Delft University of Technology

Delft Center for Systems and Control

Technical report 10-050

Model predictive control for randomly switching max-plus-linear systems using a scenario-based algorithm*

T. van den Boom and B. De Schutter

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

T. van den Boom and B. De Schutter, "Model predictive control for randomly switching max-plus-linear systems using a scenario-based algorithm," *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, Georgia, pp. 2298–2303, Dec. 2010.

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.deschutter.info/abs/10_050.html`

Model predictive control for randomly switching max-plus-linear systems using a scenario-based algorithm

Ton van den Boom* and Bart De Schutter*

***Abstract*— Switching max-plus-linear (SMPL) systems are discrete event systems that can switch between different modes of operation. In each mode the system is described by a max-plus-linear state equation and a max-plus-linear output equation, with different system matrices for each mode. The switching between from one mode to the other is a stochastic process. In the model predictive control (MPC) formulation stability is enforced by additional constraints. To reduce the computational complexity we use an algorithm based on scenario generation for such stochastic SMPL systems.**

I. INTRODUCTION

The class of discrete event systems (DES) essentially consists of man-made systems that contain a finite number of resources that are shared by several users all of which contribute to the achievement of some common goal [1]. In general, models that describe the behavior of a discrete event system are nonlinear in conventional algebra.

In this paper we consider switching max-plus-linear (SMPL) systems, discrete event systems that can switch between different modes of operation, in which the mode switching depends on the previous state, the previous mode and the input [17]. In each mode the system is described by a max-plus-linear state equation and a max-plus-linear output equation, with different system matrices for each mode. The class of randomly switching max-plus-linear (RSMPL) systems contains discrete event systems with synchronization but no concurrency, in which the order of synchronization of the event steps may vary randomly, or cannot be determined a priori [18] (see Figure 1). Typical examples of SMPL systems are flexible manufacturing systems, telecommunication networks, traffic signal controlled urban traffic networks. The random switching between different max-plus linear (MPL) modes is then due to e.g. randomly changing production recipes, varying customer demands or traffic demands, or failures in production unit, transmission lines or traffic links. In [18] we have derived a stabilizing model predictive controller for these randomly switching max-plus-linear systems. The resulting optimization problem was solved using linear programming algorithms. The main drawback of the algorithm is that the number of linear constraints and the number of optimization variables was increasing fast with the prediction horizon and the number of modes in the system. In this paper we study ways to reduce the computational complexity.

*Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands a.j.j.vandenboom@tudelft.nl, b.deschutter@dcsc.tudelft.nl

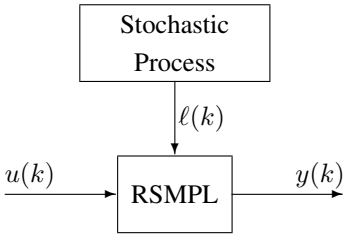

Fig. 1. Randomly switching max-plus-linear system

The paper is organized as follows. In Section II we introduce the max-plus algebra and the concept of RSMPL systems. We also recapitulate conditions for a stabilizing controller for RSMPL systems. In Section III we give conditions for a stabilizing controller, and we present a stabilizing model predictive controller for RSMPL systems. In Section IV the scenario optimization tree is discussed, and in Section V we give a worked example.

II. MAX-PLUS ALGEBRA AND SMPL SYSTEMS

A. Max-plus algebra

In this section we give the basic definition of the max-plus algebra [1], [4].
Define $\varepsilon = -\infty$ and $\mathbb{R}_\varepsilon = \mathbb{R} \cup \{\varepsilon\}$. The max-plus-algebraic addition ($\oplus$) and multiplication ($\otimes$) are defined as follows:

$$x \oplus y = \max(x, y) \qquad x \otimes y = x + y$$

for any $x, y \in \mathbb{R}_\varepsilon$, and

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

for matrices $A, B \in \mathbb{R}_\varepsilon^{m \times n}$ and $C \in \mathbb{R}_\varepsilon^{n \times p}$. The matrix $\mathcal{E}$ is the max-plus-algebraic zero matrix: $[\mathcal{E}]_{ij} = \varepsilon$ for all i, j.

A max-plus diagonal matrix $S = \operatorname{diag}_\oplus(s_1, \ldots, s_n)$ has elements $S_{ij} = \varepsilon$ for $i \neq j$ and diagonal elements $S_{ii} = s_i$ for $i = 1, \ldots, n$. If all s_i are finite, the inverse of S is equal to $S^{\otimes^{-1}} = \operatorname{diag}_\oplus(-s_1, \ldots, -s_n)$. There holds $S \otimes S^{\otimes^{-1}} = S^{\otimes^{-1}} \otimes S = E$, where $E = \operatorname{diag}_\oplus(0, \ldots, 0)$ is the max-plus identity matrix.

B. SMPL and RSMPL systems

In [17] we have introduced Switching Max-Plus-Linear (SMPL) systems, i.e. discrete event systems that can switch between different modes of operation. In each mode $\ell \in$

$\{1,\ldots,L\}$, the system is described by a max-plus-linear state equation and a max-plus-linear output equation:

$$x(k) = A^{(\ell(k))} \otimes x(k-1) \oplus B^{(\ell(k))} \otimes u(k) \quad (1)$$

$$y(k) = C^{(\ell(k))} \otimes x(k) \quad (2)$$

in which the matrices $A^{(\ell)} \in \mathbb{R}_{\varepsilon}^{n_x \times n_x}$, $B^{(\ell)} \in \mathbb{R}_{\varepsilon}^{n_x \times n_u}$, $C^{(\ell)} \in \mathbb{R}_{\varepsilon}^{n_y \times n_x}$ are the system matrices for the ℓth mode. The index k is called the event counter. For discrete event systems the state $x(k)$ typically contains the time instants at which the internal events occur for the kth time, the input $u(k)$ contains the time instants at which the input events occur for the kth time, and the output $y(k)$ contains the time instants at which the output events occur for the kth time[1].

In [18] we have introduced random switching, i.e. for the system (1)-(2), the mode switching variable $\ell(k)$ is a stochastic process. For a system with L possible modes, we assume the probability of a switching from mode i to a mode j to be given by $P_{\rm s}(i,j)$ for $i = 1,\ldots,L$, $j = 1,\ldots,L$.

C. Stability of RSMPL systems

Just like in [16], [18], we adopt the notion of stability for DES from [15], in which a DES is called stable if all its buffer levels remain bounded. Let $r(k)$ be the due date for output event $y(k)$. All the buffer levels in DES are bounded if there exist finite constants k_0, M_{yr}, M_{yx} and M_{xu} such that

$$|\, y_i(k) - r_i(k) \,| \le M_{yr}, \qquad \forall i \quad (3)$$

$$|\, y_i(k) - x_j(k) \,| \le M_{yx}, \qquad \forall i,j \quad (4)$$

$$|\, x_j(k) - u_m(k) \,| \le M_{xu}, \qquad \forall j,m \quad (5)$$

for all $k > k_0$. Condition (3) means that the delay between the actual output date $y(k)$ and the due date $r(k)$ remains bounded (for $y - r < \infty$), and on the other hand, that the stock time will remain bounded (for $r - y < \infty$). Conditions (4) and (5) mean that the throughput time (i.e. the time between the starting date $u(k)$ and the output date $y(k)$) is bounded. For a due date defined as

$$r(k) = \rho\, k + d(k), \quad \text{where } |d_i(k)| \le d_{\max}, \forall i \quad (6)$$

where r and d are vectors and ρ is a scalar, satisfying $\rho > 0$, this implies finite buffer levels.

For RSMPL systems one can compute the maximum growth rate:

Definition 1: Consider an RSMPL system of the form (1)-(2) and define the matrices $A_{\alpha}^{(\ell)}$ with $[A_{\alpha}^{(\ell)}]_{ij} = [A^{(\ell)}]_{ij} - \alpha$. The maximum growth rate λ of the RSMPL system is the smallest α for which there exists a max-plus diagonal matrix $S = \mathrm{diag}_{\oplus}(s_1,\ldots,s_n)$ with finite diagonal elements s_i, such that

$$[\, S \otimes A_{\alpha}^{(\ell)} \otimes S^{\otimes^{-1}} \,]_{ij} \le 0, \quad \forall\, i,j,\ell \quad (7)$$

[1]More specifically, for a manufacturing system, $x(k)$ contains the time instants at which the processing units start working for the kth time, $u(k)$ the time instants at which the kth batch of raw material is fed to the system, and $y(k)$ the time instants at which the kth batch of finished product leaves the system.

The maximum growth rate λ can be easily computed by solving a linear programming problem.

The set $\mathcal{L}_N = \{\, [\, \ell_1 \ \cdots \ \ell_N \,]^T \mid \ell_m \in \{1,\ldots,L\},\ m = 1,\ldots,N\}$ is the set of all possible consecutive mode switching vectors.

Definition 2: An RSMPL system is controllable if there exists a finite positive integer N such that for all $\tilde{\ell} \in \mathcal{L}_N$ the matrices

$$\Gamma_{\rho}^{N}(\tilde{\ell}) = \Big[B^{(\ell_N)} \quad A_{\rho}^{(\ell_N)} \otimes B^{(\ell_{N-1})} \quad A_{\rho}^{(\ell_N)} \otimes A_{\rho}^{(\ell_{N-1})} \otimes B^{(\ell_{N-2})} \quad \ldots \quad A_{\rho}^{(\ell_N)} \otimes \cdots \otimes A_{\rho}^{(\ell_2)} \otimes B^{(\ell_1)} \ \Big]$$

are row-finite, i.e. in each row there is at least one entry larger then ε.

Theorem 3: [18] Consider a switching MPL system with random mode switching and due-date signal (6), and a maximum grow rate λ. Define the matrices $A_{\rho}^{(\ell)}$ with $[A_{\rho}^{(\ell)}]_{ij} = [A^{(\ell)}]_{ij} - \rho$. Further assume $C^{(\ell)}$ to be row-finite. Now if

1) $\rho < \lambda$ (8)

2) the system is controllable,

then any input signal

$$u(k) = \rho\, k + \mu(k), \quad (9)$$

where $\mu_{\min} \le \mu_i(k) \le \mu_{\max}$, $\forall i$, and $\mu_{\min}$ and $\mu_{\max}$ are finite, will stabilize the SMPL system.

III. A STABILIZING MODEL PREDICTIVE CONTROLLER

Model predictive control (MPC) [3], [13] is a model-based predictive control approach that has its origins in the process industry and that has mainly been developed for linear or nonlinear time-driven systems. Its main ingredients are: a prediction model, a performance criterion to be optimized over a given horizon, constraints on inputs and outputs, and a receding horizon approach. In [5], [18] we have extended this approach to MPL systems and randomly switching MPL systems and shown that the resulting optimization problem can be solved using linear programming algorithms.

In MPC we use predictions of future signals based on the RSMPL model. The cost criterion reflects the input and output cost functions ($J_{\rm in}$ and $J_{\rm out}$, respectively) in the event period $[k, k+N_{\rm p}-1]$:

$$J(k) = \mathbb{E}\left\{ \sum_{j=0}^{N_{\rm p}-1} \sum_{i=1}^{n_y} \max(y_i(k+j) - r_i(k+j), 0) \right\} - \beta \sum_{j=0}^{N_{\rm p}-1} \sum_{i=1}^{n_u} u_i(k+j) \quad (10)$$

where $\beta \ge 0$ is a tuning parameter, $\hat{y}(k+j|k)$ denotes the prediction of $y(k+j)$ at event step $k+j$, based on knowledge at event step k, $u(k+j)$ denotes the future inputs, $\ell(k+j)$ denotes the future modes, and $N_{\rm p}$ is the prediction horizon (so it determines how many cycles we look ahead in our control law design). More about the choice of cost function J can be found in [17].

Define the prediction vectors

$$\tilde{y}(k)=\begin{bmatrix} \hat{y}(k|k) \\ \vdots \\ \hat{y}(k+N_{\rm p}-2|k) \\ \hat{y}(k+N_{\rm p}-1|k) \end{bmatrix}, \quad \tilde{u}(k)=\begin{bmatrix} u(k) \\ \vdots \\ u(k+N_{\rm p}-2) \\ u(k+N_{\rm p}-1) \end{bmatrix},$$

Now the performance index can be written as

$$J(k) = I\!E \left\{ \sum_{i=1}^{n_y N_{\rm p}} \max(\tilde{y}_i(k) - \tilde{r}_i(k), 0) \right\} - \beta \sum_{i=1}^{n_u N_{\rm p}} [\tilde{u}(k)]_i \, . \tag{11}$$

Define the mode sequence vector

$$\tilde{\ell}(k)=\begin{bmatrix} \ell(k) \\ \vdots \\ \ell(k+N_{\rm p}-2) \\ \ell(k+N_{\rm p}-1) \end{bmatrix},$$

and the matrices

$$\tilde{C}(\tilde{\ell}(k))=\begin{bmatrix} \tilde{C}_1(\tilde{\ell}(k)) \\ \vdots \\ \tilde{C}_{N_{\rm p}}(\tilde{\ell}(k)) \end{bmatrix}$$

$$\tilde{D}(\tilde{\ell}(k))=\begin{bmatrix} \tilde{D}_{11}(\tilde{\ell}(k)) & \cdots & \tilde{D}_{1N_{\rm p}}(\tilde{\ell}(k)) \\ \vdots & \ddots & \vdots \\ \tilde{D}_{N_{\rm p}1}(\tilde{\ell}(k)) & \cdots & \tilde{D}_{N_{\rm p}N_{\rm p}}(\tilde{\ell}(k)) \end{bmatrix}$$

where

$$\tilde{C}_m(\tilde{\ell}(k)) = C^{(\ell(k+m-1))} \otimes A^{(\ell(k+m-1))} \otimes \ldots \otimes A^{(\ell(k))}$$

and

$$\tilde{D}_{mn}(\tilde{\ell}(k))=\begin{cases} C^{(\ell(k+m-1))} \otimes A^{(\ell(k+m-1))} \\ \quad \otimes A^{(\ell(k+n))} \otimes B^{(\ell(k+n-1))} & \text{if } m>n \\ C^{(\ell(k+m-1))} \otimes B^{(\ell(k+m-1))} & \text{if } m=n \\ \mathcal{E} & \text{if } m<n \end{cases}$$

then the prediction model for (1)–(2) is given by:

$$\tilde{y}(k) = \tilde{C}(\tilde{\ell}(k)) \otimes x(k-1) \oplus \tilde{D}(\tilde{\ell}(k)) \otimes \tilde{u}(k). \tag{12}$$

The probability for the switching sequence $\tilde{\ell}(k) \in \mathcal{L}_{N_{\rm p}}$, given the present mode $\ell(k)$, is given by

$$P(\tilde{\ell}(k)|\ell(k)) = P_{\rm s}(\ell(k),\ell(k+1)) \cdot P_{\rm s}(\ell(k+1),\ell(k+2)) \cdots P_{\rm s}(\ell(k+N_{\rm p}-2),\ell(k+N_{\rm p}-1))$$

where $P_{\rm s}$ denotes the switching probability (see Section II-B).

The MPC problem for RSMPL systems with due-date signal (6) can be defined at event step k as minimizing (11) subject to the constraints

$$u(k+j) - u(k+j-1) \geq 0, \;\; j=0,\ldots,N_{\rm p}-1 \tag{13}$$

$$\mu_{\min} \leq u_i(k) - \rho\, k \leq \mu_{\max}, \; i=1,\ldots,n_u, \tag{14}$$

where (13) guarantees a non-decreasing input sequence, and (14) guarantees stability (cf. Theorem 3).

Theorem 4: [18] Assume that $\mathcal{L}_{N_{\rm p}}$ can be rewritten as $\mathcal{L}_{N_{\rm p}} = \{\tilde{\ell}^1, \tilde{\ell}^2, \ldots, \tilde{\ell}^M\}$ for $M = L^{N_{\rm p}-1}$. The MPC problem of minimizing (11) subject to (13)-(14) can be recast as a linear programming problem:

$$\min_{\{\tilde{u}(k),t_{i,m}\}} \sum_{i=1}^{n_y N_{\rm p}} \sum_{m=1}^{M} t_{i,m} P(\tilde{\ell}^m|\ell(k)) - \beta \sum_{i=1}^{n_u N_{\rm p}} \tilde{u}_i(k) \tag{15}$$

subject to

$$t_{i,m} \geq [\tilde{C}(\tilde{\ell}^m)]_{i,l} + x_l(k-1) - \tilde{r}_i(k) \; , \;\; \forall i,m,l \tag{16}$$

$$t_{i,m} \geq [\tilde{D}(\tilde{\ell}^m)]_{i,l} + \tilde{u}_l(k) - \tilde{r}_i(k) \; , \;\; \forall i,m,l \tag{17}$$

$$t_{i,m} \geq 0 \; , \;\; \forall i,m \tag{18}$$

$$u_i(k+j) - u_i(k+j-1) \geq 0, \;\; \forall i,j \tag{19}$$

$$\mu_{\min} \leq u_i(k+j) - \rho\, k \leq \mu_{\max}, \;\; \forall i,j \tag{20}$$

MPC uses a receding horizon strategy. So after computation of the optimal control sequence $\tilde{u}^*(k)$, only the first control sample $u(k) = u^*(k)$ will be implemented, subsequently the horizon is shifted and the model and the initial state estimate can be updated if new measurements are available, then the new MPC problem is solved, etc.

So the optimization in the MPC algorithm boils down to a linear programming problem, which is polynomially solvable [11] and for which efficient algorithms are available.

IV. SMPL-MPC USING SWITCHING SCENARIOS

In equation (15) in Theorem 4 we see that the performance index is built up from $M = L^{N_{\rm p}-1}$ terms. This number M can become very large if there are many modes and the prediction horizon is large. In this section we derive a scenario-based algorithm (inspired by the work of Bernardini and Bemporad [2]), that still gives a good approximation of the performance index, but is computationally less complex.

Instead of computing all possible realizations of $\tilde{\ell}(k)$ we only consider the most probable ones. We will now describe an algorithm to create the $n_{\max}$ ($\lll L^{N_{\rm p}-1}$) most probable realizations of $\tilde{\ell}(k)$. Note that in the approach of Bernardini and Bemporad [2] the length of the realizations is not fixed but variable since they are using standard algorithms for determining the $n_{\max}$ shortest paths in a graph[2] originating from a given node [7]; in principle, these algorithms do not return paths with the same, fixed number of edges. However, in the context of MPC it is natural to look for realization or paths with the same fixed number of modes or edges, viz. $N_{\rm p}-1$. To the authors' best knowledge there are no algorithms described in the literature that return the $n_{\max}$ shortest path with a fixed number of edges. Therefore, we propose a dedicated algorithm based on a breadth-first search in combination with an approach to cut away parts of the search tree.

To this aim we consider the search tree $\mathcal{T}$ with $N_{\rm p}-1$ levels and a root node n_0 that corresponds to the known

[2]This graph is the probability graph corresponding to the transition probability matrix $P_{\rm s}$ defined in Section II-B.

mode $\ell(k)$. The first level of $\mathcal{T}$ consists of L child nodes (corresponding to modes $1,\ldots,L$), which are connected to the root node by an edge with weight $P_{\rm s}(\ell(k),j)$ for $j=1,\ldots,L$. The next level of the search tree consists of L^2 nodes (L child nodes for each node in the preceding level) where the parent node corresponding to mode i is connected to the child node corresponding to mode j by an edge with weight $P_{\rm s}(i,j)$. In this way we can define the search tree $\mathcal{T}$ with $N_{\rm p}-1$ levels and $L^{N_{\rm p}-1}$ leaf nodes. Each leaf node $n_{\rm leaf}$ corresponds to one particular realization of $\tilde{\ell}(k)\in\mathcal{L}_{N_{\rm p}}$, which will be denoted in the sequel as $\tilde{\ell}^{\rm real}(n_{\rm leaf})$. In our approach, we will not construct $\mathcal{T}$ explicitly, but use a branch-and-bound algorithm to extract the $n_{\max}$ most probable realizations. Note that if we define[3] the weight of a path $n_0\rightarrow n_1\rightarrow\ldots\rightarrow n_{N_{\rm p}-1}$ in the tree as the product of the weights of the edges in the path then the weight of the path expresses the probability of the realization $\tilde{\ell}(k)=\tilde{\ell}^{\rm real}(n_{N_{\rm p}-1})$ corresponding to node $n_{N_{\rm p}-1}$, denoted by $P(n_{N_{\rm p}-1}):=P(\tilde{\ell}(k)|\ell(k))$. In a similar way we can define the probability $P(n)$ for each node n in the tree, where the probability $P(n_0)$ of the root node equals 1 by definition.

We now propose the following main algorithm consisting of two main steps:

- **Step 1**: Select $n_{\max}$ paths of length $N_{\rm p}-1$ in the search tree $\mathcal{T}$ using a random selection or a greedy approach. This results in a candidate set of realizations $\mathcal{L}_{N_{\rm p}}^{\rm red}=\{\bar{\ell}^1,\bar{\ell}^2,\ldots,\bar{\ell}^{n_{\max}}\}$. Define
$$\pi^{\rm red}=\min_{\bar{\ell}\in\mathcal{L}_{N_{\rm p}}^{\rm red}}P(\bar{\ell}|\ell(k))\ . \tag{21}$$
- **Step 2**: Apply a breadth-first search [12] in $\mathcal{T}$, cutting[4] a subtree originating in a node n if $P(n)\leq\pi^{\rm red}$, and updating $\mathcal{L}_{N_{\rm p}}^{\rm red}$ whenever a leaf node $n_{\rm leaf}$ is encountered such that $P(n_{\rm leaf})>\pi^{\rm red}$; in the latter case, the node $\bar{\ell}$ in $\mathcal{L}_{N_{\rm p}}^{\rm red}$ with the lowest probability is removed and replaced by $\tilde{\ell}^{\rm real}(n_{\rm leaf})$ and $\pi^{\rm red}$ is updated accordingly (cf. (21)).

Below we give a more detailed description for the algorithms of Step 1 (in case greedy search is selected) and of Step 2. It is important to note that in the algorithms we also keep track of the probability and the level of the node. In that way we do not have to construct the search tree explicitly. So whenever we select a node in either algorithm we immediately compute its probability and level, and store them along with the node; so the probability of the nodes considered in the algorithms is always assumed to be known as well as whether or not they are leaf nodes (these are characterized by a level equal to $N_{\rm p}-1$). In algorithm 1 the cardinality of the set $\mathcal{L}_{N_{\rm p}}^{\rm red}$ is denoted by $|\mathcal{L}_{N_{\rm p}}^{\rm red}|$.

Algorithm 1: Greedy search

$\mathcal{L}_{N_{\rm p}}^{\rm red}\leftarrow\emptyset$
$\mathcal{N}\leftarrow\{n_0\}$
while $|\mathcal{L}_{N_{\rm p}}^{\rm red}|<n_{\max}$
  $n_{\rm c}\leftarrow\arg\max_{n\in\mathcal{N}}P(n)$
  substitute $n_{\rm c}$ in $\mathcal{N}$ by its child nodes
  for each $n\in\mathcal{N}$
    if n is a leaf node
      $\mathcal{L}_{N_{\rm p}}^{\rm red}\leftarrow\mathcal{L}_{N_{\rm p}}^{\rm red}\cup\{\tilde{\ell}^{\rm real}(n)\}$
      $\mathcal{N}\leftarrow\mathcal{N}\setminus\{n\}$
      if $|\mathcal{L}_{N_{\rm p}}^{\rm red}|=n_{\max}$
        exit from while loop
      end if
    end if
  end for
end while

Algorithm 2: Branch-and-bound breadth-first search

$d\leftarrow 0$
$\mathcal{N}\leftarrow\{n_0\}$
while $d<N_{\rm p}-1$ and $\mathcal{N}\neq\emptyset$
  substitute each node $n\in\mathcal{N}$ by its child nodes
  for each $n\in\mathcal{N}$
    if n is a leaf node and $P(n)>\pi^{\rm red}$
      replace the node $\bar{\ell}$ in $\mathcal{L}_{N_{\rm p}}^{\rm red}$ with the lowest probability by $\tilde{\ell}^{\rm real}(n)$ and update $\pi^{\rm red}$ accordingly
    end if
    if $P(n)\leq\pi^{\rm red}$
      $\mathcal{N}\leftarrow\mathcal{N}\setminus\{n\}$
    end if
  end for
  $d\leftarrow d+1$
end while

The main algorithm presented above results in a set $\mathcal{L}_{N_{\rm p}}^{\rm red}$ containing the $n_{\max}$ most probable paths. If necessary, e.g., if the running time of the breadth-first search becomes too long, we could terminate the second algorithm prematurely or even skip Step 2 of the main algorithm altogether, and continue the MPC algorithm using the set $\mathcal{L}_{N_{\rm p}}^{\rm red}$ obtained thus far. In the MPC step we will now use optimization over the reduced set $\mathcal{L}_{N_{\rm p}}^{\rm red}$ of realizations instead of optimization over the full set $\mathcal{L}_{N_{\rm p}}$. Assume that the reduced set $\mathcal{L}_{N_{\rm p}}^{\rm red}$ can be rewritten as $\mathcal{L}_{N_{\rm p}}^{\rm red}=\{\bar{\ell}^1,\bar{\ell}^2,\ldots,\bar{\ell}^{n_{\max}}\}$. The linear programming problem with the reduced set is now given by

$$\min_{\{\tilde{u}(k),t_{i,m}\}}\sum_{i=1}^{n_yN_{\rm p}}\sum_{m=1}^{n_{\max}}t_{i,m}P(\bar{\ell}^m|\ell(k))-\beta\sum_{i=1}^{n_uN_{\rm p}}\tilde{u}_i(k) \tag{22}$$

subject to

$$t_{i,m}\geq[\tilde{C}(\bar{\ell}^m)]_{i,l}+x_l(k-1)-\tilde{r}_i(k)\ ,\ \ \forall i,m,l \tag{23}$$
$$t_{i,m}\geq[\tilde{D}(\bar{\ell}^m)]_{i,l}+\tilde{u}_l(k)-\tilde{r}_i(k)\ ,\ \ \forall i,m,l \tag{24}$$
$$t_{i,m}\geq 0\ ,\ \ \forall i,m \tag{25}$$
$$u_i(k+j)-u_i(k+j-1)\geq 0,\ \ \forall i,j \tag{26}$$

[3]Usually the weight of a path is defined as the sum of edge weights, but by considering logarithms our definition can be recast into the standard definition.

[4]The idea behind this is that any leaf node of this subtree cannot have a probability that is higher than that of the realizations that are already in $\mathcal{L}_{N_{\rm p}}^{\rm red}$; so there is no need to consider and explore the subtree any further.

$$\mu_{\min} \le u_i(k+j) - \rho\, k \le \mu_{\max}, \quad \forall i, j \tag{27}$$

We will now discuss the complexity reduction due to the scenario algorithm. Consider the linear programming problem of Theorem 4 for a system with n_x states, n_u inputs, n_y outputs, L modes, and a prediction horizon $N_{\rm p}$. The number of decision variables is equal to

the number of variables t	:	$N_{\rm p}\, n_y\, M$
the number of variables $\tilde{u}$	:	$N_{\rm p}\, n_u$
total number	:	$N_{\rm p}\, (M\, n_y + n_u)$

and the number of constraints is given by:

constraint (16)	:	$N_{\rm p}\, n_y\, M\, n_x$
constraint (17)	:	$N_{\rm p}\, n_y\, M\, n_u\, N_{\rm p}$
constraint (18)	:	$N_{\rm p}\, n_y\, M$
constraint (19)	:	$N_{\rm p}\, n_u$
constraint (19)	:	$N_{\rm p}\, n_u$
total number	:	$N_{\rm p}\, n_y\, M\, (n_x + n_u N_{\rm p} + 1) + 2 N_{\rm p}\, n_u$

Note that usually $M\, n_y \gg n_u$ and so we have about $N_{\rm p}\, n_y\, M\, (n_x + n_u N_{\rm p} + 1)$ constraints and $N_{\rm p}\, M\, n_y$ decision variables. With $M = L^{N_{\rm p}-1}$ before reduction and $M = n_{\max}$ after reduction, the reduction in constraints and decision variables is linear with $n_{\max}/L^{N_{\rm p}-1}$. Using the scenario-based approach the complexity will be drastically reduced and can still be very accurate if the limited set of switching sequences have a cumulative probability that is sufficiently close to 1.

V. EXAMPLE

Consider the production system of Figure 2. This system consists of three machines M_1, M_2, and M_3. Three products (A,B,C) can be made with this system, each with its own recipe, meaning that the order in the production sequence is different for every product.

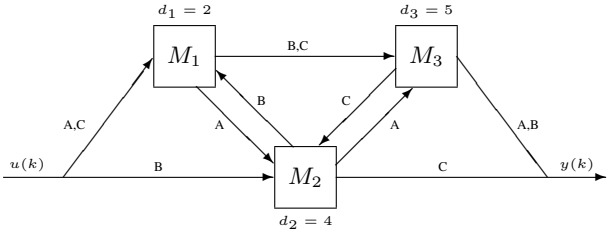

Fig. 2. A production system.

For product A the production order is M_1-M_2-M_3, which means that the raw material is fed to machine M_1 where it is processed. The intermediate product is sent to machine M_2 for further processing, and finally the product is finished in machine M_3. Similarly, for product B the processing order is M_2-M_1-M_3, and for product C the processing order is M_1-M_3-M_2. We assume that the type of the kth product (A, B, or C) becomes available just before the start of the production, so that we know $\ell(k)$ when computing $u(k)$.

Each machine starts working as soon as possible on each batch, i.e., as soon as the raw material or the required intermediate products are available, and as soon as the machine is idle (i.e., the previous batch has been finished and has left the machine). We define $u(k)$ as the time instant at which the system is fed for the kth time, $x_i(k)$ as the time instant at which machine i starts for the kth time, and $y(k)$ as time instant at which the kth product leaves the system. We assume that all the internal buffers are large enough, and no overflow will occur.

We assume the transportation times between the machines to be negligible, and the processing time of the machines M_1, M_2 and M_3 are given by $d_1 = 2$, $d_2 = 4$ and $d_3 = 5$, respectively. The system equations for x_1 and x_2 for recipe A are given by

$$\begin{aligned}
x_1(k) &= \max(x_1(k-1) + d_1, u(k))\ , \\
x_2(k) &= \max(x_1(k) + d_1, x_2(k-1) + d_2) \\
&= \max(x_1(k-1) + 2d_1, x_2(k-1) + d_2, u(k) + d_1)\ , \\
x_3(k) &= \max(x_2(k) + d_2, x_3(k-1) + d_3) \\
&= \max(x_1(k-1) + 2d_1 + d_2, x_2(k-1) + 2d_2, \\
&\quad x_3(k-1) + d_3, u(k) + d_1 + d_2)\ , \\
y(k) &= x_3(k) + d_3\ ,
\end{aligned}$$

leading to the systems matrices for recipe A:

$$A^{(1)} = \begin{bmatrix} d_1 & \varepsilon & \varepsilon \\ 2d_1 & d_2 & \varepsilon \\ 2d_1 + d_2 & 2d_2 & d_3 \end{bmatrix}, \quad B^{(1)} = \begin{bmatrix} 0 \\ d_1 \\ d_1 + d_2 \end{bmatrix},$$

$$C^{(1)} = \begin{bmatrix} \varepsilon & \varepsilon & d_3 \end{bmatrix}.$$

Similarly we derive for recipe B:

$$A^{(2)} = \begin{bmatrix} d_1 & 2d_2 & \varepsilon \\ \varepsilon & d_2 & \varepsilon \\ 2d_1 & d_1 + 2d_2 & d_3 \end{bmatrix}, \quad B^{(2)} = \begin{bmatrix} d_2 \\ 0 \\ d_1 + d_2 \end{bmatrix},$$

$$C^{(2)} = \begin{bmatrix} \varepsilon & \varepsilon & d_3 \end{bmatrix},$$

and for recipe C:

$$A^{(3)} = \begin{bmatrix} d_1 & \varepsilon & \varepsilon \\ 2d_1 + d_3 & d_2 & 2d_3 \\ 2d_1 & \varepsilon & d_3 \end{bmatrix}, \quad B^{(3)} = \begin{bmatrix} 0 \\ d_1 + d_3 \\ d_1 \end{bmatrix},$$

$$C^{(3)} = \begin{bmatrix} \varepsilon & d_2 & \varepsilon \end{bmatrix}.$$

The switching probability from one recipe to the next one is assumed to be given by:

$$\begin{array}{lll} P(1,1) = 0.7\ , & P(1,2) = 0.15\ , & P(1,3) = 0.15\ , \\ P(2,1) = 0.15\ , & P(2,2) = 0.7\ , & P(2,3) = 0.15\ , \\ P(3,1) = 0.15\ , & P(3,2) = 0.15\ , & P(3,3) = 0.7\ , \end{array}$$

which means that if we have a specific recipe in cycle k, then the probability of having the same recipe for cycle $k+1$ is 70%, and the probability of a switching to any other recipe is 15%. Note that this system is indeed an RSMPL system.

The maximum growth rate of the system is equal to $\lambda = 10$. We therefore choose a reference signal given by $r(k) =$

$\rho \cdot k$, where $\rho = 11 > \lambda$. The initial state is equal to $x(0) = \begin{bmatrix} 35 & 2.6 & 6.5 \end{bmatrix}^T$, and J is given by (10) for $N_{\mathrm{p}} = 8$, and $\beta = 10^{-4}$.

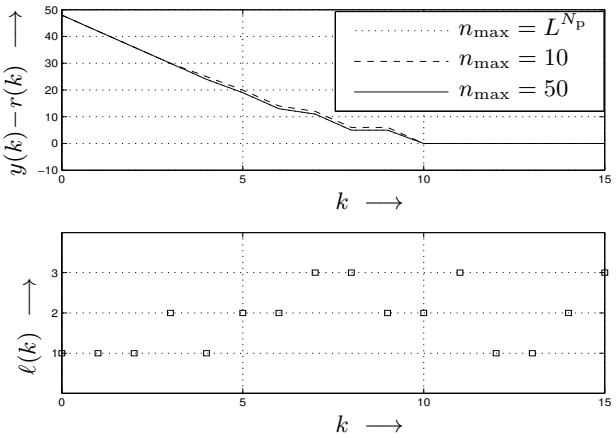

Fig. 3. (a) Tracking error $y(k)-r(k)$ and (b) switching sequence

We do the experiment for three different values of M:

- $n_{\max} = L^{N_{\mathrm{p}}} = 6561$: This means that we do not use the scenario-based algorithm, and use the complete set of possible switching sequences.
- $n_{\max} = 50$: We consider only the 50 most probable switching sequences. The cumulative probability of all selected switching sequences is 83%.
- $n_{\max} = 10$: We consider only the 10 most probable switching sequences. The cumulative probability of all selected switching sequences is 49%.

Figure 3 shows the result for the closed-loop simulation for the three different values of $n_{\max}$ and the actual switching sequence is given in Figure 3-b. Figure 3-a gives the tracking error between the reference signal and the output signal $y(k)$ for three different cases. For $n_{\max} = L^{N_{\mathrm{p}}} = 6561$ we have the dotted line (which coincides with the solid line). For the reduced case with $n_{\max} = 50$ we observe that the output is still the same as for $N_{\max} = 6561$. If we further reduce the number of sequences to $n_{\max} = 10$, a small approximation error appears. The maximum approximation error is 1. For all three responses it can be observed that $y(k)-r(k)$ is initially larger than zero, which is due to the initial state. The error decreases very rapidly and for $k \geq 10$ the error is always equal to zero, which means that the product is always delivered in time.

VI. DISCUSSION

In this paper we have considered the control of randomly switching max-plus-linear systems, a subclass of the discrete event systems, in which we can switch between different modes of operation. In each mode the system is described by max-plus-linear equations with different system matrices for each mode. The moments of switching are determined by a stochastic variable. The stabilizing model predictive control problem can be solved using linear programming algorithms. The computational complexity has been reduced by solving the problem using a scenario-based optimization tree, in which only the most relevant disturbance patterns are taken into account. The proposed approach is probably not so effective when a uniform switching probability is given.

In future research we will try to find appropriate tuning rules for $n_{\max}$ and to derive an estimation of the approximation error. Further we will study 'ordinal optimization' [8], [9]. Instead of computing the exact objective function, this method concentrates on finding the most promising control decisions.

Acknowledgments

Research partially funded by the Dutch Technology Foundation STW project "Model-predictive railway traffic management" (11025), and by the European 7th Framework Network of Excellence "Highly-complex and networked systems (HYCON2)".

REFERENCES

[1] F. Baccelli, G. Cohen, G.J. Olsder, and J.P. Quadrat. *Synchronization and Linearity*. John Wiley & Sons, New York, 1992. Text can be downloaded from *http://www-rocq.inria.fr/metalau/cohen/SED/book-online.html*.

[2] D. Bernardini and A. Bemporad. Scenario-based model predictive control of stochastic constrained linear systems. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 6333–6338, Shanghai, China, December 2009.

[3] E.F. Camacho and C. Bordons. *Model Predictive Control in the Process Industry, Advances in Industrial Control*. Springer, London, 1995.

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

[5] B. De Schutter and T. van den Boom. Model predictive control for max-plus-linear discrete event systems. *Automatica*, 37(7):1049–1056, July 2001.

[6] J. Dupačova, G. Consigli, and S. Wallace. Scenarios for multistage stochastic programs. *Annals of Operations Research*, 100(1):25–53, 2004.

[7] D. Eppstein. Finding the k shortest paths. *SIAM Journal on Computing*, 28(2):652–673, 1998.

[8] Y.C. Ho, C.C. Cassandras, C.-H. Chen, and L. Dai. Ordinal optimization and Simulation. *Journal of the Operational Research Society*, 51(4):490–500, 2000.

[9] Y.C. Ho, R.S. Sreenivas, and P. Vakili. Ordinal optimization of DEDS. *Discrete Event Dynamic Systems*, 2:61–88, 1992.

[10] K. Hoyland and S. Wallace. Generating scenario trees for multistage decision problems. *Management Science*, 47(2):295–307, 2001.

[11] L.G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20(1):191–194, January–February 1979.

[12] D.E. Knuth. *The Art of Computer Programming — Volume 1: Fundamental Algorithms*. Addison-Wesley, Boston, Massachusetts, 3rd edition, 1997.

[13] J.M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, Pearson Education Limited, Harlow, UK, 2002.

[14] D. Muñoz de la Peña, A. Bemporad, and T. Alamo. Stochastic programming applied to model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 1361–1366, Sevilla, Spain, December 2005.

[15] K.M. Passino and K.L. Burgess. *Stability Analysis of Discrete Event Systems*. John Wiley & Sons, Inc., New York, USA, 1998.

[16] T.J.J. van den Boom and B. De Schutter. Properties of MPC for max-plus-linear systems. *European Journal of Control*, 8(5):53–62, 2002.

[17] T.J.J. van den Boom and B. De Schutter. Modelling and control of discrete event systems using switching max-plus-linear systems. *Control Engineering Practice*, 14(10):1199–1211, October 2006.

[18] T.J.J. van den Boom and B. De Schutter. Stabilizing controllers for randomly switching max-plus-linear discrete event systems. In *Proceedings of the European Control Conference 2007*, pages 4952–4959, Kos, Greece, July 2007.