Efficient Mode Enumeration of Compositional Hybrid Systems

Tobias Geyer, Fabio Danilo Torrisi, and Manfred Morari

Automatic Control Laboratory, Swiss Federal Institute of Technology (ETH)
CH-8092 Zurich, Switzerland, {geyer,torrisi,morari}@aut.ee.ethz.ch

Abstract. A hyperplane arrangement is a polyhedral cell complex where the relative position of each cell of the arrangement and the composing hyperplanes are summarized by a sign vector computable in polynomial time. This tool from computational geometry enables the development of a fast and efficient algorithm that translates the composition of hybrid systems into a piecewise affine model. The tool provides also information on the real combinatorial degree of the system which can be used to reduce the size of the search tree and the computation time of the optimization algorithms underlying optimal and model predictive control.

1 Introduction

A *hybrid system* is a collection of digital programs interacting among each other and with an analog environment. Each *logic state* of the digital part of the hybrid system acts on the analog part inducing a different operational *mode*. On the other hand, the evolution of the analog part triggers switches in the states of the digital part. The practical relevance of hybrid systems is twofold. Digital controllers embedded in a continuous environment demand for adequate modelling, analysis and design tools. Moreover, many physical phenomena admit a natural hybrid description, like circuits integrating relays or diodes, biomolecular networks and TCP/IP networks.

Hybrid systems can be composed to form *compositional hybrid systems* [2, 21, 22, 24]. The resulting system is in general very complex, as the number of different operational modes depends exponentially on the number of component systems. The explosion of the size of the logic state leads to computational difficulties as the time and space complexity of many algorithms depends on the number of operational modes.

In some cases, the composition induces a structure that can be exploited, like in hierarchical hybrid systems [1]. This allows one to break the problem down into pieces and to apply the assume-guarantee approach [2, 19]. Recognizing that a system can be modelled as a hierarchical hybrid system is part of the "art" of model building. In many cases it may not be possible at all, because the cross interactions among the components are too tight. However, tight interactions often render many modes infeasible and the complexity of the system can be reduced by explicitly computing and taking into account only the feasible modes. This paper presents an efficient technique to enumerate the modes of a compositional hybrid system.

We will focus on *discrete hybrid automata* (DHA) [28]. DHA models are a mathematical abstraction of the features provided by other computation oriented and domain specific hybrid system frameworks: *Mixed logical dynamical* (MLD) *models* [7], *piecewise affine* (PWA) *systems* [25], *linear complementarity* (LC) *systems* [29], *extended linear complementarity* (ELC) *systems* [18], and *max-min-plus-scaling* (MMPS) *systems* [10, 18]. In particular, as shown first in [26] and then, with different arguments, in [18, 28] all those modelling frameworks are equivalent and it is possible to represent the same system using any of these frameworks.

DHAs are formulated in the discrete-time domain. They generalize many computation oriented models for hybrid systems and therefore represent a universal starting point for solving complex analysis and synthesis problems for hybrid systems. In particular, the enumeration of feasible modes is easily solvable for DHA systems by using algorithms that compute the cells of an arrangement of hyperplanes [16]. This is a classical problem in computational geometry [9] and admits optimal [14] and efficient [3, 15, 16] algorithms.

The impact of those techniques on applications is threefold. First, at the modelling stage, the enumeration of modes allows the designer to understand the real complexity of the compound model. This is mandatory for performing model reduction by merging modes.

Second, after the modelling stage, the model can be efficiently translated into a PWA model. This operation is trivial once all modes have been enumerated. In this respect, this paper solves a problem similar to [4] where the author computes the PWA model equivalent to an MLD system. The main difference is that the approach in [4] is based on multi-parametric programming and mixed integer linear programming and deals directly with the MLD model, while the approach presented here relies on the computation of the cells of the hyperplane arrangement and is applicable to DHAs. Note that, as shown in [28], DHAs can be automatically translated into MLD models using the tool HYSDEL and most of the MLD models that have been presented in the literature were derived from DHA descriptions using HYSDEL [5, 8, 13].

Third, during the computational stage (i.e. analysis and control), the explicit computation of the set of feasible modes of the compound system allows to prune unnecessary modes from the resulting system and to reduce the combinatorial explosion of related algorithms. This is of particular importance for *model predictive control* (MPC) of hybrid models [7], where the aim is to compute the next N inputs that optimize a performance index defined on the variables of a hybrid *prediction model*. The prediction model is the series-connection of N identical single-step prediction models where each model uses the state predicted by the previous model. The mode enumeration allows one to introduce cuts on the modes of the complete prediction model.

The paper is organized as follows: In Sect. 2, we introduce the class of DHA. Section 3 presents the problem of cell enumeration in hyperplane arrangements and in Sect. 4, the equivalence between the class of DHA and PWA systems is proved. Section 5 details how these results can be applied in the hybrid domain. Finally, Sect. 6 contains experimental results where we enumerate the modes of a hybrid model built using the HYSDEL [28] modelling language. The information collected during this mode enumeration allows one to either build efficiently

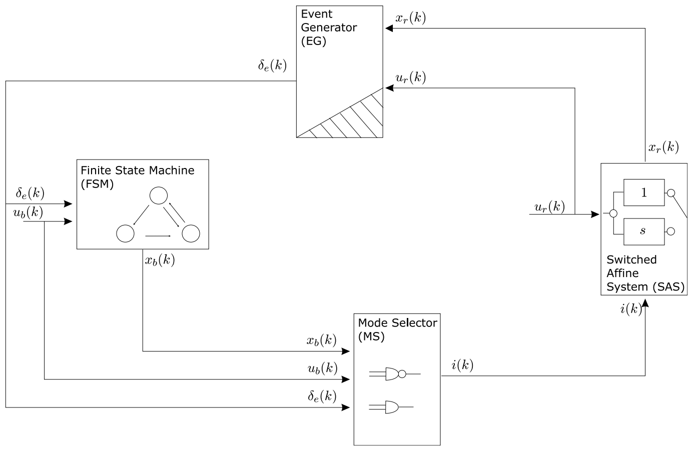


Fig. 1. A discrete hybrid automaton (DHA) is the connection of a finite state machine (FSM) and a switched affine system (SAS) through a mode selector (MS) and an event generator (EG). The output signals are omitted for clarity.

an equivalent PWA model or to speed up the computation of the hybrid MPC feedback law by automatically adding cuts to the optimization problem. Section 7 summarizes the results and points out directions for future research.

2 Discrete Hybrid Automata

Discrete hybrid automata [28] are the interconnection of a finite state machine and a switched affine system through a mode selector and an event generator (see Fig. 1).

Switched Affine System (SAS). A switched affine system is a collection of affine systems:

$$x_r(k+1) = A_{i(k)}x_r(k) + B_{i(k)}u_r(k) + f_{i(k)} \tag{1a}$$

$$y_r(k) = C_{i(k)}x_r(k) + D_{i(k)}u_r(k) + g_{i(k)}, \tag{1b}$$

where $k \in \mathbb{N}$ is the time indicator, $x_r \in \mathcal{X}_r \subseteq \mathbb{R}^{n_r}$ is the continuous state vector, $u_r \in \mathcal{U}_r \subseteq \mathbb{R}^{m_r}$ is the exogenous continuous input vector, $y_r \in \mathcal{Y}_r \subseteq \mathbb{R}^{p_r}$ is the continuous output vector, $\{A_i, B_i, f_i, C_i, D_i, g_i\}_{i\in\mathcal{I}}$ is a collection of matrices of appropriate dimensions and the mode $i(k) \in \mathcal{I} \subset \mathbb{N}$ is an input signal selecting the affine state-update dynamics and the affine output function[1].

Event Generator (EG). An event generator is a mathematical object that generates a logic signal according to the fulfillment of a constraint:

$$\delta_e(k) = f_{\mathrm{H}}(x_r(k), u_r(k)), \tag{2}$$

where $f_{\mathrm{H}} : \mathbb{R}^{n_r} \times \mathbb{R}^{m_r} \to \mathcal{D} \subseteq \{0,1\}^{n_e}$ is a vector of descriptive functions of a set of linear constraints. In particular, *threshold events* are modelled as

[1] A SAS of the form (1) preserves the value of the state when a switch occurs, however it is possible to implement reset maps on a DHA as shown in [28, Proposition 1].

$[\delta_e^i(k) = 1] \leftrightarrow [f_H^i(x_r(k), u_r(k)) \leq 0]$, and *time events* are modelled by adding a clock variable $t(k)$ in the switched affine system with dynamics $t(k{+}1) = t(k){+}T_s$ and setting $[\delta_e^i(k) = 1] \leftrightarrow [t(k) \geq t_0]$, where the superscript i denotes the i-th component of a vector and T_s is the sampling time.

Finite State Machine (FSM). A finite state machine (or automaton) is a discrete dynamic process that evolves according to a logic state-update function:

$$x_b(k+1) = f_{\mathrm{B}}(x_b(k), u_b(k), \delta_e(k)) \tag{3a}$$

$$y_b(k) = g_{\mathrm{B}}(x_b(k), u_b(k), \delta_e(k)), \tag{3b}$$

where $x_b \in \mathcal{X}_b \subseteq \{0,1\}^{n_b}$ is the binary state, $u_b \in \mathcal{U}_b \subseteq \{0,1\}^{m_b}$ the exogenous binary input, $y_b \in \mathcal{Y}_b \subseteq \{0,1\}^{p_b}$ the output, $\delta_e(k)$ the logic signal defined by the EG and $f_{\mathrm{B}} : \mathcal{X}_b \times \mathcal{U}_b \times \mathcal{D} \to \mathcal{X}_b$, $g_{\mathrm{B}} : \mathcal{X}_b \times \mathcal{U}_b \times \mathcal{D} \to \mathcal{Y}_b$ are deterministic logic functions.

Mode Selector (MS). The logic state $x_b(k)$, the binary input $u_b(k)$, and the event $\delta_e(k)$ select the dynamic mode $i(k)$ of the SAS through a Boolean function $f_{\mathrm{M}} : \mathcal{X}_b \times \mathcal{U}_b \times \mathcal{D} \to \mathcal{I}$, which is therefore called *mode selector*. The output of this function

$$i(k) = f_{\mathrm{M}}(x_b(k), u_b(k), \delta_e(k)) \tag{4}$$

is called *active mode*. We say that a *mode switch* occurs at time k if $i(k{-}1) \neq i(k)$. Note that one may associate with a state $x_b(k)$ of the FSM more than one mode $i(k)$ according to the event $\delta_e(k)$.

Definition 1. *A DHA is* well-posed *on $\mathcal{X}_r \times \mathcal{X}_b$, $\mathcal{U}_r \times \mathcal{U}_b$, $\mathcal{Y}_r \times \mathcal{Y}_b$, if for all initial conditions $x(0) = \left[\begin{smallmatrix} x_r(0) \\ x_b(0) \end{smallmatrix}\right] \in \mathcal{X}_r \times \mathcal{X}_b$ and for all inputs $u(k) = \left[\begin{smallmatrix} u_r(k) \\ u_b(k) \end{smallmatrix}\right] \in \mathcal{U}_r \times \mathcal{U}_b$, for all $k \in \mathbb{N}$, the state trajectory $x(k) \in \mathcal{X}_r \times \mathcal{X}_b$ and output trajectory $y(k) = \left[\begin{smallmatrix} y_r(k) \\ y_b(k) \end{smallmatrix}\right] \in \mathcal{Y}_r \times \mathcal{Y}_b$ are uniquely defined.*

3 Cell Enumeration in Hyperplane Arrangement

In this section, we recall the notation and the solution approach to the problem of enumerating the cells of a hyperplane arrangement using reverse search [3].

Let $\mathcal{A}$ be a collection of n distinct hyperplanes $\{H_i\}_{i=1,\ldots,n}$ in $\mathbb{R}^d$, where each hyperplane is given by a linear equality $H_i = \{x : a_i x = b_i\}$. Let $SV : \mathbb{R}^d \to \{-, 0, +\}^n$ be the sign vector defined as

$$SV_i(x) = \begin{cases} - & \text{if } a_i x < b_i \\ 0 & \text{if } a_i x = b_i \quad i \in \{1, 2, \ldots, n\} \\ + & \text{if } a_i x > b_i \end{cases} \tag{5}$$

Consider the set $\mathcal{P}_m = \{x : SV(x) = m\}$ for a given sign vector m. This set is called *cell* of the arrangement and is a polyhedron as it is defined by equalities and inequalities. Note, that each 0 element of m is associated with an equality constraint in (5) and reduces the dimension of $\mathcal{P}_m$ by one. We will refer to m as the *marking* of the polyhedron or the cell $\mathcal{P}_m$ in the *hyperplane arrangement*

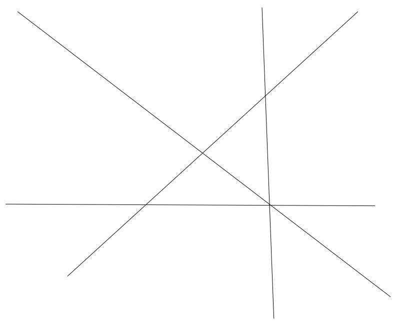


Fig. 2. Arrangement of four hyperplanes (lines) in $\mathbb{R}^2$ with the markings.

$\mathcal{A}$ (see Fig. 2). Let $M(\mathcal{R})$ be the image of the function $SV(x)$ for $x \in \mathcal{R} \subseteq \mathbb{R}^d$, namely the collection of all the possible markings of all the points in $\mathcal{R}$.

The problem of enumerating the cells of an arrangement amounts to enumerate all the elements of the set $M(\mathcal{R})$. Clearly, it is enough to enumerate the sign vectors that have all nonzero entries, as any lower dimensional cell is a face of a full dimensional cell [30, Theorem 7.16].

We say that n hyperplanes $\{H_i\}_{i=1,\dots,n}$ in $\mathbb{R}^d$ are in *general position*, if there are no two parallel hyperplanes and if any point of $\mathbb{R}^d$ belongs at most to d hyperplanes. Let $\#M(\mathcal{R})$ be the number of full dimensional cells identified by $M(\mathcal{R})$. Buck's formula defines the following upper bound: $\#M \leq \sum_{i=0}^{d} \binom{n}{i} = O(n^d)$ [9], with the equality satisfied when the hyperplanes are in general position and $\mathcal{R} = \mathbb{R}^d$.

The cell enumeration problem admits an optimal solution with time and space complexity $O(n^d)$ [14]. An alternative approach based on reverse search was presented in [3], improved in [16] and implemented in [15]. Reverse search is an exhaustive search technique which can be considered as a special graph search. This search technique has been used to design efficient algorithms for various enumeration problems such as enumeration of all spanning trees and cells in hyperplane arrangements. Before evaluating the complexity of the algorithm based on reverse search, let $\mathrm{lp}(n,d)$ denote the complexity of solving a linear program (LP) with n constraints in d variables.

Proposition 1. *[16, Theorem 4.1] There is a reverse search algorithm for enumeration of hyperplane arrangements that runs in $O(n\mathrm{lp}(n,d)\#M)$ time and $O(n,d)$ space.*

Note that in many cases of interest, the hyperplanes are not in general position and $\#M$ can be considerably smaller than the theoretical upper bound.

The following proposition follows directly from the definition of $\mathcal{P}_m$ and (5).

Proposition 2. *The collection of polyhedral sets $\{\mathcal{P}_m\}_{m\in M(\mathcal{R})}$ satisfies:*

$$(i)\quad \bigcup_{m\in M(\mathcal{R})} \mathcal{P}_m = \mathcal{R}, \qquad (ii)\quad (\mathcal{P}_i) \cap (\mathcal{P}_j) = \emptyset, \forall i \neq j.$$

A collection of polyhedral sets that satisfies points *(i)* and *(ii)* in Proposition 2 is a *polyhedral partition* of a polyhedral set $\mathcal{R}$.

4 Discrete Hybrid Automata and Piecewise Affine Systems

PWA systems [18, 25] are defined by partitioning the state space into polyhedral regions and associating with each region a different linear state-update function

$$\begin{aligned} x(k+1) &= A_{j(k)}x(k) + B_{j(k)}u(k) + f_{j(k)} && \text{(6a)}\\ y(k) &= C_{j(k)}x(k) + D_{j(k)}u(k) + g_{j(k)} && \text{(6b)}\\ j(k) \text{ such that} \quad & H_{j(k)}x(k) + J_{j(k)}u(k) \le K_{j(k)} && \text{(6c)}\\ & \tilde{H}_{j(k)}x(k) + \tilde{J}_{j(k)}u(k) < \tilde{K}_{j(k)}, && \text{(6d)} \end{aligned}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^n$, $u \in \mathcal{U} \subseteq \mathbb{R}^m$, $y \in \mathcal{Y} \subseteq \mathbb{R}^p$, the matrices $A_{j(k)}$, $B_{j(k)}$, $f_{j(k)}$, $C_{j(k)}$, $D_{j(k)}$, $g_{j(k)}$, $H_{j(k)}$, $J_{j(k)}$, $K_{j(k)}$, $\tilde{H}_{j(k)}$, $\tilde{J}_{j(k)}$, $\tilde{K}_{j(k)}$, $j(k) \in \mathcal{J}$, $\mathcal{J}$ finite, are constant and have suitable dimensions, the inequalities in (6c) and (6d) should be interpreted componentwise and the constraints (6c) and (6d) define a polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}}$ of the set $\mathcal{X} \times \mathcal{U}$.[2]

Definition 2. *Let Σ_1, Σ_2 be hybrid models with inputs $u_1(k), u_2(k) \in \mathcal{U}$, states $x_1(k), x_2(k) \in \mathcal{X}$ and outputs $y_1(k), y_2(k) \in \mathcal{Y}$, $k \in \mathbb{N}$. The hybrid models Σ_1 and Σ_2 are equivalent on $\mathcal{X}$,$\mathcal{U}$,$\mathcal{Y}$ if for all $u_1(k) = u_2(k) = u(k) \in \mathcal{U}$ the output trajectories $y_1(k)$ and $y_2(k)$ coincide and $x_1(k) = x_2(k)$ for all time-instants $k \in \mathbb{N}$.*

Lemma 1. *[28, Lemma 1] Let Σ_{PWA} be a well-posed[3] PWA model defined on a set of states $\mathcal{X} \subseteq \mathbb{R}^n$, a set of inputs $\mathcal{U} \subseteq \mathbb{R}^m$ and a set of outputs $\mathcal{Y} \subseteq \mathbb{R}^p$. Then there exists a well-posed DHA model Σ_{DHA} such that Σ_{DHA} and Σ_{PWA} are equivalent.*

The equivalence of the previous lemma allows us to call $\mathcal{J}$ modes of the PWA system (6a)-(6d).

Lemma 2. *Let Σ_{DHA} be a well-posed DHA model defined on a set of states $\mathcal{X} \subseteq \mathbb{R}^n$, a set of inputs $\mathcal{U} \subseteq \mathbb{R}^m$ and a set of outputs $\mathcal{Y} \subseteq \mathbb{R}^p$. Then there exists a well-posed PWA model Σ_{PWA} such that Σ_{PWA} and Σ_{DHA} are equivalent.*

Proof. Consider the arrangement of a set of hyperplanes defining the linear constraints of the EG. By Proposition 2 this defines a polyhedral partition. Let $\mathcal{P}_m$ be a polyhedron of that partition. By construction, $\delta_e = \bar{\delta}_e(m) = f_{\text{H}}(x_r, u_r)$ for any point $[x_r^T, u_r^T]^T \in \mathcal{P}_m$, namely all the points in $\mathcal{P}_m$ trigger the same event vector $\bar{\delta}_e(m)$. Given a marking m, the associated event $\bar{\delta}_e(m)$, a binary state $\bar{x}_b \in \mathcal{X}_b$ and a binary input $\bar{u}_b \in \mathcal{U}_b$, the MS determines the mode $\bar{i}$ using the Boolean function (4). The $\bar{i}$-th dynamic in the SAS given by (1a) and (1b) is the corresponding affine dynamic. The FSM yields the binary state-update $x_b(k+1)$

[2] Note that (6d) ensures the single definition of the functions (6a) and (6b) on borders of the cells of the partition.

[3] For PWA systems, well-posedness is defined similarly to Definition 1.

as well as the binary output $y_b(k)$ according to (3a) and (3b). Therefore, for each $m \in M$, $\bar{x}_b \in \mathcal{X}_b$ and $\bar{u}_b \in \mathcal{U}_b$, the system

$$x_r(k+1) = A_{f_{\mathrm{M}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)})} x_r(k) + B_{f_{\mathrm{M}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)})} u_r(k) + f_{f_{\mathrm{M}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)})}, \quad (7a)$$
$$x_b(k+1) = f_{\mathrm{B}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)}), \quad (7b)$$
$$y_r(k) = C_{f_{\mathrm{M}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)})} x_r(k) + D_{f_{\mathrm{M}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)})} u_r(k) + g_{f_{\mathrm{M}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)})}, \quad (7c)$$
$$y_b(k) = g_{\mathrm{B}}(\bar{x}_b,\bar{u}_b,\bar{\delta}_{e(m)}), \quad (7d)$$
$$\text{if } x_b(k) = \bar{x}_b, \quad u_b(k) = \bar{u}_b, \quad [x_r^T(k), u_r^T(k)]^T \in \mathcal{P}_m, \quad (7e)$$

defines a PWA system. In fact, by collecting $x = \left[\begin{smallmatrix} x_r \\ x_b \end{smallmatrix}\right]$ and $y = \left[\begin{smallmatrix} y_r \\ y_b \end{smallmatrix}\right]$, (7a)–(7d) are formally equivalent to (6a)–(6b) and (7e) is formally equivalent to (6c)–(6d). □

5 Algorithm

The following section presents an algorithm based on the cell enumeration summarized in Sect. 3 that efficiently enumerates the feasible modes of a composition of DHAs and derives an equivalent PWA model.

5.1 Single DHA

Consider the DHA Σ as in Sect. 2 and let $\mathcal{U}_r \times \mathcal{U}_b \times \mathcal{X}_r \times \mathcal{X}_b$ denote the input-state space of the DHA, for which we want to solve the following problem. Find the set of feasible modes $\mathcal{J} \subseteq \mathcal{I}$[4], the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}}$ and the corresponding PWA dynamics $\{\mathcal{S}_j\}_{j\in\mathcal{J}}$, where $\mathcal{S}_j = \{A_j, B_j, f_j, C_j, D_j, g_j\}$.As this is the same problem as in Lemma 2, we derive an algorithm from the constructive proof of the lemma. Note, that $\mathcal{I}$ is the image of the MS and can be computed once the set $M(\mathcal{U}_r \times \mathcal{X}_r)$ has been enumerated.

Algorithm 1

```
function SingleDHA($\Sigma$, $\mathcal{U}_r$, $\mathcal{X}_r$, $u_b(k)$, $x_b(k)$)
for $m \in M(\mathcal{U}_r \times \mathcal{X}_r)$
  get $\mathcal{P}_m$ defined by $m$ and $\mathcal{A}$, get $\delta_e(k)$ based on $m$
  $i(k) = f_{\mathrm{M}}(x_b(k), u_b(k), \delta_e(k))$
  $x_r(k+1) = A_{i(k)}x_r(k) + B_{i(k)}u_r(k) + f_{i(k)}$,  $x_b(k+1) = f_{\mathrm{B}}(x_b(k), u_b(k), \delta_e(k))$
  $y_r(k) = C_{i(k)}x_r(k) + D_{i(k)}u_r(k) + g_{i(k)}$,  $y_b(k) = g_{\mathrm{B}}(x_b(k), u_b(k), \delta_e(k))$
  push { $\mathcal{P}_m$, $S_{i(k)}$, $u_b(k)$, $x_b(k)$, $x_b(k+1)$, $y_b(k)$ } on STACK
```

Algorithm 1 enumerates the feasible modes and leads to a set of PWA models defined on $\mathcal{U}_r \times \mathcal{X}_r$, where each model is associated with a feasible combination of binary states and inputs $x_b \in \mathcal{X}_b$, $u_b \in \mathcal{U}_b$. This representation is advantageous if determining the state-update and the outputs for a given state and input is the main purpose, as choosing the respective PWA model can be done by binary search. The model can be transformed easily into a PWA model defined over $\mathcal{U}_r \times \mathcal{U}_b \times \mathcal{X}_r \times \mathcal{X}_b$ by associating additional hyperplanes with the binary inputs and states.

[4] $\mathcal{J} = \mathcal{I}$ holds, if all the modes $\mathcal{I}$ of the SAS are feasible.

5.2 Composition of DHAs

The algorithm proposed above can be extended in a natural way to deal with a composition of DHAs. Consider s DHAs denoted as Σ_i, $i = 1, \ldots, s$ with inputs $u_i = \begin{bmatrix} u_r^i \\ u_b^i \end{bmatrix} \in \mathcal{U}_r^i \times \mathcal{U}_b^i$, states $x_i = \begin{bmatrix} x_r^i \\ x_b^i \end{bmatrix} \in \mathcal{X}_r^i \times \mathcal{X}_b^i$, and outputs $y_i = \begin{bmatrix} y_r^i \\ y_b^i \end{bmatrix} \in \mathcal{Y}_r^i \times \mathcal{Y}_b^i$, $i = 1, \ldots, s$. Let $\mathcal{I}_i$ be the set of feasible modes of the DHA Σ_i. The composition has the exogenous input $u_e = \begin{bmatrix} u_r^e \\ u_b^e \end{bmatrix} \in \mathcal{U}_r^e \times \mathcal{U}_b^e$ and the exogenous output $y_e = \begin{bmatrix} y_r^e \\ y_b^e \end{bmatrix} \in \mathcal{Y}_r^e \times \mathcal{Y}_b^e$. Let the compound vector $x_b = [(x_b^1)^T, \ldots, (x_b^s)^T]^T \in \mathcal{X}_b^1 \times \ldots \times \mathcal{X}_b^s$ be the sorted aggregation of the binary states of the s DHAs. The time index k has been omitted for brevity.

Before describing the algorithm, we recall some definitions and results from graph theory [11] to describe the topology of the composition.

Definition 3. *A* directed graph *or* digraph *is an ordered pair of sets $G = (V, E)$, where V is a set of* vertices *and E is a set of ordered pairs of vertices of V called* edges. *A* directed closed walk *is defined as an alternating sequence of vertices and edges, beginning and ending with the same vertex, such that each edge is oriented from the vertex preceding it to the vertex following it. If additionally, no vertices except the initial and terminal one appear more than once, the directed closed walk is called a* directed circuit. *A digraph, that has no directed circuits is called* acyclic.

The definitions above can be applied directly to the composition of DHAs by defining DHA Σ_i, $i = 1, \ldots, s$ as vertex v_i and the connections from outputs to inputs as directed edges. In general one edge can represent several connections. Note that directed circuits correspond to loops and the lack of loops is equivalent to having an acyclic directed graph. We define the connections among the DHAs by an adjacency matrix.

Definition 4. *Let G be a digraph with s vertices containing no parallel edges. Then the adjacency matrix $W = [w_{ij}]$ of the digraph G is a $s \times s$ $(0,1)$-matrix with $w_{ij} = 1$ if there is an edge directed from the i-th vertex to the j-th vertex and $w_{ij} = 0$ otherwise.*

Given a composition of DHAs, the adjacency matrix W can be easily determined based on the connections.

Theorem 1. *[11, Theorem 9.17] Digraph G is* acyclic *if and only if* $\det(I - W)$ *is not equal to zero, where I is the identity matrix of the same size as W.*

Theorem 2. *[11, Theorem 9.16] If digraph G is acyclic, then its vertices can be ordered such that the adjacency matrix W is an upper (or lower) triangular matrix.*

Definition 5. *Given an acyclic graph, the computational order $\mathcal{O}$ is the sequence of indices of W.*

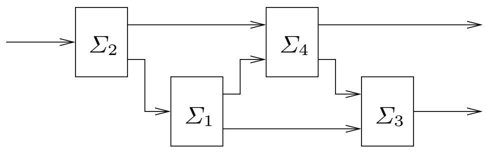


Fig. 3. Example composition of DHAs.

Example 1. Figure 3 depicts four DHAs and the connections among them. DHA 2 has an exogenous input, DHA 3 and 4 have exogenous outputs. The adjacency matrix W follows to

$$W = \begin{array}{c|c} \begin{bmatrix} 0\,0\,1\,1 \\ 1\,0\,0\,1 \\ 0\,0\,0\,0 \\ 0\,0\,1\,0 \end{bmatrix} & \begin{array}{l} 1 \\ 2 \\ 3 \\ 4, \end{array} \end{array}$$

where the indices of the corresponding DHAs are shown on the right side. As the condition $\det(I - W) \neq 0$ holds, the digraph is acyclic and we can reorder W as

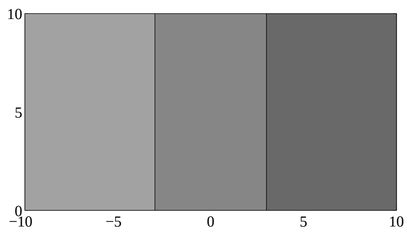
$$W' = \begin{array}{c|c} \begin{bmatrix} 0\,1\,1\,0 \\ 0\,0\,1\,1 \\ 0\,0\,0\,1 \\ 0\,0\,0\,0 \end{bmatrix} & \begin{array}{l} 2 \\ 1 \\ 4 \\ 3, \end{array} \end{array}$$

where the rows refer to Σ_2, Σ_1, Σ_4, Σ_3 and the associated sequence of indices is $\{2, 1, 4, 3\}$ which is also the computational order. This implies for example, that the inputs of Σ_1 do not depend on the outputs of Σ_4 because $\mathcal{O}(1) = 2 < \mathcal{O}(4) = 3$.

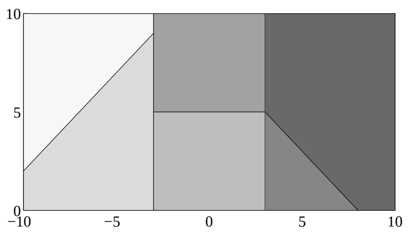
Compositions Without Loops. In a first step, we assume that the connections do not form loops and that the digraph is therefore acyclic. This can be easily determined by Theorem 1. From Theorem 2 follows, that W can be transformed into an upper triangular matrix employing for example topological sorting [11]. The computational order $\mathcal{O}$ follows from Definition 5.

According to the single DHA case, our aim is to determine for the composition of DHAs $\{\Sigma_i\}_{i=1,\dots,s}$ with the corresponding adjacency matrix W the set of feasible modes $\mathcal{J} \subseteq \mathcal{I}_1 \times \ldots \times \mathcal{I}_s$, the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}}$ and the corresponding PWA dynamics $\{\mathcal{S}_j\}_{j\in\mathcal{J}}$, where $\mathcal{S}_j = \{A_j, B_j, f_j, C_j, D_j, g_j\}$. For a given binary compound state $x_b \in \mathcal{X}_b$ and a given exogenous binary input $u_b^e \in \mathcal{U}_b^e$, the Algorithm 2 is the following.

Let $\Sigma_{\mathcal{O}(1)}$ denote the DHA with computational order 1. Its real input-state space is given by $\mathcal{R}_{\mathcal{O}(1)} = \mathcal{U}_r^{\mathcal{O}(1)} \times \mathcal{X}_r^{\mathcal{O}(1)}$, where $\mathcal{U}_r^{\mathcal{O}(1)} \subseteq \mathcal{U}_r^e$ and $\mathcal{X}_r^{\mathcal{O}(1)}$ is a property of $\Sigma_{\mathcal{O}(1)}$. Therefore, Algorithm 1 can be used to determine the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}_{\mathcal{O}(1)}}$ and the corresponding PWA dynamics $\{\mathcal{S}_j\}_{j\in\mathcal{J}_{\mathcal{O}(1)}}$ of $\Sigma_{\mathcal{O}(1)}$. Every polyhedron $\mathcal{P}_j$, $j \in \mathcal{J}_{\mathcal{O}(1)}$ defines via the connections a real input set $\mathcal{U}_r^{\mathcal{O}(2)}$ for the DHA $\Sigma_{\mathcal{O}(2)}$ with computational order 2. Thus, for a given $j \in \mathcal{J}_{\mathcal{O}(1)}$, Algorithm 1 can be used again to compute the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}_{\mathcal{O}(2)}}$ and the corresponding PWA dynamics $\{\mathcal{S}_j\}_{j\in\mathcal{J}_{\mathcal{O}(2)}}$ of $\Sigma_{\mathcal{O}(2)}$. This is repeated for all the remaining $j \in \mathcal{J}_{\mathcal{O}(1)}$.



(a) $\{\mathcal{P}_j\}_{j\in\mathcal{J}_{\mathcal{O}(1)}}$



(b) $\{\mathcal{P}_j\}_{j\in\mathcal{J}_{\mathcal{O}(2)}}$

Fig. 4. Polyhedral partitions for the first two DHAs of Example 2.

The algorithm introduced above is repeated until the DHA $\Sigma_{\mathcal{O}(k)}$ with the highest computational order is reached. The polyhedral partition of this DHA is part of the overall polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}}$ of the compound DHA system and is therefore added to it. Stepping recursively through the composition of DHAs according to their computational order leads to the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}}$ and the corresponding PWA dynamics $\{\mathcal{S}_j\}_{j\in\mathcal{J}}$.

Example 2. We want to derive the polyhedral partitions for the DHAs Σ_1 and Σ_2 of Example 1. Assume that the DHA Σ_1 yields as output the 1-norm of its input and that Σ_2 is given by[5]

$$\text{EG: } \begin{cases} \delta_1(k) = [x_2(k) \geq -3], \\ \delta_2(k) = [x_2(k) \geq 3] \end{cases} \qquad \text{MS: } i_2(k) = \begin{cases} 1 & \text{if } \bar{\delta}_1(k) \ \wedge \ \bar{\delta}_2(k), \\ 2 & \text{if } \delta_1(k) \ \wedge \ \bar{\delta}_2(k), \\ 3 & \text{if } \delta_1(k) \ \wedge \ \delta_2(k) \end{cases}$$

$$\text{SAS: } y_2(k) = \begin{cases} x_2(k) - u_2(k) + 12 & \text{if } i_2(k) = 1, \\ u_2(k) - 5 & \text{if } i_2(k) = 2, \\ x_2(k) + u_2(k) - 8 & \text{if } i_2(k) = 3. \end{cases}$$

Let the input-state space of Σ_2 be $\mathcal{R}_2 = \mathcal{U}_r^2 \times \mathcal{X}_r^2 = \{0, 10\} \times \{-10, 10\}$. Remembering that Σ_2 has computational order 1 and using Algorithm 2 leads to the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}_{\mathcal{O}(1)}}$ shown in Fig. 4(a). The PWA dynamics are defined on the polyhedral partition and are omitted due to space limitations. DHA Σ_1 with computational order 2 further divides the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}_{\mathcal{O}(1)}}$ depending on the PWA dynamics of its predecessor Σ_2 as shown in Fig. 4(b).

Compositions With Loops. The algorithm is now generalized to compositions of DHAs $\{\Sigma_i\}_{i=1,\ldots,s}$ containing loops. Having identified the adjacency matrix W and verified that the digraph is not acyclic, we first have to find the feedback arc set.

Definition 6. *Let $G = (V, E)$ be a digraph. A set $F \subseteq E$ is a* feedback arc set *(FAS) for G, if $G' = (V, E - F)$ is acyclic. The set F is a* minimum *FAS if the number of edges in F is minimum.*

[5] In general, the state-update function does not influence the polyhedral partition and is therefore omitted for brevity.

Finding the minimum FAS is NP-hard. However, for a given digraph $G = (V, E)$ a fast and effective heuristic exist [12] yielding an FAS F with upper bounded cardinality $\#F \leq \#E/2 - \#V/6$ and time complexity $O(\#E)$. This algorithm yields a vertex sequence inducing a rearranged adjacency matrix W' and an FAS. Letting the feedback arc be a directed edge from vertex v_i to vertex v_j, every feedback arc $f \in F$ is replaced by a directed edge v_i from a newly created auxiliary input vertex v_u to v_j. The auxiliary input is real if v_i corresponds to a real variable and binary if v_i relates to a binary variable. This yields a composition of DHAs with an acyclic digraph and an augmented exogenous input space $\mathcal{U}_r^e \times \bar{\mathcal{U}}_r^e \times \mathcal{U}_b^e \times \bar{\mathcal{U}}_b^e$, where $\bar{\mathcal{U}}_r^e$ and $\bar{\mathcal{U}}_b^e$ denote the auxiliary real and binary input spaces, respectively. As connections between DHAs are equivalent to equality constraints of the respective inputs and outputs, the removed connections corresponding to the FAS are added as equality constraints to a separate constraint list $\mathcal{C}$.

Using the computational order given by the sequence of indices of W' allows us to use Algorithm 2 to derive the set of feasible modes $\mathcal{J}$, the polyhedral partition $\{\mathcal{P}_j\}_{j\in\mathcal{J}}$ and the corresponding PWA dynamics $\{\mathcal{S}_j\}_{j\in\mathcal{J}}$. Adding the equality constraints in $\mathcal{C}$ removes the auxiliary inputs as well as infeasible modes together with the corresponding polyhedra and PWA dynamics.

For further details and examples on loops in compositions of DHAs, refer to the extended version of this paper [17].

6 Examples and Applications

This final section presents two examples showing how the mode enumeration algorithm can be used to efficiently derive the PWA representation of a given hybrid system and in which way it can be exploited to speed up MPC.

Car Example. In [27], the authors proposed a hybrid model of a car with robotized gear shift. This example was adopted in [4] where the author computes the MLD model using HYSDEL and the PWA system equivalent to the MLD model using multi-parametric programming. As the model is given in HYSDEL, the algorithm in Sect. 5 starts from this description to translate the car example into a PWA model. The resulting PWA model encompasses 30 polyhedra and 6 modes and is computed in 7.5 s in Matlab 5.3 on a Pentium III 650 MHz machine. This is ten times faster than the algorithm reported in [4] on a similar machine.

The reason for this is twofold. First, the algorithm presented here exploits the structure of the DHA models, while the algorithm presented in [4] deals with MLD models concealing that structural information. Second, the approach in [4] needs to remove redundant inequalities at each iteration of the exploration algorithm. This operation may dominate the total computation time in [4].

Paperboy Example. A paperboy delivers by bike two heavy and bulky mail items to two different houses within a neighborhood consisting of four properties and one road. The properties and the road have different slopes and different friction coefficients.

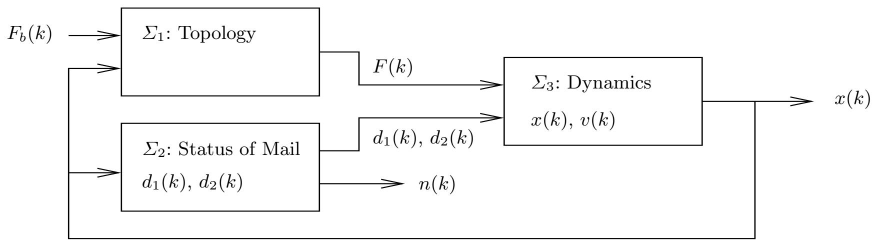


Fig. 5. Paperboy example consisting of three DHAs with the respective states.

The input of the system at time instant k is given by the force $F_b(k) \in \mathcal{U} \subset \mathbb{R}^2$, $\mathcal{U} = \{-F_{\max}, F_{\max}\}^2$, $F_{\max} = 162\,\mathrm{N}$ that the paperboy applies to his bike in order to accelerate and brake. Driven by F_b, the paperboy cycles in the 2-dimensional neighborhood $\mathcal{X} = \{-s_n, s_n\}^2$ with $s_n = 1000\,\mathrm{m}$. His position is given by $x(k) = [x_1(k), x_2(k)]^T \in \mathcal{X} \subset \mathbb{R}^2$ and his speed $v(k) \in \mathcal{V} \subset \mathbb{R}^2$ is limited by $\mathcal{V} = \{-v_{\max}, v_{\max}\}^2$, where $v_{\max} = 15\,\mathrm{m/s} = 54\,\mathrm{km/h}$. Two binary states $d_1(k), d_2(k) \in \{0, 1\}$ denote the status of the mail delivery. The outputs of the model are the position $x(k)$ and the number of delivered mail items $n(k) \in \{0, 1, 2\}$.

As depicted in Fig. 5, the paperboy problem can be decomposed in three DHAs. Each DHA is described in the following.

Topology of Neighborhood. A road of width $w_r = 4\,\mathrm{m}$ divides the neighborhood into two parts which are further partitioned into two properties yielding a total of four properties and one road. These five regions are each characterized by different slopes and friction coefficients. Given the force $F_b(k) \in \mathbb{R}^2$ that the paperboy applies, the effective force $F(k) \in \mathbb{R}^2$ acting on the bike thus depends on the partition $i(k) \in \{1, \dots, 5\}$: $F(k) = F_b(k) - \mu_{i(k)} - \nu_{i(k)} v(k)$, where $\mu_{i(k)}$ is the grade resistance corresponding to the slope and $\nu_{i(k)}$ is a friction coefficient. The parameters are given by $\mu_1 = \left[\begin{smallmatrix}2\\0\end{smallmatrix}\right]$, $\mu_2 = \left[\begin{smallmatrix}1.5\\0\end{smallmatrix}\right]$, $\mu_3 = \left[\begin{smallmatrix}-0.5\\0\end{smallmatrix}\right]$, $\mu_4 = \left[\begin{smallmatrix}-1\\0\end{smallmatrix}\right]$, $\mu_5 = \left[\begin{smallmatrix}0\\0\end{smallmatrix}\right]$ and $\nu_1 = \left[\begin{smallmatrix}2&0\\0&2\end{smallmatrix}\right]$, $\nu_2 = \left[\begin{smallmatrix}0.5&0\\0&0.5\end{smallmatrix}\right]$, $\nu_3 = \left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]$, $\nu_4 = \left[\begin{smallmatrix}1.5&0\\0&1.5\end{smallmatrix}\right]$, $\nu_5 = \left[\begin{smallmatrix}0.05&0\\0&0.05\end{smallmatrix}\right]$.

Therefore, the first DHA is static and has the real inputs $x(k)$, $v(k)$ and $F_b(k)$. The real output is $F(k)$.

$$\mathrm{EG}_1\colon \begin{cases} \delta_{x1}(k) = [x_1(k) \le -0.5 w_r], \\ \delta_{x2}(k) = [x_1(k) \ge 0.5 w_r], \\ \delta_y(k) \;= [x_2(k) \ge 0] \end{cases} \qquad \mathrm{MS}_1\colon i(k) = \begin{cases} 1 & \text{if } \delta_{x1}(k) \;\wedge\; \delta_y(k), \\ 2 & \text{if } \delta_{x2}(k) \;\wedge\; \delta_y(k), \\ 3 & \text{if } \delta_{x1}(k) \;\wedge\; \bar{\delta}_y(k), \\ 4 & \text{if } \delta_{x2}(k) \;\wedge\; \bar{\delta}_y(k), \\ 5 & \text{if } \delta_{x1}(k) \;\wedge\; \bar{\delta}_{x2}(k) \end{cases}$$

$$\mathrm{SAS}_1\colon F(k) = F_b(k) - \mu_{i(k)} - \nu_{i(k)} v(k), \quad i(k) = 1, \dots, 5$$

Status of Mail Delivery. The houses are squares of size $s_h = 10\,\mathrm{m}$ centered at $x_{h1} = [-p_h, -p_h]$ and $x_{h2} = [p_h, p_h]$, $p_h = 40\,\mathrm{m}$. The four walls of a house can be modelled by four hyperplanes with corresponding binary variables equal to 1, if the paperboy is 'within' the respective wall. The respective flag $\delta_{hi}(k)$, $i = 1, 2$ denoting that the paperboy has reached House i and delivered the mail, is the logic *and* of these four binary variables. Finally, a FSM stores the mail delivery status using the binary states $d_1(k)$, $d_2(k)$.

This leads to the following DHA with the real input $x(k)$, the binary states $d_1(k)$ and $d_2(k)$, which are also outputs and a third output $n(k) = d_1(k) + d_2(k)$ denoting the number of delivered mail items.

$$\text{EG}_2\colon \begin{cases} \delta_{h1}(k) = \Big|x(k) - x_{h1}\Big|_\infty \le 0.5 s_h, \\ \delta_{h2}(k) = \Big|x(k) - x_{h2}\Big|_\infty \le 0.5 s_h \end{cases} \quad \text{FSM}_2\colon \begin{cases} d_1(k+1) = d_1(k) \ \vee \ \delta_{h1}(k), \\ d_2(k+1) = d_2(k) \ \vee \ \delta_{h2}(k) \end{cases}$$

Dynamics of Paperboy. The weight $m(k)$ is the weight of the paperboy and his bike ($M_b = 90\,\text{kg}$) plus the weight of the undelivered mail items (each mail item weighs $M_m = 10\,\text{kg}$). Therefore, the total weight is time-dependent and decreasing as the paperboy delivers the mail. By Newton's law, the effective force $F(k)$ divided by the total weight is the acceleration and by integrating this, the velocity and the position of the paperboy are obtained. The integral is approximated by two discrete-time dynamical systems with sampling time $T_s = 1\,\text{s}$.

The third DHA has the real input $F(k) \in \mathbb{R}^2$, the two binary inputs $d_1(k)$, $d_2(k)$ denoting the status of the mail delivery and the real states $x(k)$ and $v(k)$ characterizing the position and the velocity, respectively. The outputs are the position $x(k)$ and the velocity $v(k)$.

$$\text{MS}_3\colon \ i(k) = \begin{cases} 1 & \text{if } \bar{d}_1(k) \wedge \bar{d}_2(k), \\ 2 & \text{if } (d_1(k) \wedge \bar{d}_2(k)) \vee (\bar{d}_1(k) \wedge d_2(k)), \\ 3 & \text{if } d_1(k) \wedge d_2(k) \end{cases}$$

$$\text{SAS}_3\colon \begin{bmatrix} v(k+1) \\ x(k+1) \end{bmatrix} = \begin{bmatrix} v(k) \\ x(k) \end{bmatrix} + \begin{bmatrix} F(k)/m(k) \\ v(k) \end{bmatrix} T_s, \quad \text{where } m(k) = \begin{cases} M_b + 2M_m & \text{if } i(k) = 1, \\ M_b + M_m & \text{if } i(k) = 2, \\ M_b & \text{if } i(k) = 3 \end{cases}$$

The paperboy starts the mail delivery at a random position $x(0)$ with speed $v(0) = 0$. His objective is to first deliver one mail item to House 1 centered around x_{h1} and then to move on to House 2 at position x_{h2} to deliver the second mail item. This can be expressed by the objective function

$$J(t) = \sum_{k=0}^{N-1} \|x(t+k|t) - x_{ref}(t)\|_1 \tag{8}$$

defined over the horizon N using the 1-norm. The reference $x_{ref}(t)$ is switched from x_{h1} to x_{h2} once the paperboy has reached House 1.

In the next section, we will use the paperboy example to evaluate the potential of the mode enumeration algorithm to reduce the computation time of MPC. The MPC control problem amounts to minimizing the objective function (8) subject to the evolution of the paperboy model over the prediction horizon and subject to constraints on F_b, x and v as given above. The solution of this optimization problem which is a *Mixed-Integer Linear Program* (MILP) yields the force F_b.

6.1 Model Predictive Control

When translating a composition of DHAs into an MLD model, information about the structure of the hybrid model is lost. However, the explicit computation of

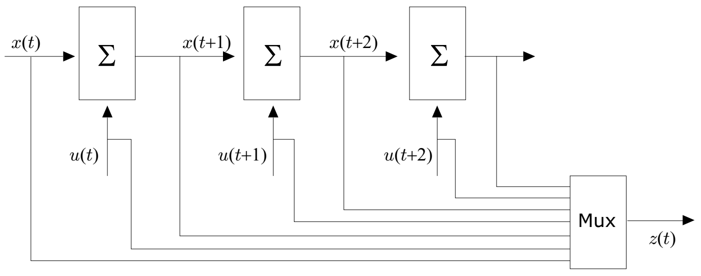


Fig. 6. Conceptual scheme of the 3-steps ahead prediction model.

the set of feasible modes of the composition of DHAs allows one to add structural information and to prune infeasible modes from the resulting system. This is of particular importance for MPC of hybrid models [7], where MPC computes the next N inputs optimizing a performance index which is defined on the variables of a hybrid prediction model. The prediction model (see Fig. 6) is the series-connection of N identical single-step prediction models, where each model uses the state predicted by the previous one as initial state. The mode enumeration allows to introduce cuts on the modes of the complete prediction model.

This information can be formulated in terms of additional logic constraints. According to [23], two methods can be used to transform logic constraints into mixed integer inequalities which can be added to the MLD model. The *Symbolical Method* converts the constraints into a canonical normal form, which is then translated into integer inequalities, whereas the *Geometrical Method* computes the convex hull of the integer points for which the constraints are fulfilled. In general, the second method is superior to the first one because the convex hull is the smallest set containing all the integer feasible points and because it introduces less additional inequalities.

We used HYSDEL to transform the paperboy example of the previous section into an MLD model. The algorithm in Sect. 5 computed the set of feasible modes which were added as additional constraints to the MLD model using the symbolical method as well as the geometrical method. We solved MPC with the objective function (8) and various prediction horizons. Using CPLEX [20] as MILP solver, Fig. 7 reports the average computation times for MPC on the two improved models normalized to the plain model produced by HYSDEL.

Note that both methods add non-trivial cuts reducing the computation time of CPLEX up to a factor of 2. This improvement is more evident when using less advanced solvers like [6], where for a prediction horizon of 3 for example, the additional information reduces the computation time by a factor of 210. Figure 7 shows clearly, that the cuts introduced by the geometrical method are more effective than the ones of the symbolical method. This is mainly due to the fact that the symbolical method needs much more constraints to define the feasible modes. For the paperboy example, the symbolical method introduces 239 additional constraints, whereas the geometrical method only adds 42. The third conclusion is, that both methods become more effective as the prediction horizon is increased, as the benefit of additional cuts grows with the number of binary variables.

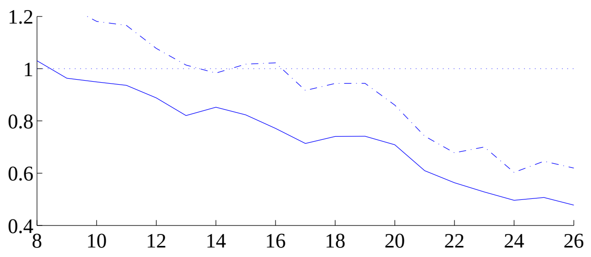


Fig. 7. Normalized average computation time vs. prediction horizon N, $N = 8, \dots, 26$, when using prediction models with additional constraints generated by either the symbolical method (dash-dotted line) or the geometrical method (straight line).

7 Conclusions

We have presented an effective method to enumerate the set of feasible modes for a given composition of DHAs. The same procedure transforms the compound model into a PWA model. The mode enumeration can be exploited in order to reduce the computation time of MPC by adding additional cuts. We have also an optimal algorithm to build an equivalent PWA model minimal in the number of polyhedra of the partition that has been omitted for space limitations. Future research will be devoted to develop fast and suboptimal algorithms.

Acknowledgements

This work was partially supported by the EU project *Control and Computation.* The authors would like to thank Prof. Alberto Bemporad and Prof. Komei Fukuda for inspiring discussions and Mato Baotic and Prof. Komei Fukuda for sharing their tools.

References

1. R. Alur, T. Dang, J. Esposito, R. Fierro, Y. Hur, F. Ivančić, V. Kumar, I. Lee, P. Mishra, G. Pappas, and O. Sokolsky. Hierarchical hybrid modeling of embedded systems. Volume 2211 of *Lecture Notes in Computer Science*, pages 14–31. Springer Verlag, 2001.
2. R. Alur and T. A. Henzinger. Modularity for timed and hybrid systems. Volume 1243 of *Lecture Notes in Computer Science*, pages 74–88, Springer Verlag, 1997.
3. D. Avis and K. Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65:21–46, 1996.
4. A. Bemporad. An efficient technique for translating mixed logical dynamical systems into piecewise affine systems. In *Proc. 41st IEEE Conf. on Decision and Control*, 2002.
5. A. Bemporad, N. Giorgetti, I.V. Kolmanovsky, and D. Hrovat. A hybrid systems approach to modeling and optimal control of disc engines. In *Proc. 41st IEEE Conf. on Decision and Control*, 2002.
6. A. Bemporad and D. Mignone. *MIQP.M: A Matlab function for solving mixed integer quadratic programs.* ETH Zurich, 2000.

7. A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, March 1999.
8. F. Borrelli, A. Bemporad, M. Fodor, and D. Hrovat. A hybrid approach to traction control. Volume 2034 of *Lecture Notes in Computer Science*, pages 162–174. Springer Verlag, 2001.
9. R.C. Buck. Partition of space. *Amer. Math. Monthly*, 50:541–544, 1943.
10. B. De Schutter and T. van den Boom. On model predictive control for max-min-plus-scaling discrete event systems. *Automatica*, 37(7):1049–1056, 2001.
11. N. Deo. *Graph Theory with Applications to Engineering and Computer Science.* Prentice-Hall, Inc., 1974.
12. P. Eades, X. Lin, and W.F. Smyth. A fast and effective heuristic for the feedback arc set problem. *Information Processing Letters*, 47:319–323, 1993.
13. M.G. Earl and R. D'Andrea. Modeling and control of a multi-vehicle system using mixed integer linear programming. In *Proc. 41st IEEE Conf. on Decision and Control*, 2002.
14. H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* 1987.
15. J.A. Ferrez and K. Fukuda. Implementations of LP-based reverse search algorithms for the zonotope construction and the fixed-rank convex quadratic maximization in binary variables using the ZRAM and the cddlib libraries. Technical report, July 2002. `http://www.cs.mcgill.ca/~fukuda/download/mink.`
16. J.A. Ferrez, K. Fukuda, and Th.M. Liebling. Cuts, zonotopes and arrangements. Technical report, EPF Lausanne, November 2001.
17. T. Geyer, F.D. Torrisi, and M. Morari. Efficient Mode Enumeration of Compositional Hybrid Systems. Technical Report AUT03-01, ETH Zurich, 2003.
18. W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, July 2001.
19. T.A. Henzinger, M. Minea, and V. Prabhu. Assume-guarantee reasoning for hierarchical hybrid systems. Volume 2034 of *Lecture Notes in Computer Science*, pages 275–290. Springer Verlag, March 2001.
20. ILOG, Inc. *CPLEX 8.0 User Manual.* Gentilly Cedex, France, 2002.
21. K.H. Johansson. Hybrid systems:modeling, analysis and control – composition of hybrid automata. Lecture notes of the class EECS 291e, Lecture 5, University of California at Berkley, 2000.
22. N. Lynch, R. Segala, and F. Vaandrager. Hybrid I/O automata revisited. Volume 2034 of *Lecture Notes in Computer Science*, pages 403–417. Springer Verlag, 2001.
23. D. Mignone. *Control and Estimation of Hybrid Systems with Mathematical Optimization.* Diss. ETH No. 14520, ETH Zurich, 2002.
24. S. Rashid and J. Lygeros. Hybrid systems:modeling, analysis and control – open hybrid automata and composition. Lecture notes of the class EECS 291e, Lecture 8, University of California at Berkley, 1999.
25. E.D. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Trans. on Aut. Control*, 26(2):346–358, April 1981.
26. E.D. Sontag. Interconnected automata and linear systems: A theoretical framework in discrete-time. Volume 1066 of *Lecture Notes in Computer Science*, pages 436–448. Springer-Verlag, 1996.
27. F.D. Torrisi and A. Bemporad. Discrete-time hybrid modeling and verification. In *Proc. 40th IEEE Conf. on Decision and Control*, pages 2899–2904, 2001.
28. F.D. Torrisi and A. Bemporad. Hysdel — a tool for generating computational hybrid models for analysis and synthesis problems. Technical Report AUT02-03, ETH Zurich, `http://control.ethz.ch/~hybrid/hysdel`, March 2002.
29. A.J. van der Schaft and J.M. Schumacher. Complementarity modelling of hybrid systems. *IEEE Trans. on Aut. Control*, 43:483–490, 1998.
30. G.M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics.* Springer, New York, 1994.