Performance Analysis of Piecewise Linear Systems and Model Predictive Control Systems

A. Bemporad[†‡], F.D. Torrisi[†], M. Morari[†]

[†]Automatic Control Laboratory, Swiss Federal Institute of Technology - bemporad,torrisi,morari@aut.ee.ethz.ch

[‡] Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Siena

Abstract

In their recent paper [2], the authors provided a tool for obtaining the explicit solution of constrained *model predictive control* (MPC) problems by showing that the control law is a continuous *piecewise affine* (PWA) function of the state vector. Therefore, the feedback interconnection between the MPC controller and a linear system, or a PWA system (e.g., a PWA approximation of a nonlinear system), is a PWA system. For discrete-time PWA and hybrid systems, we presented an algorithm for verification/reachability analysis in [5]. In this paper, we formulate the performance analysis problem of closed-loop PWA systems (including MPC feedback loops where the prediction model and the plant model could be different) as a reachability analysis problem, and use our algorithm to obtain a tool for characterizing (i) the set of states for which the evolution is feasible, (ii) the domain of stability, (iii) the performance of the closed-loop.

1 Introduction

Model Predictive Control (MPC) has become the accepted standard for complex constrained multivariable control problems in the process industries. Here at each sampling time, starting at the current state, an open-loop optimal control problem is solved over a finite horizon. At the next time step the computation is repeated starting from the new state and over a shifted horizon, leading to a moving horizon policy. The solution relies on a linear dynamic model, respects all input and output constraints, and optimizes a quadratic performance index. The big drawback of MPC was the relatively formidable on-line computational effort which limited its applicability to relatively slow and/or small problems. For discrete time linear time invariant systems with constraints on inputs and states, in [3] the authors developed an algorithm to determine explicitly the state feedback control law which minimizes a quadratic performance criterion. The control law was shown to be piecewise linear and continuous, thus reducing the on-line computation to a simple linear function evaluation instead of an expensive quadratic program.

Therefore, the feedback connection between a linear model and an MPC controller is a *piecewise affine* (PWA) system, of the form

$$x(t+1) = A_i x(t) + f_i, \text{ for } x(t) \in \mathcal{C}_i \tag{1}$$

where $x \in X \subseteq \mathbb{R}^n$, $\mathcal{C}_i \triangleq \{x : \ H_i x \leq S_i\}, i = 0, \dots, s-1$ is a polyhedral partition of the set of states X, and f_i are constant vectors. Equation (1) can be augmented with the term $B_i d(t)$, where $d \in \mathbb{R}^d$ is a vector of unmeasured disturbances entering the closed-loop. Because of continuity of the piecewise affine control law, the PWA system (1) is well-posed, in the sense that the state-update function is always uniquely defined, despite the fact that the sets $\mathcal{C}_i$ overlap on the boundaries (these will be also referred to as *guardlines*).

PWA systems belong to the class of *switched systems* [7, 16], and constitute an important framework for modeling hybrid systems, as an alternative to the class of *hybrid control systems* [10, 2, 15], which consist of the interaction between continuous dynamical systems and discrete/logic automata. PWA systems are equivalent to interconnections of linear systems and finite automata, as pointed out by Sontag [17]. Based on different arguments, a similar result was proved constructively in [1], where the authors show that PWA systems are equivalent to the hybrid *mixed logical dynamical* (MLD) systems introduced in [2]. MLD systems are hybrid systems defined by the interaction of logic, finite state machines, and linear discrete-time systems, defined by the equations

$$x(t+1) = \mathcal{A}x(t) + \mathcal{B}_1 d(t) + \mathcal{B}_2 \delta(t) + \mathcal{B}_3 z(t) \tag{2a}$$

$$\mathcal{E}_2 \delta(t) + \mathcal{E}_3 z(t) \leq \mathcal{E}_1 d(t) + \mathcal{E}_4 x(t) + \mathcal{E}_5 \tag{2b}$$

where $x \in \mathbb{R}^{n_c} \times \{0,1\}^{n_\ell}$ is a vector of continuous and binary states, $d \in \mathbb{R}^{d_c} \times \{0,1\}^{d_\ell}$ are disturbance inputs, and $\delta \in \{0,1\}^{r_\ell}$, $z \in \mathbb{R}^{r_c}$ represent auxiliary binary and continuous variables respectively, which are introduced when transforming logic relations into mixed-integer linear inequalities [2], and $\mathcal{A}$, $\mathcal{B}_{1-3}$, $\mathcal{E}_{1-5}$ are matrices of suitable dimensions.

MLD systems are capable of modeling a broad class of systems arising in many applications: linear hybrid dynamical systems, hybrid automata, nonlinear dynamic systems where the nonlinearity can be approximated by a piecewise linear function, some classes of discrete event systems, linear systems with constraints, etc. Examples of real-world applications that can be naturally modeled within the MLD framework are reported in [2].

As pointed out in [14], one important reason to study hybrid systems is to analyze stability, robust stability, and tracking properties of high-performance controllers, e.g., MPC controllers. In fact, most currently available MPC techniques guarantee stability for the *nominal* linear plant through the introduction of stability constraints, which are often removed in practical MPC schemes as they typically deteriorate performance. Moreover, an important issue is to analyze the behavior of the feedback loop when the nominal model and the *actual* plant model differ, e.g. because of the presence of nonlinearities. Robust MPC techniques partially solve this issue, by taking into account a class of linear uncertain models rather than one single prediction model, although this typically requires increased computation effort and, again, leads to deterioration of performance.

Clearly, simulation provides an answer, but this is limited to a *particular* model, initial condition, and disturbance realization. In many situations it is important to know if the specifications are met for a whole *set* of conditions. An approximate answer can be given by gridding the set of initial conditions and input signals, and by running a large number of simulations, although (i) some critical evolution might be overlooked, and (ii) the less the desired coarseness of results, the more the simulation effort.

Despite the fact that PWA systems are just a simple extension of linear systems, they can exhibit very complex behaviors as typical of nonlinear systems. Blondel and Tsitsiklis [6] showed that even in the simple case of two component subsystems, verifying the stability of autonomous discrete-time PWA systems is either an $\mathcal{NP}$-hard problem (no polynomial-time algorithm), or undecidable (the problem is not algorithmically solvable, in general). In view of these complexity results, no hope remains of finding criteria for stability of PWA systems as easy as for instance the Routh-Hurwitz rule for linear systems. Stability of each linear subsystem is not enough to guarantee stability of the overall system (and vice versa) [7], as the switching rule between linear dynamics is fundamental for the stability of the interconnection. Some criteria for stability of PWA systems were recently proposed, which are based on multiple Lyapunov functions methods [7]. However, LMI based approaches have the drawback of being conservative, the more conservative the larger the number of regions in the polyhedral partition of the state-space.

Following earlier results in [5], in this paper we formulate as a verification problem the issue of characterizing the stability of a feedback system composed of a linear (or PWA) system and a constrained MPC controller, whose explicit solution can be found in PWA form [3].

The problem of verification can be simply stated as follows: For a given set of initial conditions and disturbances, certify that all possible trajectories never enter a set of unsafe states, or possibly provide a counterexample. As for stability analysis, such a reachability analysis issue is well known to be undecidable in the context of formal verification of hybrid automata [10]. In spite of this complexity, several tools for formal verification of hybrid systems have been proposed in the literature, mainly for linear hybrid automata [10, 13].

The basic idea of this paper is to check for reachability from a bounded set $\mathcal{X}(0)$ of initial conditions to (i) a set around the origin, and (ii) the set $\mathcal{X}_u$ of states where the constraints are violated. More precisely, we label as *asymptotically stable in T steps* the trajectories that enter an invariant set around the origin within a finite time T, or as *infeasible in T steps* the trajectories which enter $\mathcal{X}_u$ within that time. Subsets of $\mathcal{X}(0)$ leading to neither of the two previous cases are called *non-classifiable in T steps*. Such a finite-time verification problem is decidable, as in the case for many other undecidable problems that can be meaningfully approximated by decidable ones (e.g., the decidable algorithm shown in [1] for analysis of observability is another example of such a philosophy).

The approach followed in this paper is related to the idea of *robust simulation* [11], which consists of simulating entire set evolutions rather than single trajectories for stability and performance analysis. In [11] the author tests for finite time stability by computing an outer approximation of the reach set via mathematical programming. In particular, an outer approximation is performed at each time step in order to keep the complexity polynomial. In this paper, we present a robust simulation algorithm that, at the expense of extra computation, provides the exact simulation. Although the worst case complexity is still exponential in the time horizon and the number of guardlines, thanks to a set of heuristics, which exploits the piecewise linear nature of the hybrid system, the performance of the algorithm is comparable to the one of [11], and does not suffer for high state-space dimensions.

2 Model Predictive Control

Consider the problem of regulating to the origin the discrete-time linear time invariant system

$$\begin{cases} x(t+1) &= Ax(t)+Bu(t) \\ y(t) &= Cx(t) \end{cases} \tag{3}$$

while fulfilling the constraints

$$x_{\min} \le x(t) \le x_{\max},\ u_{\min} \le u(t) \le u_{\max} \tag{4}$$

at all time instants $t \ge 0$. In (3)–(4), $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, and $y(t) \in \mathbb{R}^p$ are the state, input, and output vector respectively, $x_{\min} \le x_{\max}$ ($u_{\min} \le u_{\max}$) are finite $n(m)$-dimensional vectors, and the pair (A, B) is stabilizable.

Model Predictive Control (MPC) solves such a constrained regulation problem in the following way. Assume that a full measurement of the state $x(t)$ is available at the current time t. Then, the optimization problem

$$\begin{array}{ll} \min\limits_{U_t^{t+N_u-1}} & J(U,x(t)) = \|x_{t+N_y|t}\|_P^2 + \\ & \displaystyle\sum_{k=0}^{N_y-1} \|x_{t+k|t}\|_Q^2 + \|u_{t+k}\|_R^2 \\ \text{subj. to} & x_{\min} \le x_{t+k|t} \le x_{\max},\ k=0,\dots,N_c \\ & u_{\min} \le u_{t+k} \le u_{\max},\ k=0,1,\dots,N_c \\ & x_{t|t} = x(t) \\ & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k},\ k \ge 0 \\ & u_{t+k} = Kx_{t+k|t},\ N_u \le k \le N_y \end{array} \tag{5}$$

is solved at each time t, where $x_{t+k|t}$ denotes the predicted state vector at time $t+k$, obtained by applying the input sequence $U_t^{t+N_u-1} \triangleq u_t, \dots, u_{t+k-1}$ to model (3) starting from the current state $x(t)$ measured at time t. The state constraints in (5) are defined also for $k = 0$. Although such a constraint is not affected by U, it puts a limitation on the set of states for which (5) has a solution, namely, states which are infeasible for (4) are also infeasible for (5). In (5), we assume that $Q = Q' \succeq 0$, $R = R' \succ 0$, $P \succeq 0$, $(Q^{\frac{1}{2}}, A)$ detectable (for instance $Q = C'C$ with (C, A) detectable),

$N_y \geq N_u \geq N_u$, and K is a linear gain. Frequently, P and K are obtained by solving the Riccati equation with weights Q, R, which amounts to switching the control to the unconstrained LQR after N_u time-steps.

Let $U^*(t) = \{u_t^*, \ldots, u_{t+N_u-1}^*\}$ be the optimal solution of (5). Then at time t

$$u(t) = u_t^* \tag{6}$$

is applied as input to system (3). The optimization (5) is repeated at time $t+1$, based on the new state $x(t+1)$, yielding a *moving* or *receding horizon* control strategy.

The stability of MPC feedback loops was investigated by numerous researchers. Stability is, in general, a complex function of the various tuning parameters N_u, N_y, N_c, Q, R, P, and K. For applications it is most useful to impose some conditions on N_y, N_c, P, and K so that stability is guaranteed for all $Q \succeq 0$, $R \succ 0$. Then Q and R can be freely chosen as tuning parameters to affect performance. Sometimes the optimization problem (5) is augmented with a so called "stability constraint". This additional constraint imposed over the prediction horizon explicitly forces the state vector either to shrink in some norm or to reach an invariant set at the end of the prediction horizon.

Most approaches for proving stability follow in spirit the arguments of Keerthi and Gilbert [12] who establish the fact that under some conditions the value function $V(t) = J(U^*(t), t)$ attained at the minimizer $U^*(t)$ is a Lyapunov function of the system [3].

2.1 MPC Computation

By substituting

$$x_{t+k|t} = A^k x(t) + \sum_{j=0}^{k-1} A^j B u_{k-1-j} \tag{7}$$

in (5), the performance index $J(U, x(t))$ can be rewritten in the form

$$\begin{aligned} \min_U \quad & \tfrac{1}{2} U' \Psi U + x'(t) F U \\ \text{subj. to} \quad & GU \leq W + Lx(t) \end{aligned} \tag{8}$$

where the column vector $U \triangleq [u_t', \ldots, u_{t+N_u-1}']' \in \mathbb{R}^s$, $s \triangleq mN_u$, is the optimization vector, $\Psi = \Psi' \succ 0$, and Ψ, F, Y, G, W, L are easily obtained from Q, R, and (5)–(7). As only the optimizer U is needed, the term involving Y is usually removed from (8).

The optimization problem (8) is a quadratic program (QP). Because the problem depends on the current state $x(t)$, the implementation of MPC requires the on-line solution of a QP at each time step. Although efficient QP solvers based on active-set methods or interior point methods are available, computing the input $u(t)$ demands significant on-line computation effort. For this reason, the application of MPC has been limited to "slow" and/or "small" processes.

In [3] the authors presented a new approach to implement MPC, where all the computation effort is moved off-line. The idea is based on the observation that in (8) the state $x(t) \in \mathbb{R}^n$ can be considered a vector of parameters, and (8) as a multi-parametric Quadratic Program (mp-QP). An algorithm to solve mp-QP problems was presented in [3]. Once the multi-parametric problem (8) has been solved off line, i.e., the solution $U_t^* = f(x(t))$ of (8) has been found, the model predictive controller (5) is available explicitly, as the optimal input $u(t)$ consists simply of the first m components of U_t^*, $u(t) = [I\ 0\ \ldots\ 0] f(x(t))$. In [3] the authors also show that the solution $U^* = f(x)$ of the mp-QP problem is continuous and piecewise affine. Clearly, because of (9), the same properties are inherited by the controller, i.e.,

$$\begin{aligned} u(t) = \quad & F_i x(t) + g_i, \text{ for} \\ & x(t) \in \mathcal{C}_i \triangleq \{x :\ H_i x \leq S_i\},\ i = 1, \ldots, s \end{aligned} \tag{9}$$

where $\cup_{i=1}^s \mathcal{C}_i$ is the set of states for which a feasible solution to (5) exists. Therefore, the closed MPC loop is of the form (1), where $A_i = A + BF_i$, $f_i = Bg_i$, $B_i = 0$ (A vector of polyhedrally-bounded additive disturbances $d(t)$ can be taken into account by considering nonzero matrices B_i). Note that the form of the closed-loop MPC system remains PWA also when (i) the matrices A, B of the plant model are different from those used in the prediction model, and (ii) the plant model has a PWA form. Typically, the MPC law (5) is designed on a linear model obtained by linearizing the nonlinear model of the plant around some operating condition. When the nonlinear model can be approximated by a PWA system (e.g., through multiple linearizations at different operating points or by approximating nonlinear static mappings into piecewise linear functions), the closed-loop formed by the nonlinear plant model and the MPC controller (5) can be approximated by a PWA system as well.

3 Performance Characterization Problem

As mentioned in the introduction, determining the stability of PWA systems can be a complex task. Nevertheless, we aim at estimating the domain of attraction of the origin, and the set of initial conditions from which the state trajectory remains feasible for the constraints (4).

As mentioned in the previous section, the nominal MPC closed-loop is an autonomous PWA system. The origin belongs to the interior of one of the sets of the partition, namely the region where the LQ gain K is asymptotically stabilizing while fulfilling the constraints (4), which by convention will be referred to as $\mathcal{C}_0$. Denote by $\mathcal{D}_\infty(0) \subseteq \mathbb{R}^n$ the (unknown) domain of attraction of the origin. Given a bounded set $\mathcal{X}(0)$ of initial conditions, we want to characterize $\mathcal{D}_\infty(0) \bigcap \mathcal{X}(0)$.

By construction, matrix A_0, associated with the region $\mathcal{C}_0$, is strictly Hurwitz and $f_0 = 0$ (in fact, in $\mathcal{C}_0$ the feedback gain is the unconstrained LQR gain $F_0 = K$, $g_0 = 0$ [3]). Then we can compute an invariant set in $\mathcal{C}_0$. In particular, we compute the *maximum output admissible set* (MOAS) $\mathcal{X}_\infty \subseteq \mathcal{C}_0$. $\mathcal{X}_\infty$ is the largest invariant set contained in $\mathcal{C}_0$, which by construction of $\mathcal{C}_0$ is compatible with the constraints $u_{\min} \leq Kx(t) \leq u_{\max}$, $x_{\min} \leq x(t) \leq x_{\max}$. By [9, Th.4.1], MOAS is a polyhedron with a finite number of facets, and is computed through a finite number of linear programs (LP's) [9][1].

[1] If the effect of perturbations $d(t) \in \mathcal{U} \subseteq \mathbb{R}^m$, where $\mathcal{U}$ is a given bounded set of disturbances and $B_0 \neq 0$, has to be taken into account $\mathcal{X}_\infty$ is the largest invariant set under disturbance excitation, and can be computed as proposed in [8].

In order to circumvent the undecidability of stability mentioned above, we give the following

Definition 3.1 *Consider the PWA system (1), and let the origin* $0 \in \overset{\circ}{\mathcal{C}}_0 \triangleq \{x : H_0 x < S_0\}$, *and* A_0 *be strictly Hurwitz. Let* $\mathcal{X}_\infty$ *be the maximum output admissible set (MOAS) in* $\mathcal{C}_0$, *which is an invariant for the linear system* $x(t+1) = A_0 x(t)$. *Let* T *be a finite time horizon. Then, the set* $\mathcal{X}(0) \subseteq \mathbb{R}^n$ *of initial conditions is said to belong to the* domain of attraction in T steps $\mathcal{D}_T(0)$ *of the origin if* $\forall x(0) \in \mathcal{X}(0)$ *the corresponding final state* $x(T) \in \mathcal{X}_\infty$.

Note that $\mathcal{D}_T(0) \subseteq \mathcal{D}_{T+1}(0) \subseteq \mathcal{D}_\infty(0)$, and $\mathcal{D}_T(0) \to \mathcal{D}_\infty(0)$ as $T \to \infty$. The horizon T is a practical information about the speed of convergence of the PWA system to the origin.

Definition 3.2 *Consider the PWA system (1), and let* $\mathcal{X}_{\text{infeas}} \triangleq \mathbb{R}^n \setminus \cup_{i=1}^s \mathcal{C}_i$. *The set* $\mathcal{X}(0) \subseteq \mathbb{R}^n$ *of initial conditions is said to belong to the* domain of infeasibility in T steps $\mathcal{I}_T(0)$ *if* $\forall x(0) \in \mathcal{X}(0)$ *there exists* t, $0 \le t \le T$ *such that* $x(t) \in \mathcal{X}_{\text{infeas}}$.

Given a set of initial conditions $\mathcal{X}(0)$, we aim at finding subsets of $\mathcal{X}(0)$ which are safely asymptotically stable ($\mathcal{X}(0) \bigcap \mathcal{D}_T(0)$), and subsets which lead to infeasibility in T steps ($\mathcal{X}(0) \bigcap \mathcal{I}_T(0)$). Subsets of $\mathcal{X}(0)$ leading to none of the two previous cases are labeled as *non-classifiable in T steps* As we will use linear optimization tools, we assume that $\mathcal{X}(0)$ is a convex polyhedral set (or the union of convex polyhedral sets). Typically, non-classifiable subsets shrink and eventually disappear for increasing T.

3.1 Switching Sequences

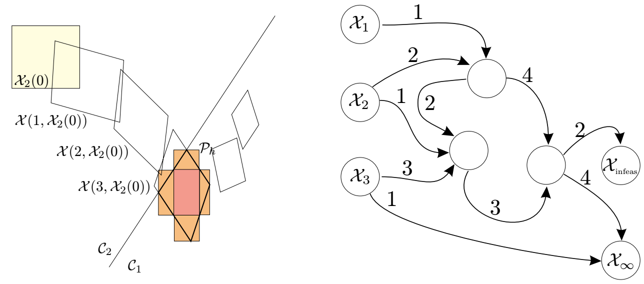
Consider the following simple case of evolution of the PWA system (1), where $u(t) = 0$, $f_i = 0$, $\forall i = 0, \dots, s-1$,

$$x(t) = A_{i(t-1)} A_{i(t-2)} \cdots A_{i(0)} x(0) \tag{10}$$

where in (10) $i(k) \in \{0, \dots, s-1\}$ is the index such that $H_{i(k)} x(k) \le S_{i(k)}$, $k = 0, \dots, t-1$, is satisfied. The previous questions of practical stability can be answered once all switching sequences $I(t) \triangleq \{i(0), \dots, i(t-1)\}$ leading to $\mathcal{X}_\infty$ or $\mathcal{X}_{\text{infeas}}$ from $\mathcal{X}(0)$ are known. In fact, for safe stability in T steps it is enough to check that the reach set at time T, $\mathcal{X}(T, \mathcal{X}(0)) \triangleq A_{i(T-1)} A_{i(T-2)} \cdots A_{i(0)} \mathcal{X}(0)$, satisfies the set inclusion $\mathcal{X}(T, \mathcal{X}(0)) \subseteq \mathcal{X}_\infty$ for all admissible switching sequences $I(T)$. However, the number of all possible switching sequences $I(T)$ is combinatorial with respect to T and s, and any enumeration method would be impractical. In the next section we show that a *verification* algorithm can be used to avoid such an enumeration.

4 Reachability Analysis of Hybrid Systems

In this section, we recall the verification algorithm presented in [5]. In order to determine admissible switching sequences $I(t)$, the algorithm exploits the special structure of the PWA system (1). This structure allows an easy computation of the reach set as long as the evolution remains within a single region $\mathcal{C}_i$. Whenever the reach set crosses a guardline and enters a new region $\mathcal{C}_j$, a new reach set computation based on the j-th linear dynamics is computed, as shown in Fig. 1(a).



(a) Reach set evolution, guardline crossing, outer approximation of a new intersection

(b) Graph of evolution G

Figure 1: Reachability Analisys

Let $\mathcal{X}(0)$ be a convex polyhedral set, and partition it into subregions $\mathcal{X}_i(0) \triangleq \mathcal{X}(0) \bigcap \mathcal{C}_i$, $i = 0, \dots, s-1$. For all nonempty sets $\mathcal{X}_i(0)$, computing the evolution $\mathcal{X}(T, \mathcal{X}_i(0))$ requires: (*i*) the reach set $\mathcal{X}(t, \mathcal{X}_i(0), \mathcal{C}_i)$, i.e., the set of evolutions at time t in $\mathcal{C}_i$ from $\mathcal{X}_i(0)$; (*ii*) crossing detection of the guardlines, $\mathcal{P}_h \triangleq \mathcal{X}(t, \mathcal{X}_i(0), \mathcal{C}_i) \bigcap \mathcal{C}_h \ne \emptyset$, $\forall h = 0, \dots, i-1, i+1, \dots, s-1$; (*iii*) elimination of redundant constraints and approximation of the polyhedral representation of the new regions $\mathcal{P}_h$ (approximation is desirable, as the number of facets of $\mathcal{P}_h$ can grow linearly with time); (*iv*) detection (1) of emptiness of $\mathcal{X}(t, \mathcal{P}_h, \mathcal{C}_i)$ (emptiness happens when all the evolutions have crossed the guardlines), (2) of safe stability $\mathcal{X}(t, \mathcal{P}_h, \mathcal{C}_i) \subseteq \mathcal{X}_\infty$, (3) of full infeasibility $\mathcal{X}(t, \mathcal{P}_h, \mathcal{C}_i) \subseteq \mathcal{X}_{\text{infeas}}$ (these three will be referred to as *fathoming* conditions).

4.1 Reach Set Computation

Let the set of initial conditions be defined by the polyhedral representation $\mathcal{X}(0) \triangleq \{x : S_0 x \le T_0\}$. The subset $S_i(t, \mathcal{X}(0))$ of $\mathcal{X}(0)$ whose evolution lies in $\mathcal{C}_i$ for t steps is given by

$$\begin{aligned} S_i(t, \mathcal{X}(0)) = \{x \in \mathbb{R}^n :\ & S_0 x \le T_0, \\ H_i A_i^k x \le S_i - H_i \textstyle\sum_{j=0}^{k-1} A_i^j f_i,\ & k = 0, \dots, t\} \end{aligned} \tag{11}$$

As $S_i(t, \mathcal{X}(0))$ is a polyhedral set, the reach set $\mathcal{X}(t, \mathcal{X}_i(0), \mathcal{C}_i)$ is a polyhedral set as well. In the presence of input disturbances, $S_i(t, \mathcal{X}(0)) = \{x \in \mathbb{R}^n : S_0 x \le T_0, H_i(A_i^k x + \sum_{j=0}^{k-1} A_i^j [B_i d(k-1-j) + f_i]) \le S_i, k = 0, \dots, t\}$, is a polyhedron in the augmented space of tuples $(x, d(0), \dots, d(t-1))$.

4.2 Guardline Crossing Detection

Switching detection amounts to finding all possible new regions $\mathcal{C}_h$'s entered by the reach set at the next time step, i.e., nonempty sets $\mathcal{P}_h \triangleq \mathcal{X}(t, \mathcal{X}_i(0), \mathcal{C}_i) \bigcap \mathcal{C}_h$, $h \ne i$. Rather than enumerating and checking nonemptiness for all $h = 0, \dots, i-1, i+1, \dots, s-1$, we can exploit the equivalence between PWA systems and MLD models (2), and solve the switching detection problem via mixed-integer linear programming. More in detail, in the MLD form the condition $x(t) \in \mathcal{C}_h$ is associated with the condition $\delta(t) = \delta_h \in \{0,1\}^{r_\ell}$, for instance $x(t) \in \mathcal{C}_5 \Leftrightarrow \delta(t) = [1\ 0\ 1]'$. Switching detection amounts to finding all feasible vectors $\delta(t) \in \{0,1\}^{r_\ell}$ which are compatible with the constraints in (2) plus the constraint $x(t-1) \in \mathcal{X}(t-1, \mathcal{X}_i(0), \mathcal{C}_i)$. Such a problem is a mixed-integer linear feasibility test (MILFT), and can be efficiently solved through standard recursive branch and bound procedures. Thus, on average the MLD form (through the branch and bound algorithm) requires only a very small number of feasibility tests, while the PWA form would require enumerating and solving a feasibility test for all the possible s regions.

4.3 Approximation of Intersections

The computation of the reach set proceeds in each region $\mathcal{C}_h$ from each new intersection $\mathcal{P}_h$. A new reach set computation is started from $\mathcal{P}_h$, unless $\mathcal{P}_h$ is contained in some larger subset of $\mathcal{C}_h$ which has already been explored. As the number of facets of $\mathcal{P}_h$ can grow linearly with time, we need to approximate $\mathcal{P}_h$ so that its complexity is bounded (and therefore the computation of the reach set from $\mathcal{P}_h$ has a limited complexity with respect to the initial region), and checking for set inclusion is a simple task. Hyper-rectangular approximations are the best candidates, as set inclusion between hyper-rectangles reduces to a simple comparison of the coordinates of the vertices. On the other hand, a crude rectangular outer approximation of $\mathcal{P}_h$ can lead to explore large regions which are not reachable from the initial set $\mathcal{X}(0)$, as they are just introduced by the approximation itself. In [4] the authors propose an iterative method for inner and outer approximation which is based on linear programming, and approximates with arbitrary precision polytopes by a collection of hyper-rectangles, as depicted in Fig. 1(a).

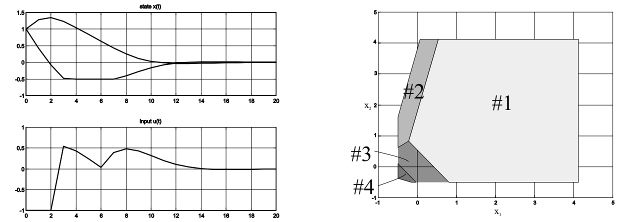
4.4 Fathoming

In Sect. 4.1 we showed how to compute the evolution of the reach set $\mathcal{X}(t, \mathcal{P}_h, \mathcal{C}_i)$ inside a region $\mathcal{C}_i$. The computation is stopped once one of the following happens: (i) The set $\mathcal{X}(t, \mathcal{P}_h, \mathcal{C}_i)$ is empty. This means that the whole evolution has left region $\mathcal{C}_i$,(ii) $\mathcal{X}(t, \mathcal{P}_h, \mathcal{C}_i) \subseteq \mathcal{X}_\infty$, i.e., all possible evolutions from $\mathcal{P}_h$ are safely stable, (iii) $\mathcal{X}(t, \mathcal{P}_h, \mathcal{C}_i) \subseteq \mathcal{X}_{\text{infeas}}$, i.e., all possible evolutions from $\mathcal{P}_h$ have violated the constraints in (4). (iv) time $t > T$. These conditions can be checked through linear programming.

4.5 Graph of Evolution

The result of the exploration algorithm detailed in the previous sections can be conveniently represented on a graph G (Fig. 1(b)). The nodes of G represent sets from which a reach set evolution is computed, and an oriented arc of G connects two nodes if a transition exists between the two correspoding sets. Each arc has an associated weight which represents the time-steps needed for the transition. The graph has initially no arc, and nonempty initial sets $\mathcal{X}_i(0)$ and $\mathcal{X}_\infty$, $\mathcal{X}_{\text{infeas}}$ as nodes. When a new intersection $\mathcal{X}(t, \mathcal{X}_i(0), \mathcal{C}_i) \bigcap \mathcal{C}_h$ is detected, it is approximated by a collection of hyper-rectangles, as described in Sect. 4.3. Each hyper-rectangle becomes a new node in G, and is connected by a weighted arc from $\mathcal{X}_i(0)$.

After the verification algorithm terminates, the oriented paths on G from initial nodes $\mathcal{X}_i(0)$ to terminal nodes $\mathcal{X}_\infty$ and $\mathcal{X}_{\text{infeas}}$ determine a superset of feasible switching sequences $I(t) = \{i(0), \ldots, i(t-1)\}$. In fact, because of the outer approximation of new intersections $\mathcal{P}_h$, not all switching sequences are feasible. Nevertheless, feasibility can be simply tested via linear programming. Once all feasible switching sequences $I(t)$ have been identified, the partition of the initial set into safely stable regions and regions where infeasibility occurs is determined by the sets $A_{i(t-1)}A_{i(t-2)} \ldots A_{i(0)}\mathcal{X}(0) \oplus \left\{ f_{i(t-1)} + \sum_{j=1}^{t-1} \left(\prod_{h=j}^{t-1} A_{i(h)} \right) f_{i(j-1)} \right\}$, $t \leq T$.



(a) Closed-loop MPC (b) Explicit solution

Figure 2: Example (12)

5 An Example

Consider the system $y(t) = \frac{s+1}{s^2+s+2}u(t)$, and sample the dynamics with $T = 0.2$ s. The task is to regulate the system to the origin while fulfilling the constraints $-1 \leq u(t) \leq 1$ and $x(t) \geq \left[\begin{smallmatrix} -0.5 \\ -0.5 \end{smallmatrix}\right]$. To this aim, we design an MPC controller based on the optimization problem

$$\begin{aligned} \min_{u_t, u_{t+1}} \quad & ||x_{t+2|t}||_P^2 + \sum_{k=0}^{1} ||x_{t+k|t}||^2 + .1||u_{t+k}||^2 \\ \text{subj. to} \quad & -2 \leq u_{t+k} \leq 2,\ k = 0, 1 \\ & x_{t+k|t} \geq x_{\min},\ x_{\min} \triangleq \left[\begin{smallmatrix} -0.5 \\ -0.5 \end{smallmatrix}\right],\ k = 0, 1 \end{aligned} \tag{12}$$

where P is the solution to the Riccati equation (in this example $Q = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$, $R = 0.1$, $N_u = N_y = N_c = 2$). Note that this choice of P corresponds to setting $u_{t+k} = Kx_{t+k|t}$ for $k \geq 2$, where K is the LQR gain, and minimizes $\sum_{k=0}^{\infty} x'_{t+k|t}x_{t+k|t} + .01u_{t+k}^2$ with respect to u_t, u_{t+1}. The closed loop response from the initial condition $x(0) = [1\ 1]'$ is shown in Fig. 2(a).

The mp-QP problem associated with the MPC law has the form (8) with

$$\Psi = \left[\begin{smallmatrix} 0.7616 & 0.3059 \\ 0.3059 & 0.6075 \end{smallmatrix}\right],\ F = \left[\begin{smallmatrix} 2.2950 & 1.5010 \\ 1.0835 & 0.5171 \end{smallmatrix}\right]$$

$$G = \begin{bmatrix} -0.1789 & 0 \\ -0.0372 & 0 \\ 1.0000 & 0 \\ -1.0000 & 0 \\ 0 & 1.0000 \\ 0 & -1.0000 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},\ W = \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0.5 \\ 0.5 \end{bmatrix},\ L = \begin{bmatrix} 0.7839 & -0.1789 \\ 0.3577 & 0.9628 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1.0000 & 0 \\ 0 & 1.0000 \end{bmatrix}$$

The solution was computed by using the mp-QP solver in [3] in 0.66 s on a PC Pentium III 650 MHz running Matlab 5.3, and the corresponding polyhedral partition of the state-space is depicted in Fig. 2(b). The MPC law is

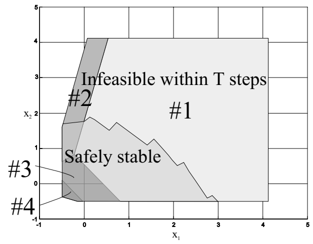


Figure 3: Partition of initial states into safely stable, and infeasible in $T = 20$ steps.

$$u = \begin{cases} -1.0000 & \text{if } \begin{bmatrix} 0.2425 & 0.0000 \\ 0.0000 & 0.2425 \\ -2.5336 & -1.3548 \\ -2.4411 & 0.5570 \\ 0.0000 & -2.0000 \end{bmatrix} x \leq \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} \quad \text{(Region \#1)} \\ \begin{bmatrix} -4.3828 & 1.0000 \end{bmatrix} x - 2.7954 & \text{if } \begin{bmatrix} 0.0000 & 0.2425 \\ -2.0000 & 0.0000 \\ 0.6615 & -0.8424 \\ -1.1548 & 0.2635 \\ 2.4411 & -0.5570 \end{bmatrix} x \leq \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#2)} \\ \begin{bmatrix} -2.5336 & -1.3548 \end{bmatrix} x & \text{if } \begin{bmatrix} -0.6615 & 0.8424 \\ -2.5336 & -1.3548 \\ 2.5336 & 1.3548 \\ -2.0000 & 0.0000 \\ 0.0000 & -2.0000 \end{bmatrix} x \leq \begin{bmatrix} 1.0000 \\ 1.0000 \\ 1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} \quad \text{(Region \#3)} \\ 1.0000 & \text{if } \begin{bmatrix} 0.0000 & -2.0000 \\ 2.5336 & 1.3548 \\ -0.6659 & -1.7922 \\ -2.0000 & 0.0000 \end{bmatrix} x \leq \begin{bmatrix} 1.0000 \\ -1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} \quad \text{(Region \#4)} \end{cases}$$

where region #3 corresponds to the unconstrained LQR controller, #1 and #4 to saturation at -1 and $+1$, respectively, and #2 is a transition region between LQR control and the saturation.

Note that the union of the regions depicted in Fig. 2(b) should not be confused with the region of attraction of the MPC closed-loop. For instance, by starting at $x(0) = [3.5\ 0]'$ (for which a feasible solution exists), the MPC controller runs into infeasibility after $t = 5$ time steps.

The reachability analysis algorithm described above was applied to determine the set of safely stable initial states and states which are infeasible in $T = 20$ steps (Fig. 3). The algorithm computes the graph of evolutions in 115 s on a Pentium II 400 running Matlab 5.3.

6 Conclusions and Acknowledgments

In this paper we proposed a technique for performance assessment of MPC closed-loop systems which is based on reachability analysis of hybrid systems. The approach can be immediately extended to set-point tracking problems and disturbance rejection, where parametric analysis with respect to set-point/disturbance values can be performed in order to determine the set of initial states which leads to safe evolutions for a given set-point/disturbance, or vice versa all the set-points/disturbance which can be safely commanded from a given set of initial states. The approach also allows the robust analysis of safe stability against norm-bounded disturbances.

This research has been supported by the Swiss National Science Foundation and Esprit Project 26270 VHS (verification of Hybrid Systems).

References

[1] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Trans. Automatic Control*, to appear. `http://control.ethz.ch/`.

[2] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, March 1999.

[3] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. In *American Control Conference*, Chicago, IL, June 2000. pdf available at `http://control.ethz.ch/~bemporad/dsi/mp_frame.html`.

[4] A. Bemporad and F.D. Torrisi. Inner and outer approximation of polytopes using hyper-rectangles. Technical Report AUT00-02, Automatic Control Lab, ETH Zurich, 2000.

[5] A. Bemporad, F.D. Torrisi, and M. Morari. Optimization-based verification and stability characterization of piecewise affine and hybrid systems. In B. Krogh and N. Lynch, editors, *Hybrid Systems: Computation and Control*, volume 1790 of *Lecture Notes in Computer Science*, pages 45–58. Springer Verlag, 2000.

[6] V.D. Blondel and J.N. Tsitsiklis. Complexity of stability and controllability of elementary hybrid systems. *Automatica*, 35:479–489, March 1999.

[7] M.S. Branicky. Multiple Lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Trans. Automatic Control*, 43(4):475–482, April 1998.

[8] E.G. Gilbert and I. Kolmanovsky. Maximal output admissible sets for discrete-time systems with disturbance inputs. In *Proc. American Contr. Conf.*, pages 2000–2005, 1995.

[9] E.G. Gilbert and K. Tin Tan. Linear systems with state and control constraints: the theory and applications of maximal output admissible sets. *IEEE Trans. Automatic Control*, 36(9):1008–1020, 1991.

[10] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. HyTech: a model checker for hybrid systems. *Software Tools for Technology Transfer*, 1:110–122, 1997.

[11] M. Kantner. Robust stability of piecewise linear discrete time systems. In *Proc. American Contr. Conf.*, pages 1241–1245, Evanston, IL, USA, 1997.

[12] S.S. Keerthi and E.G. Gilbert. Optimal infinite-horizon feedback control laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations. *J. Opt. Theory and Applications*, 57:265–293, 1988.

[13] S. Kowalewski, O. Stursberg, M. Fritz, H. Graf, I. Hoffmann, J. Preußig, M. Remelhe, S. Simon, and H. Treseler. A case study in tool-aided analysis of discretely controlled continuos systems: the two tanks problem. In *Hybrid Systems V*, volume 1567 of *Lecture Notes in Computer Science*, pages 163–185. Springer-Verlag, 1999.

[14] D. Liberzon and A.S. Morse. Basic problems in stability and design of switched systems. *IEEE Control Systems Magazine*, 19(5):59–70, October 1999.

[15] J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3):349–370, 1999.

[16] E.D. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Trans. Automatic Control*, 26(2):346–358, April 1981.

[17] E.D. Sontag. Interconnected automata and linear systems: A theoretical framework in discrete-time. In R. Alur, T.A. Henzinger, and E.D. Sontag, editors, *Hybrid Systems III - Verification and Control*, number 1066 in Lecture Notes in Computer Science, pages 436–448. Springer-Verlag, 1996.