IEEE TRANSACTIONS
ON AUTOMATIC CONTROL

Dr. Roberto Tempo
Editor, Technical Notes and Correspondence

IEIIT-CNR
Politecnico di Torino
Corso Duca degli Abruzzi 24
10129 Torino, Italy
Phone +39 011 564 5418
Fax +39 011 564 5429
Email tn-tac@polito.it

October 18, 2007

Professor Teodoro Alamo
Departamento de Ingeniería de Sistemas y Automática
Universidad de Sevilla
Spain

Dear Prof. Alamo:

I am very pleased to inform you that your paper, listed below, has received final approval for publication in the *Technical Notes Section* of the *IEEE Transactions on Automatic Control*. I am tentatively scheduling it for the February 2008 issue of the *Transactions*.

To complete the processing of your paper, I will need the items below mailed to this office. Please do this at your earliest convenience, and in no case later than **one month** from the date of this letter. You should expect to receive the galley proofs of your paper directly from IEEE, approximately four weeks before the scheduled publication date. Should your address change between now and then, please inform this office immediately. When preparing the final version of your paper, to avoid publication delays, I recommend to carefully follow the instructions regarding pages limits available at http://control.bu.edu/ieee/information.html and to use the appropriate style file.

I congratulate you for publication of your work in the *Transactions.*

Sincerely,
Roberto Tempo

Editor, Technical Notes and Correspondence
IEEE Transactions on Automatic Control

cc: Associate Editor

Ref. No.: TN07-02-07
Title: An Efficient Maximization Algorithm with Implications in Min-Max Predictive Control
Authors: T. Alamo, D. Muñoz de la Peña and E. F. Camacho

Please mail the following items:
1. Completed IEEE Copyright Transfer form (if not submitted yet)
2. Completed Author's Checklist form
3. Three (3) additional copies of the paper
4. Original drawings and figures, and a caption page
5. Electronic version of your paper *
6. Two to five index terms (keywords) for your paper

* The electronic version of your paper should be sent as an attachment to Bill Colacchio at IEEE Publications (w.colacchio@ieee.org) with copy to tn-tac@polito.it . Please include the paper's reference number in the subject line.Information and forms for items 1, 2 and 5 can be downloaded from http://control.bu.edu/ieee/information.html, at the bottom of the orange box.

An efficient maximization algorithm with implications in min-max predictive control

T. Alamo, D. Muñoz de la Peña and E.F. Camacho

***Abstract*— In this note an algorithm for binary quadratic programs defined by matrices with band structure is proposed. It was shown by the authors in [2] that this class of problems arise in robust model predictive control when min-max techniques are applied. Although binary quadratic problems belongs to a class of NP-complete problems, the computational burden of the proposed maximization algorithm for band matrices is polynomial with the dimension of the optimization variable and exponential with the band size. Computational results and comparisons on several hundred test problems demonstrate the efficiency of the algorithm.**

***Index Terms*— Combinatorial optimization; Binary quadratic programming; Band matrices; Model predictive control; Min-max techniques.**

I. INTRODUCTION

The objective of binary quadratic programming (BQP) is to find a binary vector that maximizes a quadratic function. These kind of problems belong to a class of NP-complete combinatorial problems that have many interesting applications. Capital budgeting and financial analysis [20], traffic message problems [9] and machine scheduling [3] can be formulated as BQP problems. In model predictive control (MPC), BQP problems arise when min-max techniques are applied to linear systems with bounded additive uncertainties; see [7], [15], [17]. In this case, a BQP problem has to be solved in order to evaluate the inner maximization problem at each iteration of the minimization algorithm.

There is a vast literature on BQP that goes back to the 70's, see [10], [19], [11] and the references therein. Solution approaches include linear programming-based methods [4], branch and bound with preprocessing [18], eigenvalue-based approaches [16] and semi-definite relaxations [11]. These techniques deal with large scale (possibly sparse) problems and are able to solve problems with hundred of variables. However, in general are not appropriate for application to MPC because in this case, a high number of low order (less than 50 variables) BQP problems have to be solved. Therefore, techniques that depend on linear or semi-definite programming solvers are too cumbersome. This make the implementation of min-max controllers a hard issue.

In [2] several approximate techniques were proposed to solve the inner maximization problem in polynomial time. In particular, it was proved that the maximization problem can be approximated by an BQP problem with band matrix, i.e. the matrix $M = \{m_{ij}\}$ satisfies $m_{ij} = 0$ if $|i-j| \geq L$ where L is the band size. This problem is highly structured and can be solved efficiently. In this note the algorithm that solves this class of problems is presented. The computational burden of the proposed maximization algorithm is polynomial with the dimension of the optimization variable and exponential with L, the band size. The algorithm uses no multiplications or divisions and it is appropriate to implement the inner optimization of the min-max problem. The primary goal is to provide a detailed description and computational experiments of the algorithm for BQP problems with band structure. The results in this note complements the previous work [2].

The paper is organized as follows: In Section II some preliminary notation is introduced. The problem formulation is presented in Section III. Section IV presents the main contribution of the paper. An example of application is shown is Section V. The computational burden of the proposed algorithm is analyzed in Section VI. The paper ends with a section of conclusions.

II. PRELIMINARY NOTATION

Given a vector w, $w(k)$ denotes the k-th component of the vector. Vector $\bar{0}_n$ is a vector of zeros of dimension n. Given n, $B_n = \{w \in \mathbb{R}^n : w(k) \in \{-1, 1\},\ k = 1, \ldots, n\}$ denotes the set of vertices of the unit-hypercube in $\mathbb{R}^n$. Given a vector $w \in \mathbb{R}^n$ and an integer k ($1 \leq k \leq n$), $s_k(w) = [w(n+1-k),\ \ldots\ , w(n-1),\ w(n)]^T \in \mathbb{R}^k$ denotes the suffix of length k of w.

III. PROBLEM FORMULATION

The objective of binary quadratic programming (BQP) is to find, given a symmetric matrix M and a vector q, a binary vector w of dimension N that maximizes:

$$F(w) = w^T M w + q^T w. \tag{1}$$

In this paper an algorithm to solve efficiently BQP problems in which the matrix $M = \{m_{ij}\}$ has a band structure, that is, $m_{ij} = 0$ if $|i-j| \geq L$, where L is the band size, is presented. The algorithm exploits the structure of the matrix to build a set of 2^L hypotheses that contains the maximum. To build this set and evaluate the maximum, the algorithm requires a number of evaluations of $F(w)$ equal to $(N-L)2^L$. The maximization problem, that will be referred to as ***"L-Band"*** problem from now on, is:

$$\gamma^* = \max_{w \in B_N} F(w) = \max_{w \in B_N} w^T M w + q^T w, \tag{2}$$

where M is a band matrix with band size L.

The authors acknowledge MCYT-Spain for funding this work (project DPI2007-66718-C04-01).

The authors are with the Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Spain.

Email: alamo@cartuja.us.es (T. Alamo, corresponding author), davidmps@cartuja.us.es (D. Muñoz de la Peña), eduardo@cartuja.us.es (E.F. Camacho)

A preliminary version of this work was presented in [1].

A. Min-Max Model Predictive Control

The algorithm proposed in this note complements the results presented in [2] on min-max model predictive control. Min-max robust receding horizon control was first proposed by Witsenhausen [21]. In the context of robust MPC, the problem was tackled by Campo and Morari [8]. See [14] and [13] for a discussion on the estability properties of this class of controllers. The problem that has to be solved in order to obtain the optimum input is of the form:

$$J^*(x) = \min_{u} \max_{\|w\|_\infty \leq 1} J(x,u,w), \tag{3}$$

subject to linear constraints in x and u. The state vector is x. The optimization vector u is the input trajectory that is used to minimize the worst case cost. Vector $w \in \mathbb{R}^N$ is the uncertainty trajectory. The uncertainty is supposed to lie in the unit box[1], i.e. $\|w\|_\infty \leq 1$. The objective function when a quadratic cost function is taken into account can be expressed as follows:

$$J(x,u,w) = w^T M_{MPC} w + q_{MPC}(x,u)^T w + J(x,u,\bar{0}_N).$$

This cost function depends on the original weighting matrices of the cost function, the linear system description and the uncertainty bound. See for example [7] for details on how to compute these matrices.

Therefore, the computation the inner maximization problem has the same form as the BQP cost function stated in (1). And because the maximum of a convex function can be found in at least one vertex of the feasibility region, i.e. $w^* \in B_N$ (see [5], theorem 3.4.6), this problem indeed belongs to the class of BQP problems. This makes solving (3) a hard issue. Although it is a convex problem, each time the maximum has to be evaluated, a BQP problem has to be solved. It follows that in general, large scale BQP solving techniques are not appropriate and specific solvers are needed.

If matrix M_{MPC} is a band matrix, the maximization function can be done efficiently (in polynomial time) using the algorithm proposed in this note and the control law can be evaluated using standard optimization algorithms for convex problems. In [2] an approximate formulation that guarantees that matrix M_{MPC} is a band matrix was presented. This controller guarantees that the closed-loop system is ultimately bounded in a closed region that contains the origin. The size of this region depends of the approximation error, which can be made arbitrarily small increasing the band size. Note that increasing the band size increases the computational burden of solving the corresponding BQP problems. This implies that there is a tradeoff between precision and computational complexity.

The L-Band approximation is appropriate when the control action of a min-max MPC is obtained computing a sequence of correction control actions to a given stabilizing control law (see [6], [12]). In this case the $|m_{ij}|$ decreases in an exponential way with $|i-j|$ and the original problem can be approximated efficiently by an BQP problem with band structure. Moreover, if the stabilizing control law is chosen in such a way that all the eigenvalues of the closed loop system are at the origin (dead-beat control), then matrix M_{MPC} has a band structure and no approximation is required. See [2] for more details on the approximation technique.

[1]Arbitrary box constraints can be scaled to the unit box

IV. L-Band Maximization Algorithm

In this section the main results of the note are presented. First a general procedure to solve a quadratic maximization problem (regardless of the structure of matrix M) is introduced. The algorithm for L-Band problems follows this procedure exploiting the structure of matrix M to implement it in an efficient way. The following definitions are used to describe the general procedure.

Definition 1: The solution set S_N is defined as the set of vertices of B_N for which the maximum is attained.

$$S_N = \{w \in B_N : F(w) = \gamma^*\}.$$

Definition 2: Given the integer k, $1 \leq k < N$, we define the solution set of order k, denoted S_k, as the following set in B_k:

$$S_k = \{w \in B_k : \text{ There is } \alpha \in B_{N-k} \text{ such that } [w^T \ \alpha^T]^T \in S_N\}.$$

Algorithm 1: General BQP solving algorithm ($2 \leq L \leq N$):

1) $C_{L-1} = B_{L-1}$
2) For $k = L$ to N
   a) Obtain the hypothesis set H_k from C_{k-1} as follows:
   $$H_k = \{[w^T \ \beta]^T : w \in C_{k-1}, \ \beta \in B_1\}.$$
   b) Obtain a candidates subset $C_k \subseteq H_k$ such that $C_k \bigcap S_k$ is not the empty set.
3) end for

Property 1: The candidates set C_N obtained from Algorithm 1 satisfies:

$$\gamma^* = \max_{w \in C_N} F(w).$$

Proof:

As $S_{L-1} \subseteq B_{L-1} = C_{L-1}$ and the algorithm assures at each iteration t k, $C_k \bigcap S_k$ is not the empty set, there exists at least a vector $w \in C_N$ such that $F(w) = \gamma^*$. ■

The hypotheses set contains all possible vectors of size k that can be obtained from the candidates set of size $k-1$. From this set a smaller subset is built rejecting those vertices that are known not to be part of the solution set S_k. This is the candidates set. The higher the number of rejected vertices, the more efficient is the algorithm.

In general, without any prior knowledge, it is not possible to reject any vertex that is not full length. In this case $C_N = B_N$ and no gain is obtained. For L-Band problems, it is possible to reject half of the hypothesis vectors in each step. In this case the number of components of C_N is much lower than the number of vectors in B_N so the solution of the maximization problem is obtained in an efficient manner. In particular, it will be shown that C_k has 2^{L-1} elements for $k = L-1, \ldots, N$.

A. Eliminating vertices from the hypothesis set

In this section an algorithm to obtain the candidates subset from the hypothesis subset for L-Band problems is presented. The algorithm is based on the following result:

Property 2: Consider function (1) where $M \in \mathbb{R}^{N\times N}$ is a matrix with band structure and band size L. If $s \in B_{L-1}$ and p_a,$p_b \in B_r$ then:

$$F(\begin{bmatrix}p_a\\ s\\ y\end{bmatrix}) - F(\begin{bmatrix}p_b\\ s\\ y\end{bmatrix}) = F(\begin{bmatrix}p_a\\ s\\ \bar{0}_k\end{bmatrix}) - F(\begin{bmatrix}p_b\\ s\\ \bar{0}_k\end{bmatrix}),$$

for all $y \in B_k$ with $k = N - r - L + 1$.

Proof: Given the structure of M if $s \in B_{L-1}$, the objective function can be rewritten as:

$$F(\begin{bmatrix}p\\ s\\ y\end{bmatrix}) = \begin{bmatrix}p\\ s\\ y\end{bmatrix}^T \begin{bmatrix}M_{pp} & M_{ps} & \bar{0}_{py}\\ M_{ps}^T & M_{ss} & M_{sy}\\ \bar{0}_{py}^T & M_{sy}^T & M_{yy}\end{bmatrix} \begin{bmatrix}p\\ s\\ y\end{bmatrix} + q^T \begin{bmatrix}p\\ s\\ y\end{bmatrix},$$

where $M_{pp}, M_{ps}, M_{ss}, M_{sy}$ and M_{yy} are submatrices of M of appropriate size and $\bar{0}_{py} = \bar{0}_r\bar{0}_k^T$. The equality is obtained operating. ■

Property 2 can be used to eliminate possible beginnings of the maximum solution. Given all possible vectors w of length $n \geq L$ with a given suffix $s(w)$ of length $L-1$, only one of them needs to be considered to obtain the optimal solution. This idea is applied in the following procedure.

Algorithm 2: Algorithm to build the candidates set C_k from the hypotheses set H_k.

- $C_k = H_k$.
- While there exists $w_a, w_b \in C_k$ such that $s_{L-1}(w_a) = s_{L-1}(w_b)$:
  - If $F(\begin{bmatrix}w_a\\ \bar{0}_{N-k}\end{bmatrix}) \geq F(\begin{bmatrix}w_b\\ \bar{0}_{N-k}\end{bmatrix})$ then $C_k = C_k/w_b$ else $C_k = C_k/w_a$.
- end while

Note that the operator / is the set subtraction operator.

Property 3: If H_k is such that $H_k \bigcap S_k$ is not empty then the candidate set $C_k \subseteq H_k$ obtained using Algorithm 2 satisfies that $C_k \bigcap S_k$ is not the empty set.

Proof: It suffices to show that no element of $H_k \bigcap S_k$ is rejected when compared to an element of H_k that does not belongs to S_k. Suppose $w^* \in H_k \bigcap S_k$, $w_b \notin S_k$, and $s_{L-1}(w^*) = s_{L-1}(w_b)$. Then it follows that

$$F(\begin{bmatrix}w^*\\ \alpha\end{bmatrix}) = \gamma^*, \quad F(\begin{bmatrix}w^*\\ \alpha\end{bmatrix}) - F(\begin{bmatrix}w_b\\ \alpha\end{bmatrix}) > 0.$$

and applying Property 2

$$F(\begin{bmatrix}w^*\\ \alpha\end{bmatrix}) - F(\begin{bmatrix}w_b\\ \alpha\end{bmatrix}) = F(\begin{bmatrix}w^*\\ \bar{0}_{N-k}\end{bmatrix}) - F(\begin{bmatrix}w_b\\ \bar{0}_{N-k}\end{bmatrix}) > 0.$$

This implies that when $w^* \in H_k \bigcap S_k$ is compared to $w_b \notin S_k$, w^* is not subtracted from the candidates set. ■

B. Algorithm Implementation

The algorithm proposed in this section constructs in a recursive way a set of candidates for the maximum solution to Problem (2). The algorithm starts with a set of 2^{L-1} different vectors of $L-1$ components. These are all the possible beginnings of a vector. Therefore, it is guaranteed that the maximum will have one of this beginnings. This is the set of candidates C_{L-1}.

In each iteration, the algorithm builds a candidate set of 2^{L-1} vectors with one more component than the previous iteration. When the vectors have dimension N the algorithm finishes and the maximum can be evaluated.

When a new component is added, the number of vectors is doubled, one for each of the possible values of the new component. This is the hypothesis set H_k (the subindex k denotes the dimension of the vectors) and it is made of 2^L vectors. To eliminate half of these vectors we use Property 2. This property allows us to compare vectors with the same suffix of length $L-1$, eliminating one of them. We obtain the new set of candidates vectors C_k, which has 2^{L-1} elements.

Algorithm 3: Implementation of Algorithms 1-2. Note that in this algorithm the sub-index indicates enumeration and not componentwise value of a vector. In each iteration the dimension of the vector is increased until the candidates set of full dimension is obtained. The dimension of each vector is denoted by its super-index. The candidate vectors are denoted as w_i^k while the hypothesis vectors are denoted as h_i^k. Proper enumeration is vital to this algorithm in order to ensure that the set of candidates is mutually exclusive and yet collectively exhaustive. In the algorithm this is achieved by connecting the enumeration to the binary digits of the numbers 0 to $2^L - 1$.

1) Initial candidates set $C_{L-1} = \{w_0^{L-1}, w_1^{L-1}, \ldots, w_{2^{L-1}-1}^{L-1}\} = B_{L-1}$.

$$w_i^{L-1} = \begin{bmatrix} w_i^{L-1}(1)\\ \vdots \\ w_i^{L-1}(L-1)\end{bmatrix}$$

$$\text{with } w_i^{L-1}(j) = \begin{cases} 1 & \text{if } b_i^{L-1}(j) = 0\\ -1 & \text{if } b_i^{L-1}(j) = 1\end{cases}$$

where $b_i^{L-1}(j), j = 1, \ldots, L-1$ are the $L-1$ binary digits of i, i.e. $i = \sum_{j=1}^{L-1} b_i^{L-1}(j)2^{j-1}$.

2) For $k = L$ to N

   a) Build the hypothesis set $H_k = \{h_0^k, h_1^k, \ldots, h_{2^L-1}^k\}$ from $C_{k-1} = \{w_0^{k-1}, w_1^{k-1}, \ldots, w_{2^{L-1}-1}^{k-1}\}$.

$$h_i^k = \begin{cases} \begin{bmatrix} w_i^{k-1}\\ 1\end{bmatrix} & \text{if } i \leq 2^{L-1}-1\\ \\ \begin{bmatrix} w_{i-2^{L-1}}^{k-1}\\ -1\end{bmatrix} & \text{otherwise}\end{cases}$$

   b) Obtain a candidates subset $C_k = \{w_0^k, w_1^k, \ldots, w_{2^{L-1}-1}^k\} \subseteq H_k$ such that $C_k \bigcap S_k$

is not the empty set.

$$w_i^k = \begin{cases} h_{2i}^k & \text{if } F(\begin{bmatrix} h_{2i}^k \\ \bar{0}_{N-k} \end{bmatrix}) \geq F(\begin{bmatrix} h_{2i+1}^k \\ \bar{0}_{N-k} \end{bmatrix}) \\ h_{2i+1}^k & \text{otherwise} \end{cases}$$

3) end for

Property 4: The candidates set

$$C_N = \{w_0^N, w_1^N, \ldots, w_{2^{L-1}-1}^N\}$$

obtained applying Algorithm 3 satisfies:

$$\gamma^* = \max_{i=0,\ldots,2^{L-1}-1} F(w_i^N).$$

Proof: In order to prove Property 4, it must be proved that Algorithm 3 implements correctly Algorithms 1-2. Part of the proof is is presented in the appendix. Algorithm 3 enumerates each vector in a defined way which allows one to obtain the hypothesis set in an efficient way. Step 1 implements step 1 of Algorithm 1, that is, $C_{L-1} = B_{L-1}$. An appropriate enumeration of each vertex is given.

Step 2(a) implements step 2(a) of Algorithm 1. Given C_{k-1}, step 2(a) obtains

$$H_k = \{[w^T \ \beta]^T : w \in C_{k-1}, \ \beta \in B_1\}.$$

An appropriate enumeration of the hypothesis vectors is also given.

In step 2(b) is implemented the vertex rejection algorithm (Algorithm 2). The value function of the hypothesis vectors are compared by pairs. Vector h_{2i}^k is compared to h_{2i+1}^k. In the appendix is shown that both vectors share the same suffix of size $L-1$, that is

$$s_{L-1}(h_{2i}^k) = s_{L-1}(h_{2i+1}^k), \quad (4)$$
$$i = 0, \ldots, 2^{L-1}-1, \quad k = L-1, \ldots, N.$$

Thus, the assumptions of Property 3 are satisfied and the construction of the candidates set C_k (step 2(b) of Algorithm 1) is done in a correct way. ■

Note that the cost function for a given hypothesis vector may be efficiently evaluated using the value of the cost function previously evaluated of the candidates set. At each iteration of the algorithm, these values can be obtained with a number of operations proportional to ($\sim O(N)$).

V. Example

Consider the following example in which $N = 4$ and $L = 3$:

$$M = \begin{bmatrix} 6 & 1 & -2 & 0 \\ 1 & 6 & 1 & -2 \\ -2 & 1 & 5 & 2 \\ 0 & -2 & 2 & 4 \end{bmatrix}, \quad q = \begin{bmatrix} 2 \\ -7 \\ 8 \\ -1 \end{bmatrix}.$$

The initial candidates set C_2 is made up of all possible beginnings of length $L-1 = 2$.

$C_2 =$

w_0^2	w_1^2	w_2^2	w_3^2
1	−1	1	−1
1	1	−1	−1

The hypothesis set H_3 has the double of vectors because another component has been added with the two possible values. The functional is evaluated for each vector, and then, the next candidates set is made of the ones with the higher functional between those that have the same suffix of length 2 (recall that $L = 3$). Entry F_i^k denotes the value of the cost function $F(\begin{bmatrix} h_i^{kT} & \bar{0}_{N-k}^T \end{bmatrix}^T)$. Note that (4) holds throughout the example.

$H_3 =$

h_0^3	h_1^3	h_2^3	h_3^3	h_4^3	h_5^3	h_6^3	h_7^3
1	−1	1	−1	1	−1	1	−1
1	1	−1	−1	1	1	−1	−1
1	1	1	1	−1	−1	−1	−1
F_0^3	F_1^3	F_2^3	F_3^3	F_4^3	F_5^3	F_6^3	F_7^3
20	20	26	**34**	**8**	−8	**22**	14

The new candidates set has again $2^{L-1} = 4$ vectors, but now with one more component.

$C_3 =$

w_0^3	w_1^3	w_2^3	w_3^3
1	−1	1	1
1	−1	1	−1
1	1	−1	−1

Again the hypothesis set is constructed and the functional evaluated.

$H_4 =$

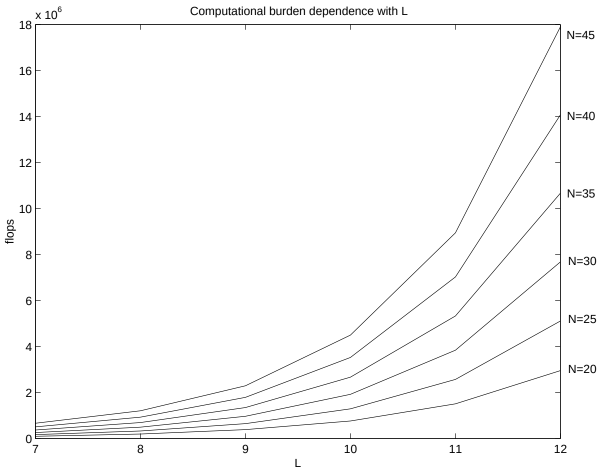
h_0^4	h_1^4	h_2^4	h_3^4	h_5^4	h_5^4	h_6^4	h_7^4
1	−1	1	1	1	−1	1	1
1	−1	1	−1	1	−1	1	−1
1	1	−1	−1	1	1	−1	−1
1	1	1	1	−1	−1	−1	−1
F_0^4	F_1^4	F_2^4	F_3^4	F_4^4	F_5^4	F_6^4	F_7^4
23	**45**	3	**25**	25	**31**	21	**27**

The maximum is guaranteed to be in the last candidates set C_4 that is made up of $w_0^4 = h_1^4$, $w_2^4 = h_3^4$, $w_2^4 = h_5^4$ and $w_3^4 = h_7^4$. The maximum is $w_1^4 = h_2^4$ and the value of the functional is 45.

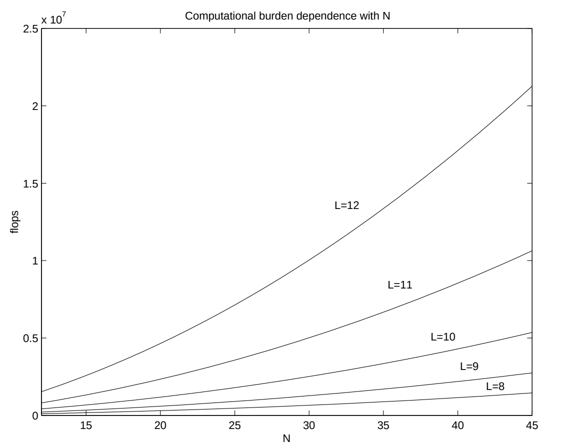
VI. Computational efficiency

Generally, the BQP problem is an NP-complete optimization problem. Without any prior knowledge the complexity of the computational effort is exponential with N, the dimension of the optimization variable. The algorithm presented for band matrices has a computational effort exponential with L, the band size, but quadratic with N, namely $\sim O(N^2 2^L)$. In Figure 1 the computational burden of the L-Band algorithm is shown. The estimation of the order of the complexity is useful for choosing the band size of the approximated min-max controller. It is also important to note that the computational burden is always the same because the algorithm is a sort of exhaustive efficient search. This is an interesting property for a control implementation.

As explained in the introduction, there are several different solution strategies for BQP in the literature. However, most of them are designed to deal with large scale problems and often rely on linear programming or semi-definite programming solvers. For small problems, these techniques are too cumbersome. In order to demonstrate that the proposed algorithm is indeed appropriate for LBand problems, we have compared it with the well known branch and bound technique proposed



(a)



(b)

Fig. 1. Computational burden of the L-Band algorithm (a) for fixed N and varying L and (b) for fixed L and varying N.

in [18]. This technique is appropriate for the comparison because it uses no multiplications or divisions (as the LBand algorithm), can be easily implemented and outperforms most solutions techniques (see [11]) for small sparse problems (dimension less than 50). This is the class of problems we are interested (L-Band matrices are sparse). Both algorithms have been programmed fully in Matlab 6.3. Table I shows computation results for problems of different dimension N and band size L. For each dimension and band size a hundred different problems have been generated using the technique proposed in [18] for generating random band symmetric matrices with elements in -100,100. Entry *Tlb* is the mean computation time of the LBand algorithm, while *Tbb*, *std*, *min* and *max* are the mean, standard deviation, minimum and maximum time of the branch and bound algorithm respectively. All times are in seconds.

Note that the LBand algorithm always has the same computational burden, while the branch and bound algorithm has a standard deviation of the same order of magnitude of the mean computation time. In the results it can be seen that the LBand algorithm outperforms the branch and bound algorithm when the band size is small enough. For increasing values of L for a dimension of 20, it can be seen that the branch and bound algorithm is faster.

TABLE I
COMPARISON WITH THE BRANCH AND BOUND ALGORITHM PRESENTED IN [18].

N	L	Tlb	Tbb	std	min	max
20	10	0.1410	0.1756	0.0895	0.0310	0.4690
20	15	0.4690	0.3498	0.1657	0.1090	0.9370
20	16	0.7970	0.4428	0.2072	0.1410	1.2970
20	17	1.3590	0.4757	0.2166	0.0630	1.1720
20	18	2.1720	0.4609	0.2111	0.0940	0.9680
20	19	5.5630	0.5189	0.2677	0.1250	1.2660
30	10	0.2340	3.1389	2.4634	0.2820	12.7660
30	15	1.5000	7.3387	4.8364	1.2970	24.0940
35	10	0.0470	12.9916	8.3566	1.3280	46.6250
35	15	2.1250	43.7799	34.3757	3.0470	206.6720
40	10	0.0620	63.2074	58.9678	1.9840	338.8280

VII. CONCLUSIONS

In this note an efficient algorithm for solving BQP problems with band structure is presented. This algorithm is relevant in the robust MPC context and complements the results presented in [2]. The algorithm uses no multiplications or divisions and the computational complexity is shown to depend exponentially on the band size and polynomially on the dimension of the problem. These properties make the algorithm appropriate for using it to solve the inner maximization of a min-max optimization problem. Computational results demonstrate the efficiency of the algorithm. A comparison with a branch and bound algorithm is presented.

REFERENCES

[1] T. Alamo, D. Muñoz de la Peña, and E.F. Camacho. An efficient maximization algorithm with implications in min-max predictive control. In *European Control Conference*, Cambridge, England, 2003.

[2] T. Alamo, D. Muñoz de la Peña, D. Limon, and E.F. Camacho. Constrained min-max predictive control: Modifications of the objective function leading to polynomial complexity. *IEEE Transactions on Automatic Control*, 50(5):710–714, May 2005.

[3] B.Alidaee, B. Kochenberger, and A. Ahmadiam. 01-quadratic programming approach for the optimal solution of two scheduling problems. *Int. Journal of Systems Science*, 25:401–408, 1994.

[4] F. Barahona, M. Junger, and G. Reinelt. Experiments in quadratic 0-1 programming. *Mathematical Programming*, 44:127–137, 1989.

[5] M.S. Bazaraa and C.M. Shetty. *Nonlinear Programming. Theory and Algorithms*. John Wiley & Sons, 1979.

[6] A. Bemporad. Reducing conservativeness in predictive control of constrained systems with disturbances. In *Proceedings of the 37st IEEE Conference on Decision and Control*, pages 1384–1389, Tampa, Florida USA, December 1998.

[7] E.F. Camacho and C. Bordóns. *Model Predictive Control in the Process Industry, 2nd Edition*. Springer-Verlag, 2005.

[8] P. J. Campo and M. Morari. Robust model predictive control. In *Proceedings of the American Control Conference*, pages 1021–1026, Green Valley AZ, USA, 1987.

[9] G.Gallo, P. Hammer, and B. Simeone. Quadratic knapsack problems. *Mathematical Programming*, 12:132–149, 1980.

[10] P. Hansen. Methods of nonlinear 0-1 programming. *Annals of Discrete Mathematics*, 5:53–70, 1979.

[11] C. Helmberg and F. Rendl. Solving quadratic (0,1)-problems by semidefinite programs and cutting planes. *Mathematical Programming*, 82:291–315, 1998.

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

[13] M. Lazar, D. Muñoz de la Peña, W. P. M. H. Heemels, and T. Alamo. On input-to-state stability of min-max nonlinear model predictive control. *Systems and Control Letters*, 57:39–48, 2008.

[14] D. Limon, T. Alamo, F. Salas, and Camacho E. F. Input to state stability of min-max mpc controllers for nonlinear systems with bounded uncertainties. *Automatica*, 42:797–803, 2006.

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

[16] B. Mohar and S. Poljak. Eigenvalues and the max-cut problem. *Czechoslovak Mathematical Journal*, 115:343–352, 1990.

[17] D. Muñoz de la Peña, T. Alamo, A. Bemporad, and E. F. Camacho. A decomposition algorithm for feedback min-max model predictive control. *IEEE Transactions on Automatic Control*, 51:1688–1692, 2006.

[18] P.M. Pardalos and G.P. Rodgers. Computational aspects of a branch and bound algorithm for quadratic zero-one programming. *Computing*, 45:131–144, 1990.

[19] P.M.Pardalos and J.Xue. The maximum clique problem. *Journal of Global Optimization*, 4:301–328, 1994.

[20] R.Bridge and J. Yormark. An evolutionary heuristic for quadratic 0-1 programming. *Management Science*, 26(3):282–296, 1980.

[21] H.S. Witsenhausen. A min-max control problem for sampled linear systems. *IEEE Transactions on Automatic Control*, 13:5–21, January 1968.

APPENDIX

In the following some preliminary results needed to prove that (4) holds are presented. These results are based on the properties of the binary digits of a number.

Property 5: The L binary digits $b_i^L(j), j=1,\dots,L$ of i, i.e. $i=\sum_{j=1}^{L} b_i^L(j)2^{j-1}$ have the following properties:

(a) Given $0 \le i \le 2^L - 1$ (value of most significant bit),

$$b_i^L = \begin{cases} \begin{bmatrix} b_i^{L-1} \\ 0 \end{bmatrix} & \text{if } \ i \le 2^{L-1}-1 \\ \\ \begin{bmatrix} b_{i-2^{L-1}}^{L-1} \\ 1 \end{bmatrix} & \text{otherwise} \end{cases}$$

(b) Given $0 \le i \le 2^{L-1} - 1$ (value of least significant bit),

$$b_{2i}^L(j) = b_{2i+1}^L(j), \quad j = 2,\dots,L.$$

(c) Given $0 \le i \le 2^{L-1} - 1$ (shift left bitwise operation),

$$b_i^{L-1}(j) = b_{2i}^L(j+1), \quad j = 1,\dots,L-1.$$

These properties follow from the definition of $b_i^L(j)$.

Property 6: Consider the following enumeration of vertices of the unit hypercube B_n,

$$\vartheta_i^n = \begin{bmatrix} \vartheta_i^n(1) \\ \vdots \\ \vartheta_i^n(n) \end{bmatrix} \text{ with } \vartheta_i^n(j) = \begin{cases} 1 & \text{if } b_i^n(j) = 0 \\ -1 & \text{if } b_i^n(j) = 1 \end{cases} \tag{5}$$

where $b_i^n(j), j=1,\dots,n$ are the n binary digits of i. The following statements hold:

(a) The vertices ϑ_i^L can be obtained from the vertices ϑ_i^{L-1} as follows:

$$\vartheta_i^L = \begin{cases} \begin{bmatrix} \vartheta_i^{L-1} \\ 1 \end{bmatrix} & \text{if } \ i \le 2^{L-1}-1 \\ \\ \begin{bmatrix} \vartheta_{i-2^{L-1}}^{L-1} \\ -1 \end{bmatrix} & \text{otherwise} \end{cases}$$

(b) $s_{L-1}(\vartheta_{2i}^L) = s_{L-1}(\vartheta_{2i+1}^L) = \vartheta_i^{L-1}$.

Proof: Statement (a) stems directly from Property 5 (a) and the definition of $\vartheta_i^L(j)$ given in (5). Taking into account this definition, Property 5 (b) implies that given $0 \le i \le 2^{L-1} - 1$,

$$\vartheta_{2i}^L(j) = \vartheta_{2i+1}^L(j), \quad j = 2,\dots,L.$$

Recalling the definition of suffix, it follows that $s_{L-1}(\vartheta_{2i}^L) = s_{L-1}(\vartheta_{2i+1}^L)$ and taking into account Property 5 (c) and again (5), $s_{L-1}(\vartheta_{2i}^L) = \vartheta_i^{L-1}$. ■

Equation (4) will be proved in a recursive way. Suppose that $s_{L-1}(w_i^{k-1}) = \vartheta_i^{L-1}$. Then it is easy to see that:

$$s_L(h_i^k) = \begin{cases} \begin{bmatrix} s_{L-1}(w_i^{k-1}) \\ 1 \end{bmatrix} & \text{if } \ i \le 2^{L-1}-1 \\ \\ \begin{bmatrix} s_{L-1}(w_{i-2^{L-1}}^{k-1}) \\ -1 \end{bmatrix} & \text{otherwise} \end{cases}$$

$$= \begin{cases} \begin{bmatrix} \vartheta_i^{L-1} \\ 1 \end{bmatrix} & \text{if } \ i \le 2^{L-1}-1 \\ \\ \begin{bmatrix} \vartheta_{i-2^{L-1}}^{L-1} \\ -1 \end{bmatrix} & \text{otherwise} \end{cases} = \vartheta_i^L.$$

Note that the last equality is due to the properties of the proposed enumeration of the vertices of a hyper-cube (see Property 6). Thus, it is inferred that $s_L(h_i^k) = \vartheta_i^L$. Recall (see Property 6) that $s_{L-1}(\vartheta_{2i}^L) = s_{L-1}(\vartheta_{2i+1}^L) = \vartheta_i^{L-1}$. Summing up, the following equality holds:

$$s_{L-1}(h_{2i}^k) = s_{L-1}(h_{2i+1}^k) = \vartheta_i^{L-1}. \tag{6}$$

Now from the definition of w_i^k it is easy to see that:

$$s_{L-1}(w_i^k) = \begin{cases} s_{L-1}(h_{2i}^k) & \text{if } F(\begin{bmatrix} h_{2i}^k \\ \bar{0}_{N-k} \end{bmatrix}) \ge F(\begin{bmatrix} h_{2i+1}^k \\ \bar{0}_{N-k} \end{bmatrix}) \\ s_{L-1}(h_{2i+1}^k) & \text{otherwise} \end{cases}$$

Taking into account equation (6) it results that

$$s_{L-1}(w_i^k) = \vartheta_i^{L-1}. \tag{7}$$

For $k = L$ the initial assumption $s_{L-1}(w_i^{L-1}) = \vartheta_i^{L-1}$ is verified by construction. Applying equation (7) recursively and taking into account (6), equation (4) holds.■