Robust Hybrid Control from a Behavioural Perspective

T. Moor *, J.M. Davoren †, B.D.O. Anderson*

Abstract

This paper investigates the synthesis of discrete supervisors for hybrid systems where the control objective is to enforce a language inclusion specification in the presence of plant uncertainty. The discussion is set within Willems' behavioural system theory, where we find a relationship between robustness of closed-loop performance and earlier results on abstraction based synthesis. From this relationship, we develop our main result: a method for the synthesis of robust supervisory controllers. Note that virtually any engineering system *must* possess some amount of robustness in order to fulfil even minimal reliability requirements. This commonly accepted fact is of a particular importance for hybrid control systems, since the motivating application areas are safety-critical and high-confidence systems as air traffic control or medical instrumentation.

Keywords: hybrid systems, behavioural systems theory, robust control, supervisory control.

1 Introduction

Hybrid control systems are mathematical models of heterogeneous systems consisting of digital components interacting in real-time with continuous processes. In particular, a variety of controller design problems for such systems has received extensive attention e.g. [2, 4, 7, 8, 10, 11, 14]. In this paper, we investigate the synthesis of a discrete event controller for continuous time and continuous valued control systems; see Fig. 1. The conversion between continuous signals and discrete events is performed in a similar way as it occurs within the widely accepted hybrid automata model [1]. As in [4, 11, 14], we assume that the actuator (D/A-map) and the sensor (A/D-map) are given, and our synthesis problem is the construction of a discrete event controller that enforces a language inclusion specification.

In [11, 13], this synthesis problem is discussed within J.C. Willems' behavioural systems theory, and it is shown that a solution can be obtained in two steps: (i) construct a plant abstraction that can be realised by a finite automaton; (ii) apply slightly modified tools from P.J. Ramagde and W.M. Wonham's DES supervisory control theory [15, 16].

*Research School of Information Sciences and Engineering, Australian National University, Canberra, Australia, email: {thomas.moor | brian.anderson}@anu.edu.au. Research partially supported by the Australian Research Council, Project ID: DP0208553.

†Department of Electrical and Electronic Engineering, University of Melbourne, Australia, email: davoren@unimelb.edu.au

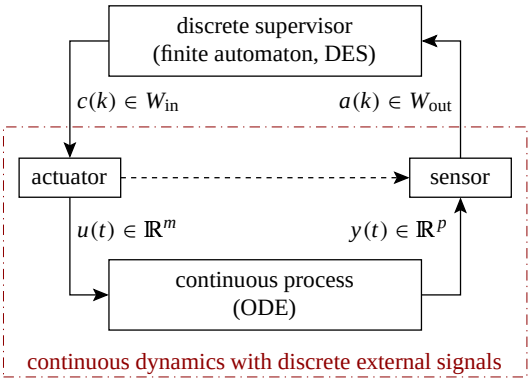

Figure 1: Hybrid control configuration

In this paper, we extend the general methodology of [11, 13] in order to synthesise supervisors that are robust w.r.t. a quantified parameter uncertainty in the hybrid plant model. It is commonly accepted that every engineering system *must* be robust in order to provide a vital level of reliability, and this requirement is addressed by e.g. virtually all classical continuous feedback designs. However, little is known about the robust design of hybrid systems e.g. [10, 5]. In particular, hybrid closed-loop systems that have been designed to fulfil a language inclusion specification by the methods in e.g. [4, 7, 11, 14], in general, fail to possess any robustness margin: even under the smallest perturbations of any plant component, the closed loop may cease to fulfil the performance criteria it has been designed for. Our contribution addresses this problem for a broad class of parameter uncertainties.

The paper is organised as follows. In Section 2, we recall definitions and facts from Willems' behavioural framework, including links to DES theory. A detailed model of the hybrid systems under consideration is given in Section 3. In Section 4, we present an adapted version of the core results from [11, 13]. This allows for an accessible treatment of a general class of robust hybrid control problems in Section 5, where we basically allow all plant components to depend on an uncertain parameter with known range.

2 Behaviours and states machines

For the readers convenience, we collect some basic definitions from Willems' behavioural systems theory; a comprehensive introduction is given in [17, 18].

Definition 2.1. (See [18], Def. II.1) A *dynamical system* is a triple $\Sigma = (T,\ W,\ \mathfrak{B})$, with $T \subseteq \mathbb{R}$ the *time axis*, W the

signal space, and $\mathfrak{B} \subseteq W^T$ the *behaviour*. [1] □

The behaviour is viewed as the set of all trajectories which are compatible with the phenomena modelled by the system: trajectories $w \notin \mathfrak{B}$ cannot occur. In this paper, we focus attention on *discrete time behaviours* with $T = \mathbb{N}_0$ [2]. Note that the discrete time case is *not* restricted to sampling with a constant sampling period (clock time), but also accounts for scenarios in which a discrete time axis is derived from counting events (logic time). In the latter case, a behaviour models a phenomenon very much in the way *formal languages* are used in DES theory; e.g. [3]. This link is further elaborated by the following definition of state machine realisations of discrete time behaviours.

Definition 2.2. A *state machine* is a tuple $P = (X,\ W,\ \delta,\ X_0)$ with W the *external signal space*, X the *state space*, X_0 the *set of initial conditions*, and with $\delta \subseteq X \times W \times X$ the *next state relation*. If $|W| \in \mathbb{N}$ and $|X| \in \mathbb{N}$ (both sets are finite), P is said to be a *finite automaton*. The *external behaviour* $\mathfrak{B}$ induced by P is defined as

$$\mathfrak{B} := \{w \in W^{\mathbb{N}_0} \mid \exists\, x \in X^{\mathbb{N}_0} :\ \forall\, k \in \mathbb{N}_0 : \\ (x(k),\ w(k),\ x(k+1)) \in \delta \text{ and } x(0) \in X_0\}. \quad (1)$$

Conversely, a state machine P' with induced external behaviour $\mathfrak{B}'$ is called a *realisation* of $\mathfrak{B}'$. □

We recall some basic terminology for state machines:

Definition 2.3. Consider state machines $P_a = (A,\ W,\ \alpha,\ A_0)$ and $P_b = (B,\ W,\ \beta,\ B_0)$. A state $a' \in A$ is *reachable* if there exists a state $a \in A_0$ and a sequence of transitions (elements in the next state relation) from α connecting a with a'. The state machine P_a is *reachable* if every state $a' \in A$ is reachable. The state machine P_a is *nonblocking*, if for every reachable state $a \in A$ there exists $\omega \in W$ and $a' \in A$ such that $(a,\ \omega,\ a') \in \alpha$. The *parallel composition* of P_a and P_b is defined by $P_a \parallel P_b := (A \times B,\ W,\ \lambda,\ A_0 \times B_0)$, where $((a,\ b),\ \omega,\ (a',\ b')) \in \lambda$ if and only if $(a,\ \omega,\ a') \in \alpha$ and $(b,\ \omega,\ b') \in \beta$. The state machine P_a is *pastinduced* if $|A_0| = 1$ and if for all reachable $a \in A$ and all $\omega \in W$, $(a, \omega, a') \in \delta$ and $(a, \omega, a'') \in \delta$ implies $a' = a''$. For $W = W_{\text{in}} \times W_{\text{out}}$, the state machine P_a is an *I/S/- machine*, if for every reachable $a \in A$ and every $\mu \in W_{\text{in}}$, there exist $\nu \in W_{\text{out}}$ and $a' \in A$ such that $(a,\ (\mu,\ \nu),\ a') \in \alpha$. □

Reachability, the nonblocking property and the parallel composition are standard definitions from DES theory; e.g. [3]. The pastinduced property is a particular form of determinism, as it requires that at any instance of time the current state is uniquely determined by the past of the external trajectory; see [17] for the corresponding definition in terms of full (state) behaviours. I/S/- machines conform with the traditional notion of inputs and outputs on the product space $W = W_{\text{in}} \times W_{\text{out}}$, as e.g. in Moore automata or discrete-time linear continuous systems. As in these examples, the hybrid systems considered in this paper evolve on a state trajectory that is uniquely defined by the initial condition and the sequence of applied input events. However, our notion of I/S/- machines does *not require* that the input drives the state in unique way. This is utilised by the plant abstractions considered in Section 5, which, in general, for a given state and input pair, fail to produce a unique successor state. Note that this is allowed by pastinducedness, and this is precisely why we need to distinguish pastinducedness form other notions of determinism.

It is easy to see that any discrete time behaviour $\mathfrak{B}$ can be realised as a state machine (e.g. use $X = \mathfrak{B} \times \mathbb{N}_0$, $X_0 = \mathfrak{B} \times \{0\}$, $\delta := \{((w,k),\ w(k),\ (w,k+1)) \mid k \in \mathbb{N}_0,\ w \in \mathfrak{B}\}$ for a constructive proof). There is a natural interest in how properties of state machines relate to properties defined in terms of behaviours, and we shall recall some definitions and facts regarding I/S/- machines and pastinducedness.

Definition 2.4. (See [18], Def. II.4) A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ is *complete* if for all $w \in W^{\mathbb{N}_0}$ the following holds: [3]

$$w \in \mathfrak{B} \iff \forall\, k \in \mathbb{N}_0 :\ w\big|_{[0,k]} \in \mathfrak{B}\big|_{[0,k]}. \quad (2)$$

□

A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ is complete if and only if it can be realised by a pastinduced state machine; e.g. [9], Theorem 2.2.9. Note that any finite state machine can be transformed to a pastinduced finite state machine without affecting the external behaviour. Hence, a behaviour induced by a finite state machine is complete. As another example, finite-dimensional discrete-time linear systems are seen to induce a complete external behaviour. However, not all behaviours are complete; e.g. $\mathfrak{B} = \{w \in \mathbb{R}^{\mathbb{N}_0} \mid \lim_{k\to\infty} w(k) = 0\}$.

Definition 2.5. (see [18], Defs. VIII.1, VIII.4) A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ over the signal space $W = W_{\text{in}} \times W_{\text{out}}$ is an *I/- behaviour* if: [4] (i) the input is *free*, i.e. $\mathcal{P}_{\text{in}}\mathfrak{B} = U^{\mathbb{N}_0}$; and (ii) the output *does not anticipate* the input, i.e. for all $k \in \mathbb{N}_0$ and $\tilde{w},\ \hat{w} \in \mathfrak{B}$, the following implication holds:

$$\mathcal{P}_{\text{in}}\tilde{w}\big|_{[0,k]} = \mathcal{P}_{\text{in}}\hat{w}\big|_{[0,k]} \implies \\ \exists\, w \in \mathfrak{B} :\ \mathcal{P}_{\text{out}}w\big|_{[0,k]} = \mathcal{P}_{\text{out}}\tilde{w}\big|_{[0,k]},\ \mathcal{P}_{\text{in}}w = \mathcal{P}_{\text{in}}\hat{w}. \quad (3)$$

□

Any behaviour that is induced by an I/S/- machine is an I/- behaviour; any pastinduced state machine that realises an I/- behaviour is an I/S/- machine; e.g. [11], Prop. 24.

3 A hybrid control configuration

We provide a detailed model for hybrid plants that consist of a continuous process, an actuator and a sensor; i.e. the dashed box in Fig. 1. While the hybrid closed-loop system

[1] The set of maps from T to W is denoted $W^T := \{w \mid w : T \to W\}$.

[2] $\mathbb{N}$ denotes the positive integers (without zero); let $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$.

[3] The restriction operator $(\cdot)|_{[k_1,k_2]}$: $W^{\mathbb{N}_0} \to W^{k_2-k_1+1}$ is defined by $w|_{[k_1,k_2]} = (w(k_1),\ w(k_{k+1}),\ \ldots\ w(k_2))$ for all $k_1,\ k_2 \in \mathbb{N}_0$, $k_1 \le k_2$, and all w: $\mathbb{N}_0 \to W$.

[4] By $\mathcal{P}_{\text{in}}$ and $\mathcal{P}_{\text{out}}$ we denote the natural projections from $W = W_{\text{in}} \times W_{\text{out}}$ to the input and output component, respectively; i.e. $\mathcal{P}_{\text{in}}w = c$ and $\mathcal{P}_{\text{out}}w = a$ for all $w = (c,\ a)$, $c \in W_{\text{in}}^{\mathbb{N}_0}$, $a \in W_{\text{out}}^{\mathbb{N}_0}$.

could be modelled within the widely accepted hybrid automata framework [1], a more explicit notion of inputs and outputs is required for our discussion of controller synthesis. We therefore construct an I/S/- machine that realises the external plant behaviour.

Continuous process. We model the continuous dynamics by a time invariant control system with input $u(t)$, state $x(t)$ and output $y(t)$:

$$\dot{x}(t) = f(x(t),\, u(t))\,, \tag{4}$$

$$y(t) = g(x(t),\, u(t))\,, \tag{5}$$

where $f\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $g\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$, $u\colon \mathbb{R}_0^+ \to \mathbb{R}^m$, $x\colon \mathbb{R}_0^+ \to \mathbb{R}^n$, $y\colon \mathbb{R}_0^+ \to \mathbb{R}^p$. Let $\mathcal{U} \subseteq \{u |\, u\colon \mathbb{R}_0^+ \to \mathbb{R}^m\}$ denote a set of *globally admissible inputs* such that for all $u \in \mathcal{U}$ and all initial conditions $x_0 \in \mathbb{R}^n$ the ODE (4) exhibits a unique solution $x(t)$ on the entire time axis with $x(0) = x_0$. We denote this solution $\varphi(x_0, u, \cdot)\colon \mathbb{R}_0^+ \to \mathbb{R}^n$. Note that all relevant continuous dynamics are to be summarised by the ODE (4) and this in technical applications typically includes the one or the other low-level continuous controllers e.g. output tracking. Here, unique existence of solutions appears a modest assumption.

Actuator. The actuator translates *discrete control events* from the finite control alphabet W_{in} to continuous input signals. We think of a control event as activating a particular *elementary manoeuvre* that is then executed by the continuous dynamics, perhaps supported by a suitable low-level continuous controller. If e.g. for a physical system the actuator exhibits continuous dynamics, these should be incorporated in (4). Here, the actuator is formalised by a *D/A-map* $\alpha\colon W_{\text{in}} \to \mathcal{U}$. Let $c\colon \mathbb{N}_0 \to W_{\text{in}}$ denote a sequence of control events where the k-th event is applied at continuous time $t_k \in \mathbb{R}_0^+$, $t_{k+1} > t_k$, $t_0 = 0$. Then the D/A-map transforms this timed sequence of input events to the input signal u with $u(t) = [\alpha(c(k))](t - t_k)$ for all $t \in [t_k, t_{k+1})$ and all $k \in \mathbb{N}_0$; i.e. the actuator interprets continuous time relative to the most recent discrete input event. Given a continuous initial condition $x(0) = x_0 \in \mathbb{R}^n$, the continuous system then evolves on the state trajectory $x\colon [0, \sup_k t_k) \to \mathbb{R}^n$, $x(t) = \varphi(x(t_k), \alpha(c(k)), t - t_k)$ for $t \in [t_k, t_{k+1})$.

Sensor. The task of the sensor is to generate discrete output events from the continuous output trajectory y. This is modelled by an *A/D-map* $\beta\colon \mathbb{R}^p \to W_{\text{out}}$, where W_{out} is the finite alphabet of *measurement events*. With each measurement event $a \in W_{\text{out}}$ we associate the region $\mathcal{Y}_a := \beta^{-1}(a) := \{\zeta \in \mathbb{R}^p |\, a = \beta(\zeta)\}$ in the continuous output space. A measurement event is generated when the output signal exits a restricted domain $\text{Inv} \subseteq \mathbb{R}^p$ of evolution, which for historical reasons is referred to as the *mode invariant*. In general, the mode invariant may depend on the most recent control event. However, in our setting the continuous output depends on the continuous input, and we may without loss of generality use one common invariant $\text{Inv} \subseteq \mathbb{R}^p$ for all control events. We assume that $\text{Inv} \subseteq \mathbb{R}^p$ is open and that g is continuous. For any given initial condition $x_0 \in \mathbb{R}^n$ and any input signal $u \in \mathcal{U}$, the time that elapses until a measurement event is triggered is defined by

$$t^+(x_0,\, u) := \sup\{t |\, \forall\, \tau \in [0, t)\colon\ g(\varphi(x_0,\, u,\, \tau), u(\tau))) \in \text{Inv}\}\,. \tag{6}$$

Equation (6) exhibits two special cases: $t^+(x_0,\, u) = \infty$ indicates that the continuous output evolves within Inv for all future; $t^+(x_0,\, u) = 0$ corresponds to $g(x_0, u(0))) \notin \text{Inv}$. Both special cases are considered as errors and trigger a distinguished output event $a_{\text{err}} \in W_{\text{out}}$. [5] In the nominal case $t^+(x_0,\, u) \in (0, \infty)$, a quantised version of the continuous output $y(t_0 + t^+(x_0,\, u))$ will be generated as an output event. This mechanism of event generation can be conveniently summarised by two maps $F\colon \mathbb{R}^n \times \mathcal{U} \to \mathbb{R}^n$ and $G\colon \mathbb{R}^n \times \mathcal{U} \to W_{\text{out}}$, where

$$F(x_0,\, u) := \varphi(x_0,\, u,\, t^+(x_0,\, u))\,, \tag{7}$$

$$G(x_0,\, u) := \beta(\, g(F(x_0, u),\, u(t^+(x_0,\, u)))\,)\,. \tag{8}$$

for $t^+(x_0,\, u) \in (0, \infty)$ and

$$F(x_0,\, u) := x_0\,, \qquad G(x_0,\, u) := a_{\text{err}}\,, \tag{9}$$

for $t^+(x_0,\, u) \in \{0, \infty\}$.

External plant behaviour. The components listed so far constitute continuous dynamics over continuous time with a discrete event interface via measurement and control events. Note that in our configuration Fig. 1 we assume that a control event can only occur as an immediate reaction to a measurement event. Therefore the continuous input signal $u \in \mathcal{U}$ remains the same between successive measurement events. The external plant behaviour $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\text{in}} \times W_{\text{out}}$, is defined as the set of event sequences that can occur according to the detailed model developed above. For a realisation of $\mathfrak{B}_{\text{p}}$ by a state machine $P = (X,\, W,\, \delta,\, X_0)$, we let $X := X_0 := \mathbb{R}^n$, and $(\xi,\, (\mu, \nu),\, \xi') \in \delta$ if and only if both $\xi' = F(\xi,\, \alpha(\mu))$ and $\nu = G(\xi,\, \alpha(\mu))$. It is readily observed that P is an I/S/- machine and, hence, $\mathfrak{B}_{\text{p}}$ is an I/- behaviour. Note however that, in general, $\mathfrak{B}_{\text{p}}$ fails to be complete. Hence, a pastinduced realisation of $\mathfrak{B}_{\text{p}}$ may fail to exist.

4 Supervisory controller synthesis

Adapting the concepts of Ramadge and Wonham's DES supervisory control theory [15, 16], the task of a supervisor is to restrict a plant behaviour $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$ so that the closed loop is guaranteed to only evolve on acceptable signals. This specification can be formally represented by the set of acceptable external signals, denoted $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$. As indicated in Fig. 1, we aim for a finite automaton realisation of a supervisor. To this end, however, we can represent the supervisor by its induced behaviour $\mathfrak{B}_{\text{sup}}$. The closed-loop behaviour is defined as the intersection $\mathfrak{B}_{\text{cl}} = \mathfrak{B}_{\text{p}} \cap \mathfrak{B}_{\text{sup}}$,

[5] It will be the task of the supervisors to prevent a_{err} by applying control events $c \in W_{\text{in}}$ for which $t^+(x_0,\, \alpha(c)) \in (0, \infty)$. The device of the error event a_{err} is crucial to formalise this control objective. However, the plant behaviour for *after* the occurrence of a_{err} can be padded arbitrarily.

and $\mathfrak{B}_{\text{sup}}$ is said to *enforce the specification* $\mathfrak{B}_{\text{spec}}$ if the inclusion $\mathfrak{B}_{\text{cl}} \subseteq \mathfrak{B}_{\text{spec}}$ is satisfied. However, two conditions apply for the interconnection of a supervisor with a plant and we shall state and motivate these *admissibility criteria* in terms of behaviours.

The first admissibility criterion addresses the requirement that the two systems must not "get stuck" temporally. That is, if the two behaviours can agree on a common trajectory up to time k, then there should also be some common future evolution on the entire discrete time axis.

Definition 4.1. (See [11]) Two behaviours $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$ and $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$ are said to be *nonconflicting* if $\mathfrak{B}_{\text{p}}|_{[0,k]} \cap \mathfrak{B}_{\text{sup}}|_{[0,k]} = (\mathfrak{B}_{\text{p}} \cap \mathfrak{B}_{\text{sup}})|_{[0,k]}$ for all $k \in \mathbb{N}_0$. □

A similar notion of nonconflictingness can be found for formal languages; e.g. [3]. Two pastinduced state machines induce nonconflicting behaviours if and only if their parallel composition is nonblocking. If the parallel composition of two state machines is nonblocking, then the induced behaviours are nonconflicting. In general, the converse implication does not hold.

Our second condition on behavioural interconnection specifically addresses I/- behaviours: the supervisor may enable or disable certain plant input events at any time but no restrictions on the plant outputs are allowed.

Definition 4.2. [6] A behaviour $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\text{in}} \times W_{\text{out}}$, is *generically implementable* if $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_{\text{sup}}|_{[0,k]}$, $\tilde{w}|_{[0,k]} \in W^{k+1}$, $\tilde{w}|_{[0,k]} \approx_{\text{y}} w|_{[0,k]}$ implies $\tilde{w}|_{[0,k]} \in \mathfrak{B}_{\text{sup}}|_{[0,k]}$. □

Our notion of generic implementability differs slightly from *implementability w.r.t. a particular plant* as defined in [11]. This adjustment leads to admissibility criteria that are independent of particular plant dynamics. This becomes crucial for the synthesis of robust supervisors, where one considers a parametrised set of plants; see Section 5. Our supervisory control problem is defined as follows: [7]

Definition 4.3. (See [11], Def. 16; also [12]) Given a plant $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\text{in}} \times W_{\text{out}}$, and a specification $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$, the pair $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$ is a *supervisory control problem*. A supervisor $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$ is *admissible* to the plant $\mathfrak{B}_{\text{p}}$, if $\mathfrak{B}_{\text{p}}$ and $\mathfrak{B}_{\text{sup}}$ are nonconflicting and $\mathfrak{B}_{\text{sup}}$ is generically implementable. A supervisor $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$ *enforces the specification* $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$ if $\mathfrak{B}_{\text{cl}} := \mathfrak{B}_{\text{p}} \cap \mathfrak{B}_{\text{sup}} \subseteq \mathfrak{B}_{\text{spec}}$. A supervisor $\mathfrak{B}_{\text{sup}}$ that is admissible to $\mathfrak{B}_{\text{p}}$ and that enforces the $\mathfrak{B}_{\text{spec}}$ is said to be a *solution* of $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$. A solution $\mathfrak{B}_{\text{sup}}$ is *nontrivial* if it imposes a nontrivial closed loop behaviour $\mathfrak{B}_{\text{cl}} := \mathfrak{B}_{\text{p}} \cap \mathfrak{B}_{\text{sup}} \neq \emptyset$. □

[6] We use $\tilde{w}|_{[0,k]} \approx_{\text{y}} w|_{[0,k]}$ as an abbreviation for the two restricted trajectories to be identical up to the last output event, i.e. $\mathcal{P}_{\text{in}}\tilde{w}|_{[0,k]} = \mathcal{P}_{\text{in}}w|_{[0,k]}$ and $\mathcal{P}_{\text{out}}\tilde{w}|_{[0,k)} = \mathcal{P}_{\text{out}}Yw|_{[0,k)}$.

[7] For a single fixed plant behaviour, it can be shown that the two alternative notions of implementability lead to precisely the same closed-loop behaviours. In this sense, the supervisory control problem in Definition 4.3 is equivalent to that in [11].

Observe that $\mathfrak{B}_{\text{sup}} = \emptyset$ is a trivial solution to any supervisory control problem. Moreover, $\mathfrak{B}_{\text{sup}} = \emptyset$ is the *only* trivial solution:

Proposition 4.4. (See [12], Prop. 2.6) Let $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$ be admissible w.r.t. an I/- behaviour $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\text{in}} \times W_{\text{out}}$. If $\mathfrak{B}_{\text{sup}} \neq \emptyset$, then $\mathfrak{B}_{\text{p}} \cap \mathfrak{B}_{\text{sup}} \neq \emptyset$. □

Very much in the spirit of [15, 16], the following theorem uses a set-theoretic lattice argument to establish the unique existence of a *least restrictive* supervisor.

Theorem 4.5. (See [12], Thm. 2.7) Let $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$ be a supervisory control problem. The set of all solutions of $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$ is a complete upper semi-lattice with the join operator "$\cup$" and the partial order "$\subseteq$". The supremal element $\mathfrak{B}^{\uparrow}_{\text{sup}}$ of that lattice is referred to as *least restrictive solution* of $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$. □

The least restrictive supervisor $\mathfrak{B}^{\uparrow}_{\text{sup}}$ contains all other solutions $\mathfrak{B}_{\text{sup}}$ of $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$; i.e. $\mathfrak{B}_{\text{sup}} \subseteq \mathfrak{B}^{\uparrow}_{\text{sup}}$. In particular, $\mathfrak{B}^{\uparrow}_{\text{sup}}$ is a nontrivial solution if and only if a nontrivial solution exists. Another aspect of practical relevance is that, as we now show, $\mathfrak{B}^{\uparrow}_{\text{sup}}$ is complete whenever $\mathfrak{B}_{\text{spec}}$ is complete. The latter can be ensured by requiring that $\mathfrak{B}_{\text{spec}}$ is realised by a finite automaton, and this assumption is very common in applications. Note that Proposition 4.6 does not require the plant $\mathfrak{B}_{\text{p}}$ to be complete.

Proposition 4.6. (See [12], Prop. 3.4) Let $\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$. If $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$ is complete, then the least restrictive solution $\mathfrak{B}^{\uparrow}_{\text{sup}}$ of the supervisory control problem $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$ is also complete. □

It can be shown that the parallel composition of any I/S/-machine with any pastinduced realisation of a generically implementable supervisor is nonblocking. The existence of such a pastinduced realisation can be ensured by Proposition 4.6. Obviously, a nonblocking closed loop is highly desirable for engineering applications and it justifies the general layout of our supervisory control problem.

5 Abstraction based synthesis and robust control

We develop a natural link between abstraction based controller synthesis and robust control, and extend it to investigate robust supervisory controller synthesis. While the literature gives some account of robustness of hybrid closed-loop systems e.g. [6], it is only recently that design procedures for robust hybrid control have been proposed e.g. [10, 5]. While our main target is the hybrid control configuration from Section 3, but our reasoning applies to *arbitrary* behaviours, including those that are realised by finite automata.

If both $\mathfrak{B}_{\text{p}}$ and $\mathfrak{B}_{\text{spec}}$ are realised by pastinduced finite automata, a realisation of the least restrictive solution $\mathfrak{B}^{\uparrow}_{\text{sup}}$ to the problem $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$ can be computed with a slight

modification of DES tools. However, since hybrid plant behaviours $\mathfrak{B}_p$ almost never have a finite realisation, we instead work with an approximation $\mathfrak{B}_{ca}$ that is realised by a finite automaton. We say $\mathfrak{B}_{ca}$ is an *abstraction* of $\mathfrak{B}_p$ if $\mathfrak{B}_p \subseteq \mathfrak{B}_{ca}$. Under this condition, we can guarantee that solutions for $\mathfrak{B}_{ca}$ carry over to $\mathfrak{B}_p$. To prove this claim, we first show that a complete supervisor that is generically implementable is also admissible to any plant that is realisable by an I/S/- machine.

Lemma 5.1. Let $\mathfrak{B}_{sup} \subseteq W^{\mathbb{N}_0}$, $W = W_{in} \times W_{out}$, be complete and generically implementable. If a plant $\mathfrak{B}_p \subseteq W^{\mathbb{N}_0}$ is realisable by an I/S/- machine then $\mathfrak{B}_p$ and $\mathfrak{B}_{sup}$ are nonconflicting.

Proof. Let $P = (X, W_{in} \times W_{out}, \delta, X_0)$ be an I/S/- machine that realises $\mathfrak{B}_p$. Pick any $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_p|_{[0,k]} \cap \mathfrak{B}_{sup}|_{[0,k]}$. Without loss of generality we may assume $w \in \mathfrak{B}_p$. Pick $x \in X^{\mathbb{N}_0}$ such that $(x(\kappa), w(\kappa), x(\kappa+1)) \in \delta$ for all $\kappa \in \mathbb{N}_0$ and $x(0) \in X_0$. Pick $\tilde{w} \in \mathfrak{B}_{sup}$ such that $\tilde{w}|_{[0,k]} = w|_{[0,k]}$. Let $\mu := \mathcal{P}_{in}\tilde{w}(k+1)$. Since P is an I/S/- machine, there exists $\xi' \in X$ and $\nu \in Y$ such that $(x(k), (\mu, \nu), \xi') \in \delta$. In consequence, we can construct a trajectory $\hat{x} \in X^{\mathbb{N}_0}$, $\hat{w} \in W^{\mathbb{N}_0}$, such that $(\hat{x}(\kappa), \hat{w}(\kappa), \hat{x}(\kappa+1)) \in \delta$ for all $\kappa \in \mathbb{N}_0$ and $\hat{x}|_{[0,k]} = x|_{[0,k]}$, $\hat{w}|_{[0,k+1]} \approx_y \tilde{w}|_{[0,k+1]}$. We use generic implementability of $\mathfrak{B}_{sup}$ to observe that $\hat{w}|_{[0,k+1]} \in \mathfrak{B}_p|_{[0,k+1]} \cap \mathfrak{B}_{sup}|_{[0,k+1]}$. Thus our construction can be carried out iteratively, and thereby constitutes a sequence of trajectories $(w_\kappa, x_\kappa) \in (W \times X)^{\mathbb{N}_0}$, $\kappa \in \mathbb{N}_0$, with $(w_\kappa, x_\kappa)|_{[0,k+\kappa]} = (w_{\kappa+1}, x_{\kappa+1})|_{[0,k+\kappa]}$. This implies that, for each $j \in \mathbb{N}_0$, the sequences $(w_\kappa(j))_{\kappa\in\mathbb{N}_0}$ and $(x_\kappa(j))_{\kappa\in\mathbb{N}_0}$ converge as $\kappa \to \infty$. Thus we obtain limit trajectories $w_\infty \in W^{\mathbb{N}_0}$ and $x_\infty \in X^{\mathbb{N}_0}$. Observe that $w_\infty|_{[0,\kappa]} \in \mathfrak{B}_{sup}|_{[0,\kappa]}$ for all $\kappa \in \mathbb{N}_0$, and, by completeness of $\mathfrak{B}_{sup}$, we obtain $w_\infty \in \mathfrak{B}_{sup}$. Similarly, observe that $(x_\infty(\kappa), w_\infty(\kappa), x_\infty(\kappa+1)) \in \delta$ for all κ, and, hence, $w_\infty \in \mathfrak{B}_p$. The last two observations imply $w|_{[0,k]} \in (\mathfrak{B}_p \cap \mathfrak{B}_{sup})|_{[0,k]}$. □

The above lemma leads to our central theorem on abstraction based supervisory controller synthesis:

Theorem 5.2. Let $\mathfrak{B}_{ca} \subseteq W^{\mathbb{N}_0}$ be an abstraction of a plant $\mathfrak{B}_p \subseteq \mathfrak{B}_{ca}$ and let $\mathfrak{B}_{sup}$ be a complete and nontrivial solution to the supervisory control problem $(\mathfrak{B}_{ca}, \mathfrak{B}_{spec})$, $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$. If $\mathfrak{B}_p$ is realisable in I/S/- plant form, then $\mathfrak{B}_{sup}$ is a nontrivial solution of $(\mathfrak{B}_p, \mathfrak{B}_{spec})$.

Proof. Generic implementability does not depend on the particular plant, and, by Lemma 5.1, we obtain that $\mathfrak{B}_{sup}$ is admissible w.r.t. $\mathfrak{B}_p$. Clearly, $\mathfrak{B}_{sup}$ enforces the specification for $\mathfrak{B}_{ca}$, and, hence, $\mathfrak{B}_{sup}$ solves $(\mathfrak{B}_p, \mathfrak{B}_{spec})$. Nontriviality is a consequence of Prop. 4.4. □

This contrasts with the basic DES setting [15, 16], where the signal space is a union of controllable and uncontrollable events, and a controllable sublanguage of an abstraction may very well fail to be a controllable sublanguage of the actual plant. Theorem 5.2 exploits the input-output structure of our framework and thereby reduces the problem of hybrid controller synthesis to the construction of a plant abstraction that can be realised by a finite automaton. The latter problem has been discussed from various perspectives e.g. [7, 4, 14]. In [11, 13] so called *l-complete approximations* $\mathfrak{B}_l$, $l \in \mathbb{N}$, are proposed as a particular suitable class of abstractions: (i) accuracy is monotone in the parameter $l \in \mathbb{N}$, i.e. $\mathfrak{B}_p \subseteq \mathfrak{B}_{l+1} \subseteq \mathfrak{B}_l$; and (ii) a pastinduced finite automaton that realises $\mathfrak{B}_l$ can be computed from the *finite* set $\mathfrak{B}|_{[0,l]}$, provided that $|W| \in \mathbb{N}$.

Reading Theorem 5.2 from a different point of view, it relates to robust control in a broad sense: the theorem states a sufficient condition under which a controller achieves a control objective not only for one particular plant, but for a family of plants. Further elaboration of this line of thought will enable us to give a not only sufficient but also necessary condition. As a first step, we formally define a problem of robust supervisory control.

Definition 5.3. Let $(\mathfrak{B}_\theta)_{\theta\in\Theta}$ denote a family of behaviours $\mathfrak{B}_\theta \subseteq W^{\mathbb{N}_0}$, indexed by the *uncertain parameter* $\theta \in \Theta$. Given a specification $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$, the pair $((\mathfrak{B}_\theta)_{\theta\in\Theta}, \mathfrak{B}_{spec})$ is a *supervisory control problem under parameter uncertainty*. If a candidate supervisor $\mathfrak{B}_{sup} \subseteq W^{\mathbb{N}_0}$ is a solution of $(\mathfrak{B}_\theta, \mathfrak{B}_{spec})$ for all $\theta \in \Theta$, then $\mathfrak{B}_{sup}$ is said to be a *robust solution* of $((\mathfrak{B}_\theta)_{\theta\in\Theta}, \mathfrak{B}_{spec})$. If in addition $\mathfrak{B}_\theta \cap \mathfrak{B}_{sup} \neq \emptyset$ for all $\theta \in \Theta$, then $\mathfrak{B}_{sup}$ is said to be a *robustly nontrivial solution*. □

Clearly, if any of the components of the hybrid plant model from Section 3 depends on an uncertain parameter of which only its range is known, this constitutes a family of plant behaviours in the sense of Definition 5.3. The prototypical example is the case in which the ODE (4) is uncertain, i.e. the right hand side f is replaced by a parameter dependent map $\hat{f}_\theta : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ with $\theta \in \Theta$. Another example that is covered by our general concept is the case of measurement noise. Here, we replace Eq. (5) by

$$y(t) = \hat{g}(x(t), u(t), v(t)), \tag{10}$$

where $v : \mathbb{R}_0^+ \to \mathbb{R}^q$ belongs to a specified class of disturbances, say $v \in \mathcal{V} := \{v \mid \forall\, t \in \mathbb{R}_0^+ : ||v(t)|| < \gamma\}$ for some norm $||\cdot||$ and some $\gamma > 0$. For any fixed disturbance $v \in \mathcal{V}$, let $\mathfrak{B}_v$ denote the behaviour induced by the hybrid plant. We then ask for a supervisor that enforces the specification $\mathfrak{B}_{spec}$ for all $v \in \mathcal{V}$. In terms of Definition 5.3, we ask for a solution to $((\mathfrak{B}_v)_{v\in\mathcal{V}}, \mathfrak{B}_{spec})$.

Let $(\mathfrak{B}_\theta)_{\theta\in\Theta}$ be a family of plants $\mathfrak{B}_\theta \subseteq W^{\mathbb{N}_0}$ that for any fixed $\theta \in \Theta$ are realisable by some I/S/- machine. Then Theorem 5.2 implies that if a complete supervisor $\mathfrak{B}_{sup}$ solves the (ordinary) supervisory control problem $(\cup_{\theta\in\Theta}\mathfrak{B}_\theta, \mathfrak{B}_{spec})$ then $\mathfrak{B}_{sup}$ also solves the supervisory control problem under parameter uncertainty $((\mathfrak{B}_\theta)_{\theta\in\Theta}, \mathfrak{B}_{spec})$. The next theorem establishes the converse: we can characterise the complete solutions under parameter uncertainty as solutions of $(\cup_{\theta\in\Theta}\mathfrak{B}_\theta, \mathfrak{B}_{spec})$.

Theorem 5.4. Let $(\mathfrak{B}_\theta)_{\theta\in\Theta}$ be a family of behaviours $\mathfrak{B}_\theta \subseteq W^{\mathbb{N}_0}$ that are realisable by I/S/- machines. Let $\mathfrak{B}_\cup := \cup_{\theta\in\Theta}\mathfrak{B}_\theta$ and $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$. A complete supervisor $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$ is a robust solution of $((\mathfrak{B}_\theta)_{\theta\in\Theta}, \mathfrak{B}_{\text{spec}})$ if and only if $\mathfrak{B}_{\text{sup}}$ is a solution of $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$. A robust solution $\mathfrak{B}_{\text{sup}}$ of $((\mathfrak{B}_\theta)_{\theta\in\Theta}, \mathfrak{B}_{\text{spec}})$ is robustly nontrivial if and only if $\mathfrak{B}_{\text{sup}}$ is a nontrivial solution of $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$.

Proof. First assume that $\mathfrak{B}_{\text{sup}}$ is a solution of $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$. If $\mathfrak{B}_{\text{sup}}$ is a nontrivial solution of $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$, then Theorem 5.2 implies that $\mathfrak{B}_{\text{sup}}$ is a robustly nontrivial solution for $((\mathfrak{B}_\theta)_{\theta\in\Theta}, \mathfrak{B}_{\text{spec}})$. If $\mathfrak{B}_{\text{sup}}$ is a trivial solution of $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$, we refer to Proposition 4.4 and the fact that the I/- property is retained under arbitrary unions of behaviours, to obtain that $\mathfrak{B}_{\text{sup}} = \emptyset$. In this case, $\mathfrak{B}_{\text{sup}}$ is a trivial robust solution. To prove the converse implications, assume that $\mathfrak{B}_{\text{sup}}$ is a robust solution for $((\mathfrak{B}_\theta)_{\theta\in\Theta}, \mathfrak{B}_{\text{spec}})$. Obviously, $\mathfrak{B}_\cup \cap \mathfrak{B}_{\text{sup}} = \cup_{\theta\in\Theta}(\mathfrak{B}_\theta \cap \mathfrak{B}_{\text{sup}}) \subseteq \mathfrak{B}_{\text{spec}}$, and hence $\mathfrak{B}_{\text{sup}}$ enforces the specification on $\mathfrak{B}_\cup$. To establish admissibility, we need to show that that $\mathfrak{B}_\cup$ and $\mathfrak{B}_{\text{sup}}$ are nonconflicting. Pick any $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_\cup|_{[0,k]} \cap \mathfrak{B}_{\text{sup}}|_{[0,k]}$. Then there exists an $\theta \in \Theta$ such that $w|_{[0,k]} \in \mathfrak{B}_\theta|_{[0,k]} \cap \mathfrak{B}_{\text{sup}}|_{[0,k]}$ and hence $w|_{[0,k]} \in (\mathfrak{B}_\theta \cap \mathfrak{B}_{\text{sup}})|_{[0,k]} \subseteq (\mathfrak{B}_\cup \cap \mathfrak{B}_{\text{sup}})|_{[0,k]}$. Therefore, $\mathfrak{B}_\cup$ and $\mathfrak{B}_{\text{sup}}$ are nonconflicting, and we conclude that $\mathfrak{B}_{\text{sup}}$ solves $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$. If in addition $\mathfrak{B}_{\text{sup}}$ is assumed to be a robustly nontrivial solution, nontriviality of $\mathfrak{B}_{\text{sup}}$ as a solution of $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$ follows from the simple observation that $\mathfrak{B}_\cup \cap \mathfrak{B}_{\text{sup}} \supseteq \mathfrak{B}_\theta \cap \mathfrak{B}_{\text{sup}} \neq \emptyset$ for any $\theta \in \Theta$. □

Note that in the context of our hybrid plant, where we may assume that $\mathfrak{B}_{\text{spec}}$ is realised by a finite automaton, we can focus on the least restrictive solution $\mathfrak{B}^\uparrow_{\text{sup}}$ of $(\mathfrak{B}_\cup, \mathfrak{B}_{\text{spec}})$ and then appeal to Proposition 4.6 for the completeness of $\mathfrak{B}^\uparrow_{\text{sup}}$. In this case, it is seen as a simple consequence of Theorem 5.4 that the supervisory problem under parameter uncertainty exhibits a unique least restrictive solution and that this least restrictive solution coincides with $\mathfrak{B}^\uparrow_{\text{sup}}$. Theorem 5.4 says that we can, in principle, approach the robust control problem by the same methods that have proved useful for the ordinary control problem [11, 13]. In particular, if we can compute the *finite* set $\mathfrak{B}_\cup|_{[0,l]}$, for some $l \in \mathbb{N}$, we can apply an l-complete approximation and derive a finite automaton P_l that realises an abstraction of $\mathfrak{B}_\cup$. Supervisory controller synthesis for P_l can then be carried out as indicated in [11, 13].

Conclusions

It is commonly accepted that every engineering system must be robust in order to provide a vital level of reliability. In this paper, we have addressed this requirement for a broad class of control problems in which we ask for a discrete event supervisor that enforces a language inclusion specification for an uncertain hybrid plant model. Our discussion is set within the framework of Willems' behavioural systems theory, and includes —but is not restricted to— the prototypical case in which the uncertainty effects the continuous plant dynamics. As our main result, we are able to characterise the desired robust supervisors as solutions of an ordinary (non-robust) supervisory control problem. Thus, a robust supervisor can be derived by our abstraction based methods from earlier work.

References

[1] R. Alur, T.A. Henzinger, G. Lafferriere, and G. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88:971–984, 2000.

[2] E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli. Effective synthesis of switching controllers for linear systems. *Proceedings of the IEEE*, 88:1011–1025, 2000.

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

[4] J.E.R. Cury, B.A. Krogh, and T. Niinomi. Synthesis of supervisory controllers for hybrid systems based on approximating automata. *IEEE Trans. on Automatic Control*, 43:564–568, 1998.

[5] E. Frazzoli, M.A. Dahleh, and E. Feron. Robust hybrid control for autonomous vehicle motion planning. Technical report, LIDS-P-2468, Massachusetts Institute of Technology, May 2000.

[6] C. Horn and P.J. Ramadge. Robustness issues for hybrid systems. In *IEEE Proc. of the 34th International Conference on Decision and Control, CDC'95*, pages 1467–1472, 1995.

[7] X. Koutsoukos, P.J. Antsaklis, J.A. Stiver, and M.D. Lemmon. Supervisory control of hybrid systems. *Proceedings of the IEEE*, 88:1026–1049, July 2000.

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

[9] T. Moor. *Approximationsbasierter Entwurf diskreter Steuerungen für gemischtwertige Regelstrecken*, volume 2 of *Forschungsberichte aus dem Max-Planck-Institut für Dynamik komplexer technischer Systeme*. Shaker-Verlag, Aachen, Germany, 2000. Also PhD thesis, Fachbereich Elektrotechnik, Universität der Bundeswehr Hamburg.

[10] T. Moor and J.M. Davoren. Robust controller synthesis for hybrid systems using modal logic. In *Hybrid Systems: Computation and Control (HSCC'01)*, LNCS 2034, pp. 433–446, 2001.

[11] T. Moor and J. Raisch. Supervisory control of hybrid systems within a behavioural framework. *Systems and Control Letters*, 38:157–166, 1999.

[12] T. Moor and J. Raisch. Think continuous, act discrete: DES techniques for continuous systems. *Proc. 10th Mediterranean Conference on Control and Automation*, Lisbon, July 2002.

[13] T. Moor, J. Raisch, and S.D. O'Young. Discrete supervisory control of hybrid systems based on l-complete approximations. *J. of Discrete Event Dynamic Systems*, 12:83–107, 2002.

[14] J. Raisch and S.D. O'Young. Discrete approximation and supervisory control of continuous systems. *IEEE Transactions on Automatic Control*, 43:569–573, 1998.

[15] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event systems. *SIAM J. Control and Optimization*, 25:206–230, 1987.

[16] P.J. Ramadge and W.M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77:81–98, 1989.

[17] J.C. Willems. Models for dynamics. *Dynamics Reported*, 2:172–269, 1989.

[18] J.C. Willems. Paradigms and puzzles in the theory of dynamic systems. *IEEE Transactions on Automatic Control*, 36:258–294, 1991.