Delft University of Technology

Delft Center for Systems and Control

Technical report 03-012

Model predictive control for discrete-event and hybrid systems*

B. De Schutter and T.J.J. van den Boom

August 2003

Paper for the *Workshop on Nonlinear Predictive Control (Workshop S-5)* at the 42nd IEEE Conference on Decision and Control, Maui, Hawaii, Dec. 2003.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: https://www.dcsc.tudelft.nl

* This report can also be downloaded via https://pub.bartdeschutter.org/abs/03_012

Model Predictive Control for Discrete-Event and Hybrid Systems

B. De Schutter, T.J.J. van den Boom
Delft Center for Systems and Control, Delft University of Technology
Mekelweg 2, 2628 CD Delft, The Netherlands
{b.deschutter,t.j.j.vandenboom}@dcsc.tudelft.nl

***Abstract*— Model predictive control (MPC) is a very popular controller design method in the process industry. A key advantage of MPC is that it can accommodate constraints on the inputs and outputs. Usually MPC uses linear or nonlinear discrete-time models. In this paper we give an overview of some results in connection with model predictive control (MPC) approaches for some tractable classes of discrete-event systems and hybrid systems. In general the resulting optimization problems are nonlinear and nonconvex. However, for some classes tractable solution methods exist. In particular, we discuss MPC for max-plus-linear systems, for mixed logical dynamical systems, and for continuous piecewise-affine systems.**

I. Introduction

Model predictive control (MPC) was pioneered simultaneously by Richalet *et al.* [65], and Cutler and Ramaker [25]. In the last decades MPC has shown to respond effectively to control demands imposed by tighter product quality specifications, increasing productivity demands, new environmental regulations, and fast changes in the market. As a result, MPC is now widely accepted in the process industry. There are several other reasons why MPC is probably the most applied advanced control technique in this industry:

- MPC is a model-based controller design procedure that can easily handle multi-input multi-output processes, processes with large time-delays, nonminimum phase processes, and unstable processes.
- It is an easy-to-tune method: in principle only three parameters have to be tuned.
- MPC can handle constraints on the inputs and the outputs of the process (due to, e.g., limited capacity of buffers, actuator saturation, output quality specifications, etc.) in a systematic way during the design and the implementation of the controller.
- MPC can handle structural changes, such as sensor or actuator failures, and changes in system parameters or system structure, by adapting the model and by using a moving horizon approach, in which the model and the control strategy are regularly updated.

Conventional MPC uses discrete-time models (i.e., models consisting of a system of difference equations). In this paper we propose some extensions and adaptations of the MPC framework to classes of discrete-event systems and hybrid systems that ultimately result in "tractable" control approaches. For each of these cases the proposed MPC approach has the following ingredients (which are also present in conventional MPC): a prediction horizon, a receding horizon procedure, and a regular update of the model and re-computation of the optimal control input.

This paper is organized as follows. In Section II we give a brief overview of conventional MPC for discrete-time systems. Next, we introduce discrete-event systems and max-plus-linear discrete-event systems in Section III. Section IV then considers MPC for max-plus-linear discrete-event systems. In Section V we present hybrid systems and some specific subclasses of hybrid systems: piecewise affine systems, mixed logical dynamical systems, and max-min-plus-scaling systems. An MPC approach for these classes of hybrid systems is then presented in Sections VI and VII.

For easy reference we here already list the abbreviations used in this paper:

ELCP	: extended linear complementarity problem
LP	: linear programming
MIQP	: mixed integer quadratic programming
MLD	: mixed logical dynamical
MMPS	: max-min-plus-scaling
MPC	: model predictive control
MPL	: max-plus linear
PWA	: piecewise affine
QP	: quadratic programming
SQP	: sequential quadratic programming.

II. Model predictive control

In this section we give a short and simplified introduction to conventional model predictive control (MPC) for nonlinear discrete-time systems. Since we will only consider the deterministic, i.e., noiseless, case for discrete-event and hybrid systems in this paper (cf. Remark 3.2), we will also omit the noise terms in this brief introduction to MPC for nonlinear systems. More extensive information on MPC can be found in [1], [10], [17], [22], [38], [53] and the references therein.

A. Prediction model

Consider a plant with m inputs and l outputs that can be modeled by a nonlinear discrete-time state space description of the following form:

$$x(k+1) = f(x(k), u(k)) \tag{1}$$

$$y(k) = h(x(k), u(k)) \tag{2}$$

where f and h are smooth functions of x and u.

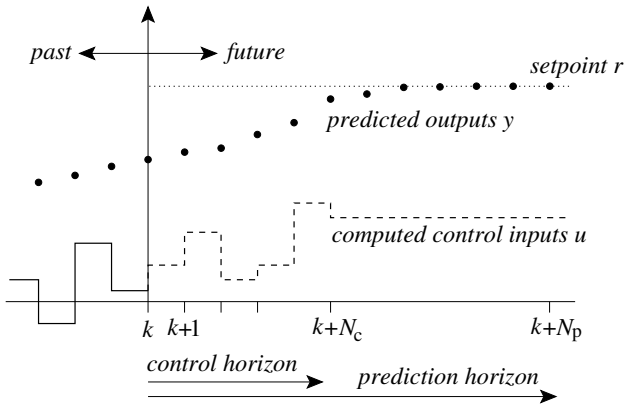

Fig. 1. Representation of the MPC control scheme.

In MPC we consider the future evolution of the system over a given prediction period $[k+1, k+N_\mathrm{p}]$, which is characterized by the prediction horizon N_p, and where k is the current sample step (see Figure 1). For the system (1)–(2) we can make an estimate $\hat{y}(k+j|k)$ of the output at sample step $k+j$ based on the state[1] $x(k)$ at step k and the future input sequence $u(k), u(k+1), \ldots, u(k+j-1)$. Using successive substitution, we obtain an expression of the form

$$\hat{y}(k+j|k) = F_j(x(k), u(k), u(k+1), \ldots, u(k+j-1)) \quad (3)$$

for $j = 1, \ldots, N_\mathrm{p}$. If we define the vectors

$$\tilde{u}(k) = \left[u^\mathrm{T}(k) \ \ldots \ u^\mathrm{T}(k+N_\mathrm{p}-1)\right]^\mathrm{T} \quad (4)$$

$$\tilde{y}(k) = \left[\hat{y}^\mathrm{T}(k+1|k) \ \ldots \ \hat{y}^\mathrm{T}(k+N_\mathrm{p}|k)\right]^\mathrm{T} \ , \quad (5)$$

we obtain the following prediction equation:

$$\tilde{y}(k) = \tilde{F}(x(k), \tilde{u}(k)) \ . \quad (6)$$

B. Cost criterion and constraints

The cost criterion J used in MPC reflects the reference tracking error (J_out) and the control effort (J_in):

$$\begin{aligned} J(k) &= J_\mathrm{out}(k) + \lambda J_\mathrm{in}(k) \\ &= \|\tilde{y}(k) - \tilde{r}(k)\|_Q^2 + \lambda \|\tilde{u}(k)\|_R^2 \\ &= \big(\tilde{y}(k) - \tilde{r}(k)\big)^\mathrm{T} Q \big(\tilde{y}(k) - \tilde{r}(k)\big) + \lambda \tilde{u}^\mathrm{T}(k) R \tilde{u}(k) \end{aligned}$$

where λ is a nonnegative integer, and $\tilde{r}(k)$ contains the reference signal (defined similarly to $\tilde{y}(k)$ (cf. (5))), and Q, R are positive definite matrices.

In practical situations, there will be constraints on the input and output signals (caused by limited capacity of buffers, limited transportation rates, saturation, etc.). This is reflected in the nonlinear constraint function

$$C_\mathrm{c}(k, \tilde{u}(k), \tilde{y}(k)) \leqslant 0 \ , \quad (7)$$

[1]For the sake of simplicity, we assume that all the components of the state can be measured, or that the system is observable such that the current state can be reconstructed from the past output sequence.

which is often taken to be linear:

$$A_\mathrm{c}(k)\tilde{u}(k) + B_\mathrm{c}(k)\tilde{y}(k) \leqslant c_\mathrm{c}(k) \ . \quad (8)$$

The MPC problem at sample step k consists in minimizing $J(k)$ over all possible future input sequences subject to the constraints. This is usually a nonconvex optimization problem. To reduce the complexity of the optimization problem a control horizon N_c is introduced in MPC, which means that the input is taken to be constant beyond sample step $k+N_\mathrm{c}$:

$$u(k+j) = u(k+N_\mathrm{c}-1) \ \text{ for } \ j = N_\mathrm{c}, \ldots, N_\mathrm{p}-1. \quad (9)$$

In addition to a decrease in the number of optimization parameters and thus also the computational burden, a smaller control horizon N_c also gives a smoother control signal, which is often desired in practical situations.

C. Receding horizon approach

MPC uses a receding horizon principle. At time step k the future control sequence $u(k), \ldots, u(k+N_\mathrm{p}-1)$ is determined such that the cost criterion is minimized subject to the constraints. At time step k the first element of the optimal sequence ($u(k)$) is applied to the process. At the next time instant the horizon is shifted, the model is updated with new information of the measurements, and a new optimization at time step $k+1$ is performed.

D. The standard MPC problem

So the MPC problem at sample step k for the nonlinear discrete-time system described by (1)–(2) is defined as follows:

> Find the input sequence $\{u(k), \ldots, u(k+N_\mathrm{p}-1)\}$ that minimizes the cost criterion $J(k)$ subject to the evolution equations (1)–(2) of the system, the nonlinear constraint (7), and the control horizon constraint (9).

For *linear* discrete-time systems and with linear constraints (8) only, the MPC problem boils down to a convex quadratic programming problem, which can be solved very efficiently. Furthermore, in this case the solution can be even computed off-line and reduces to the simple evaluation of an explicitly defined piecewise linear function [8].

Traditionally MPC uses linear discrete-time models for the process to be controlled. In this paper we consider the extension and adaptation of the MPC framework to discrete-event systems and hybrid systems. In general, MPC for discrete-event systems and hybrid systems results in hard nonconvex nonlinear and often even nonsmooth optimization problems with integer and real-valued variables, but — as we shall see — for some classes of discrete-event systems and hybrid systems tractable solution methods exist.

MPC AND DISCRETE-EVENT SYSTEMS

III. DISCRETE-EVENT SYSTEMS

A. General discrete-event systems

Typical examples of discrete-event systems are flexible manufacturing systems, telecommunication networks, parallel processing systems, traffic control systems and logistic systems. The class of discrete-event systems essentially consists of man-made systems that contain a finite number of resources (e.g., machines, communications channels, or processors) that are shared by several users (e.g., product types, information packets, or jobs) all of which contribute to the achievement of some common goal (e.g., the assembly of products, the end-to-end transmission of a set of information packets, or a parallel computation) [5].

One of the most characteristic features of a discrete-event system is that its dynamics are *event-driven* as opposed to time-driven: the behavior of a discrete-event system is governed by events rather than by ticks of a clock. An event corresponds to the start or the end of an activity. For a production system possible events are: the completion of a part on a machine, a machine breakdown, or a buffer becoming empty. Events occur at discrete time instants. Intervals between events are not necessarily identical; they can be deterministic or stochastic.

There exist many different modeling and analysis frameworks for discrete-event systems such as Petri nets, finite state machines, queuing networks, automata, (extended) state machines, semi-Markov processes, max-plus algebra, formal languages, temporal logic, perturbation analysis, process algebra, and computer models (see [5], [18], [35], [46]–[48], [62], [76] and the references therein).

Although in general discrete-event systems lead to a nonlinear description in conventional algebra, there exists a subclass of discrete-event systems for which this model becomes "linear" when we formulate it in the max-plus algebra [5], [23], [24], which has maximization and addition as its basic operations. Discrete-event systems in which only synchronization[2] and no concurrency[3] or choice occur can be modeled using the operations maximization (corresponding to synchronization: a new operation starts as soon as all preceding operations have been finished) and addition (corresponding to durations: the finishing time of an operation equals the starting time plus the duration). This leads to a description that is "linear" in the max-plus algebra. Therefore, discrete-event systems with synchronization but no concurrency are called *max-plus-linear discrete-event systems*. Typical examples are serial production lines, production systems with a fixed routing schedule, and railway networks.

[2]Synchronization requires the availability of several resources at the same time (e.g., before we can assemble a product on a machine, the machine has to be idle and the various parts have to be available).

[3]Concurrency appears when at a certain time there is a choice among several resources (e.g., a job may be executed on one of the several machines that can handle that job and that are idle at that time).

B. Max-plus algebra and max-plus-linear discrete-event systems

1) Max-plus algebra: The basic operations of the max-plus algebra are maximization and addition, which will be represented by $\oplus$ and $\otimes$ respectively:

$$x \oplus y = \max(x,y) \quad \text{and} \quad x \otimes y = x + y$$

for $x, y \in \mathbb{R}_\varepsilon \stackrel{\text{def}}{=} \mathbb{R} \cup \{-\infty\}$. Define $\varepsilon = -\infty$. The structure $(\mathbb{R}_\varepsilon, \oplus, \otimes)$ is called the max-plus algebra [5], [24]. The operations $\oplus$ and $\otimes$ are called the max-plus-algebraic addition and max-plus-algebraic multiplication respectively since many properties and concepts from linear algebra can be translated to the max-plus algebra by replacing $+$ by $\oplus$ and $\times$ by $\otimes$ [5], [24]. The rules for the order of evaluation of the max-plus-algebraic operators are similar to those of conventional algebra. So $\otimes$ has a higher priority than $\oplus$.

The matrix $\boldsymbol{\varepsilon}_{m\times n}$ is the $m \times n$ max-plus-algebraic zero matrix: $(\boldsymbol{\varepsilon}_{m\times n})_{ij} = \varepsilon$ for all i, j. The matrix E_n is the $n \times n$ max-plus-algebraic identity matrix: $(E_n)_{ii} = 0$ for all i and $(E_n)_{ij} = \varepsilon$ for all i, j with $i \neq j$. The basic max-plus-algebraic operations are extended to matrices as follows. If $A, B \in \mathbb{R}_\varepsilon^{m\times n}$, $C \in \mathbb{R}_\varepsilon^{n\times p}$ then

$$(A \oplus B)_{ij} = a_{ij} \oplus b_{ij} = \max(a_{ij}, b_{ij})$$
$$(A \otimes C)_{ij} = \bigoplus_{k=1}^{n} a_{ik} \otimes c_{kj} = \max_k (a_{ik} + c_{kj})$$

for all i, j. Note the analogy with the definitions of matrix sum and product in conventional linear algebra. The max-plus-algebraic matrix power of $A \in \mathbb{R}_\varepsilon^{n\times n}$ is defined as follows: $A^{\otimes^0} = E_n$ and $A^{\otimes^k} = A \otimes A^{\otimes^{k-1}}$ for $k = 1, 2, \ldots$

2) Max-plus-linear discrete-event systems: Discrete-event systems with only synchronization and no concurrency can be modeled by a max-plus-algebraic model of the following form [5] (see also Example 3.3):

$$x(k+1) = A \otimes x(k) \oplus B \otimes u(k) \tag{10}$$
$$y(k) = C \otimes x(k) \tag{11}$$

with $A \in \mathbb{R}_\varepsilon^{n\times n}$, $B \in \mathbb{R}_\varepsilon^{n\times m}$ and $C \in \mathbb{R}_\varepsilon^{l\times n}$ where m is the number of inputs and l the number of outputs.

For a manufacturing system, $u(k)$ would typically represent the time instants at which raw material is fed to the system for the $(k+1)$th time, $x(k)$ the time instants at which the machines start processing the kth batch of intermediate products, and $y(k)$ the time instants at which the kth batch of finished products leaves the system.

Note the analogy of the description (10)–(11) with the state space model for conventional linear time-invariant discrete-time systems. This analogy is another reason why the symbols $\oplus$ and $\otimes$ are used to denoted max and $+$.

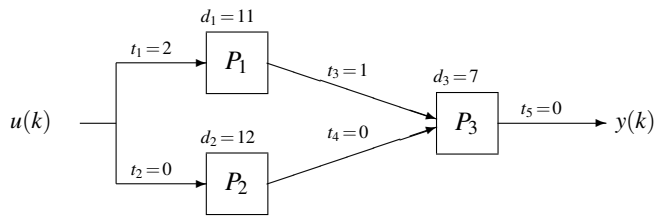

Fig. 2. A simple manufacturing system.

Remark 3.1 Apart from the fact that in (10)–(11) the components of the input, output and state vector are event times, an important difference between the descriptions (10)–(11) for discrete-event systems and (1)–(2) for discrete-time systems is that the counter k in (10)–(11) is an event counter (and event occurrence instants are in general not equidistant), whereas in (1)–(2) k is a sample counter that increases each clock cycle. □

A discrete-event system that can be modeled by (10)–(11) will be called a *max-plus-linear* (MPL) time-invariant discrete-event system.

Remark 3.2 For (linear) discrete-time systems the influence of noise is usually modeled by adding an extra noise term to the state and/or output equation. For MPL models the entries of the system matrices correspond to production times or transportation times. So instead of modeling noise (i.e., the variation in the processing times), by adding an extra max-plus-algebraic term in (10) or (11), noise should rather be modeled as an additive term to these system matrices. However, this would not lead to a nice model structure. Therefore, and for the sake of simplicity, we will use the MPL model (10)–(11) as an approximation of a discrete-event system with uncertainty and/or modeling errors when we present the extension of the MPC framework to MPL systems. This also motivates the use of a receding horizon strategy when we define MPC for MPL systems, since then we can regularly update our model of the system as new measurements become available. Note, however, that the approach given in Section IV below can also be extended to the case where noise is present (see [70], [71]). □

Example 3.3 Consider the production system of Fig. 2. This manufacturing system consists of three processing units: P_1, P_2 and P_3, and works in batches (one batch for each finished product). Raw material is fed to P_1 and P_2, processed and sent to P_3 where assembly takes place. The processing times for P_1, P_2 and P_3 are respectively $d_1 = 11$, $d_2 = 12$ and $d_3 = 7$ time units. It takes $t_1 = 2$ time units for the raw material to get from the input source to P_1, and $t_3 = 1$ time unit for a finished product of P_1 to get to P_3. The other transportation times and the set-up times are assumed to be negligible. At the input of the system and between the processing units there are buffers with a capacity that is large enough to ensure that no buffer overflow will occur. A processing unit can only start working on a new product if it has finished processing the previous product. Each processing unit starts working as soon as all parts are available.

Let $u(k)$ be the time instant at which a batch of raw material is fed to the system for the $(k+1)$th time, $x_i(k)$ the time instant at which P_i starts working for the kth time, and $y(k)$ the time instant at which the kth finished product leaves the system. Now consider $x_1(k+1)$, the time instant at which P_1 starts processing the $(k+1)$st batch. As we have to wait for the $(k+1)$st batch of raw material to arrive at P_1 (which happens at time instant $u(k)+t_1 = u(k)+2$), and for the kth batch to be processed completely at P_1 (which happens at time instant $x_1(k)+d_1 = x_1(k)+11$), and since we assume that P_1 starts processing a new batch as soon as the raw material is available and as the processing unit is idle again, we have

$$\begin{aligned} x_1(k+1) &= \max(x_1(k)+11, u(k)+2) \\ &= 11 \otimes x_1(k) \oplus 2 \otimes u(k) \ . \end{aligned}$$

In a similar way we find

$$\begin{aligned} x_2(k+1) &= \max(x_2(k)+12, u(k)+0) \\ &= 12 \otimes x_2(k) \oplus 0 \otimes u(k) \\ x_3(k+1) &= \max(x_1(k+1)+11+1, x_2(k+1)+12, \\ &\qquad x_3(k)+7) \\ &= \max(x_1(k)+11+11+1, u(k)+11+1+2, \\ &\qquad x_2(k)+12+12, u(k)+12+0, x_3(k)+7) \\ &= \max(x_1(k)+23, x_2(k)+24, x_3(k)+7, u(k)+14) \\ &= 23 \otimes x_1(k) \oplus 24 \otimes x_2(k) \oplus 7 \otimes x_3(k) \oplus 14 \otimes u(k) \\ y(k) &= x_3(k)+7 \\ &= 7 \otimes x_3(k), \end{aligned}$$

or, in max-plus-algebraic matrix notation (with $\varepsilon = -\infty$):

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 11 & \varepsilon & \varepsilon \\ \varepsilon & 12 & \varepsilon \\ 23 & 24 & 7 \end{bmatrix} \otimes x(k) \oplus \begin{bmatrix} 2 \\ 0 \\ 14 \end{bmatrix} \otimes u(k) \\ y(k) &= \begin{bmatrix} \varepsilon & \varepsilon & 7 \end{bmatrix} \otimes x(k) \ . \end{aligned}$$

□

IV. MPC FOR MPL DISCRETE-EVENT SYSTEMS

A. Prediction

Consider a discrete-event system modeled by a MPL model of the form (10)–(11). We assume that $x(k)$, the state at event step k, can be measured or estimated using previous measurements. We can then use (10)–(11) to estimate the evolution of the output of the system for the input sequence $u(k), \ldots, u(k+N_\mathrm{p}-1)$ (cf. (3)):

$$\hat{y}(k+j|k) = C \otimes A^{\otimes^j} \otimes x(k) \oplus \bigoplus_{i=0}^{j-1} C \otimes A^{\otimes^{j-i}} \otimes B \otimes u(k+i) \ ,$$

or, in matrix notation, $\tilde{y}(k) = H \otimes \tilde{u}(k) \oplus g(k)$ (cf. (6)) with

$$H = \begin{bmatrix} C\otimes B & \varepsilon & \dots & \varepsilon \\ C\otimes A\otimes B & C\otimes B & \dots & \varepsilon \\ \vdots & \vdots & \ddots & \vdots \\ C\otimes A^{\otimes^{N_\mathrm{p}-1}}\otimes B & C\otimes A^{\otimes^{N_\mathrm{p}-2}}\otimes B & \dots & C\otimes B \end{bmatrix},$$

$$g(k) = \begin{bmatrix} C\otimes A \\ C\otimes A^{\otimes^2} \\ \vdots \\ C\otimes A^{\otimes^{N_\mathrm{p}}} \end{bmatrix} \otimes x(k).$$

where $\tilde{u}(k)$ and $\tilde{y}(k)$ are defined by (4)–(5). Note the analogy between these expressions and the corresponding expressions for conventional linear time-invariant discrete-time systems.

B. Cost criterion

Just as in conventional MPC we define a cost criterion of the form

$$J(k) = J_\mathrm{out}(k) + \lambda J_\mathrm{in}(k) \ .$$

For the tracking error or output cost criterion J_out and the input cost criterion J_in several criteria are possible in a discrete-event systems context.

A straightforward translation of the cost criterion used in conventional MPC systems (with Q the identity matrix) would yield

$$\begin{aligned} J_\mathrm{out} &= \big(\tilde{y}(k) - \tilde{r}(k)\big)^\mathrm{T} \otimes \big(\tilde{y}(k) - \tilde{r}(k)\big) \\ &= 2 \bigoplus_{j=1}^{N_\mathrm{p}} \bigoplus_{i=1}^{l} \big(\hat{y}_i(k+j|k) - r_i(k+j)\big) \\ &= 2 \max_{j=1,\dots,N_\mathrm{p}} \max_{i=1,\dots,l} \big(\hat{y}_i(k+j|k) - r_i(k+j)\big) \ . \end{aligned} \tag{12}$$

This objective function does not force the difference between $\hat{y}(k+j|k)$ and $r(k+j)$ to be small since there is no absolute value in (12). Therefore, it is not very useful in practice.

If the due dates r for the finished products are known and if we have to pay a penalty for every delay, a well-suited cost criterion is the tardiness:

$$J_\mathrm{out,tard} = \sum_{j=1}^{N_\mathrm{p}} \sum_{i=1}^{l} \max(\hat{y}_i(k+j|k) - r_i(k+j), 0) \ .$$

If we have perishable goods, then we could want to minimize the differences between the due dates and the actual output time instants. This leads to

$$J_\mathrm{out,1} = \sum_{j=1}^{N_\mathrm{p}} \sum_{i=1}^{l} |\hat{y}_i(k+j|k) - r_i(k+j)| = \|\tilde{y}(k) - \tilde{r}(k)\|_1 \ .$$

If we want to balance the output rates, we could consider the following cost criterion:

$$J_\mathrm{out,\Delta} = \sum_{j=2}^{N_\mathrm{p}} \sum_{i=1}^{l} |\Delta^2 \hat{y}_i(k+j|k)|$$

where

$$\Delta^2 s(k) = \Delta s(k) - \Delta s(k-1) = s(k) - 2s(k-1) + s(k-2) \ .$$

A straightforward translation of the conventional MPC input cost criterion $\tilde{u}^\mathrm{T}(k)\tilde{u}(k)$ (where we have taken R equal to the identity matrix) would lead to a minimization of the input time instants. Since this could result in input buffer overflows, a better objective is to *maximize* the input time instants. For a manufacturing system, this would correspond to a scheme in which raw material is fed to the system as late as possible. As a consequence, the internal buffer levels are kept as low as possible. This also leads to a notion of stability if we let instability for the manufacturing system correspond to internal buffer overflows. So for MPL systems an appropriate cost criterion is

$$J_\mathrm{in,2} = -\tilde{u}^\mathrm{T}(k)\tilde{u}(k) \ .$$

Note that this is exactly the opposite of the input effort cost criterion for conventional discrete-time systems. Another objective function that leads to a maximization of the input time instants is

$$J_\mathrm{in,\Sigma} = -\sum_{j=1}^{N_\mathrm{p}} \sum_{i=1}^{m} u_i(k+j-1) \ .$$

If we want to balance the input rates we could take

$$J_\mathrm{in,\Delta} = \sum_{j=1}^{N_\mathrm{p}-1} \sum_{i=1}^{l} |\Delta^2 u_i(ik+j| \ .$$

C. Constraints

Just as in conventional MPC we can consider the linear constraint (8). Furthermore, it is easy to verify that typical constraints for discrete-event systems such as minimum or maximum separation between input and output events:

$$a_1(k+j) \leqslant \Delta u(k+j) \leqslant b_1(k+j) \ \text{ for } j = 0,\dots,N_\mathrm{c}-1, \tag{13}$$

$$a_2(k+j) \leqslant \Delta \hat{y}(k+j|k) \leqslant b_2(k+j) \ \text{ for } j = 1,\dots,N_\mathrm{p}, \tag{14}$$

or maximum due dates for the output events:

$$\hat{y}(k+j|k) \leqslant r(k+j) \quad \text{ for } j = 1,\dots,N_\mathrm{p}, \tag{15}$$

can also be recast as a linear constraint of the form (8).

Since for MPL systems the input and output sequences correspond to occurrence times of consecutive events, they should be nondecreasing. Therefore, we should always add the condition $\Delta u(k+j) \geqslant 0$ for $j = 0,\dots,N_\mathrm{p}-1$ to guarantee that the input sequences are nondecreasing.

A straightforward translation of the conventional control horizon constraint would imply that the input should stay constant from event step $k+N_\mathrm{c}$ on, which is not very useful for MPL systems since there the input sequences should normally be increasing. Therefore, we change this condition as follows: the feeding rate should stay constant beyond

event step $k+N_c$, i.e., $\Delta u(k+j) = \Delta u(k+N_c-1)$ for $j = N_c, \ldots, N_p - 1$, or

$$\Delta^2 u(k+j) = 0 \quad \text{for } j = N_c, \ldots, N_p - 1 . \tag{16}$$

This condition introduces regularity in the input sequence and it prevents the buffer overflow problems that could arise when all resources are fed to the system at the same time instant as would be implied by the conventional control horizon constraint (9).

D. The MPL MPC problem

If we combine the material of previous subsections, we finally obtain the following problem:

$$\min_{\tilde{u}(k)} J_{\text{out}}(k) + \lambda J_{\text{in}}(k) \tag{17}$$

subject to

$$\tilde{y}(k) = H \otimes \tilde{u}(k) \oplus g(k) \tag{18}$$
$$A_c(k)\tilde{u}(k) + B_c(k)\tilde{y}(k) \leqslant c_c(k) \tag{19}$$
$$\Delta u(k+j) \geqslant 0 \qquad \text{for } j = 0, \ldots, N_p - 1 \tag{20}$$
$$\Delta^2 u(k+j) = 0 \qquad \text{for } j = N_c, \ldots, N_p - 1. \tag{21}$$

This problem will be called the MPL MPC problem for event step k. MPL MPC also uses a receding horizon principle.

E. Algorithms to solve the MPL MPC problem

1) Nonlinear optimization: In general the problem (17)–(21) is a nonlinear nonconvex optimization problem: although the constraints (19)–(21) are convex in $\tilde{u}$ and $\tilde{y}$, the constraint (18) is in general not convex. So we could use standard multi-start nonlinear nonconvex local optimization methods to compute the optimal control policy.

The feasibility of the MPC-MPL problem can be verified by solving the system of (in)equalities (18)–(21)[4]. If the problem is found to be infeasible we can use the same techniques as in conventional MPC and use constraint relaxation [17].

2) The ELCP approach: Now we discuss an alternative approach which is based on the Extended Linear Complementarity problem (ELCP) [28]. Consider the ith row of (18) and define $\mathscr{J}_i = \{ j \,|\, h_{ij} \neq \varepsilon \}$. We have

$$\tilde{y}_i(k) = \max_{j \in \mathscr{J}_i} (h_{ij} + \tilde{u}_j(k), g_i(k)) \ ,$$

or, equivalently,

$$\tilde{y}_i(k) \geqslant h_{ij} + \tilde{u}_j(k) \qquad \text{for } j \in \mathscr{J}_i$$
$$\tilde{y}_i(k) \geqslant g_i(k)$$

[4]In general this is a nonlinear system of equations but if the constraints depend monotonically on the output, the feasibility problem can be recast as a linear programming problem (cf. Theorem 4.2).

with the extra condition that at least one inequality should hold with equality (i.e., at least one residue or slack variable should be equal to 0):

$$(\tilde{y}_i(k) - g_i(k)) \cdot \prod_{j \in \mathscr{J}_i} (\tilde{y}_i(k) - h_{ij} - \tilde{u}_j(k)) = 0 \ . \tag{22}$$

Hence, (18) can be rewritten as a system of the form

$$A_{\text{elcp}}\tilde{y}(k) + B_{\text{elcp}}\tilde{u}(k) + c_{\text{elcp}}(k) \geqslant 0 \tag{23}$$
$$\prod_{j \in \phi_i} (A_{\text{elcp}}\tilde{y}(k) + B_{\text{elcp}}\tilde{u}(k) + c_{\text{elcp}}(k))_j = 0 \qquad \text{for } i = 1, \ldots, lN_p \tag{24}$$

for appropriately defined matrices and vectors $A_{\text{elcp}}, B_{\text{elcp}}, c_{\text{elcp}}$ and index sets ϕ_i. We can rewrite the linear constraints (19)–(21) as

$$D_{\text{elcp}}(k)\tilde{y}(k) + E_{\text{elcp}}(k)\tilde{u}(k) + f_{\text{elcp}}(k) \geqslant 0 \tag{25}$$
$$G_{\text{elcp}}\tilde{u}(k) + h_{\text{elcp}} = 0 \ . \tag{26}$$

So the feasible set of the MPC problem (i.e., the set of feasible system trajectories) coincides with the set of solutions of the system (23)–(26), which is a special case of an Extended Linear Complementarity Problem (ELCP) [28]. In [28] we have also developed an algorithm to compute a compact parametric description of the solution set of an ELCP. In order to determine the optimal MPC policy we can use nonlinear optimization algorithms to determine for which values of the parameters the objective function J over the solution set of the ELCP (23)–(26) reaches its global minimum. The algorithm of [28] to compute the solution set of a general ELCP requires exponential execution times, which that the ELCP approach is not feasible if N_c is large.

3) Monotonically nondecreasing objective functions: Now consider the *relaxed* MPL MPC problem, which is also defined by (17)–(21) but with the =-sign in (18) replaced by a $\geqslant$-sign. Note that whereas in the original problem $\tilde{u}(k)$ is the only independent variable, since $\tilde{y}(k)$ can be eliminated using (18), the relaxed problem has both $\tilde{u}(k)$ and $\tilde{y}(k)$ as independent variables. It is easy to verify that the set of feasible solutions of the relaxed problem coincides with the set of solutions of the system of linear inequalities (23), (25), (26). So the feasible set of the relaxed MPC problem is convex. Hence, the relaxed problem is much easier to solve numerically.

A function $F : y \to F(y)$ is a monotonically nondecreasing function if $\bar{y} \leqslant \check{y}$ implies that $F(\bar{y}) \leqslant F(\check{y})$. Now we show that if the objective function J and the linear constraints are monotonically nondecreasing as a function of $\tilde{y}$ (this is the case for $J = J_{\text{out,tard}}$, $J_{\text{in},2}$, $J_{\text{in},\Sigma}$, or $J_{\text{in},\Delta}$, and, e.g., $(B_c)_{ij} \geqslant 0$ for all i, j), then the optimal solution of the relaxed problem can be transformed into an optimal solution of the original MPC problem. So in that case the optimal MPC policy can be computed very efficiently. If in addition the objective function is convex (e.g., $J = J_{\text{out,tard}}$ or $J_{\text{in},\Sigma}$), we finally get a convex optimization problem.

Remark 4.1 Note that $J_{\text{in},\Sigma}$ is a linear function. So for $J = J_{\text{in},\Sigma}$ the problem even reduces to a linear programming (LP) problem, which can be solved very efficiently. It easy to verify that for $J = J_{\text{out,tard}}$ the problem can also be reduced to an LP problem by introducing some additional dummy variables. □

Theorem 4.2: Let the objective function J and mapping $\tilde{y} \to B_{\text{c}}(k)\tilde{y}$ be monotonically nondecreasing functions of $\tilde{y}$. Let $(\tilde{u}^*, \tilde{y}^*)$ be an optimal solution of the relaxed MPC problem. If we define $\tilde{y}^\sharp = H \otimes \tilde{u}^* \oplus g(k)$ then $(\tilde{u}^*, \tilde{y}^\sharp)$ is an optimal solution of the original MPC problem.

Proof: First we show that $(\tilde{u}^*, \tilde{y}^\sharp)$ is a feasible solution of the original problem. Clearly, $(\tilde{u}^*, \tilde{y}^\sharp)$ satisfies (18), (20) and (21). Since $\tilde{y}^* \geqslant H \otimes \tilde{u}^* \oplus g(k) = \tilde{y}^\sharp$ and since $\tilde{y} \to B_{\text{c}}(k)\tilde{y}$ is monotonically nondecreasing, we have

$$A_{\text{c}}(k)\tilde{u}^* + B_{\text{c}}(k)\tilde{y}^\sharp \leqslant A_{\text{c}}(k)\tilde{u}^* + B_{\text{c}}(k)\tilde{y}^* \leqslant c_{\text{c}}(k) \ .$$

So $(\tilde{u}^*, \tilde{y}^\sharp)$ also satisfies the constraint (19). Hence, $(\tilde{u}^*, \tilde{y}^\sharp)$ is a feasible solution of the original problem. Since the set of feasible solutions of the original problem is a subset of the set of feasible solutions of the relaxed problem, we have $J(\tilde{u}, \tilde{y}) \geqslant J(\tilde{u}^*, \tilde{y}^*)$ for any feasible solution $(\tilde{u}, \tilde{y})$ of the original problem. Hence, $J(\tilde{u}^*, \tilde{y}^\sharp) \geqslant J(\tilde{u}^*, \tilde{y}^*)$. On the other hand, we have $J(\tilde{u}^*, \tilde{y}^\sharp) \leqslant J(\tilde{u}^*, \tilde{y}^*)$ since $\tilde{y}^\sharp \leqslant \tilde{y}^*$ and since J is a monotonically nondecreasing function of $\tilde{y}$. As a consequence, we have $J(\tilde{u}^*, \tilde{y}^\sharp) = J(\tilde{u}^*, \tilde{y}^*)$, which implies that $(\tilde{u}^*, \tilde{y}^\sharp)$ is an optimal solution of the original MPC problem. ■

F. Example

Consider the production system of Example 3.3. Let us now compare the efficiency of the methods discussed in Section IV-E when solving one step of the MPC problem for the objective function $J(k) = J_{\text{out,tard}}(k) + J_{\text{in},\Sigma}(k)$ (so $\lambda = 1$) with the additional constraints $2 \leqslant \Delta u(k+j) \leqslant 12$ for $j = 0, \ldots, N_{\text{c}} - 1$. We take $N_{\text{c}} = 5$ and $N_{\text{p}} = 8$. Assume that $k = 0$, $x(0) = [\ 0\ \ 0\ \ 10\ ]^{\text{T}}$, $u(-1) = 0$, and $\tilde{r}(k) = [\,40\ 45\ 55\ 66\ 75\ 85\ 90\ 100\,]^{\text{T}}$.

The objective function J and the linear constraints are monotonically nondecreasing as a function of $\tilde{y}$ so that we can apply Theorem 4.2. We have computed a solution $\tilde{u}_{\text{elcp}}$ obtained using the ELCP method and the ELCP algorithm of [28], a solution $\tilde{u}_{\text{nlcon}}$ using nonlinear constrained optimization, a solution $\tilde{u}_{\text{penalty}}$ using linearly constrained optimization with a penalty function for the nonlinear constraints, a solution $\tilde{u}_{\text{relaxed}}$ for the relaxed MPC problem, and an LP solution $\tilde{u}_{\text{lp}}$ (cf. Remark 4.1). For the nonlinear constrained optimization we have used a sequential quadratic programming algorithm, and for the linear optimization a variant of the simplex algorithm. All these methods result in the same optimal input sequence:

$$\{u_{\text{opt}}\}_{k=0}^{7} = 12, 24, 35, 46, 58, 70, 82, 94.$$

TABLE I

THE CPU TIME NEEDED TO COMPUTE THE OPTIMAL INPUT SEQUENCE VECTORS FOR THE EXAMPLE OF SECTION IV-F FOR $N_{\text{c}} = 4, 5, 6, 7$. FOR $N_{\text{c}} = 7$ WE HAVE NOT COMPUTED THE ELCP SOLUTION SINCE IT REQUIRES TOO MUCH CPU TIME.

$\tilde{u}_{\text{opt}}$	CPU time			
	$N_{\text{c}} = 4$	$N_{\text{c}} = 5$	$N_{\text{c}} = 6$	$N_{\text{c}} = 7$
$\tilde{u}_{\text{elcp}}$	5.525	106.3	287789	—
$\tilde{u}_{\text{nlcon}}$	0.870	1.056	1.319	1.470
$\tilde{u}_{\text{penalty}}$	0.826	0.988	1.264	1.352
$\tilde{u}_{\text{relaxed}}$	0.431	0.500	0.562	0.634
$\tilde{u}_{\text{lp}}$	0.029	0.030	0.031	0.032

The corresponding output sequence is

$$\{y_{\text{opt}}(k)\}_{k=1}^{8} = 33, 45, 56, 67, 79, 91, 103, 115,$$

and the corresponding value of the objective function is $J = -381$.

In Table I we have listed the CPU time needed to compute the various input sequence vectors $\tilde{u}$ for $N_{\text{c}} = 4, 5, 6, 7$ on a Pentium II 300 MHz PC with the optimization routines called from MATLAB and implemented in C (average values over 10 experiments). For the input sequence vectors that have to be determined using a nonlinear optimization algorithm selecting different (feasible) initial points always leads to the same numerical value of the final objective function (within a certain tolerance). Therefore, we have only performed one run with a random feasible initial point for each of these cases.

The CPU time for the ELCP algorithm of [28] increases exponentially as the number of variables increases (see also Table I). So in practice the ELCP approach cannot be used for on-line computations if the control horizon or the number of inputs or outputs are large. In that case one of the other methods should be used instead. Looking at Table I we see that the $\tilde{u}_{\text{lp}}$ solution — which is based on Theorem 4.2 and an LP approach — is clearly the most interesting. For more results we refer the interested reader to [30].

G. Extensions and related research

The approach presented above can also be extended to the case where modeling errors and disturbances are present [70], [71]. Tuning rules and properties such as stability, timing issues, etc. have been discussed in [69]. The method derived above can also be used for MPC for discrete-event systems with hard and soft synchronization constraints [33] such as railway networks (where some connections may be broken — but at a cost — if delays become too large), or logistic systems.

Above we have presented an MPC framework for MPL discrete-event systems. Several other authors have already

developed methods to compute optimal control sequences for MPL discrete-event systems [5], [12], [51], [54]–[56]. Compared to these methods one of the main advantages of the MPL MPC approach is that it allows to include general linear inequality constraints on the inputs and outputs of the system such as (19), or even simple constraints of the form (13) or (14).

MPC AND HYBRID SYSTEMS

V. HYBRID SYSTEMS

A. General hybrid systems

Hybrid systems arise throughout business and industry in areas such as interactive distributed simulation, traffic control, plant process control, aircraft and robot design, and path planning. There are several possible definitions of hybrid systems. For some authors a hybrid system is a coupling of a continuous-time or analog system and a discrete-time or digital system (in practice often a continuous-time, analog plant and an asynchronous, digital controller). We shall use a somewhat different definition.

For us, hybrid systems arise from the interaction between continuous-variable systems[5] and discrete-event systems. In general we could say that a hybrid system can be in one of several modes of operation, whereby in each *mode* the behavior of the system can be described by a system of difference or differential equations, and that the system switches from one mode to another due to the occurrence of events (see Figure 3). The mode transitions may be caused by an external control signal, by an internal control signal (if the controller is already included in the system under consideration), or by the dynamics of the system itself, i.e., when a certain boundary in the state space is crossed. At a switching time instant there may be a reset of the state (i.e., a jump in the values of the state variables) and/or the dimension of the state may change.

There are many modeling and analysis techniques for hybrid systems. Typical modeling techniques are predicate calculus, real-time temporal logics, timed communicating sequential processes, hybrid automata, timed automata, timed Petri nets, and object-oriented modeling languages such as Modelica, SHIFT or Chi. Current analysis techniques for hybrid systems include formal verification, perturbation analysis, and computer simulation. Furthermore, special mathematical analysis techniques have been developed for specific subclasses of hybrid systems. We shall only discuss some of these methods. For more information on the other methods the interested reader is referred to [3], [4], [45], [57], [68], [73] and the references cited therein. An important trade-off in this context is that of modeling power versus

[5]Continuous variable systems are systems that can be described by a difference or differential equation.

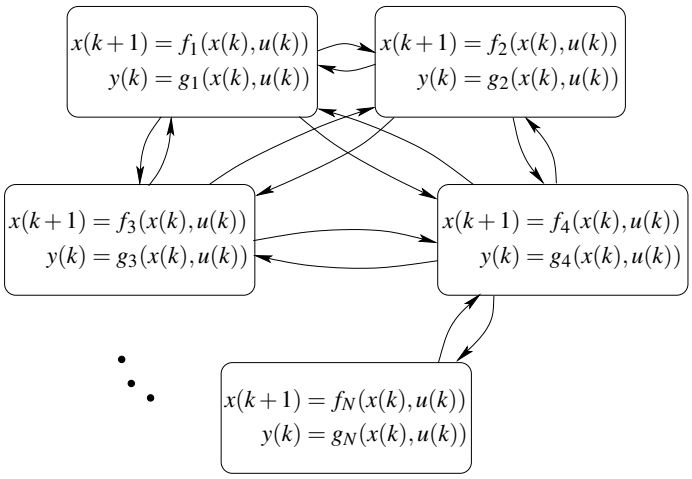

Fig. 3. Schematic representation of a hybrid system with N modes. In each mode the behavior of the hybrid system is described by a system of difference (or differential) equations. The system goes from one mode to another due to the occurrence of an event (this is indicated by the arrows).

decision power: the more accurate the model is the less we can analytically say about its properties. Furthermore, many analysis and control problems lead to computationally hard problems for even the most elementary hybrid systems [11]. As tractable methods to analyze general hybrid systems are not available, several authors have focused on special subclasses of hybrid dynamical systems for which analysis and/or control design techniques are currently being developed. Some examples of such subclasses are:

- mixed logical dynamical (MLD) systems [6], [7],
- piecewise-affine (PWA) systems [67],
- linear complementarity systems [44], [72],
- extended linear complementarity systems [43],
- max-min-plus scaling (MMPS) systems [26],
- timed automata [2],
- timed Petri nets [58], [63].

In this paper we will consider PWA systems, MLD systems, and MMPS systems. Note that some of these classes (in particular MLD, PWA, (extended) linear complementarity and constrained MMPS systems) are equivalent [43], possibly under mild additional assumptions related to well-posedness and boundedness of input, state, output or auxiliary variables. Each subclass has its own advantages over the others. E.g., stability criteria were proposed for PWA systems [49], control and verification techniques for MLD hybrid models [6], [7], [9] and for MMPS systems [26], [29], [31], and conditions of existence and uniqueness of solution trajectories (well-posedness) for linear complementarity systems [44], [72]. So it really depends on the application, which of these classes is best suited.

In the next sections we will discuss some tractable classes of hybrid systems, for which MPC control design methods are available as we will see in Sections VI and VII.

Note that in the next sections k is again a sample step counter and not on event step counter k as in Sections III and IV, which dealt with discrete-event systems.

B. PWA systems

PieceWise Affine (PWA) systems [67] are described by

$$\begin{aligned} x(k+1) &= A_i x(k) + B_i u(k) + f_i \\ y(k) &= C_i x(k) + D_i u(k) + g_i \end{aligned} \quad \text{for} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \Omega_i, \tag{27}$$

for $i = 1, \ldots, N$ where $\Omega_1, \ldots, \Omega_N$ are convex polyhedra (i.e., given by a finite number of linear inequalities) in the input/state space. PWA systems have been studied by several authors (see [6], [21], [49], [50], [67] and the references therein) as they form the "simplest" extension of linear systems that can still model nonlinear and nonsmooth processes with arbitrary accuracy and are capable of handling hybrid phenomena.

Example 5.1 As a very simple example of a PWA model we can consider an integrator with upper saturation:

$$\begin{aligned} x(k+1) &= \begin{cases} x(k) + u(k) & \text{if } x(k) + u(k) \leqslant 1 \\ 1 & \text{if } x(k) + u(k) \geqslant 1 \,. \end{cases} \\ y(k) &= x(k) \end{aligned} \tag{28}$$

If we rewrite (28) as in (27) then we have

$$\begin{aligned} &\Omega_1 = \left\{ (x(k), u(k)) \in \mathbb{R}^2 \,\middle|\, x(k) + u(k) \leqslant 1 \right\} \\ &\Omega_2 = \left\{ (x(k), u(k)) \in \mathbb{R}^2 \,\middle|\, x(k) + u(k) \geqslant 1 \right\} \\ &A_1 = 1, \quad A_2 = 0, \quad B_1 = 1, \quad B_2 = 0 \\ &f_1 = 0, \quad f_2 = 1, \quad C_1 = C_2 = 1 \\ &D_1 = D_2 = 0, \quad g_1 = g_2 = 0 \ . \end{aligned}$$

□

C. MLD systems[6]

1) Preliminaries: First, we will provide some tools to transform logical statements involving continuous variables into mixed-integer linear inequalities.

We will use capital letters X_1 to represent statements, e.g., "$x \leq 0$" or "temperature is hot"; X_i is commonly referred to as a literal, and has a truth value of either "T" (true) or "F" (false). We use the following notation for boolean connectives: "$\wedge$" (and), "$\vee$" (or), "$\sim$" (not), "$\Rightarrow$" (implies), "$\Leftrightarrow$" (if and only if), "**xor**" (exclusive or). These connectives are defined by means of the truth table given in Table II, and they satisfy several properties (see, e.g., [20]), such as:

$$X_1 \Rightarrow X_2 \quad \text{is the same as} \quad \sim X_1 \vee X_2 \tag{29}$$

$$X_1 \Rightarrow X_2 \quad \text{is the same as} \quad \sim X_2 \Rightarrow \sim X_1 \tag{30}$$

$$X_1 \Leftrightarrow X_2 \quad \text{is the same as} \quad (X_1 \Rightarrow X_2) \wedge (X_2 \Rightarrow X_1) \,. \tag{31}$$

One can associate with a literal X_i a logical variable $\delta_i \in \{0,1\}$, which has a value of either 1 if $X_i = \text{T}$, or 0 if $X_i = \text{F}$. A propositional logic problem, where a statement X must be proved to be true given a set of (compound) statements involving literals $X_1, \ldots, X_n$, can thus be solved by means of an integer linear program, by suitably translating the original compound statements into linear inequalities involving logical variables $\delta_1, \ldots, \delta_n$. In fact, the following propositions and linear constraints can easily be seen to be equivalent [74, p. 176]:

$$X_1 \wedge X_2 \quad \text{is equivalent to} \quad \delta_1 = \delta_2 = 1 \tag{32}$$

$$X_1 \vee X_2 \quad \text{is equivalent to} \quad \delta_1 + \delta_2 \geq 1 \tag{33}$$

$$\sim X_1 \quad \text{is equivalent to} \quad \delta_1 = 0 \tag{34}$$

$$X_1 \Rightarrow X_2 \quad \text{is equivalent to} \quad \delta_1 - \delta_2 \leq 0 \tag{35}$$

$$X_1 \Leftrightarrow X_2 \quad \text{is equivalent to} \quad \delta_1 - \delta_2 = 0 \tag{36}$$

$$X_1 \,\mathbf{xor}\, X_2 \quad \text{is equivalent to} \quad \delta_1 + \delta_2 = 1 \ . \tag{37}$$

We can use this computational inference technique to model logical parts of processes (on/off switches, discrete mechanisms, combinational and sequential networks), and heuristics knowledge about plant operation as integer linear inequalities. In this way we can construct models of hybrid systems.

As we are interested in systems which contain both logic and continuous dynamics, we wish to establish a link between the two worlds. As will be shown next, we end up with mixed-integer linear inequalities, i.e., linear inequalities involving both continuous variables $x \in \mathbb{R}^n$ and logical variables $\delta \in \{0,1\}^{n_\delta}$.

Consider the statement $X \stackrel{\text{def}}{=} [f(x) \leq 0]$, where $f : \mathbb{R}^n \mapsto \mathbb{R}$. Assume that $x \in \mathscr{X}$, where $\mathscr{X}$ is a given bounded set, and define

$$M = \max_{x \in \mathscr{X}} f(x), \qquad m = \min_{x \in \mathscr{X}} f(x) \ . \tag{38}$$

Theoretically, an over-estimate (under-estimate) of M (m) suffices for our purpose. However, more realistic estimates provide computational benefits [74, p. 171]. Now it is easy to verify that

$$[f(x) \leq 0] \wedge [\delta = 1] \quad \text{is true if and only if} \quad f(x) - \delta \leq -1 + m(1-\delta) \tag{39}$$

$$[f(x) \leq 0] \vee [\delta = 1] \quad \text{is true if and only if} \quad f(x) \leq M\delta \tag{40}$$

$$\sim [f(x) \leq 0] \quad \text{is true if and only if} \quad f(x) \geq \varepsilon_{\text{tol}} \ , \tag{41}$$

where ε_{tol} is a small tolerance (typically the machine precision), beyond which the constraint is regarded as violated. By (29) and (40), it also follows that

$$[f(x) \leq 0] \Rightarrow [\delta = 1] \quad \text{is true if and only if} \quad f(x) \geq \varepsilon + (m - \varepsilon)\delta \tag{42}$$

$$[f(x) \leq 0] \Leftrightarrow [\delta = 1] \quad \text{is true if and only if}$$

[6]This section is based on [7].

TABLE II
TRUTH TABLE.

X_1	X_2	$X_1 \wedge X_2$	$X_1 \vee X_2$	$\sim X_1$	$X_1 \Rightarrow X_2$	$X_1 \Leftrightarrow X_2$	X_1 **xor** X_2
T	T	T	T	F	T	T	F
T	F	F	T	F	F	F	T
F	T	F	T	T	F	F	T
F	F	F	F	T	F	T	T

$$\begin{cases} f(x) \leq M(1-\delta) \\ f(x) \geq \varepsilon + (m-\varepsilon)\delta \,. \end{cases} \tag{43}$$

Finally, we report procedures to transform products of logical variables, and of continuous and logical variables, in terms of linear inequalities, which however require the introduction of auxiliary variables [74, p. 178]. The product term $\delta_1\delta_2$ can be replaced by an auxiliary logical variable $\delta_3 = \delta_1\delta_2$. Then, $[\delta_3 = 1] \Leftrightarrow [\delta_1 = 1] \wedge [\delta_2 = 1]$, and therefore

$$\delta_3 = \delta_1\,\delta_2 \quad \text{is equivalent to} \quad \begin{cases} -\delta_1 + \delta_3 \leq 0 \\ -\delta_2 + \delta_3 \leq 0 \\ \delta_1 + \delta_2 - \delta_3 \leq 1 \,. \end{cases}$$

Moreover, the term $\delta f(x)$, where $f : \mathbb{R}^n \mapsto \mathbb{R}$ and $\delta \in \{0,1\}$, can be replaced by an auxiliary real variable $y = \delta f(x)$ that satisfies $[\delta = 0] \Rightarrow [y = 0]$, $[\delta = 1] \Rightarrow [y = f(x)]$. Therefore, by defining M and m as in (38), $y = \delta f(x)$ is equivalent to

$$y \leq M\delta \tag{44}$$
$$y \geq m\delta \tag{45}$$
$$y \leq f(x) - m(1-\delta) \tag{46}$$
$$y \geq f(x) - M(1-\delta) \;\; . \tag{47}$$

2) MLD systems: The results of the previous section will now be used now to express relations describing the evolution of systems where physical laws, logic rules, and operating constraints are interdependent. Before giving a general definition, we first consider an example.

Example 5.2 Consider the following PWA system:

$$x(k+1) = \begin{cases} 0.8\,x(k) + u(k) & \text{if } x(k) \geq 0 \\ -0.8\,x(k) + u(k) & \text{if } x(k) < 0 \end{cases} \tag{48}$$

where $x(k) \in [-10, 10]$, and $u(k) \in [-1, 1]$. The condition $x(k) \geq 0$ can be associated with a binary variable $\delta(k)$ such that

$$[\delta(k) = 1] \Leftrightarrow [x(k) \geq 0] \;\; .$$

By using the transformation (43), this equation can be expressed by the inequalities

$$\begin{aligned} -m\delta(k) &\leq x(k) - m \\ -(M+\varepsilon)\delta &\leq -x - \varepsilon \;\; , \end{aligned}$$

where $M = -m = 10$, and ε is a small positive scalar. Then (48) can be rewritten as

$$x(k+1) = 1.6\,\delta(k)x(k) - 0.8\,x(k) + u(k) \;\; .$$

By defining a new variable $z(k) = \delta(k)x(k)$ which, by (44)–(47), can be expressed as

$$z(k) \leq M\delta(k) \tag{49}$$
$$z(k) \geq m\delta(k) \tag{50}$$
$$z(k) \leq x(k) - m(1-\delta(k)) \tag{51}$$
$$z(k) \geq x(k) - M(1-\delta(k)) \;\; , \tag{52}$$

the evolution of system (48) is ruled by the linear equation

$$x(k+1) = 1.6z(k) - 0.8x(k) + u(k)$$

subject to the linear constraints (49)–(52). □

This example can be generalized by describing systems through the following linear relations:

$$x(k+1) = Ax(k) + B_1u(k) + B_2\delta(k) + B_3z(k) \tag{53}$$
$$y(k) = Cx(k) + D_1u(k) + D_2\delta(k) + D_3z(k) \tag{54}$$
$$E_1x(k) + E_2u(k) + E_3\delta(k) + E_4z(k) \leqslant g_5, \tag{55}$$

where $x(k) = [\,x_{\rm r}^{\rm T}(k) \;\; x_{\rm b}^{\rm T}(k)\,]^{\rm T}$ with $x_{\rm r}(k) \in \mathbb{R}^{n_{\rm r}}$ and $x_{\rm b}(k) \in \{0,1\}^{n_{\rm b}}$ ($y(k)$ and $u(k)$ have a similar structure), and where $z(k) \in \mathbb{R}^{r_{\rm r}}$ and $\delta(k) \in \{0,1\}^{r_{\rm b}}$ are auxiliary variables. Systems of the form (53)–(55) are called *Mixed Logical Dynamical* (MLD) systems.

The MLD formalism allows specifying the evolution of continuous variables through linear dynamic equations, of discrete variables through propositional logic statements and automata, and the mutual interaction between the two. As explained above the key idea of the approach consists of embedding the logic part in the state equations by transforming boolean variables into 0-1 integers, and by expressing the relations as mixed-integer linear inequalities. MLD systems are therefore capable of modeling a broad class of systems, such as PWA systems, linear hybrid systems, finite state machines, (bi)linear systems with discrete inputs, etc. [7].

Remark 5.3 It is assumed that for all $x(k)$ with $x_{\rm b}(k) \in \{0,1\}^{n_{\rm b}}$, all $u(k)$ with $u_{\rm b}(k) \in \{0,1\}^{m_{\rm b}}$, all $z(k) \in \mathbb{R}^{r_{\rm r}}$, and all $\delta(k) \in \{0,1\}^{r_{\rm b}}$ satisfying (55) it holds that $x(k+1)$ and $y(k)$ determined from (53)–(54) are such that $x_{\rm b}(k+1) \in \{0,1\}^{n_{\rm b}}$ and $y_{\rm b}(k) \in \{0,1\}^{l_{\rm b}}$. This is without loss of generality, as we

can take binary components of states and outputs (if any) to be auxiliary variables as well (see proof of [6, Prop. 1]). □

D. MMPS systems

In [31] a class of discrete-event systems and hybrid systems has been introduced that can be modeled using the operations maximization, minimization, addition and scalar multiplication. Expressions that are built using these operations are called *Max-Min-Plus-Scaling* (MMPS) expressions.

Definition 5.4: An MMPS expression f of the variables $x_1, \ldots, x_n$ is defined by the grammar[7]

$$f := x_i | \alpha | \max(f_k, f_l) | \min(f_k, f_l) | f_k + f_l | \beta f_k \tag{56}$$

with $i \in \{1, 2, \ldots, n\}$, α, $\beta \in \mathbb{R}$, and where f_k, f_l are again MMPS expressions.
An MMPS example of an expression is, e.g., $3x_1 - 8x_2 + 9 + \max(\min(3x_1, -9x_2), -x_2 - 3x_3)$.

Consider now systems that can be described by

$$x(k+1) = \mathscr{M}_x(x(k), u(k), d(k)) \tag{57}$$
$$y(k) = \mathscr{M}_y(x(k), u(k), d(k)), \tag{58}$$

where $\mathscr{M}_x$, $\mathscr{M}_y$ are MMPS expressions in terms of the components of $x(k)$, $u(k)$ and the auxiliary variables $d(k)$, which are all real-valued. Such systems will be called MMPS systems. If in addition, we have a condition of the form

$$\mathscr{M}_c(x(k), u(k), d(k)) \leqslant c(k) \ ,$$

with $\mathscr{M}_c$ an MMPS expression, we speak about *constrained* MMPS systems. A typical example of an (unconstrained) MMPS is a traffic-signal controlled intersection [26].

Example 5.5 It is easy to verify that if we recast the PWA model (28) of Example 5.1 into an MMPS model we obtain

$$x(k+1) = \min(x(k) + u(k), 1)$$
$$y(k) = x(k) \ . \qquad \square$$

E. Equivalence between continuous PWA systems and MMPS systems

In [43] it has been shown that PWA systems, MLD systems and constrained MMPS systems are equivalent under mild additional assumptions related to well-posedness and boundedness of input, state, output or auxiliary variables. Now we consider another equivalence between *continuous* PWA systems and (*unconstrained*) MMPS systems in more detail as it will be the basis for the MPC approach derived in Section VII below.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be a *continuous* PWA function if and only if the following conditions hold [21]:

1) The domain space $\mathbb{R}^n$ is divided into a finite number of polyhedral regions $R_{(1)}, \ldots, R_{(N)}$.
2) For each $i \in \{1, \ldots, N\}$, f can be expressed as
$$f(x) = \alpha_{(i)}^{\mathrm{T}} x + \beta_{(i)} \tag{59}$$
for any $x \in R_{(i)}$ with $\alpha_{(i)} \in \mathbb{R}^n$ and $\beta_{(i)} \in \mathbb{R}$.
3) f is continuous on any boundary between two regions.

A continuous PWA system is a system of the form

$$x(k) = \mathscr{P}_x(x(k-1), u(k)) \tag{60}$$
$$y(k) = \mathscr{P}_y(x(k), u(k)) \ , \tag{61}$$

where $\mathscr{P}_x$ and $\mathscr{P}_y$ are vector-valued continuous PWA functions. Note that the main difference with the PWA systems introduced in Section V-B is that now we require the PWA functions to be continuous on the boundary between the regions that make up the partition of the domain.

Theorem 5.6 ([40], [60]*):* If f is a continuous PWA function of the form (59), then there exist index sets $I_1, \ldots, I_\ell \subseteq \{1, \ldots, N\}$ such that

$$f = \max_{j=1,\ldots,\ell} \min_{i \in I_j} (\alpha_{(i)}^{\mathrm{T}} x + \beta_{(i)}) \ .$$

From the definition of MMPS functions it follows that (see also [40], [60]):

Lemma 5.7: Any MMPS function is also a continuous PWA function.

From Theorem 5.6 and Lemma 5.7 it follows that continuous PWA systems and (unconstrained) MMPS systems are equivalent, i.e., for a given continuous PWA model there exists an MMPS model (and vice versa) such that the input-output behavior of both models coincides.

Corollary 5.8: Continuous PWA models and (unconstrained) MMPS models are equivalent.

Note that this is an extension of the results of [43], which prove an equivalence between (not necessarily continuous) PWA models and MMPS models, but there some extra auxiliary variables and some additional algebraic MMPS constraints between the states, the inputs and the auxiliary variables were required to transform the PWA model into an MMPS model.

VI. MPC FOR MLD SYSTEMS[8]

A. The MLD MPC problem

An important control problem for MLD systems is to stabilize the system to an equilibrium state or to track a desired reference trajectory. In general finding a control law that attains these objectives for an MLD system is not an easy task, as in general MLD systems are neither linear[9] nor even smooth. MPC provides a successful tool to perform these task, as will be shown next. For the sake of brevity we will concentrate on the stabilization to an equilibrium state.

Consider the MLD system (53)–(55) and an equilibrium state/input/output triple $(x_{\mathrm{eq}}, u_{\mathrm{eq}}, y_{\mathrm{eq}})$, and let $(\delta_{\mathrm{eq}}, z_{\mathrm{eq}})$ be the corresponding pair of auxiliary variables. Now we consider

[7] The symbol | stands for OR and the definition is recursive.

[8] This section is based on [7].

[9] Due to the integer constraints $\delta_i \in \{0, 1\}$, the linear inequality (55) results in a nonlinear relation between δ and x, u, and between z and x, u.

the MLD MPC problem with objective function

$$J(k) = \sum_{j=1}^{N_\mathrm{p}} \|\hat{x}(k+j|k) - x_\mathrm{eq}\|_{Q_x}^2 + \|u(k+j-1) - u_\mathrm{eq}\|_{Q_u}^2 + \|\hat{y}(k+j|k) - y_\mathrm{eq}\|_{Q_y}^2 + \|\hat{\delta}(k+j-1|k) - \delta_\mathrm{eq}\|_{Q_\delta}^2 + \|\hat{z}(k+j-1|k) - z_\mathrm{eq}\|_{Q_z}^2$$

where Q_u, Q_x are positive definite matrices, and Q_y, Q_δ, Q_z are nonnegative definite matrices. Furthermore, we have the end-point condition

$$\hat{x}(k+N_\mathrm{p}|k) = x_\mathrm{eq}$$

in addition to the MLD system equations, and possibly also a control horizon constraint[10] of the form (9). Now we have

Theorem 6.1 ([7]): Consider an MLD system (53)–(55) and an equilibrium state/input/output triple $(x_\mathrm{eq}, u_\mathrm{eq}, y_\mathrm{eq})$, and let $(\delta_\mathrm{eq}, z_\mathrm{eq})$ be the corresponding pair of auxiliary variables. Assume that the initial state $x(0)$ is such that a feasible solution of the MLD MPC problem exists for sample step 0. The input signal resulting from applying the optimal MLD MPC input signal in a receding horizon approach stabilizes the MLD system in the sense that

$$\lim_{k\to\infty} x(k) = x_\mathrm{eq}\,, \qquad \lim_{k\to\infty} \|y(k) - y_\mathrm{eq}\|_{Q_y} = 0\,,$$
$$\lim_{k\to\infty} u(k) = u_\mathrm{eq}\,, \qquad \lim_{k\to\infty} \|\delta(k) - \delta_\mathrm{eq}\|_{Q_\delta} = 0\,,$$
$$\lim_{k\to\infty} \|z(k) - z_\mathrm{eq}\|_{Q_z} = 0\ .$$

B. Algorithms

Let us now show that the MLD MPC problem can be recast as a mixed integer quadratic programming (MIQP) problem. For the MLD case using successive substitution for (53) results in the following prediction equation for the state:

$$\hat{x}(k+j|k) = A^j x(k) + \sum_{i=0}^{j-1} A^{j-i}\big(B_1 u(k+i) + B_2\delta(k+i) + B_3 z(k+i)\ .$$

For $\hat{y}(k+j|k)$ we have a similar expression. If we define $\tilde{\delta}(k)$ and $\tilde{z}(k)$ in a similar way as $\tilde{u}(k)$ (cf. (4)), and if we define

$$\tilde{V}(k) = \left[\tilde{u}^\mathrm{T}(k)\ \tilde{y}^\mathrm{T}(k)\ \tilde{\delta}^\mathrm{T}(k)\ \tilde{z}^\mathrm{T}(k)\right]^\mathrm{T}\ ,$$

we obtain the following equivalent formulation for the MLD MPC problem:

$$\min_{\tilde{V}(k)} \tilde{V}^\mathrm{T}(k) S_1 \tilde{V}(k) + 2(S_2 + x^\mathrm{T}(k) S_3)\tilde{V}(k) \tag{62}$$

$$\text{subject to } F_1\tilde{V}(k) \leqslant F_2 + F_3 x(k)\ , \tag{63}$$

for appropriately defined matrices S_1, S_2, S_3, F_1, F_2, F_3. Note that $\tilde{V}(k)$ contains both real-valued and integer-valued components. As the objective function is quadratic, the equivalent problem is an MIQP problem.

MIQP problems are classified as NP-hard [39], [66], which — loosely speaking — means that, in the worst case, the solution time grows exponentially with the problem size. Several algorithmic approach have been applied successfully to medium and large-size application problems [37], the four major ones being cutting plane methods, decomposition methods, logic-based methods, and branch-and-bound methods. In [7] the authors use a branch-and-bound method as several authors seems to agree on the fact that branch-and-bound methods are the most successful for mixed integer programming problems [36].

As described by [7], [36], the branch-and-bound algorithm for MIQP consists of solving and generating new quadratic programming (QP) problems in accordance with a tree search, where the nodes of the tree correspond to QP subproblems. The QP subproblems involve real-valued variables only, and are thus efficiently solvable using a modified simplex method or an interior point method [59], [61], [75].

For a worked example of the MLD MPC approach we refer the interested reader to [7].

VII. MPC for continuous PWA and MMPS systems

In the previous section we have already discussed how the MPC problem for MLD systems (and thus also PWA systems) can be recast as an MIQP problem. In this section we will define the MMPS MPC problem and show that for (unconstrained) MMPS systems, and thus also for *continuous* PWA systems, the MPC problem can be transformed into solving a sequence of real (so not integer!) LP problems. The approach we propose is based on canonical forms for MMPS functions, which are introduced below, and is similar to the cutting-plane algorithm for convex optimization problems.

A. Canonical forms of MMPS functions

Theorem 7.1: Any MMPS function $f : \mathbb{R}^n \to \mathbb{R}$ can be rewritten in the min-max canonical form

$$f = \min_{i=1,\ldots,K} \max_{j=1,\ldots,n_i} \left(\alpha_{(i,j)}^\mathrm{T} x + \beta_{(i,j)}\right) \tag{64}$$

or in the max-min canonical form

$$f = \max_{i=1,\ldots,L} \min_{j=1,\ldots,m_i} \left(\gamma_{(i,j)}^\mathrm{T} x + \delta_{(i,j)}\right) \tag{65}$$

for some integers K, L, $n_1,\ldots,n_K$, $m_1,\ldots,m_L$, vectors $\alpha_{(i,j)}, \gamma_{(i,j)}$, and real numbers $\beta_{(i,j)}, \delta_{(i,j)}$.

Proof: We will only sketch the proof of the theorem (see [34] for the full proof). Moreover, we only consider the min-max canonical form since the proof for the max-min canonical form is similar.

It is easy to verify that if f_k and f_l are affine functions, then the functions that results from applying the basic constructors of an MMPS function (max, min, +, and scaling — cf. (56))

[10] While in other contexts introducing a control horizon constraint amounts to hugely down-sizing the optimization problem at the price of a reduced performance, for MLD systems the computational gain is only partial, since all the (auxiliary) variables $\hat{\delta}(k+\ell|k)$ and $\hat{z}(k+\ell|k)$ for $\ell = N_\mathrm{c},\ldots,N_\mathrm{p}-1$ remain in the optimization.

are in min-max canonical form[11].

Now we can use a recursive argument that consists in showing that if we apply the basic constructors of an MMPS function to two (or more) MMPS functions in min-max canonical form, then the result can again be transformed into min-max canonical form. Consider two MMPS functions f and g in min-max canonical form[12]: $f = \min(\max(f_1, f_2), \max(f_3, f_4))$ and $g = \min(\max(g_1, g_2), \max(g_3, g_4))$. Using the following properties (with $\alpha, \beta, \gamma, \delta \in \mathbb{R}$):

- minimization is distributive w.r.t. maximization, i.e., $\min\big(\alpha, \max(\beta, \gamma)\big) = \max\big(\min(\alpha, \beta), \min(\alpha, \gamma)\big)$. So
$$\min\big(\max(\alpha, \beta), \max(\gamma, \delta)\big) = \max\big(\min(\alpha, \gamma), \min(\alpha, \delta), \min(\beta, \gamma), \min(\beta, \delta)\big).$$
- the max operation is distributive w.r.t. min. Hence,
$$\max\big(\min(\alpha, \beta), \min(\gamma, \delta)\big) = \min\big(\max(\alpha, \gamma), \max(\alpha, \delta), \max(\beta, \gamma), \max(\beta, \delta)\big).$$
- $\min(\alpha, \beta) + \min(\gamma, \delta) = \min(\alpha + \gamma, \alpha + \delta, \beta + \gamma, \beta + \delta)$
  $\max(\alpha, \beta) + \max(\gamma, \delta) = \max(\alpha + \gamma, \alpha + \delta, \beta + \gamma, \beta + \delta)$
  $\max(\alpha, \beta) = -\min(-\alpha, -\beta)$
- if $\rho \in \mathbb{R}$ is positive, then $\rho \max(\alpha, \beta) = \max(\rho\alpha, \rho\beta)$ and $\rho \min(\alpha, \beta) = \min(\rho\alpha, \rho\beta)$;

it can be shown that $\max(f, g)$, $\min(f, g)$, $f + g$ and βf can again be written in min-max canonical form. ■

B. The MMPS MPC problem

We can use the deterministic model (57)–(58) either as a model of an MMPS system, as the equivalent model of a continuous PWA system, or as an approximation of a general smooth nonlinear system. Note that we do not include modeling errors or uncertainty in the model. However, since MPC uses a receding finite horizon approach, we can regularly update the model and the state estimate as new information and measurements become available.

We can make an estimate $\hat{y}(k + j|k)$ of the output of the system (57)–(58) at sample step $k + j$ based on the state $x(k)$ and the future input sequence $u(k), \ldots, u(k + j - 1)$. Using successive substitution, we obtain an expression of the following form:

$$\hat{y}(k + j|k) = F_j(x(k), u(k), \ldots, u(k + j - 1))$$

for $j = 1, \ldots, N_\mathrm{p}$. Clearly, $\hat{y}(k + j|k)$ is an MMPS function of $x(k), u(k), \ldots, u(k + j - 1)$.

[11] We allow "void" min or max statements of the form $\min(s)$ or $\max(s)$, which by definition are equal to s for any expression s. Alternatively, we can write $\min(s, s)$ or $\max(s, s)$.

[12] For the sake of simplicity we only consider two min-terms in f and g, each of which consists of the maximum of two affine functions. However, the proof also holds if more terms are considered.

In this section we consider the following output and input cost functions (see also Section IV-B):

$$J_{\mathrm{out},1}(k) = \|\tilde{y}(k) - \tilde{r}(k)\|_1, \quad J_{\mathrm{in},1}(k) = \|\tilde{u}(k)\|_1, \tag{66}$$
$$J_{\mathrm{out},\infty}(k) = \|\tilde{y}(k) - \tilde{r}(k)\|_\infty, \quad J_{\mathrm{in},\infty}(k) = \|\tilde{u}(k)\|_\infty \ . \tag{67}$$

Since we have $|x| = \max(x, -x)$ for all $x \in \mathbb{R}$, it is easy to verify that these cost functions are also MMPS functions.

Just as in conventional MPC and MPL MPC we can define (non)linear constraints (7) or (8), and a control horizon constraint (9) or (16). This then results in the MMPS MPC problem.

C. Algorithms for the MMPS MPC optimization problem

1) Nonlinear or ELCP optimization: In general the MMPS MPC optimization problem is a nonlinear, nonconvex optimization problem. Some of the methods discussed in Section IV-E can also be used to solve the MMPS MPC optimization problem: we can use multi-start nonlinear optimization based on sequential quadratic programming (SQP), or we can use a method based on the extended linear complementarity problem (ELCP). However, both methods have their disadvantages.

If we use the SQP approach, then we usually have to consider a large number of initial starting points and perform several optimization runs to obtain (a good approximation of) the global minimum. In addition, the objective functions that appear in the MMPS MPC optimization problem are nondifferentiable and PWA (if we use the cost criteria given in (66)–(67)), which makes the SQP algorithm less suitable for them.

The main disadvantage of the ELCP approach is that the execution time of this algorithm increases exponentially as the size of the problem increases. This implies that this approach is not feasible if N_c or the number of inputs and outputs of the system are large.

An alternative option consists in transforming the MMPS system into an MLD system since (constrained) MMPS systems are equivalent to MLD systems [43]. The main difference between MLD MPC and MMPS MPC is that MLD MPC requires the solution of *mixed integer-real* optimization problems. In general, these are also computationally hard optimization problems.

Now we will present another method to solve the MMPS MPC optimization problem that is similar to the cutting-plane method used in convex optimization.

2) An LP-based algorithm: We assume that the cost criteria given in (66)–(67) are used[13]. Recall that these objective functions (and any linear combination of them) are MMPS functions. The same holds for the estimate of future output $\tilde{y}(k)$. So if we substitute $\tilde{y}(k)$ in the expression for $J(k)$, we finally obtain an MMPS function of $\tilde{u}(k)$ as objective

[13] The result below also holds for any other cost criterion that is an MMPS function of $\tilde{y}(k)$ and $\tilde{u}(k)$. So it follows from Theorem 5.6 that any continuous PWA norm function can also be used.

function. From Theorem 7.1 it follows that this objective function can be written in min-max canonical form as follows (where — for the sake of simplicity of notation — we drop the index k):

$$J = \min_{i=1,\ldots,\ell} \max_{j=1,\ldots,n_i} (\alpha_{(i,j)}^{\mathrm{T}} \tilde{u} + \beta_{(i,j)})$$

for appropriately defined integers ℓ, $n_1, \ldots, n_\ell$, vectors $\alpha_{(i,j)}$ and integers $\beta_{(i,j)}$. Note that in general the expression obtained by straightforwardly applying the manipulations of the proof of Theorem 7.1 will contain a large number of affine arguments $\alpha_{(i,j)}^{\mathrm{T}} \tilde{u} + \beta_{(i,j)}$. However, many of these terms are redundant[14], and can thus be removed. This reduces the number of affine arguments. Also note that the transformation into canonical form only has to be performed once — provided that we explicitly consider all arguments that depend on k as additional variables when performing the transformation, — and that it can be done off-line.

The derivation below is similar to the cutting-plane algorithm for convex optimization (see, e.g., [13]). Hence, it requires constraints that are linear (or convex) in $\tilde{u}$. Note that the control horizon constraints (9) or (16) satisfy this condition. However, even if the original MPC constraint (7) is linear in $\tilde{u}(k)$ and $\tilde{y}(k)$, then in general this constraint is not linear any more after substitution of $\tilde{y}(k)$. Therefore, from now on we assume that there are only linear[15] constraints on the input $\tilde{u}(k)$:

$$P\tilde{u} + q \geqslant 0 \ . \tag{68}$$

In practice, constraints of the form (68) occur if we have to guarantee that the control signal $\tilde{u}(k)$ or the control signal rate $\Delta\tilde{u}(k)$ stay within certain bounds. Note that in general P and q may depend on $x(k)$ and k, but for the sake of simplicity of notation we do not explicitly indicate this dependence.

To obtain the optimal MMPS MPC input signal at sample step k, we have to solve an optimization problem of the following form:

$$\begin{aligned} &\min_{\tilde{u}} \min_{i=1,\ldots,\ell} \max_{j=1,\ldots,n_i} (\alpha_{(i,j)}^{\mathrm{T}} \tilde{u} + \beta_{(i,j)}) \\ &\text{subject to } P\tilde{u} + q \geqslant 0 . \end{aligned}$$

or equivalently

$$\min_{i=1,\ldots,\ell} \min_{\tilde{u}} \max_{j=1,\ldots,n_i} (\alpha_{(i,j)}^{\mathrm{T}} \tilde{u} + \beta_{(i,j)}) \tag{69}$$

$$\text{subject to } P\tilde{u} + q \geqslant 0 \ . \tag{70}$$

Now let $i \in \{1, \ldots, \ell\}$ and consider

$$\min_{\tilde{u}} \max_{j=1,\ldots,n_i} (\alpha_{(i,j)}^{\mathrm{T}} \tilde{u} + \beta_{(i,j)})$$

$$\text{subject to } P\tilde{u} + q \geqslant 0 \ .$$

It is easy to verify that this problem is equivalent to the following LP problem:

$$\min t \tag{71}$$

$$\text{subject to } \quad t \geqslant \alpha_{(i,j)}^{\mathrm{T}} \tilde{u} + \beta_{(i,j)} \quad \text{for } j = 1, \ldots, n_i \tag{72}$$

$$P\tilde{u} + q \geqslant 0 \ . \tag{73}$$

This LP problem can be solved efficiently using (variants of) the simplex method or an interior-point algorithm (see, e.g., [59], [75]).

To obtain the solution of (69)–(70), we solve (71)–(73) for $i = 1, \ldots, \ell$ and afterward we select the solution $\tilde{u}_{(i)}^{\mathrm{opt}}$ for which $\max_{j=1,\ldots,n_i} (\alpha_{(i,j)}^{\mathrm{T}} \tilde{u}_{(i)}^{\mathrm{opt}} + \beta_{(i,j)})$ is the smallest[16]. This results in an algorithm to solve the MMPS MPC problem that is more efficient than the SQP or the ELCP approach.

For a worked example of the MMPS MPC approach we refer the interested reader to [32].

VIII. Related work in connection with MPC for hybrid systems

A similar approach as the one derived in Section IV can also be applied to MPC for first-order hybrid systems with saturation [26], [27] (an example of these systems is a traffic-signal controlled intersection).

Note that MPC is related to optimal control. In this context, optimal control of a classes of manufacturing systems is considered in [19]. Other methods for optimal control of hybrid systems are presented [14]–[16], [41], [42], [52], [64].

IX. Conclusions

In this paper we have presented an overview of some results in connection with MPC for some tractable classes of discrete-event systems and hybrid systems. Computational complexity and the search for good and efficient approximations and solution methods are among the major current research topics in this field.

Acknowledgments
Many of the results described in this paper have been obtained by or in cooperation with A. Bemporad and W.M.P.H. Heemels.
Research partially funded by the Dutch Technology Foundation STW project "Model predictive control for hybrid systems" (DMR.5675) and by the European IST project "Modelling, Simulation and Control of Nonsmooth Dynamical Systems (SICONOS)" (IST-2001-37172).

[14] E.g., since they appear twice, or since there are other arguments in the max (min) expression that are always larger (smaller) than the given argument.

[15] The optimization algorithm used below, which is based on the cutting plane algorithm for convex optimization, can also deal with convex constraints. So we can also allow convex constraints instead of (68).

[16] If we use a primal-dual simplex method or an interior-point method to solve the LP problems, we can improve the efficiency of the approach even further by stopping the optimization if we obtain a lower bound for the objective function of the current LP problem that is larger than the smallest final objective function of the LP problems that have already been solved.

X. REFERENCES

[1] F. Allgöwer, T. Badgwell, J. Qin, J. Rawlings, and S. Wright, "Nonlinear predictive control and moving horizon estimation – An introductory overview," in *Advances in Control: Highlights of ECC '99*, P. Frank, Ed. London, UK: Springer, 1999, pp. 391–449.

[2] R. Alur and D. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[3] P. Antsaklis, W. Kohn, M. Lemmon, A. Nerode, and S. Sastry, Eds., *Hybrid Systems V*, ser. Lecture Notes in Computer Science, vol. 1567. Berlin, Germany: Springer-Verlag, 1999, (Proceedings of the 5th International Hybrid Systems Workshop, Notre Dame, Indiana, Sept. 1997).

[4] P. Antsaklis and A. Nerode, eds., "Special issue on hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, Apr. 1998.

[5] F. Baccelli, G. Cohen, G. Olsder, and J. Quadrat, *Synchronization and Linearity*. New York: John Wiley & Sons, 1992.

[6] A. Bemporad, G. Ferrari-Trecate, and M. Morari, "Observability and controllability of piecewise affine and hybrid systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 10, pp. 1864–1876, 2000.

[7] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, Mar. 1999.

[8] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, Jan. 2002.

[9] A. Bemporad, F. Torrisi, and M. Morari, "Optimization-based verification and stability characterization of piecewise affine and hybrid systems," in *Hybrid Systems: Computation and Control*, ser. Lecture Notes in Computer Science, B. Krogh and N. Lynch, Eds., vol. 1790. Springer Verlag, 2000, pp. 45–58.

[10] L. Biegler, "Efficient solution of dynamic optimization and NMPC problems," in *Nonlinear Model Predictive Control*, ser. Progress in Systems and Control Theory, F. Allgöwer and A. Zheng, Eds., vol. 26. Basel, Switzerland: Birkhäuser Verlag, 2000.

[11] V. Blondel and J. Tsitsiklis, "Complexity of stability and controllability of elementary hybrid systems," *Automatica*, vol. 35, no. 3, pp. 479–489, Mar. 1999.

[12] J. Boimond and J. Ferrier, "Internal model control and max-algebra: Controller design," *IEEE Transactions on Automatic Control*, vol. 41, no. 3, pp. 457–461, Mar. 1996.

[13] S. Boyd and C. Barratt, *Linear Controller Design: Limits of Performance*. Englewood Cliffs, New Jersey: Prentice Hall, 1991.

[14] M. Branicky, "Analyzing and synthesizing hybrid control systems," in *Lectures on Embedded Systems*, ser. Lecture Notes in Computer Science, G. Rozenberg and F. Vaandrager, Eds. Berlin: Springer, 1998, vol. 1494, pp. 74–113.

[15] M. Branicky, V. Borkar, and S. Mitter, "A unified framework for hybrid control: Model and optimal control theory," *IEEE Transactions on Automatic Control*, vol. 43, no. 1, pp. 31–45, Jan. 1998.

[16] M. Branicky and G. Zhang, "Solving hybrid control problems: Level sets and behavioral programming," in *Proceedings of the 2000 American Control Conference*, Chicago, IL, June 2000, pp. 1175–1180.

[17] E. Camacho and C. Bordons, *Model Predictive Control in the Process Industry*. Berlin, Germany: Springer-Verlag, 1995.

[18] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Boston: Kluwer Academic Publishers, 1999.

[19] C. Cassandras, D. Pepyne, and Y. Wardi, "Optimal control of a class of hybrid systems," *IEEE Transactions on Automatic Control*, vol. 46, no. 3, pp. 398–415, Mar. 2001.

[20] D. Christiansen, *Electronics Engineers' Handbook*, 4th ed. New York: IEEE Press/McGraw Hill, 1997.

[21] L. Chua and A. Deng, "Canonical piecewise-linear representation," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 1, pp. 101–111, Jan. 1988.

[22] D. Clarke, C. Mohtadi, and P. Tuffs, "Generalized predictive control – Part I. The basic algorithm," *Automatica*, vol. 23, no. 2, pp. 137–148, Mar. 1987.

[23] G. Cohen, D. Dubois, J. Quadrat, and M. Viot, "A linear-system-theoretic view of discrete-event processes and its use for performance evaluation in manufacturing," *IEEE Transactions on Automatic Control*, vol. 30, no. 3, pp. 210–220, Mar. 1985.

[24] R. Cuninghame-Green, *Minimax Algebra*, ser. Lecture Notes in Economics and Mathematical Systems. Berlin, Germany: Springer-Verlag, 1979, vol. 166.

[25] C. Cutler and B. Ramaker, "Dynamic matrix control – a computer control algorithm," in *Proceedings of the 86th AIChE National Meeting*, Houston, Texas, Apr. 1979.

[26] B. De Schutter, "Optimal control of a class of linear hybrid systems with saturation," *SIAM Journal on Control and Optimization*, vol. 39, no. 3, pp. 835–851, 2000.

[27] ——, "Optimizing acyclic traffic signal switching sequences through an extended linear complementarity problem formulation," *European Journal of Operational Research*, vol. 139, no. 2, pp. 400–415, June 2002.

[28] B. De Schutter and B. De Moor, "The extended linear complementarity problem," *Mathematical Programming*, vol. 71, no. 3, pp. 289–325, Dec. 1995.

[29] B. De Schutter and T. van den Boom, "Model predictive control for max-min-plus systems," in *Discrete Event Systems: Analysis and Control*, ser. The Kluwer International Series in Engineering and Computer Science, R. Boel and G. Stremersch, Eds. Boston: Kluwer Academic Publishers, 2000, vol. 569, pp. 201–208.

[30] ——, "Model predictive control for max-plus-linear discrete event systems," *Automatica*, vol. 37, no. 7, pp. 1049–1056, July 2001.

[31] ——, "Model predictive control for max-min-plus-scaling systems — Efficient implementation," in *Proceedings of the 6th International Workshop on Discrete Event Systems (WODES'02)*, M. Silva, A. Giua, and J. Colom, Eds., Zaragoza, Spain, Oct. 2002, pp. 343–348.

[32] ——, "MPC for continuous piecewise-affine systems," Control Systems Engineering, Fac. of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands, Tech. Rep. CSE02-004, Mar. 2002, provisionally accepted for publication in *Systems & Control Letters*.

[33] ——, "MPC for discrete-event systems with soft and hard synchronisation constraints," *International Journal of Control*, vol. 76, no. 1, pp. 82–94, 2003.

[34] B. De Schutter, T. van den Boom, and G. Benschop, "MPC for continuous piecewise-affine systems," in *Proceedings of the 15th IFAC World Congress*, Barcelona, Spain, July 2002, paper 229 / T-Fr-A16.

[35] G. Fishman, *Discrete-Event Simulation — Modeling, Programming, and Analysis*. New York: Springer-Verlag, 2001.

[36] R. Fletcher and S. Leyffer, "Numerical experience with lower bounds for MIQP branch-and-bound," *SIAM Journal on Optimization*, vol. 8, no. 2, pp. 604–616, May 1998.

[37] C. Floudas, *Nonlinear and mixed-integer optimization*. Oxford, UK: Oxford University Press, 1995.

[38] C. García, D. Prett, and M. Morari, "Model predictive control: Theory and practice — A survey," *Automatica*, vol. 25, no. 3, pp. 335–348, May 1989.

[39] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: W.H. Freeman and Company, 1979.

[40] V. Gorokhovik and O. Zorko, "Piecewise affine functions and polyhedral sets," *Optimization*, vol. 31, pp. 209–221, 1994.

[41] S. Hedlund and A. Rantzer, "Hybrid control laws from convex dynamic programming," in *Proceedings of IEEE Conference of Decision and Control*, Dec. 2000.

[42] ——, "Convex dynamic programming for hybrid systems," *IEEE Transactions on Automatic Control*, vol. 47, no. 9, pp. 1536–1540, Sept. 2002.

[43] W. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, no. 7, pp. 1085–1091, July 2001.

[44] W. Heemels, J. Schumacher, and S. Weiland, "Linear complementarity systems," *SIAM Journal on Applied Mathematics*, vol. 60, no. 4, pp. 1234–1269, 2000.

[45] T. Henzinger and S. Sastry, Eds., *Hybrid Systems: Computation and Control*, ser. Lecture Notes in Computer Science, vol. 1386. Berlin, Germany: Springer-Verlag, 1998, (Proceedings of the First International Workshop on Hybrid Systems: Computation and Control (HSCC'98), Berkeley, California, Apr. 1998).

[46] Y. Ho, Ed., *Discrete Event Dynamic Systems: Analyzing Complexity and Performance in the Modern World*. Piscataway, New Jersey: IEEE Press, 1992.

[47] Y. Ho and X. Cao, *Perturbation Analysis of Discrete Event Dynamic Systems*. Boston: Kluwer Academic Publishers, 1991.

[48] Y. Ho, ed., "Special issue on dynamics of discrete event systems," *Proceedings of the IEEE*, vol. 77, no. 1, Jan. 1989.

[49] M. Johansson and A. Rantzer, "Computation of piecewise quadratic Lyapunov functions for hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 555–559, Apr. 1998.

[50] D. Leenaerts and W. van Bokhoven, *Piecewise Linear Modeling and Analysis*. Boston: Kluwer Academic Publishers, 1998.

[51] L. Libeaut and J. Loiseau, "Admissible initial conditions and control of timed event graphs," in *Proceedings of the 34th IEEE Conference on Decision and Control*, New Orleans, Louisiana, Dec. 1995, pp. 2011–2016.

[52] B. Lincoln and A. Rantzer, "Optimizing linear system switching," in *Proceedings of the 40th Conference on Decision and Control*, 2001.

[53] J. Maciejowski, *Predictive Control with Constraints*. Harlow, England: Prentice Hall, 2002.

[54] E. Menguy, J. Boimond, and L. Hardouin, "A feedback control in max-algebra," in *Proceedings of the European Control Conference (ECC'97)*, Brussels, Belgium, paper 487, July 1997.

[55] ——, "Adaptive control for linear systems in max-algebra," in *Proceedings of the International Workshop on Discrete Event Systems (WODES'98)*, Cagliari, Italy, Aug. 1998, pp. 481–488.

[56] ——, "Optimal control of discrete event systems in case of updated reference input," in *Proceedings of the IFAC Conference on System Structure and Control (SSC'98)*, Nantes, France, July 1998, pp. 601–607.

[57] A. Morse, C. Pantelides, S. Sastry, and J. Schumacher, eds., "Special issue on hybrid systems," *Automatica*, vol. 35, no. 3, Mar. 1999.

[58] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[59] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. Philadelphia, Pennsylvania: SIAM, 1994.

[60] S. Ovchinnikov, "Max-min representation of piecewise linear functions," *Beiträge zur Algebra und Geometrie/Contributions to Algebra and Geometry*, vol. 43, no. 1, pp. 297–302, 2002.

[61] P. Pardalos and M. Resende, Eds., *Handbook of Applied Optimization*. Oxford, UK: Oxford University Press, 2002.

[62] K. Passino and K. Burgess, *Stability Analysis of Discrete Event Systems*. New York: John Wiley & Sons, 1998.

[63] J. Peterson, *Petri Net Theory and the Modeling of Systems*. Englewood Cliffs, New Jersey: Prentice-Hall, 1981.

[64] A. Rantzer and M. Johansson, "Piecewise linear quadratic optimal control," *IEEE Transactions on Automatic Control*, vol. 45, no. 4, pp. 629–637, Apr. 2000.

[65] J. Richalet, A. Rault, J. Testud, and J. Papon, "Model predictive heuristic control: Applications to industrial processes," *Automatica*, vol. 14, no. 5, pp. 413–428, Sept. 1978.

[66] A. Schrijver, *Theory of Linear and Integer Programming*. Chichester, UK: John Wiley & Sons, 1986.

[67] E. Sontag, "Nonlinear regulation: The piecewise linear approach," *IEEE Transactions on Automatic Control*, vol. 26, no. 2, pp. 346–358, Apr. 1981.

[68] F. Vaandrager and J. van Schuppen, Eds., *Hybrid systems: Computation and Control*, ser. Lecture Notes in Computer Science, vol. 1569. Berlin, Germany: Springer, 1999, (Proceedings of the Second International Workshop on Hybrid Systems: Computation and Control (HSCC'99), Berg en Dal, The Netherlands, Mar. 1999).

[69] T. van den Boom and B. De Schutter, "Properties of MPC for max-plus-linear systems," *European Journal of Control*, vol. 8, no. 5, pp. 453–462, 2002.

[70] ——, "Model predictive control for perturbed max-plus-linear systems: A stochastic approach," in *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, Florida, Dec. 2001, pp. 4535–4540.

[71] ——, "Model predictive control for perturbed max-plus-linear systems," *Systems & Control Letters*, vol. 45, no. 1, pp. 21–33, Jan. 2002.

[72] A. van der Schaft and J. Schumacher, "Complementarity modeling of hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 483–490, Apr. 1998.

[73] ——, *An Introduction to Hybrid Dynamical Systems*, ser. Lecture Notes in Control and Information Sciences. London: Springer-Verlag, 2000, vol. 251.

[74] H. Williams, *Model Building in Mathematical Programming*, 3rd ed. New York: Wiley, 1993.

[75] S. Wright, *Primal-Dual Interior Point Methods*. Philadelphia, Pennsylvania: SIAM, 1997.

[76] M. Zhou and F. DiCesare, *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*. Boston: Kluwer Academic Publishers, 1991.