singularity problem is effectively solved by using the integral Lyapunov function in avoiding the nonlinear parametrization from entering into the adaptive control and repetitive control. Asymptotic convergence of the tracking error is established in the presence of periodic uncertainty, while global stability of the closed-loop system is ensured. Error bounds have been provided to characterize the control performance.

APPENDIX

Note that for $a, b \in R^m$ and $\Lambda \in R^{m\times m}$, if $a_i b_i \le 0$, $i = 1, 2, \ldots, m$, and Λ is diagonal and positive definite, then $a^T \Lambda b \le 0$, where a_i and b_i are the components of a and b, respectively. Thus, we need only to prove for each component

$$\left[(\gamma+1)a_i - \left(\gamma b_i + \mathrm{sat}_{\bar{b}_i}(b_i)\right)\right]\left[b_i - \mathrm{sat}_{\bar{b}_i}(b_i)\right] \le 0 \qquad (27)$$

where $\gamma \ge 0$, and a_i satisfies that $\bar{b}_i^1 \le a_i \le \bar{b}_i^2$. There are three possible cases which we should consider in order to prove (27). Case $\bar{b}_i^1 \le b_i \le \bar{b}_i^2$: It follows that $(\gamma+1)a_i - (\gamma b_i + \mathrm{sat}_{\bar{b}_i}(b_i)) = (\gamma+1)(a_i - b_i)$ and $b_i - \mathrm{sat}_{\bar{b}_i}(b_i) = 0$. Hence, (27) is true for this case. Case $b_i < \bar{b}_i^1$: It follows that $\gamma b_i + \mathrm{sat}_{\bar{b}_i}(b_i)) = \gamma b_i + \bar{b}_i^1 < (\gamma+1)\bar{b}_i^1 \le (\gamma+1)a_i$. Since $b_i - \mathrm{sat}_{\bar{b}_i}(b_i) = b_i - \bar{b}_i^1 < 0$, then $(\gamma b_i + \bar{b}_i^1)(b_i - \bar{b}_i^1) > (\gamma+1)a_i(b_i - \bar{b}_i^1)$. Hence, (27) also holds for this case. Case $b_i > \bar{b}_i^2$: It follows that $\gamma b_i + \mathrm{sat}_{\bar{b}_i}(b_i)) = \gamma b_i + \bar{b}_i^2 > (\gamma+1)\bar{b}_i^2 \ge (\gamma+1)a_i$. Since $b_i - \mathrm{sat}_{\bar{b}_i}(b_i) = b_i - \bar{b}_i^2 > 0$, then $(\gamma b_i + \bar{b}_i^2)(b_i - \bar{b}_i^2) > (\gamma+1)a_i(b_i - \bar{b}_i^2)$. In summary, (27) is true for all three cases.

REFERENCES

[1] B. Armstrong, P. Dupont, and C. C. de Wit, "A survey of analysis tools and compensation methods for control of machines with friction," *Automatica*, vol. 30, no. 7, pp. 1083–1138, 1994.

[2] J. D. Boskovic, "Stable adaptive control of a class of first-order nonlinearly parameterized plants," *IEEE Trans. Autom. Control*, vol. 40, no. 2, pp. 347–350, Feb. 1995.

[3] R. Marino and P. Tomei, "Global adaptive output-feedback control of nonlinear systems, Part II: Nonlinear parameterization," *IEEE Trans. Autom. Control*, vol. 38, no. 1, pp. 17–48, Jan. 1993.

[4] A. M. Annaswamy, F. P. Skantze, and A. P. Loh, "Adaptive control of continuous time systems with convex/concave parametrization," *Automatica*, vol. 34, no. 1, pp. 33–49, 1998.

[5] S. S. Ge, C. C. Hang, and T. Zhang, "A direct adaptive controller for dynamic systems with a class of nonlinear parameterizations," *Automatica*, vol. 35, pp. 741–747, 1999.

[6] Z. Qu, "Adaptive and robust controls of uncertain systems with nonlinear parameterization," *IEEE Trans. Autom. Control*, vol. 48, no. 10, pp. 1817–1823, Oct. 2003.

[7] S. Arimoto, "Learning control theory for robotic motion," *Int. J. Adapt. Control Signal Process.*, vol. 4, pp. 543–564, 1990.

[8] S. Hara, Y. Yamamoto, T. Omata, and M. Nakano, "Repetitive control system: A new type servo system for periodic exogenous signals," *IEEE Trans. Autom. Control*, vol. 33, no. 7, pp. 659–668, Jul. 1988.

[9] M. Tomizuka, T.-C. Tsao, and K.-K. Chew, "Analysis and synthesis of discrete-time repetitive controllers," *ASME J. Dyna. Syst. Measure. Control*, vol. 11, pp. 353–358, 1989.

[10] W. Messner, R. Horowitz, W. W. Kao, and M. Boals, "A new adaptive learning law," *IEEE Trans. Autom. Control*, vol. 36, no. 2, pp. 188–197, Feb. 1991.

[11] D. Gorinevsky, "Loop shaping for iterative control of batch processes," *IEEE Control Syst. Mag.*, vol. 22, no. 6, pp. 55–65, Dec. 2002.

[12] N. Sadegh, R. Horowitz, W. W. Kao, and M. Tomizuka, "A unified approach to design of adaptive and repetitive controllers for robotic manipulators," *ASME J. Dyna. Syst. Measure. Control*, vol. 112, pp. 618–629, 1990.

[13] W. E. Dixon, E. Zergeroglu, D. M. Dawson, and B. T. Costic, "Repetitive learning control: a Lyapunov-based approach," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 32, no. 4, pp. 538–545, Aug. 2002.

[14] L. Xu and B. Yao, "Adaptive robust repetitive control of a class of nonlinear systems in normal form with applications to motion control of linear motors," in *Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechtronics*, Como, Italy, Jul. 2001, pp. 527–532.

[15] Z. Qu and H. Zhuang, "Non-linear learning control of robot manipulators without requiring acceleration measurement," *Int. J. Adapt. Control Signal Process.*, vol. 7, pp. 77–90, 1993.

[16] Z. Qu and J.-X. Xu, "Model-based learning controls and their comparisons using Lyapunov direct method," *Special Issue: Iterative Learning Control, Asian J. Control*, vol. 4, no. 1, pp. 99–110, 2002.

[17] D. D. Vecchio, R. Marino, and P. Tomei, "Adaptive learning control for feedback linearizable systems," *Eur. J. Control*, vol. 9, pp. 479–492, 2003.

[18] S. J. Lee and T. C. Tsao, "Repetitive learning of backstepping controlled nonlinear electrohydraulic material testing systems," *Control Eng. Pract.*, vol. 12, pp. 1393–1408, 2004.

[19] B. H. Park, T. Y. Kuc, and J. S. Lee, "Adaptive learning control of uncertain robotic systems," *Int. J. Control*, vol. 65, no. 5, pp. 725–744, 1996.

[20] J. Y. Choi and J. S. Lee, "Adaptive iterative learning control of uncertain robotic systems," *Proc. Inst. Elect. Eng. D*, vol. 147, no. 2, pp. 217–223, 2000.

[21] T.-Y. Kuc and W.-G. Han, "An adaptive PID learning control of robot manipulators," *Automatica*, vol. 36, pp. 717–725, 2000.

[22] J.-X. Xu, Y. Tan, and T.-H. Lee, "Iterative learning control design based on composite energy function with input saturation," *Automatica*, vol. 40, pp. 1371–1377, 2004.

[23] S. S. Ge, C. C. Hang, T. H. Lee, and T. Zhang, *Stable Adaptive Neural Network Control*. Boston, MA: Kluwer, 2001.

[24] S. S. Ge, F. Hong, and T. H. Lee, "Adaptive neural control of nonlinear time-delay systems with unknown virtual control coefficients," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 34, no. 1, pp. 499–516, Feb. 2004.

[25] M. Sun and S. S. Ge, "Adaptive rejection of periodic and non-periodic disturbances for a class of nonlinearly parametrized systems," in *Proc. Amer. Control Conf.*, Boston, MA, Jun. 2004, pp. 1229–1234.

A Decomposition Algorithm for Feedback Min–Max Model Predictive Control

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

***Abstract*—An algorithm for solving feedback min–max model predictive control for discrete-time uncertain linear systems with constraints is presented in this note. The algorithm is based on applying recursively a decomposition technique to solve the min–max problem via a sequence of low complexity linear programs. It is proved that the algorithm converges to the optimal solution in finite time. Simulation results are provided to compare the proposed algorithm with other approaches.**

***Index Terms*—Optimization algorithms, predictive control for linear systems, uncertain systems.**

Manuscript received October 12, 2005; revised February 23, 2006 and June 9, 2006. Recommended by Associate Editor M. Kothare. This work supported in part by the European Community through the HYCON NoE under Contract FP6-IST-511368. The work of D. Muñoz de la Peña was supported in part by a European Community Marie Curie Fellowship and in the framework of the CTS under Contract HPMT-CT-2001-00278.

D. Muñoz de la Peña, T. Alamo and E. F. Camacho are with the Departamento de Ingeniería de Sistemas y Automática, University of Seville, Seville 41092, Spain (e-mail: davidmps@cartuja.us.es; alamo@cartuja.us.es; eduardo@cartuja.us.es).

A. Bemporad is with the Dipartamento di Ingegneria dell'Informazione, University of Siena, Siena 53100, Italy (e-mail: bemporad@dii.unisi.it).

I. Introduction

Most control design methods need a model of the process to be controlled. These models are always subject to uncertainties and only describe the dynamics of the process in an approximate way. Model predictive control (MPC) is one of the control strategies that is able to deal with uncertainty in an explicit way. One approach used to design robust control laws is to minimize the cost function for the worst possible uncertainty realization. This min–max approach was introduced by Witsenhausen [19]. Early model predictive control approaches can be found in [2], [8]. These works deal with open-loop predictions and optimize a single sequence of control inputs for the worst possible trajectory of the uncertain variables. Further results address the feedback min–max problem, where the optimization is done over a sequence of control laws in order to take into account that more information about uncertain variables will be available in the future through feedback (see [14], [18] and the references therein). In both formulations, the resulting min–max optimization problems can be computationally very demanding. Different strategies have been proposed in the literature to overcome this problem; see, for example, [1], [13], and [15].

This note deals with feedback min–max MPC-based on a linear performance index, namely an index that leads to the optimization of a linear function. In [3] multiparametric programming and dynamic programming is used to compute an explicit solution. The method proposed in [9] treats the same problem class, but uses a dual instead of a primal approach to perform the dynamic programming recursion. In [10], it is shown that the min–max problem can be cast as a large-scale linear program. However, both methods can only be applied to problems with very small horizon, as the complexity of the explicit solutions and the size of the LP problems grow exponentially and very fast with the prediction horizon.

In this note, a novel algorithm to solve the min–max problem is proposed. The approach is able to handle some of the problems mentioned above. The algorithm exploits the structure and the convexity properties of the min–max optimization problem. It applies a nested decomposition procedure to solve the problem via a sequence of linear programs of lower complexity. Due to the combinatorial nature of the problem, the computational burden of the algorithm still grows exponentially with the prediction horizon, however, it provides an improved way to implement min–max controllers online on a broader family of plants.

II. Problem Formulation

Consider the discrete-time linear system

$$x(t+1) = A\left(w(t)\right)x(t) + B\left(w(t)\right)u(t) + D\left(w(t)\right) \quad (1)$$

subject to constraints

$$G_x x(t) + G_u u(t) \le g \quad (2)$$

where $x(t) \in R^{n_x}$ is the state vector, $u(t) \in R^{n_u}$ is the input vector, and $w(t) \in R^{n_w}$ is the uncertainty vector that is supposed to be bounded, namely $w(t) \in \mathcal{W}$ where $\mathcal{W}$ is a compact polyhedron. The system matrices are defined by the uncertainty as $M(w(t)) = M_0 + \sum_{k=1}^{n_w} e_k^T w(t) M_k$, where e_k is the kth column of the identity matrix of size n_w.

Digital Object Identifier 10.1109/TAC.2006.883062

Feedback min–max MPC obtains a sequence of feedback control laws that minimizes the worst case cost, while assuring robust constraint handling. A polyhedral terminal region χ^f is taken into account to constraint the state at the end of the prediction.

The min–max problem can be solved by taking into account not all possible values of the uncertainty (which leads to an infinite dimension problem), but only the extreme realizations. See [10], [18]. The enumeration of all the possible extreme realizations of the uncertainty vector w along the prediction horizon gives rise to what is called a *scenario tree*. This tree is used to solve the min–max problem as a finite dimensional deterministic problem. The root node of the tree represents the initial time-step. Level n of the tree stands for time-step n and contains all possible extreme uncertainty trajectories of length n, i.e., all the possible combinations of n vertices of the uncertainty set $\mathcal{W}$. Each node has q children, one for each vertex of $\mathcal{W}$. Each node is then defined by an uncertainty vector w_i which characterizes the uncertainty realization from the parent node. By definition, w_i is one of the vertices of $\mathcal{W}$.

A possible uncertain trajectory defining a scenario is then a path from the root node of the tree down to a leaf. All the nodes of the tree are numbered, starting from the root node (node 0) to the leaf nodes, stage by stage (so the enumeration of the nodes of a given stage is lower than their children nodes). M is the total number of nodes. The time step of the node is denoted by $n(i)$. Each node i has a set of children $I(i)$ and a parent node $p(i)$. The set of children is empty for the leafs nodes and the root node has no parent.

To each node a cost function $\hat{V}_i(x_{p(i)}, u_{p(i)})$ is assigned. This function depends on the previous decision variables $x_{p(i)}$, $u_{p(i)}$, i.e. the variables of the father node, and is defined as the optimum of the following optimization problem[1]:

$$\min_{x_i, u_i} \left\{ \|Qx_i\|_\infty + \|Ru_i\|_\infty + \max_{j \in I(i)} \hat{V}_j(x_i, u_i) \right\} \quad (3a)$$

$$\text{s.t.} \quad (3b)$$

$$x_i = A(w_i)x_{p(i)} + B(w_i)u_{p(i)} + D(w_i) \quad (3c)$$

$$G_x x_i + G_u u_i \le g \quad (3d)$$

$$A(w_j)x_i + B(w_j)u_i + D(w_j) \in \chi^{n(i)+1} \quad \forall j \in I(i). \quad (3e)$$

The subindex i of the variables x_i, u_i denotes node enumeration. The set $\chi^{n(i)+1}$ corresponds to the feasible set of the problem for the next step and is a polyhedron. For the leaf nodes, $\chi^N = \chi^f$. Note that the optimization variable x_i of each node is fixed by (3c).

To obtain the control input, the cost function of the root node $\hat{V}_0(x)$ is minimized for a given initial state x, i.e., the following optimization problem is solved:

$$\min_{x_0, u_0} \left\{ \|Qx_0\|_\infty + \|Ru_0\|_\infty + \max_{j \in I(0)} \hat{V}_j(x_0, u_0) \right\} \quad (4a)$$

$$\text{s.t.} \quad (4b)$$

$$x_0 = x \quad (4c)$$

$$G_x x_0 + G_u u_0 \le g \quad (4d)$$

$$A(w_j)x_0 + B(w_j)u_0 + D(w_j) \in \chi^1 \quad \forall j \in I(0). \quad (4e)$$

[1]Although only the infinity norm is taken into account, the proposed approach can be applied to any feedback min–max control problem with a stage cost that can be evaluated via an LP problem, such as costs based on 1-norms, or, more generally, convex piecewise linear costs.

The definition of $\hat{V}_0(x)$ takes into account that the state of the root node is given by the measured state of the system. See constraint (4c). The boundary conditions are $\hat{V}_i(x_{p(i)}, u_{p(i)})$ equal to

$$\begin{aligned}&\min_{x_i} \|Px_i\|_\infty \\ &\text{s.t.} \quad x_i = A(w_i)x_{p(i)} + B(w_i)u_{p(i)} + D(w_i)\end{aligned}$$

for all leaf nodes (nodes such that $I(i)$ is empty). The value of the cost function at the leaf nodes do not depend on any other node, i.e., can be formulated as an LP problem. The cost function is defined by full-rank matrices Q, R, and P.

The control law is applied in a receding horizon scheme. At each sampling time the problem must be solved for the current state x and the optimum value $\hat{V}_0^*(x)$ is obtained. The controller applies the optimal control input for the first time-step u_0^*. Note that the optimization problem formulated above is of very high complexity. In the following sections, an algorithm that takes advantage of the structure of the problem by means of a decomposition method is presented.

III. Multistage Min–Max Linear Programming

In this section, the multistage min–max linear program in standard form

$$V_i^*\left(z_{p(i)}\right) = \min_{z_i} \left\{ c_i^T z_i + \max_{j \in I(i)} V_j^*(z_i) \right\} \tag{5a}$$

$$\text{s.t.} \quad W_i z_i = h_i - T_i z_{p(i)} \tag{5b}$$

$$z_i \geq 0. \tag{5c}$$

is considered. The main result of this note is an algorithm that solves problem (5). The standard form is introduced to simplify the notation. As problem (3) belongs to the class (5), the algorithm presented can be applied to evaluate the feedback min–max MPC controller described in the previous section.

Problem (5) is defined by a scenario tree. Each node i has a set of children $I(i)$ and a parent node $p(i)$ and is defined by matrices and vectors c_i, W_i, h_i, and T_i. All these parameters are deterministic and can be different for each node. The objective is to minimize V_0^*, the cost function at the root node

$$\begin{aligned}V_0^* = &\min_{z_0} \left\{ c_0^T z_0 + \max_{j \in I(0)} V_j^*(z_0) \right\} \\ &\text{s.t.} \quad W_0 z_0 = h_0 \\ &\qquad z_0 \geq 0.\end{aligned}$$

The boundary conditions are given by the problem solved at each leaf node

$$\begin{aligned}V_i^*\left(z_{p(i)}\right) = &\min_{z_i} c_i^T z_i \\ &\text{s.t.} \quad W_i z_i = h_i - T_i z_{p(i)} \\ &\qquad z_i \geq 0.\end{aligned}$$

Multistage min–max linear programs are related to multistage stochastic linear programs [6]. In the latter, instead of the worst case, the expected value of the cost function for a given discrete distribution of the uncertainty is minimized.

Problem (3) can be formulated as a multistage min–max linear program because the infinity norm can be evaluated using a linear program [17]. The matrices and vectors c_i, W_i, h_i and T_i depend on the system, the cost function, the constraints, and the value of the uncertainty from the parent node to node i (what in the previous section was defined as w_i). The initial state of the system defines the constraints in the root node which does not depends on any previous decision, namely h_0 depends only on x. Recall constraint (4c).

The set of variables z_i corresponding to each node includes the state, the input, the auxiliary variables introduced to evaluate the cost function, and the slack variables needed to represent the feedback min–max problem in standard form.

The constraints represent both the system constraints from (2) and the constraints introduced to model the cost function. The value $V_i^*(z_{p(i)})$ represents the cost function at node i depending on the previous decision vector $z_{p(i)}$ (recall that $p(i)$ is the index of the parent node of i).

IV. Nested Decomposition Algorithm

This section presents an algorithm for solving the multistage min–max linear program (5) by using nested decomposition. This algorithm exploits the specific structure of the problem and the convexity properties of the objective function. It is based on the ideas first introduced by Benders in [4] for solving mixed-integer problems which also have been successfully applied to stochastic programming [5], [6].

The method consists on solving a sequence of LP problems that approximate the value of the original problem. In (5), the value of the function $V_j^*(z_i)$ has to be evaluated. These functions are piecewise linear functions over polyhedral regions (see the results in [3]). In the proposed algorithm, these functions are substituted by an outer linearization, i.e. a lower bound that can be evaluated by linear programming. This lower bound is improved at each iteration and converges to the exact value of $V_j^*(z_i)$.

To simplify the algorithm, we address here a particular case of multistage min–max linear programs. It is assumed that $V_i^*(z)$ is bounded and greater than zero for all nodes. This assumption holds in the case of feedback min–max problems, where the objective function satisfies this condition by definition.

At each step m of the algorithm, subproblems P_i^m are solved, one for each node i of the scenario tree, to obtain the lower bound of the cost function $V_i^*(z)$. As each of the functions of the children nodes are also approximated by an outer linearization, the resulting subproblem is a linear program. Problems P_i^m are defined as

$$V_i^m\left(z_{p(i)}\right) = \min_{z_i, \theta_i, \theta_{i,j}} c_i^T z_i + \theta_i \tag{6a}$$

$$\text{s.t.} \quad W_i z_i = h_i - T_i z_{p(i)} \tag{6b}$$

$$D_{i,j}^k z_i + \theta_{i,j} \geq d_{i,j}^k \quad \forall j \in I(i),\ k \leq r_{i,j}^m \tag{6c}$$

$$\theta_i \geq \theta_{i,j},\ \forall j \in I(i) \tag{6d}$$

$$z_i,\ \theta_i \geq 0,\ \theta_{i,j} \geq 0,\ \forall j \in I(i). \tag{6e}$$

These problems are modified at each iteration. The number of constraints in (6c) for each children node j at step m is $r_{i,j}^m$. These constraints are added in order to evaluate a lower bound on the value of $V_j^m(z_i)$, namely$V_j^m(z_i) \geq \theta_{i,j} \forall j \in I(i)$. At the first iteration $r_{i,j}^0 = 0$ for all nodes and $j \in I(i)$. This means that for the first iteration, the value of $\theta_{i,j}$ of the children of each node i is zero (recall that $\theta_{i,j} \geq 0$). Each time a new optimality cut is added ($r_{i,j}^m$ increases), the approximation provided by $V_j^m(z_i)$ is tighter.

Each $\theta_{i,j}$ is a lower bound on the value of $V_j^m(z_i)$. To evaluate the maximization over the set of childrens, the auxiliary variable θ_i is included. Constraints (6d) evaluate the maximization over all the children of node i using an epigraph approach.

In this section, we prove that $V_i^m(z)$ is a lower bound on $V_i^*(z)$ and that, when the algorithm stops, both values are equal.

As in the previous section, when solving the root node, constraint (6b) is replaced by $W_0 z_0 = h_0$, because the root node has no parent. For the feedback min–max problem, h_0 depends on the initial state of the system x_0.

When solving a leaf node, variables $\theta_{i,j}$ and constraints (6c) are omitted because these nodes do not have children. Note that this means that, by definition, $V_i^m(z) = V_i^*(z)$ for each leaf node and at every algorithm iteration m.

The algorithm solves problems with relative complete recourse [6], i.e., feasibility of the root problem P_0^0 assures feasibility of all the problems of the nodes of the scenario tree for all steps m. For general problems, feasibility cuts can be added to the algorithm as in stochastic linear programming [5], [6]. Note that feedback min–max is formulated to have relatively complete recourse. By definition, if x_0 lies in χ^0, there exists a sequence of control laws that drive x_0 to the terminal region while satisfying the constraints for all possible uncertainties because constraint (3e) holds for each node.

The algorithm is based on adding optimality cuts at each time step. These optimality cuts are obtained from feasible solutions to the dual problem of P_i^m.

Theorem 1 [c.f. [7]]: Define $D_i = \lambda_i^T T_i$ and $d_i = \lambda_i^T T_i z_{p(i)} + V_i^m(z_{p(i)})$, where λ_i are the dual variables of the equality constraints (6b) for a given $z_{p(i)}$, and $V_i^m(z_{p(i)})$ is the optimal value determined by solving problem (6) for $z_{p(i)}$. Then, it holds that for all z

$$V_i^m(z) \geq d_i - D_i z. \tag{7}$$

Note that as the number of optimality cuts $r_{i,j}^m$ is increased at each step m for each children node j, the set of dual constraints of a previous optimum solution may not be optimal, but still remains feasible if new zero variables $\mu_{i,j}^k$ of the new optimality cuts are added. Hence, although problems P_i^m differ at each iteration because optimality cuts are added, the lower bounds on the optimal value remain valid.

The proposed algorithm can be summarized as follows.

Algorithm 1

INITIALIZATION $m = 0, r_{i,j}^0 = 0, e_i = 0, i = 0, \ldots, M-1, j \in I(i)$.

IF P_0^0 is infeasible

Multistage min–max problem is infeasible. Stop.

END IF

DO

FOR $i = 0, \ldots, M-1$

Solve P_i^m using $z_{p(i)}$ and obtain $V_i^m(z_{p(i)}), z_i, \lambda_i$ [the dual variables of the equality constraints (6b)] and $\theta_{i,j}$, $j \in I(i)$.

END FOR

FOR $i = M-1, \ldots, 0$

FOR $j \in I(i)$

IF $\theta_{i,j} < V_j^m(z_i)$.

$$r_{i,j}^{m+1} = r_{i,j}^m + 1.$$
$$D_{i,j}^{r_{i,j}^{m+1}} = \lambda_j^T T_j.$$
$$d_{i,j}^{r_{i,j}^{m+1}} = \lambda_j^T T_j z_i + V_j^m(z_i).$$

ELSE

$$r_{i,j}^{m+1} = r_{i,j}^m.$$

END IF

END FOR

IF $I(i)$ is not empty

$$e_i = \max_{j \in I(i)} V_j^m(z_i) - \theta_{i,j} + e_j.$$

END IF

END FOR

$m = m + 1.$

WHILE $e_0 > 0$

Stop.

Theorem 2: The solution obtained by applying Algorithm 1, denoted by z_0^*, is an optimal solution of (5). The algorithm converges in finite time.

Proof: The optimality cuts are obtained from a set of optimal dual variables of P_j^m as sub-gradients of the optimal solution $V_j^m(z)$. These sub-gradients are defined using the dual variables of the equality constraints (6b) λ_j and the optimal value of the cost function $V_j^m(z)$ which includes the remaining dual variables for different values of z. By taking into account Theorem 1 and the definition of $D_{i,j}^k$ and $d_{i,j}^k$, the following inequality holds at each iteration m and node j:

$$V_j^m(z) \geq \max_{k=1,\ldots,r_{p(j),j}^m} d_{p(j),j}^k - D_{p(j),j}^k z. \tag{8}$$

By construction, if $I(i)$ is empty it holds that $V_i^m(z) = V_i^*(z)$ because in this case (5) and (6) are equal. Taking this into account and applying (8) backwards from the leaf nodes, it holds that at each iteration m and node j

$$V_j^*(z) \geq V_j^m(z). \tag{9}$$

The algorithm finishes when $e_0 = 0$. By construction, if $e_0 = 0$, then $e_i = 0$ for $i = 0, \ldots, M-1$, i.e., for the values z_i of the algorithm solution it holds

$$V_j^m\left(z_{p(j)}\right) = \max_{k=1,\ldots,r_{p(j),j}^m} d_{p(j),j}^k - D_{p(j),j}^k z_{p(j)} \tag{10}$$

for $j = 0, \ldots, M-1$.

By definition, all the previous functions of z are convex so the minimization problems have an unique minimum value. If the following conditions are satisfied: a) z_i is a minimizer of P_i^m (and so minimizes θ_i), b) $e_i = 0$, and c) $V_j^*(z_i) = V_j^m(z_i)$, then the following holds:

$$V_i^*\left(z_{p(i)}\right) = V_i^m\left(z_{p(i)}\right). \tag{11}$$

By taking into account that for the leaf nodes the hypothesis $V_j^*(z_i) = V_j^m(z_i)$ is true, by applying backwards (11) it is proved that

$$V_0^* = V_0^m. \tag{12}$$

Finite termination of the algorithm is assured because the optimality cuts are generated from the dual variables of the optimal solutions of

TABLE I
COMPUTATIONAL ASPECTS FOR DIFFERENT PREDICTION HORIZONS

N	node	iter	Dec(s)	mem(Dec)	LP(s)	mem(LP)
2	21	2.2	0.03	11Kb	0.02	31.6Kb
3	85	2.6	0.13	41Kb	0.37	203.Kb
4	341	3.1	0.69	163Kb	25.2	1.41Mb
5	1365	4.3	3.34	648Kb	203.2	8.4Mb
6	5461	4.4	12.70	2.6Mb	-	-

P_i^m. As P_i^m are LP problems, the optimal dual variables are attained at the vertices of the feasibility set of the dual problem. As the number of vertices is finite and no cut can be repeated, the maximum number of iterations of the algorithm is finite. ■

The computational burden of the algorithm presented in this section depends on the number of nodes of the worst-case scenario tree, which grows in an exponential manner with the prediction horizon. This means that the computational burden can be very high for large horizons. It is important to remark that at the end of any iteration m, a feasible solution z_0^m with a bound on the error e_0^m is obtained, so suboptimal solutions could be used when limits on the computation time are imposed.

V. EXAMPLE

The computation time and the memory requirements of the proposed algorithm are compared with the one corresponding to solving the min–max problem with a single large scale LP problem [10]. Consider the problem of robustly regulating to the origin the system (c.f. [3])

$$x(t+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w(t) \qquad (13)$$

subject to constraints on the state and the input, namely, $\|x(t)\|_\infty \leq 10$ and $\|u(t)\|_\infty \leq 3$. No terminal region is taken into account. The disturbance in supposed to be bounded in the hypercube $\|w(t)\|_\infty \leq 1.5$. We consider the performance measure based on infinity norm with $P = Q = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $R = 1.8$ and different predictions horizons.

Table I shows different results for prediction horizons from $N = 2$ to 6. The results are obtained from over a hundred randomly selected initial states.[2] In Table I, entry node denotes the number of nodes of the scenario tree, *iter* the mean number of iterations of the decomposition algorithm, and *Dec* the time in seconds. Entry *mem(Dec)* is the size of the file that the proposed algorithm needs to solve the optimization problem. The SMPS format [12] has been used to store the problem. Entry *mem(LP)* is the size of the MPS [11] file of the large scale LP and *LP* is the time needed to solve the problem. Note that for $N = 6$ we could not solve the LP problem.

In the simulation results, it is seen that the proposed algorithm gives promising results. It outperforms the LP solver in computation time and memory requirements. For a prediction horizon of 5, the computation time is two orders of magnitude smaller.

VI. CONCLUSION

The note has shown how to compute the solution of a multistage min–max linear program by taking advantage of the structure of the problem. The proposed algorithm can be applied to implement feedback min–max controllers. The decomposition algorithm also gives an insight into the underlying structure of the problem formulation.

[2]The simulations have been realized in Matlab in a AMD Athlon Xp 2800+ using [16].

REFERENCES

[1] 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 Trans. Autom. Control*, vol. 50, no. 5, pp. 710–714, May 2005.

[2] J. C. Allwright and G. C. Papavasiliou, "On linear programming and robust model-predictive control using impulse-responses," *Syst. Control Lett.*, vol. 18, pp. 159–164, 1992.

[3] A. Bemporad, F. Borreli, and M. Morari, "Min–max control of constrained uncertain discrete-time linear systems," *IEEE Trans. Autom. Control*, vol. 48, no. 9, pp. 1600–1606, Sep. 2003.

[4] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numer. Math.*, vol. 4, pp. 238–252, 1962.

[5] J. R. Birge and F. V. Louveaux, "A multicut algorithm for twostage stochastic linear programs," *Eur. J. Oper. Res.*, vol. 34, no. 3, pp. 384–392, 1988.

[6] ——, *Introduction to Stochastic Programming*. New York: Springer-Verlag, 1997.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[8] P. J. Campo and M. Morari, "Robust model predictive control," in *Proc. Amer. Control Conf.*, 1987, vol. 2, pp. 1021–1026.

[9] M. Diehl and J. Björnberg, "Robust dynamic programming for min–max model predictive control of constrained uncertain systems," *IEEE Trans. Autom. Control*, vol. 49, no. 12, pp. 2253–2257, Dec. 2004.

[10] E. C. Kerrigan and J. M. Maciejowski, "Feedback min–max model predictive control using a single linear program: robust stability and the explicit solution," *Int. J. Robust Nonlinear Control*, vol. 14, pp. 395–413, 2004.

[11] J. Edwards, J. Birge, A. King, and L. Nazareth, "A standard input format for computer codes which solve stochastic programs with recourse and a library of utilities to simplify its use," Working Paper WP-85-03, International Institute for Applied Systems Analysis. Laxenburg, Austria, 1985.

[12] H. I. Gassmann, The SMPS format for stochastic linear programs [Online]. Available: http://www.mgmt.dal.ca/sba/profs/hgassmann/SMPS2.htm

[13] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, pp. 1361–1379, 1996.

[14] J. H. Lee and Z. Yu, "Worst-case formulations of model predictive control for systems with bounded parameters," *Automatica*, vol. 33, no. 5, pp. 763–781, 1997.

[15] J. Löfberg, "Minimax approaches to robust model predictive control," Ph.D. dissertation, Dept. Elect. Eng., Linköping Univ., Linköping, Sweden, Apr. 2003 [Online]. Available: http://www.control.isy.liu.se/publications/doc?id=1466

[16] A. Makhorin, GNU linear programming kit [Online]. Available: http://www.gnu.org/software/glpk 2005

[17] A. I. Propoi, "Use of linear programming methods for synthesizing sampled-data automatic systems," *Automat. Remote Control*, vol. 24, no. 7, pp. 837–844, 1963.

[18] P. O. M. Scokaert and D. Q. Mayne, "Min–max feedback model predictive control for constrained linear systems," *IEEE Trans. Autom. Control*, vol. 43, no. 8, pp. 1136–1142, Aug. 1998.

[19] H. S. Witsenhausen, "A min–max control problem for sampled linear systems," *IEEE Trans. Autom. Control*, vol. AC-13, no. 1, pp. 5–21, Jan. 1968.