Delft University of Technology

Delft Center for Systems and Control

Technical report 04-003

Model predictive control for discrete-event and hybrid systems – Part I: Discrete-event systems*

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

If you want to cite this report, please use the following reference instead:

B. De Schutter and T.J.J. van den Boom, "Model predictive control for discrete-event and hybrid systems – Part I: Discrete-event systems," *Proceedings of the 16th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2004)*, Leuven, Belgium, 10 pp., July 2004. Paper 312.

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/04_003.html

Model Predictive Control for Discrete-Event and Hybrid Systems
Part I: Discrete-Event 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 and its companion paper ("Part II: Hybrid Systems") we give an overview of some results in connection with MPC approaches for discrete-event systems and hybrid systems. In general the resulting optimization problems are nonlinear and nonconvex. However, for some classes of discrete-event systems and hybrid systems tractable solution methods exist.**

In this paper we consider discrete-event systems, i.e., asynchronous systems with event-driven dynamics. In particular, we discuss MPC for a special class of discrete-event systems, viz. max-plus-linear discrete-event systems, for both the noise-free and perturbed case (i.e., with modeling errors and/or noise).

In the companion paper we will discuss MPC for some classes of hybrid systems.

I. Introduction

Model predictive control (MPC) was pioneered simultaneously by Richalet *et al.* [28], and Cutler and Ramaker [11]. 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, non-minimum 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 receding 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 and the companion paper [15] 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.

For easy reference we here already list the abbreviations used in this paper and in the companion paper [15]:

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. For the sake of simplicity we will only consider the deterministic, i.e., noiseless, case in this brief introduction. More extensive information on MPC can be found in [1], [4], [6], [8], [17], [22] 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:

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

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

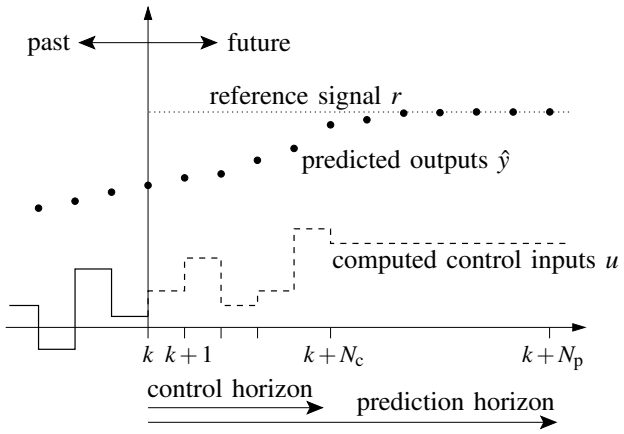

Fig. 1. Representation of the MPC control scheme.

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

In MPC we consider the future evolution of the system over a given prediction period $[k+1, k+N_p]$, 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 $j = 1, \ldots, N_p$. If we define the vectors

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

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

we obtain the following prediction equation:

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

B. Cost criterion and constraints

The cost criterion J used in conventional MPC reflects the reference tracking error (J_{out}) and the control effort (J_{in}):

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

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 of the plant (caused by limited capacity of buffers, limited transportation rates, saturation, etc.). This is reflected in the nonlinear constraint function

$$C_c(k, \tilde{u}(k), \tilde{y}(k)) \leqslant 0 \;, \quad (7)$$

which is often taken to be linear:

$$A_c(k)\tilde{u}(k) + B_c(k)\tilde{y}(k) \leqslant c_c(k) \;. \quad (8)$$

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 $k+N_c$:

$$u(k+j) = u(k+N_c-1) \;\text{ for }\; j = N_c, \ldots, N_p - 1. \quad (9)$$

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 $u(k), \ldots, u(k+N_p-1)$ 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

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 $\{u(k), \ldots, u(k+N_p-1)\}$ 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 affine function [3].

Traditionally MPC uses linear discrete-time models for the process to be controlled. In this paper and the companion paper [15] 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.

[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 (using, e.g., an (extended) Kalman filter).

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) [2].

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 algebras, and computer models (see [2], [7], [16], [18]–[20], [27], [33] 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 [2], [9], [10], 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 [2], [10]. 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 $\otimes$ [2], [10]. 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 $\mathcal{E}_{m\times n}$ is the $m \times n$ max-plus-algebraic zero matrix: $(\mathcal{E}_{m\times n})_{ij} = \varepsilon$ 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 $A, B \in \mathbb{R}_\varepsilon^{m\times n}$ and $C \in \mathbb{R}_\varepsilon^{n\times p}$, 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 [2] (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 $+$.

Remark 3.1 Apart from the fact that in (10)–(11) the components of the input, output and state vector are event

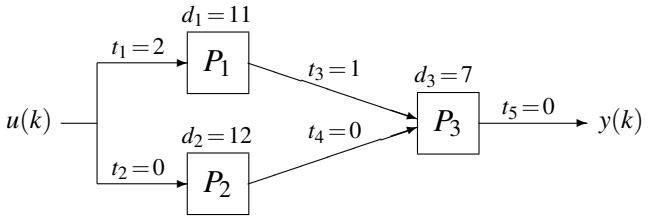

Fig. 2. A simple manufacturing system.

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 Section VII and [30], [31]). □

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

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

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 an 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 (cf. (6)),

$$\tilde{y}(k) = H\otimes\tilde{u}(k)\oplus g(k) \tag{12}$$

with

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

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

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

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

at event step 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

$$\begin{aligned} J_\mathrm{out}(k) &= \big(\tilde{y}(k)-\tilde{r}(k)\big)^\mathrm{T}\otimes\big(\tilde{y}(k)-\tilde{r}(k)\big) \\ &= 2\bigoplus_{j=1}^{N_\mathrm{p}}\bigoplus_{i=1}^{l}\big(\hat{y}_i(k+j|k)-r_i(k+j)\big) \\ &= 2\max_{j=1,\ldots,N_\mathrm{p}}\max_{i=1,\ldots,l}\big(\hat{y}_i(k+j|k)-r_i(k+j)\big) \ . \end{aligned} \tag{13}$$

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 (13). 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:

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

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

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

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

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

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

$$\begin{aligned} J_\mathrm{in}(k) &= \tilde{u}^\mathrm{T}(k)\otimes\tilde{u}(k) \\ &= 2\bigoplus_{j=0}^{N_\mathrm{p}-1}\bigoplus_{i=1}^{m}u_i(k+j) \\ &= 2\max_{j=0,\ldots,N_\mathrm{p}-1}\max_{i=1,\ldots,m}u_i(k+j) \ , \end{aligned}$$

i.e., we get 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

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

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

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

If we want to balance the input rates, we could take

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

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:

$$a_1(k+j) \leqslant \Delta u(k+j) \leqslant b_1(k+j) \ \text{ for } j=0,\ldots,N_\mathrm{c}-1, \tag{14}$$

$$a_2(k+j) \leqslant \Delta\hat{y}(k+j|k) \leqslant b_2(k+j) \ \text{ for } j=1,\ldots,N_\mathrm{p}, \tag{15}$$

or maximum due dates for the output events:

$$\hat{y}(k+j|k) \leqslant r(k+j) \quad \text{for } j=1,\ldots,N_\mathrm{p}\,, \tag{16}$$

can also be recast as a linear constraint of the form (8). The same holds for the straightforward translation of the linear constraint (8) into its max-plus-algebraic equivalent

$$A_c(k) \otimes \tilde{u}(k) \oplus B_c(k) \otimes \tilde{y}(k) \leqslant c_c(k) \ .$$

This condition 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 $j = 0, \ldots, N_p - 1$ 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 $k+N_c$ 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

$$\Delta^2 u(k+j) = 0 \quad \text{for } j = N_c, \ldots, N_p - 1 \,. \tag{17}$$

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:

$$\min_{\tilde{u}(k)} J_{out}(k) + \lambda J_{in}(k) \tag{18}$$

subject to

$$\tilde{y}(k) = H \otimes \tilde{u}(k) \oplus g(k) \tag{19}$$
$$A_c(k)\tilde{u}(k) + B_c(k)\tilde{y}(k) \leqslant c_c(k) \tag{20}$$
$$\Delta u(k+j) \geqslant 0 \qquad \text{for } j = 0, \ldots, N_p - 1 \tag{21}$$
$$\Delta^2 u(k+j) = 0 \qquad \text{for } j = N_c, \ldots, N_p - 1. \tag{22}$$

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

V. Algorithms to solve the MPL-MPC problem

A. Nonlinear optimization

In general the problem (18) – (22) is a nonlinear nonconvex optimization problem: although the constraints (20) – (22) are convex in $\tilde{u}$ and $\tilde{y}$, the constraint (19) 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 (19) – (22)[4]. If the problem is found to be infeasible we can use the same techniques as in conventional MPC and use constraint relaxation [6].

B. The ELCP approach

Now we discuss an alternative approach which is based on the Extended Linear Complementarity problem (ELCP) [12]. Consider the ith row of (19) 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,

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

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):

$$(\tilde{y}_i(k) - g_i(k)) \cdot \prod_{j \in \mathscr{J}_i} (\tilde{y}_i(k) - h_{ij} - \tilde{u}_j(k)) = 0 \ . \tag{23}$$

Hence, (19) can be rewritten as a system of the form

$$A_{elcp}\tilde{y}(k) + B_{elcp}\tilde{u}(k) + c_{elcp}(k) \geqslant 0 \tag{24}$$
$$\prod_{j \in \phi_i} (A_{elcp}\tilde{y}(k) + B_{elcp}\tilde{u}(k) + c_{elcp}(k))_j = 0 \qquad \text{for } i = 1, \ldots, lN_p \tag{25}$$

for appropriately defined matrices and vectors A_{elcp}, B_{elcp}, c_{elcp}, and index sets ϕ_i. We can rewrite the linear constraints (20) – (22) as

$$D_{elcp}(k)\tilde{y}(k) + E_{elcp}(k)\tilde{u}(k) + f_{elcp}(k) \geqslant 0 \tag{26}$$
$$G_{elcp}\tilde{u}(k) + h_{elcp} = 0 \ . \tag{27}$$

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 (24) – (27), which is a special case of an Extended Linear Complementarity Problem (ELCP) [12]. In [12] 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 (24) – (27) reaches its global minimum. The algorithm of [12] 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.

[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 5.2).

C. Monotonically nondecreasing objective functions

Now consider the *relaxed* MPL-MPC problem, which is also defined by (18)–(22) but with the =-sign in (19) 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 (19), 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 (24), (26), (27). 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., $(B_{\text{c}})_{ij} \geqslant 0$ 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 5.1 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 5.2: 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 (19), (21) and (22). 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 (20). 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. ■

TABLE I

THE CPU TIME NEEDED TO COMPUTE THE OPTIMAL INPUT SEQUENCE VECTORS FOR THE EXAMPLE OF SECTION VI 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 (s)			
	$N_{\text{c}} = 4$	$N_{\text{c}} = 5$	$N_{\text{c}} = 6$	$N_{\text{c}} = 7$
$\tilde{u}_{\text{elcp}}$	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

VI. EXAMPLE

Consider the production system of Example 3.3. Let us now compare the efficiency of the methods discussed in Section V 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 5.2. We have computed a solution $\tilde{u}_{\text{elcp}}$ obtained using the ELCP method and the ELCP algorithm of [12], 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 5.1). 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:

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

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 $J(0) = -381$.

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 [12] 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 $\tilde{u}_{\mathrm{lp}}$ solution — which is based on Theorem 5.2 and an LP approach — is clearly the most interesting. For more results we refer the interested reader to [13].

VII. MPL-MPC: The perturbed case

The approach presented in the previous sections can be extended to the perturbed case as follows. We extend the noise-free deterministic MPL model (10)–(11) to include uncertainty, such as modeling errors and disturbances [30], [31]. Therefore, we consider the following (time-varying) MPL system:

$$x(k+1) = A(k) \otimes x(k) \oplus B(k) \otimes u(k) \tag{28}$$

$$y(k) = C(k) \otimes x(k) \oplus D(k) \otimes u(k) \tag{29}$$

where $A(k)$, $B(k)$ and $C(k)$ represent uncertain system matrices due to modeling errors or disturbances. Usually fast changes in the system matrices will be considered as noise and disturbances, whereas slow changes or permanent errors are considered as model mismatch. Both features will be treated in one single framework here. The uncertainty caused by disturbances and errors in the estimation of physical variables is gathered in the uncertainty vector $e(k)$. We assume that the uncertainty either is bounded or has stochastic properties. We also assume that the uncertainty vector $e(k)$ captures the complete time-varying aspect of the system. We collect the uncertainty over the interval $[k, k+N_{\mathrm{p}}-1]$ in one vector

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

Now it is easy to verify that the prediction model, i.e., the prediction of the future outputs for the system (28)–(29) can be written in the following form[5] (cf. (12)):

$$\tilde{y}(k) = H(\tilde{e}(k)) \otimes \tilde{u}(k) \oplus g(\tilde{e}(k)) \ . \tag{30}$$

Now we distinguish between two cases: bounded perturbations and stochastic perturbations.

A. Bounded perturbations

In this case we assume that $e(k)$ is bounded, and/or, in the case that, e.g., consecutive noise samples $e(k)$ and $e(k-1)$ are related, that the change $\Delta e(k) = e(k) - e(k-1)$ is bounded. This implies that $\tilde{e}(k) \in \mathscr{E}_{\tilde{e}}$, with $\mathscr{E}_{\tilde{e}}$ a bounded set. Recall that in MPL-MPC we want to minimize the criterion

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

where J_{out} represents the output error and J_{in} is related to the input dates. We aim to find the optimal input-output pair $(\tilde{u}(k), \tilde{y}(k))$ that minimizes $J(k)$, where $\tilde{y}(k)$ and $\tilde{u}(k)$ are related by (30). Note that, in contrast to the noise-free case, the relation between $\tilde{u}(k)$ and $\tilde{y}(k)$ is not unique any longer in the perturbed case because of the perturbation $\tilde{e}(k)$.

As $\tilde{y}(k)$ and therefore also $J_{\mathrm{out}}(k)$ depends on the perturbation $\tilde{e}(k)$, we therefore introduce a worst-case criterion for the bounded perturbation case:

$$J_{\mathrm{wc}}(k) = \max_{\tilde{e}(k) \in \mathscr{E}_{\tilde{e}}} J_{\mathrm{out}}(k) + \lambda J_{\mathrm{in}}(k) \ .$$

In combination with (30) this leads to the worst-case MPL-MPC problem at event step k, which is defined as follows:

$$\min_{\tilde{u}(k), \tilde{y}(k)} J_{\mathrm{wc}}(k) \tag{31}$$

subject to

$$\tilde{y}(k) = H(\tilde{e}(k)) \otimes \tilde{u}(k) \oplus g(\tilde{e}(k)) \tag{32}$$

$$A_{\mathrm{c}}(k)\tilde{u}(k) \leqslant c_{\mathrm{c}}(k) \tag{33}$$

$$\Delta u(k+j) \geqslant 0 \qquad \text{for } j = 0, \ldots, N_{\mathrm{p}}-1 \tag{34}$$

$$\Delta^2 u(k+j) = 0 \qquad \text{for } j = N_{\mathrm{c}}, \ldots, N_{\mathrm{p}}-1. \tag{35}$$

Note that compared to linear constraint (20) on both inputs and outputs used in the deterministic, noise-free MPL-MPC problem (18)–(22) we now only consider linear input constraints (i.e., $B_{\mathrm{c}} = 0$). The following proposition can then be given for the worst-case MPL-MPC problem:

Proposition 7.1 ([31]*):* Assume that J_{out} is a monotonically nondecreasing, convex function of $\tilde{y}$ and that J_{in} is convex in $\tilde{u}$. In that case $J_{\mathrm{wc}}(k)$ is convex in $\tilde{e}(k)$ (for a given value of $\tilde{u}(k)$) and the worst-case MPC problem is convex in $\tilde{u}(k)$.

B. Stochastic perturbations

In the stochastic perturbation case $e(k)$ is a stochastic variable. Therefore, we introduce a stochastic cost criterion

$$J_{\mathrm{st}}(k) = I\!E[J_{\mathrm{out}}(k)] + \lambda J_{\mathrm{in}}(k) \ ,$$

where $I\!E[\cdot]$ denotes the expected value. More specifically, we could consider, e.g.,

$$J_{\mathrm{st},1}(k) = \sum_{i=1}^{l} \sum_{j=0}^{N_{\mathrm{p}}-1} I\!E[\eta_i(k+j)] - \lambda \sum_{i=1}^{m} \sum_{j=0}^{N_{\mathrm{p}}-1} u_i(k+j) \tag{36}$$

where $\eta_i(k)$ denotes the ith "tardiness" given by

$$\eta_i(k) = \max(\, y_i(k) - r_i(k) \,,\, 0 \,) \ . \tag{37}$$

This leads to the stochastic MPL-MPC problem:

$$\min_{\tilde{u}(k), \tilde{y}(k)} J_{\mathrm{st}}(k)$$

[5] The full expressions for the matrix $H(\tilde{e}(k))$ and the vector $g(\tilde{e}(k))$ are given in [31].

subject to

$$\tilde{y}(k) = H(\tilde{e}(k)) \otimes \tilde{u}(k) \oplus g(\tilde{e}(k)) \tag{38}$$

$$A_{\mathrm{c}}(k)\tilde{u}(k) + B_{\mathrm{c}}(k)\mathbb{E}[\tilde{y}(k)] \leqslant c_{\mathrm{c}}(k) \tag{39}$$

$$\Delta u(k+j) \geqslant 0 \qquad \text{for } j = 0, \ldots, N_{\mathrm{p}} - 1 \tag{40}$$

$$\Delta^2 u(k+j) = 0 \qquad \text{for } j = N_{\mathrm{c}}, \ldots, N_{\mathrm{p}} - 1. \tag{41}$$

The following proposition can be given for the stochastic MPL-MPC problem:

Proposition 7.2 ([30]*):* Consider the stochastic MPL-MPC problem with cost criterion (36). Assume the linear constraints (39) to be monotonically nondecreasing as a function of $\mathbb{E}[\tilde{y}(k)]$ (i.e., $(B_{\mathrm{c}})_{ij} \geq 0$ for all i, j). In that case the cost criterion (36) and the constraint (39) become convex in $\tilde{u}(k)$, and so the stochastic MPL-MPC problem will be a convex problem in $\tilde{u}(k)$.
Furthermore, a subgradient of the constraints and a subgradient of the cost criterion can easily be derived (see [30]).

Note that under the monotonicity and convexity conditions stated in Propositions 7.1 and 7.2 both the worst-case MPL-MPC problem and the stochastic MPL-MPC problem turn out to be convex optimization problems. These type of problems can be solved using reliable and efficient optimization algorithms based on, e.g., interior point methods [26], [32].

VIII. Extensions and related research

Tuning rules for MPL-MPC and properties such as stability, timing issues, etc. have been discussed in [29]. The MPL-MPC method derived above can also be extended to MPC for discrete-event systems with hard and soft synchronization constraints [14] 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 [2], [5], [21], [23]–[25]. 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 (20), or even simple constraints of the form (14) or (15).

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. In the sequel paper [15] we will present MPC for some tractable classes of hybrid systems.

Acknowledgments
Research partially funded by the Dutch Technology Foundation STW projects "Model predictive control for hybrid systems" (DMR.5675) and "Multi-agent control of large-scale hybrid systems" (DWV.6188), and by the European IST project "Modelling, Simulation and Control of Nonsmooth Dynamical Systems (SICONOS)" (IST-2001-37172).

X. References

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

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

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

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

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

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

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

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

[9] G. Cohen, D. Dubois, J.P. 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.

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

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

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

[13] B. De Schutter and T. van den Boom, "Model predictive control for max-plus-linear discrete event systems," *Automatica*, vol. 37, no. 7, pp. 1049–1056, July 2001.

[14] B. De Schutter and T.J.J. van den Boom, "MPC for discrete-event systems with soft and hard synchronisation constraints," *International Journal of Control*, vol. 76, no. 1, pp. 82–94, Jan. 2003.

[15] B. De Schutter and T.J.J. van den Boom, "Model predictive control for discrete-event and hybrid systems – Part II: Hybrid systems," in *Proceedings of the 16th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2004)*, Leuven, Belgium, July 2004. Paper 313.

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

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

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

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

[20] Y.C. Ho, ed., "Special issue on dynamics of discrete event systems," *Proceedings of the IEEE*, vol. 77, no. 1, Jan. 1989.
[21] L. Libeaut and J.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, pp. 2011–2016, Dec. 1995.
[22] J.M. Maciejowski, *Predictive Control with Constraints*. Harlow, UK: Prentice Hall, 2002.
[23] E. Menguy, J.L. 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.
[24] E. Menguy, J.L. Boimond, and L. Hardouin, "Adaptive control for linear systems in max-algebra," in *Proceedings of the International Workshop on Discrete Event Systems (WODES'98)*, Cagliari, Italy, pp. 481–488, Aug. 1998.
[25] E. Menguy, J.L. Boimond, and L. Hardouin, "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, pp. 601–607, July 1998.
[26] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. Philadelphia, Pennsylvania: SIAM, 1994.
[27] K.M. Passino and K.L. Burgess, *Stability Analysis of Discrete Event Systems*. New York: John Wiley & Sons, 1998.
[28] J. Richalet, A. Rault, J.L. Testud, and J. Papon, "Model predictive heuristic control: Applications to industrial processes," *Automatica*, vol. 14, no. 5, pp. 413–428, Sept. 1978.
[29] 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.
[30] T.J.J. van den Boom and B. De Schutter, "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, pp. 4535–4540, Dec. 2001.
[31] T.J.J. van den Boom and B. De Schutter, "Model predictive control for perturbed max-plus-linear systems," *Systems & Control Letters*, vol. 45, no. 1, pp. 21–33, Jan. 2002.
[32] S.J. Wright, *Primal-Dual Interior Point Methods*. Philadelphia, Pennsylvania: SIAM, 1997.
[33] M.C. Zhou and F. DiCesare, *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*. Boston, Massachusetts: Kluwer Academic Publishers, 1991.