A Numerically Robust Mixed-Integer Quadratic Programming Solver for Embedded Hybrid Model Predictive Control

Alberto Bemporad * Vihangkumar V. Naik *

* IMT School for Advanced Studies Lucca, Italy
(email: {alberto.bemporad, vihangkumar.naik}@imtlucca.it).

Abstract: The deployment of hybrid model predictive control (MPC) in practical applications requires primarily an efficient and robust on-line Mixed-Integer Quadratic Programming (MIQP) solver that runs in real time. In this paper we propose a new algorithm for solving MIQP problems which is particularly tailored to solve small-scale MIQPs, such as those that arise in embedded hybrid MPC applications. The algorithm couples a branch and bound (B&B) scheme with a recently proposed numerically robust Quadratic Programming (QP) solver based on nonnegative least squares (NNLS) and proximal-point iterations. The resulting MIQP solver supports positive semidefinite Hessian matrices, often appearing in hybrid MPC formulations, and warm starts with respect to both binary and real variables. We show that the speed of execution of our solver is comparable with state-of-the-art commercial solvers, while it is relatively simple to code in an embedded control system.

Keywords: Mixed-integer quadratic programming, quadratic programming, active-set methods, nonnegative least squares, model predictive control, hybrid systems.

1. INTRODUCTION

Since hybrid Model Predictive Control (MPC) was introduced almost two decades ago (Bemporad and Morari, 1999), it has attracted a lot of attention in both academia and industry. The widespread popularity of hybrid systems is mainly due to their ability of modeling a very large spectrum of practical systems where physical processes coexist with switching dynamics, discrete actuators, logic rules, and constraints on system variables. MPC based on hybrid models provides an optimized way of controlling such systems. However, the real-time implementation of hybrid MPC on embedded platforms is still a challenge, as it requires solving a mixed-integer quadratic programming (MIQP) problem at every sample time in a numerically efficient and robust manner.

For this reason, several solution approaches to solve MIQP problems in an embedded control setting have been investigated. In “desktop” applications the numerical package is used to solve (sometimes large-scale) MIQPs with large memory and computing resources available and no stringent limits on execution time. Conversely, in embedded applications, severe restrictions on CPU/memory/time resources pose considerable differences: the number of variables (especially binary variables) should be small, the code must be simple and library-free, the algorithm must be numerically robust also when executed in single precision arithmetic.

As suboptimal solutions are often enough for practical control purposes, several heuristic approaches were introduced to approximately solve the mixed-integer problem in a very quick manner. The feasibility pump (FP) heuristic was proposed for mixed-integer linear constraints (Fischetti et al., 2005), and for binary and general-integers (Bertacco et al., 2007). Recently, heuristic methods were used with the alternating direction method of multipliers (ADMM) (Takapoui et al., 2016), and accelerated dual gradient projection (GPAD) (Naik and Bemporad, 2017) in order to approximately solve the MIQP problem. However, heuristic approaches are not guaranteed to converge to a feasible solution. Under some assumptions, an approach for finding suboptimal solutions based on operator splitting with convergence guarantees was presented in (Frick et al., 2016).

The Branch and Bound (B&B) approach (Floudas, 1995) to solve MIQP problems to optimality has been used in combination with various algorithms to solve the QP relaxations including dual active-set methods (Axehill and Hansson, 2006), interior-point methods (Frick et al., 2015), an active-set method based on nonnegative least squares (NNLS) (Bemporad, 2015), GPAD (Naik and Bemporad, 2017), and ADMM based on the OSQP solver (Stellato et al., 2018). The algorithms in (Bemporad, 2015), (Naik and Bemporad, 2017), (Stellato et al., 2018) are relatively simple to code, and are demonstrated to be quite effective for small-scale problems. The first two approaches however require a strictly convex objective function, with their numerical performance deteriorating with the condition number of the Hessian matrix. While regularizing the cost function would improve numerical robustness, it would bias the solution away from optimality.

In this paper we propose an MIQP algorithm based on B&B and the numerically robust solver for positive
semidefinite QPs recently introduced in Bemporad (2018). The QP solver is based on an active-set method for solving nonnegative least-squares (NNLS) problems, but differently from (Bemporad, 2016) it uses proximal-point iterations that greatly improve robustness without worsening execution time. Compared to the MIQP approach in (Bemporad, 2015), the numerical robustness of the resulting MIQP solver is largely improved and can handle positive semidefinite Hessian matrices. Moreover, it handles equality constraints, supports warm starting of QP relaxations from parent nodes in the search tree, and more general bilateral inequality constraints.

In addition, we include a warm start strategy for binary variables within the B&B setup, which allows prioritizing the exploration of a specific section of the binary tree. This is especially useful in hybrid MPC (Bemporad et al., 1999), moving-horizon estimation (Ferrari-Trecate et al., 2002) and piecewise affine regression (Naik et al., 2017; Mejari et al., 2018), where a good initial guess for the binary variables is available by shifting the optimal solution computed at the previous sample step.

We will show in numerical experiments that the resulted approach gives quite comparable results against well-known commercial solvers when tested on small-scale MIQPs, such as those arising in embedded hybrid MPC applications.

2. PROBLEM FORMULATION

Consider the Mixed-Integer Quadratic Programming (MIQP) problem

\[
\begin{align*}
\min \quad & V(z) \triangleq \frac{1}{2} z^T Q z + c^T z \\
\text{s.t.} \quad & \ell \leq Az \leq u \\
& Gz = g \\
& A_i z \in \{\ell_i, u_i\}, \ i = 1, \ldots, p \\
\end{align*}
\]

(1a)

(1b)

(1c)

(1d)

where \( z \in \mathbb{R}^n \) is the vector of optimization variables, \( Q \in \mathbb{R}^{n \times n} \) is the positive semidefinite Hessian matrix, \( Q \succeq 0 \), \( c \in \mathbb{R}^n \), \( A \in \mathbb{R}^{m \times n} \) and \( \ell, u \in \mathbb{R}^m \) describe the linear inequality constraints, \( \ell \leq u \), and \( G \in \mathbb{R}^{q \times n} \), \( g \in \mathbb{R}^q \) the linear equality constraints. Binary equality constraints are described by \( A \in \mathbb{R}^{p \times n}, \ell, u \in \mathbb{R}^p, p \leq m \). Binary variables \( z_i \in \{0, 1\} \) are a special case of \((1d)\), corresponding to setting \( A_i \) as the \( i \)-th row of the identity matrix, \( \ell_i = 0 \), and \( u_i = 1 \). The QP relaxation of problem \((1)\) is obtained by replacing the integrality constraint \((1d)\) with the following linear inequality constraint of the form \((1b)\) as

\[
\ell_i \leq A_i z \leq u_i, \ i = 1, \ldots, p.
\]

(1e)

Hybrid systems can efficiently be modeled using the Mixed Logical Dynamical (MLD) framework (Bemporad and Morari, 1999). The tool HYDSDEL (Torrisi and Bemporad, 2004) allows one to describe a hybrid dynamical model and get the equivalent MLD transformation. The hybrid MPC problem based on MLD models can be recast as a MIQP problem of the form \((1a)-(1d)\). Very often the associated Hessian matrix \( Q \) is only positive semidefinite. The MIQP solution approach described in this paper does not require regularizing \( Q \) to make the problem strictly convex.

3. SOLUTION OF QP RELAXATIONS

Solving problem \((1a)-(1d)\) using B&B requires solving QP relaxations, in which constraints \((1d)\) are either relaxed to inequality constraints \( \ell_i \leq A_i z \leq u_i \) or to equality constraints \( A_i z = \ell_i \) or \( A_i z = u_i \). To solve QP relaxations of the form \((1a)-(1c)\), we slightly extend the solver in (Bemporad, 2018) to handle bilateral constraints of the form \((1b)\), as summarized in Algorithm 1. It is based on the idea of solving proximal-point iterations, where each iteration consists of solving a regularized QP. The latter that is recast as a least distance problem (LDP) and solved using a nonnegative least squares (NNLS) algorithm.

3.1 Outer proximal-point iterations

Let \( Q = Q' \succeq 0 \) in \((1)\). For any \( \epsilon > 0 \), the sequence \( \{z_k\} \) of solutions to the following strictly convex quadratic programs

\[
\begin{align*}
\min & \quad \frac{1}{2} \| z - z_k \|^2 \\
\text{s.t.} & \quad \ell \leq Az \leq u \\
& \quad Gz = g \\
\end{align*}
\]

(2)

converges to a solution \( z^* \) of the QP relaxation of the form \((1a)-(1c)\) as \( k \) tends to infinity (Bemporad, 2018, Corollary 1).

3.2 Inner active-set solver

The dual of the QP problem \((2)\) is the following convex QP

\[
\begin{align*}
\min & \quad \frac{1}{2} \lambda_T^T [ -A^T (Q + \epsilon I) -1 [ -A^T \lambda_u ] + [ d_T^T \lambda_u ] + [ d_T^T \lambda_u ] \\
\text{s.t.} & \quad \lambda_T, \lambda_u \geq 0, \mu \text{ free},
\end{align*}
\]

(3a)

(3b)

where \( \lambda_T, \lambda_u \in \mathbb{R}^m, \mu \in \mathbb{R}^q, \) and \( d_T^T, d_k^T \in \mathbb{R}^m, f_k^T \in \mathbb{R}^q \) are defined as follows:

\[
\begin{align*}
d_T & \triangleq \left[ \frac{G'T}{g} \right] \\
d_k & \triangleq \left[ \frac{G'T}{g} \right] (Q + \epsilon I)^{-1} (c - \epsilon z_k).
\end{align*}
\]

(3c)

The following theorem shows how the QP problem \((2)\) can be recast as a least-squares problem in which some of the variables are constrained to be nonnegative, extending (Bemporad, 2016, Th. 1) to the case of equality constraints, bilateral inequalities as in (Bemporad, 2015) and subsequently to scaling of \( d_T, d_k, f_k \) for improved numerical robustness as in (Bemporad, 2018).

Theorem 1. Consider the QP \((1a)-(1c)\) and let \( z_k \geq 0, \epsilon > 0 \). Let \( L' \) be a Cholesky factorization of \( Q + \epsilon I \) and define \( M = AL^{-1}, N = GL^{-1} \). Let \( \gamma, \beta \) be any positive scalars. Consider the Partially Nonnegative Least Squares (PNLNS) problem

\[
\begin{align*}
\min & \quad \frac{1}{2} \left[ \frac{1}{2} (\beta(d_k^T) - y) + \beta(d_k^T) y - M' y - M' y \right] \\
\text{s.t.} & \quad y, y \geq 0, \nu \text{ free},
\end{align*}
\]

(4a)

(4b)

with \( y, y \in \mathbb{R}^m, \nu \in \mathbb{R}^q \), and let

\[
\begin{align*}
\delta^* & \triangleq \beta (\gamma + d_k^Ty) y + (d_k^T y) y + (f_k) \nu \strut^* \\
r^* & \triangleq \left[ \frac{M' (y - y^*) - N' \nu} {\delta^*} \right]
\end{align*}
\]

(5a)

(5b)
where $r^* \in \mathbb{R}^{n+1}$ is the residual obtained at the optimal solution $(y^*_L, y^*_U, \nu^*)$ of (4), where $y^*_L, y^*_U \in \mathbb{R}^m$, $\nu^* \in \mathbb{R}^q$.

The following statements hold:

i) If $r^* = 0$ then QP (1a)-(1c),(1e) is feasible;

ii) If $r^* \neq 0$ then

$$z^* = -(Q + \epsilon I)^{-1} \left( c - c z_k + \frac{A'(y^*_U - y^*_L) + G'\nu^*}{\beta \delta} \right)$$

and

$$\lambda^*_L = \frac{1}{\beta \delta} y^*_L, \quad \lambda^*_U = \frac{1}{\beta \delta} y^*_U, \quad \mu^* = \frac{1}{\beta \delta} \nu^*$$

solve QP (2) and its dual (3), respectively.

The proof is a slight modification in (Bemporad, 2018, Appendix A), specifically $M$ is replaced by $[M, M']$, $d_k$ by $[d_k^L, d_k^U]$, and $y$ by $[w, w_z]$, $w$ by $[w, w_z]$, and $\lambda$ by $[\lambda, \lambda_z]$.

### 3.3 Warm starting

At each proximal-point iteration, as well as when starting to solve a new QP relaxation after a solution is available from the parent node in the B&B search tree, the QP problem (2) can be warm-started from an initial guess $z_k$ and the corresponding problem (4) from sets of active constraints $P_L, P_U \subseteq \{1, \ldots, m\}$, $P_L \cap P_U = \emptyset$, along with the initial guess $y^*_L \geq 0, y^*_U \geq 0, w^*_L, w^*_U \in \mathbb{R}^m$ and $\nu^* \in \mathbb{R}^q$ which satisfies the following conditions:

$$w^*_L = -M(y^*_U - y^*_L + N'\nu^*) + \beta \delta d^*_L$$  \hspace{1cm} (7a)

$$w^*_U = M(y^*_U - y^*_L + N'\nu^*) + \beta \delta d^*_U$$  \hspace{1cm} (7b)

$$y^*_L w^*_L = y^*_U w^*_U = 0$$  \hspace{1cm} (7c)

$$y^*_L \geq 0, \quad w^*_L = 0, \quad \forall i \in P_L$$  \hspace{1cm} (7d)

$$y^*_U \geq 0, \quad w^*_U = 0, \quad \forall i \in P_U$$  \hspace{1cm} (7e)

$$y^*_L = 0, \quad \forall i \in \{1, \ldots, m\} \setminus P_U$$  \hspace{1cm} (7f)

$$y^*_U = 0, \quad \forall i \in \{1, \ldots, m\} \setminus P_L$$  \hspace{1cm} (7g)

$$\nu = -\left[ N' \beta (f^*) \right]^T \left[ M(y^*_U - y^*_L) \right]$$  \hspace{1cm} (7h)

where $\#$ denotes pseudoinversion, and

$$\delta = \beta + (d_k^L)^T y^*_L + (d_k^U)^T y^*_U + (f^*)^T \nu$$  \hspace{1cm} (7i)

Terms (7a)-(7g), (7i) are derived from the Karush-Kuhn-Tucker (KKT) conditions of problem (4). Condition (7i) is derived from the proof of (Bemporad, 2018, Appendix A) with the modifications mentioned in Section 3.2.

### 3.4 Parameter selection and scaling

The parameters $\beta, \gamma > 0$ are selected as in (Bemporad, 2018)

$$\gamma = 1 + \|f^k\|_1 + \|d^k_{P_L}\|_1 + \|d^k_{P_U}\|_1, \quad \beta = \frac{1}{\gamma},$$

which are known to provide good numerical conditioning. Unlike first-order methods which are well-known to be very sensitive to scaling, our numerical experiments suggest that this algorithm is less sensitive to the scaling. However, in order to ensure better numerical robustness of the overall MIQP solver, we use the scaling applied to the constraints of type (1b), (1c) as

$$\begin{align*}
\bar{M}_L &\triangleq \frac{1}{\|M_L\|_2}, & \bar{N}_L &\triangleq \frac{1}{\|N_L\|_2}, \\
\bar{M}_U z \leq \bar{M}_U u &\leq M_1 u, \\
\bar{N}_L G_{i,z} \leq \bar{N}_L G_{i,z}.
\end{align*}$$

### Algorithm 1 Robust QP solver based on NNLS and proximal-point iterations

**Input:** QP matrices $Q, c, A, G$, vectors $u, \ell, g$, regularization term $\epsilon > 0$, feasibility tolerance $\sigma > 0$, stop tolerance $\eta > 0$, initial guess $y_0$.

1. Compute inverse Cholesky factor $L^{-1}, L^T L = (Q + \epsilon I)$;

2. $M \triangleq AL^{-1}, N \triangleq GL^{-1}, P_L, P_U \ni \emptyset$;

3. Factorize $NN^T$ as $LDL^T = NN^T$;

4. $k, h \leftarrow 0$;

5. $v_k \leftarrow L^{-T}(c - c z_k)$; $W \leftarrow \begin{bmatrix} -M & N \end{bmatrix}$; $d_k \leftarrow -\ell + M v_k$;

6. $[L, D] \leftarrow \text{rankone}(L, D, 1, \beta \delta^k)$;

7. $(y_k, y_U, \nu, P_L, P_U, \gamma, \beta) \leftarrow \text{get_feasible}(P_L, P_U); h \leftarrow h + 1$;

8. $\delta \leftarrow \beta (\gamma + |\beta \delta|^T + |\beta \delta|^V)$, $a \leftarrow W \left[\begin{array}{c} y_k \\ y_U \\ \nu \end{array}\right]$;

9. $w_k \leftarrow [y_k]^T \gamma + \beta \delta^k [d_k^L, d_k^U]$;

10. if $k = 0$ then QP problem (1) is infeasible; go to Step 18;

11. if min$\{w_k, w_u\} \geq - \beta \delta, \forall i \in \{1, \ldots, m\}$, or $P_L \cup P_U = \{1, \ldots, m\}$ then go to Step 14;

12. $i_k \leftarrow \arg \min_{i \in \{1, \ldots, m\}} \{w_k \}, w_u [\ldots]$;

13. update $LDL^T$ factorization go to Step 7;

14. $k \leftarrow k + 1$;

15. $\lambda_{i_k} \leftarrow \frac{1}{\beta \delta} y_k$, $u_k \leftarrow \frac{1}{\beta} z_k; z_k \leftarrow L^{-1}(u_k - v_k)$;

16. if $\|z_k - z_{k-1}\| >> \eta$ then

17. $z^* \leftarrow z_k$, $\lambda_{i_k} \leftarrow \lambda_{i_k}$, $\lambda_{i_k} \leftarrow \lambda_{i_k}$, $\mu^* \leftarrow \mu^*$, $P_L \leftarrow P_L$, $P_U \leftarrow P_U$;

18. end procedure.

**Output:** Primal solution $z^*$ solving (1a)-(1d) dual solution $(\lambda^*_L, \lambda^*_U, \nu^*)$, optimal active sets $P_L, P_U$ or infeasibility status.
3.5 Stopping criteria and optimality

At Step 16 the iterates of Algorithm 1 stop when \( \|z_k - z_{k-1}\| \leq \eta \).

This condition corresponds to satisfying the optimality condition
\[
\|Qz_k + c + A'(\lambda_{uk} - \lambda_{tk}) + G'\mu_k\| \leq \epsilon \eta
\]
(9) of the QP problem of the form (1a)-(1c). In fact, from (6), we have
\[
(Q + \epsilon I)z_k = -c + \epsilon z_{k-1} - A'(\lambda_{uk} - \lambda_{tk}) - G'\mu_k
\]
\[
\epsilon(z_k - z_{k-1}) = -Qz_k - c - A'(\lambda_{uk} - \lambda_{tk}) - G'\mu_k
\]
and hence \( \|z_k - z_{k-1}\| \leq \eta \) implies (9).

In Algorithm 1, Steps 1-4 performs initialization, followed by active-set iterations starting at Step 5. The active sets \( P_u, P_I \) change for every iteration while solving problem (4) at a given proximal-point iteration \( k \); this is done via rank-one updates of LDL^T factorization (Bemporad, 2018, Section III. C). Feasible values as in (7) are computed at Steps 7, 8. Steps 9, 10 check for infeasibility. The inner active set iterations continue from Step 5 until \( w \) is feasible (within tolerance) at Step 11 or if all inequality constraints have entered the active set, followed by computation of solution \( z_k \) at Steps 14, 15. Step 16 continues executing proximal-point iterations until the criteria (9) is satisfied followed by computing the optimal solution at Step 17.

4. MIQP SOLVER

Algorithm 2 describes a standard B&B method to solve the MIQP problem (1a)-(1d). It uses Algorithm 1 for solving the QP relaxations (1a)-(1c),(1e). It consists of three basic sets \( J, I^f, I_u \), contains the indices of binary constraints of type \( \ell_j \leq \ell_j \leq \ell_j \leq \ell_j \) and \( A_I z = \ell_j \) respectively where \( J = \{1, \ldots, p\} \setminus (I'_f \cup I'_u) \), \( I_u \cap I_u = \emptyset \). The tuple \( T \) holds the sets \( I^f, I_u \) for each relaxed QP subproblem. The stack \( S \) holds these tuples with the order of remaining QP subproblems to be solved. The best cost associated with an integer feasible solution is denoted by \( V_0 \), \( z^* \) denotes the optimal value of the MIQP problem. Vector \( z_k \) is the initial value of the optimization vector, which is initialized with an arbitrary initial guess \( z_0 \).

At Step 1, the set \( J \) is initialized with all \( p \) binary constraints, which is equivalent to (1e) and represents the root-node of the B&B tree. At Step 2.1, the last tuple \( T \) is popped from the stack \( S \). Step 2.2 solves this QP relaxation using Algorithm 1.

Step 2.3 checks if the solution of the QP problem is feasible and the solution \( V^* \leq V_0 \). Step 2.3.1 checks if all the binary constraints are satisfied and updates the optimal values \( V^* \) and \( \xi^* \). The binary constraint to be branched upon \( j \) is picked from the set \( J \) having the largest fractional part at \( \ell_j \). The binary constraints \( j^* \) having the largest fractional part at \( \ell_j \) are branched from the two subproblems \( T_0, T_1 \) based on the value of \( A_I z^* \) of the relaxed solution. Once all QP problems are solved, the

Algorithm 2 MIQP solver

Input: MIQP problem matrices \( Q = Q' \geq 0, c, A, G \) and vectors \( \ell, u, g, y \), initial value \( z_0 \); feasibility tolerance \( \epsilon \geq 0 \).

1. set \( V_0 \leftarrow +\infty \); \( z^* \leftarrow \emptyset \); \( J \leftarrow \{1, \ldots, p\} \); \( I_f \leftarrow \emptyset \); \( I_u \leftarrow \emptyset \); \( z_k \leftarrow z_0 \); \( T \leftarrow \{I_f, I_u, z_k\} \); \( S \leftarrow \{T\} \).
2. while \( S \neq \emptyset \) do:
   2.1. \( \{I_f, I_u, z_k\} \) \leftarrow last element \( T \) of \( S \); \( S \leftarrow S \setminus \{T\} \).
   2.2. execute Algorithm 1 to solve QP problem for \( z^*, V^* \).
   2.3. if the QP problem is feasible and \( V^* \leq V_0 \) then
       2.3.1. if \( A_I z^* \leftarrow \{I_f, I_u\}, \forall i \in J \) or \( I_f \cup I_u = \{1, \ldots, p\} \) then \( V_0 \leftarrow V^* \), \( \xi^* \leftarrow z^* \); otherwise
       2.2.3.1. \( j \leftarrow \arg\min_{i \in I_J} \left| A_I z^* - \ell_j + u_j \right| \);
       2.2.3.2. \( J \leftarrow J \setminus j \), \( T_0 \leftarrow \{I_f \cup \{j\}, I_u, z^*\} \);
       2.2.3.3. if \( A_I z^* \leq \ell_j + u_j \) then append \( \{T_0, T_0\} \) to \( S \) otherwise append \( \{T_0, T_1\} \) to \( S \).
3. if \( V_0 = +\infty \) then \( 1 \) infeasible otherwise \( V^* \leftarrow V_0 \); 4. end.

Output: Solution \( \xi^* \) of the MIQP problem (1), optimal cost \( V^* \), or infeasibility status.

value of \( V_0 \) is checked at Step 3. If \( V_0 \) is still having +∞ then (1) is infeasible, in the other case the optimal cost \( V^* \) and the optimal solution \( \xi^* \) is returned.

5. WARM STARTING BINARY VARIABLES

For simplicity we focus on the case of constraints (1d) having the form \( z_i \in \{0, 1\} \) for some indices \( i \in B \), \( B \subseteq \{1, \ldots, n\} \), with \( \text{card} B = p \). We want to introduce a strategy in the MIQP solver that exploits warm starts on (all or some of) the binary variables \( z_i \), which can be immediately extended to more general constraints of the form (1d).

Let \( I_f, I_u \subseteq B \) be the sets containing the indices of warm started binary variables set equal to 0 or 1 respectively, \( I_f \cap I_u = \emptyset, I_f \cup I_u \neq \emptyset \). After solving the relaxed problem (1a)-(1c), (1e) at the root node to compute the optimal solution \( z^* \) (steps up to Step 2.2 of Algorithm 2), and assuming that \( z^*_i \notin \{0, 1\} \) for some \( i \in B \), Steps 2.2.3.1-2.2.3.3 of Algorithm 2 are replaced by Algorithm 3, which is described below.

Steps 1.1-1.3 are executed if the warm started binary variables have fractional values. In this case, the priority for branching is first given to these warm started variables at Step 1.1. At Step 1.2, the selected \( j \) is removed from the set \( J \), and two new subproblems are created. These subproblems are pushed on \( S \) with the order defined in Step 1.3. Once all the warm started binaries are branched upon, Steps 1.4-1.6 are executed for further branching on the binary variables which were not warm started.

Step 2 is executed right after the relaxed problem at the root node is solved and two subproblems are pushed on the stack, and executed only once as ensured by condition in 2. At Step 2.1, all the warm started values are removed from set \( J \). The binary constraints \( j \) having the largest fractional part at is picked (from the indices of non warm
Algorithm 3 Warm start for binary variables

**Input:** Sets $\tilde{I}^e, \tilde{I}^a$ containing the values corresponding to the warm start, $J, I^e, I^a$; Optimal solution $z^*$ of the QP problem;

1. if $J \cap (\tilde{I}^e \cup \tilde{I}^a) \neq \emptyset$ then:  
   1.1. $j \leftarrow \inf_{i \in J} (J \cap (\tilde{I}^e \cup \tilde{I}^a))$;  
   1.2. $J \leftarrow J \setminus \{j\}$, $T_0 \leftarrow \{I^e \cup \{j\}, I^a, z^*\}$; $T_1 \leftarrow \{I^e, I^a \cup \{j\}, z^*\}$;  
   1.3. if $j \in \tilde{I}^e$ then append $\{T_1, T_0\}$ to $S$ otherwise append $\{T_0, T_1\}$ to $S$; 
   otherwise
   1.4. $j \leftarrow \arg\min_{i \in J} |z_i^* - \frac{1}{2}|$;  
   1.5. $J \leftarrow J \setminus \{j\}$, $T_0 \leftarrow \{I^e \cup \{j\}, I^a, z^*\}$; $T_1 \leftarrow \{I^e, I^a \cup \{j\}, z^*\}$;  
   1.6. if $z_j^* \leq 0$ then append $\{T_1, T_0\}$ to $S$ otherwise append $\{T_0, T_1\}$ to $S$;

2. if card$(J) = p - 1$ then (just once below the root node)
   2.1. set $J \leftarrow \{1, \ldots, p\} \setminus (\tilde{I}^e \cup \tilde{I}^a)$;  
   2.2. $j \leftarrow \arg\min_{i \in J} |z_i^* - \frac{1}{2}|$;  
   2.3. $J \leftarrow J \setminus \{j\}$, $T_0 \leftarrow \{I^e \cup \{j\}, I^a, z^*\}$; $T_1 \leftarrow \{I^e, I^a \cup \{j\}, z^*\}$;  
   2.4. if $z_j^* \leq 0$ then append $\{T_1, T_0\}$ to $S$ otherwise append $\{T_0, T_1\}$ to $S$; (pushed at the top of $S$ with the highest priority)

**Output:** Two subproblems $T_1, T_0$ (in appropriate order) are pushed at the top of the stack $S$.

We illustrate the flow of Algorithm 2+3 with an example from beneath the root node. Fig. 1 illustrates example with 3 binary variables and $(1,0,*)$ as a binary warm start. The numbers indicate the order in which the QP relaxations are solved, dashed nodes correspond to QP subproblems that are ignored.

When solving hybrid MPC in receding horizon fashion, at every sampling instance a sequence $z$ is optimized over the whole horizon of length $N$, the first computed value of the input is applied to the system, then the horizon window is shifted by one sample time and the procedure repeated. Let $z^* = (v^*_{00}, \delta^*_0, \zeta^*_0, \ldots, v^*_{N-1\ell}, \delta^*_N, \zeta^*_N)$ be the optimal solution computed at time $t$, where $v_k, \zeta_k, \delta_k$ are the vectors of inputs, real-valued and binary auxiliary variables respectively, corresponding to the optimal trajectory of the MLD model used by hybrid MPC (Bemporad and Morari, 1999). We can exploit the shifted binary variables optimized at the previous step as the binary warm start, in particular $\delta^*_{\ell t+1} = \delta^*_{\ell t+1}$, $\ldots$, $\delta^*_{N-2\ell t+1} = \delta^*_{N-\ell t}$ (and similarly for binary inputs, if any), and use Algorithm 2 with Steps 2.2.3.1–2.2.3.3 replaced by Algorithm 3.

6. NUMERICAL RESULTS

We report numerical experiments carried out on a desktop computer with Intel Core i7-4700MQ CPU 2.40 GHz and 8 GB RAM, using MATLAB R2015a. Algorithms 2, 3 are implemented in interpreted MATLAB code and Algorithm 1 in compiled Embedded MATLAB code.

We consider the hybrid MPC problem from (Bemporad and Morari, 1999, Example 5.1), with the default settings as in bm99sim.m which is a part of the the Hybrid Toolbox for MATLAB (Bemporad, 2003). This demo considers unit weight on the output of the system and all the other weights are zero. This hybrid MPC problem has been tested for the prediction horizon $N$ from 2 to 15.
Motivated by embedded hybrid MPC applications, this paper has proposed a new MIQP solver based on branch and bound that is able to exploit warm start on binary variables. A numerically efficient and robust QP solver based on nonnegative least squares and proximal point iterations is used for solving the QP relaxations, which does not require the Hessian matrix to be positive definite; this is particularly useful in hybrid MPC problems based on quadratic costs where the quadratic cost is only positive semidefinite due to some variables having zero weight in the MPC cost function. The reported numerical results demonstrate that presented framework for warm starting binary variables for hybrid MPC problems using the same QP solution algorithm provides better performance in terms of both average and maximum computation time.

7. CONCLUSIONS

Motivated by embedded hybrid MPC applications, this paper has proposed a new MIQP solver based on branch and bound that is able to exploit warm start on binary variables. A numerically efficient and robust QP solver based on nonnegative least squares and proximal point iterations is used for solving the QP relaxations, which does not require the Hessian matrix to be positive definite; this is particularly useful in hybrid MPC problems based on quadratic costs where the quadratic cost is only positive semidefinite due to some variables having zero weight in the MPC cost function. The reported numerical results demonstrate that presented framework for warm starting binary variables for hybrid MPC problems using the same QP solution algorithm provides better performance in terms of both average and maximum computation time.

REFERENCES


