbb{R} \) be an arbitrary function with \( \hat{F}, \hat{J} \subseteq \mathbb{R}^{n} \). For a given region \( R \subseteq (\hat{F} \cap \Gamma) \) it holds that \( \exists \gamma \in \mathbb{R} : J(\gamma) \leq \hat{J}(\gamma) \Leftrightarrow \forall \gamma \in \mathbb{R} : J(\gamma) \leq \hat{J}(\gamma) \forall p, \text{ and } J(\gamma) \geq \hat{J}(\gamma) \forall \gamma \in R \Rightarrow \forall p \in R, \exists p : J(\gamma) \geq \hat{J}(\gamma) \).

Algorithm 3 PWQ function comparison

Input: \( J(\gamma), \hat{J}(\gamma), L, p = 1, \ldots, n_p \)

Output: \( L, L_p \)

1: \( L \leftarrow \{1, \ldots, n\} \)
2: for \( p = 1, \ldots, n_p \) do
3: for \( J \subseteq L \) do
4: \( J_p = \{j : (R_j \cap R_i) \neq \emptyset\} \)
5: if \( J(\gamma) \geq \hat{J}(\gamma) \forall \gamma \in (R_i \cap R_j), \forall \gamma \in L_j \) then
6: if \( R_j \subseteq \{R_i\}_{i=1}^n \), then
7: \( L \leftarrow L \backslash J_p \)
8: end if
9: end if
10: end for
11: end for
12: \( L \subseteq \{ \{j : (R_j \cap R_i) \neq \emptyset\} \} \)
13: if \( J(\gamma) \leq \hat{J}(\gamma) \forall \gamma \in (R_i \cap R_j), \forall \gamma \in L_j \) then
14: if \( R_i \subseteq \{R_j\}_{j=1}^n \), then
15: \( L_p \leftarrow L_p \backslash \{J\} \)
16: end if
17: end if
18: end for
19: end for

Based on Corollary 12, the comparison of PWQ functions can be executed as outlined in Algorithm 3. The algorithm takes a PWQ function \( J(\gamma) \) and compares it to a set of PWQ functions \( J_p(\gamma) \), partition-by-partition and region-by-region. The lists \( L \) and \( L_p \) indicate non-redundant regions, while the lists \( L \) and \( L_p \) indicate which regions intersect with each other. Algorithm 3 can be used in line 10 as well as in line 19 of Algorithm 2. After line 10, the best-so-far solution can be updated by simply appending \( J(\gamma) \) to \( J_p(\gamma) \), and \( L \) to \( L_p \). In line 19, we do not want to update the best-so-far solution, thus it is sufficient to compute \( L \) without updating \( L_p \), and lines 11–18 of Algorithm 3 do not have to be executed. If \( L = \emptyset \) after the execution of Algorithm 3, the condition in line 19 of Algorithm 2 is fulfilled. The proposed procedure is only identifying regions which are dominated by a single partition. The conditions of redundancy in Corollary 12 are thus strengthened to \( \exists \gamma \in \mathbb{R} : J(\gamma) \leq \hat{J}(\gamma) \forall p \Rightarrow \forall p, \exists \gamma \in \mathbb{R} : J(\gamma) \leq \hat{J}(\gamma), \forall \gamma \in R_e, \exists p : J(\gamma) \geq \hat{J}(\gamma) \forall \gamma \in R. \) In fact \( J(\gamma) \) may also contain redundant regions which are not dominated by a single partition, but by a set of partitions. Identifying also these regions is conceptually straightforward, but the computational effort would grow significantly.

At the core of Algorithm 3, redundant regions are determined by pairwise comparisons of quadratic cost functions. This is equivalent to checking if the difference of two quadratic functions, and thus its minimum, is non-negative on the common domain. We are proposing to compare the quadratic cost functions by solving the quadratic program

\[
\min_{\gamma} \{J(\gamma) - \hat{J}(\gamma) \mid \gamma \in (R_1 \cap R_2)\}. \tag{7}
\]

Note that the difference of two quadratic functions is not necessarily convex. If the difference function is convex, the quadratic program (7) can be solved by a standard QP-solver. If the difference function is non-convex, the optimization problem is a non-convex quadratic program, which is known to be \( \mathcal{NP} \)-hard, Pardalos and Vavasis (1991). In this situation, a spatial branch and bound algorithm is employed built on the ideas in McCormick (1976), for which an implementation is available in YALMIP, Löfberg (2004).

Since the optimization problem in (7) does not have to be solved to optimality, the minimization can be terminated if one of the stopping criteria, i.e., a lower bound greater than zero or an upper bound less than zero, is met.

7. Suboptimal hybrid MPC scheme

Explicit MPC for MLD systems requires the solution of an mp-MIQP problem, which we now know can be efficiently solved optimally and suboptimally using Algorithm 2. Given this new framework, it is interesting to discuss different choices of the suboptimality tolerance \( \sigma(x) = \epsilon(x) + (1 + \rho)\xi(x) + \rho f^{\star}(x) \).

I. Optimal solution: The optimal solution to the CFTOC problem (2) is obtained by selecting \( \sigma(x) = 0 \).

II. Absolute performance bound: In general the absolute performance bound \( \epsilon(x) \) can take on the form of any polyhedral piecewise quadratic function. By selecting a large enough absolute suboptimality tolerance, the algorithm terminates after computing any solution feasible for all \( x_0 \in \tilde{X}_T \).

III. Relative performance bound: The user can also specify a performance bound relative to the optimal cost function \( \sigma(x_0) = \rho f^{\star}(x_0) \) with \( \rho \in \mathbb{R}_+ \).

IV. Suboptimal solutions to relaxations: If the relaxations are computed \( \xi(\gamma) \)-suboptimally, the returned solution in general becomes suboptimal. See Theorem 8 for more details.

V. Stability: Depending on the choice of the suboptimality tolerance, closed-loop stability can also be guaranteed.

In order to guarantee stability, the set \( X_T \) and the terminal penalty \( V_T \) are required to satisfy the following assumption.

Assumption 13. The design parameters \( V_T \) and \( X_T \) in (2) are such that, given an auxiliary control law \( \kappa(x) \), a \( K \)-function \( \alpha_2(\kappa(x)) \), and a neighborhood of the origin \( \mathcal{N} \subseteq X_T \)

- \( \kappa(x) \in \mathcal{U}, \forall x \in \tilde{X}_T \)
- \( \tilde{X}_T \) is invariant for system (1) in closed loop with \( \kappa(x) \)
- \( V_T(x) \leq \alpha_2(x), \forall x \in \mathcal{N} \)
- \( V_T(f(x, \kappa(x))) = V_T(x) \leq -|\alpha_1(x) + \alpha_2(x)|, \forall x \in \tilde{X}_T \)

Assumption 13 is a standard stabilization condition for hybrid MPC, see Lazar and Heemels (2009). We now recall Theorem 16 in Lazar and Heemels (2009), whose assumptions hold for the suboptimal MPC scheme here proposed (for the meaning of epsilon–asymptotic stability (epsilon–AS) see Definition 3 in Lazar and Heemels (2009)).

Theorem 14 (Lazar & Heemels, 2009). Under Assumption 13, given a suboptimality tolerance \( \sigma(x) = \sigma \), the origin of system (1) in closed loop with \( u_{\text{bound}}(x) \) is \( \epsilon \)-AS in \( \tilde{X}_T \). Moreover, if the suboptimality tolerance is state dependent, and there exists a \( K \)-function \( \alpha_2(x) \) such that \( -\alpha_1(x) + \sigma(x) \leq -\alpha_2(x) \) for all \( x \in \tilde{X}_T \), then the origin of system (1) in closed loop with \( u_{\text{bound}}(x) \) is asymptotically stable in \( \tilde{X}_T \).

8. Numerical examples

In this section Algorithm 2 is tested in numerical experiments using Matlab 7.10 on a computer with four processors of type Intel Xeon E5540 with 24 GB RAM running Debian 5.0.6. The CPLEX version used was 11.1, the MPT version 2.6.2, and the YALMIP version was 3. In the example, the algorithm was applied to 9 stable random hybrid MPC problems of the type in (2) with 4 real-valued states, 2 real-valued control signals, 3 binary-valued control signals, and no auxiliary variables. The real-valued inputs were constrained by random upper and lower bounds, whereas no state
constraints were enforced. The prediction horizon length $N$ was varied in the range 1–7 steps, resulting in $2^N$ possible switching sequences. The matrices $Q$ and $R$ were chosen as a random positive semidefinite and a random positive definite matrix, respectively. $P$ was chosen equal to $Q$, and a relative performance bound $\rho$ was used as described as Option III in Section 7. The problem was solved parametrically for all $\|x_k\|_\infty \leq 1$. Explicit enumeration is also shown as a comparison. The average computation time and the average number of partitions from the 9 examples for each prediction horizon length are shown in Fig. 1. All computation times presented include removal of redundant regions using the algorithm presented in Section 6. The traditional explicit enumeration approach was not able to finish the 9 examples for $N = 4$ within the time of the experiments. Furthermore, it can also be seen that if Algorithm 2 is run with the aim for an optimal solution ($\rho = 0$), the number of partitions is typically the same as from explicit enumeration after overlap reduction.

The conclusions drawn from the experiment are that, firstly, Algorithm 2 outperforms traditional explicit enumeration in terms of computation time for computing an optimal solution to problems already with modest complexity and, secondly, the introduction of suboptimality reduces the computation times even further as well as the complexity (number of partitions) of the resulting explicit solution. This effect becomes more pronounced for longer prediction horizons, with a large number of possible switching sequences. It also follows from the result of the experiment that this reduction continues as the level of suboptimality is increased and the maximum reasonable level is eventually given by what performance of the closed-loop system that is acceptable in a particular application.

References


Davide Martino Raimondo was born in Pavia, Italy, in 1981. He received the B.Sc. and M.Sc. in Computer Engineering, and the Ph.D. in Electronic, Computer Science and Electric Engineering from the University of Pavia, Italy, in 2003, 2005, and 2009, respectively. As a Ph.D. student he held a visiting position at the Department of Automation and Systems Engineering, University of Seville, Spain. From January 2009 to December 2010 he was a post-doctoral fellow in the Automatic Control Laboratory, ETH Zürich, Switzerland. From March 2012 to June 2012 and from August 2013 to September 2013 he was a visiting scholar in Prof. Braatz’s Group, Department of Chemical Engineering, MIT, USA. Since December 2010 he has been an Assistant Professor at University of Pavia, Italy. He is the author or coauthor of more than 50 papers published in refereed journals, edited books, and refereed conference proceedings. His current research interests include optimization-based control, fault-tolerant control, autonomous surveillance and control of glycaemia in diabetic patients.

Manfred Morari was head of the Department of Information Technology and Electrical Engineering at ETH Zurich from 2009 to January 2012. He was head of the Automatic Control Laboratory from 1994 to 2008. Before that he was the McCollium–Corcoran Professor of Chemical Engineering and Executive Officer for Control and Dynamical Systems at the California Institute of Technology. He obtained the diploma from ETH Zürich and the Ph.D. from the University of Minnesota, both in chemical engineering. His interests are in hybrid systems and the control of biomedical systems. In recognition of his research contributions he received numerous awards, among them the Donald P. Eckman Award, the John R. Ragazzini Award and the Richard E. Bellman Control Heritage Award of the American Automatic Control Council, the Allan P. Colburn Award and the Professional Progress Award of the AIChE, the Curtis W. McGraw Research Award of the ASEE, Doctor Honoris Causa from Babes-Bolyai University, Fellow of IEEE, IFAC and AIChE, the IEEE Control Systems Technical Field Award, and was elected to the National Academy of Engineering (US). Manfred Morari has held appointments with Exxon and ICI plc and serves on the technical advisory boards of several major corporations.