Learning by doing: systematic abstraction refinement for hybrid control synthesis

T. Moor, J.M. Davoren and J. Raisch

Abstract: The synthesis of discrete event controllers for given continuous dynamics is studied within the hybrid system theory and its applications. A common approach involves the generation of a discrete abstraction of the continuous plant model, thus transforming the hybrid control problem into a purely discrete one that is then addressable using methods from the discrete event systems theory. In previous work, conditions were derived guaranteeing that successful synthesis on the abstraction level would provide a solution for the underlying hybrid problem. If synthesis failed, however, the abstraction was in need of refinement. This resulted in an iterative procedure alternating abstraction refinement with trial controller synthesis. The authors now use a temporal decomposition of the control problem to extract relevant diagnostic information when the synthesis step fails. In contrast with standard unfocused and global refinement strategies, the new iteration 'learns by doing' in the synthesis step and implements a refinement that is tailored to the particular hybrid plant and specification at hand.

1 Introduction

A basic hybrid control configuration is considered, in which a continuous process is controlled by a discrete supervisor where an actuator and a sensor translate discrete input events to continuous input signals and continuous output signals to discrete output events, respectively (Fig. 1). Typically, the actuator exhibits a hold characteristic and the sensor generates output events by quantisation. This configuration has received extensive attention and detailed mathematical models have been developed in the work of Cury *et al.* [1] and Koutsoukos *et al.* [2]. It can also be seen that the closed-loop system is representable within the hybrid automata framework [3, 4]. In this paper, we have taken the perspective of the discrete supervisor that exclusively interacts with the hybrid plant. Here, an adequate model of the hybrid plant is the external behaviour, which is defined as the set of all sequences of pairs of input events and output events that can be generated by the system. This definition conforms with Willems' [5, 6] behavioural systems theory, which has proved fruitful for the formulation and solution of supervisory control problems for hybrid systems [7, 8]. The present paper is entirely set within the behavioural framework and the reader is kindly referred to earlier work [8, 9], in which we have shown how the external plant behaviour is formally derived from a detailed model based on the individual components.

IEE Proceedings online no. 20050347

doi:10.1049/ip-cta:20050347

Paper first received 27th January 2005 and in revised form 1st March 2006

T. Moor is with the Lehrstuhl für Regelungstechnik, Friedrich-Alexander-Universität, Erlangen 91058, Germany

J.M. Davoren is with the Department of Electrical and Electronic Engineering, University of Melbourne, VIC 3010, Australia

J. Raisch is with Fachgebiet Regelungssysteme, Technische Universität Berlin, 10587 Berlin, and the Systems and Control Theory Group, Max-Planck-Institut für Dynamik Komplexer Technischer Systeme, Magdeburg 39106, Germany

E-mail: thomas.moor@rt.eei.uni-erlangen.de

According to Ramadge and Wonham's [10, 11] supervisory control theory for discrete event systems (DES), the task of the supervisor is to restrict the external plant behaviour so that the closed loop is guaranteed to only evolve on acceptable trajectories. This control specification is formalised as a language over the same alphabet of pairs of input and output symbols. Given a hybrid plant and a closed-loop specification, we investigate the supervisory controller synthesis problem; that is, the construction of a supervisor that enforces the specification when interconnected with the plant. A crucial feature of the hybrid setting is that state machine realisations of the plant behaviour typically evolve on a real-vector valued, and hence uncountable, state space. Therefore one cannot directly apply known synthesis procedures from DES theory. One possibility to circumvent this issue is to first construct a finite automaton abstraction from the external behaviour of the hybrid plant and then design a supervisor for that abstraction. In the work of Moor and co-workers [7, 8], we develop so called *l*-complete approximations of hybrid plants, and constructively demonstrate that they are realisable by finite automata. We also show that if a supervisor has been successfully synthesised for an *l*-complete approximation then that supervisor indeed solves the original hybrid control problem.

If an abstraction is too coarse it will not retain sufficient information about the original plant dynamics and no relevant supervisory controller will be found for that abstraction. In such a case, one needs to refer back to the original hybrid plant in order to obtain a refinement of the abstraction. As it is not clear beforehand how accurate the abstraction needs to be, one ends up with an iteration that alternates trial controller synthesis with abstraction refinement. This type of iteration is not unique for the method developed in the work of Moor and co-workers [7, 8], but is characteristic for abstraction-based approaches to hybrid control problems in general; the readers are referred to various research works [2, 12–15] for studies of a variety of hybrid plant classes and variety of kinds of closed-loop specifications.

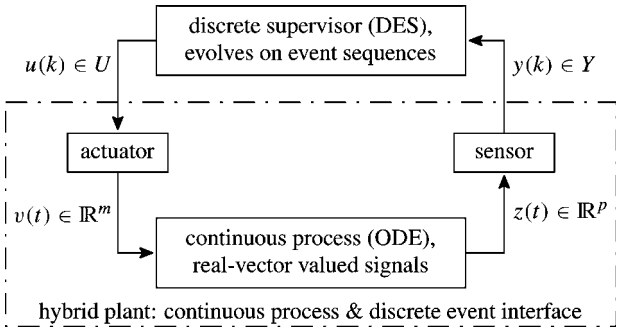

Fig. 1 *Hybrid control configuration*

In the present paper, we study the strategic use of refinements of abstractions for supervisory controller synthesis. Two basic questions motivate our discussion. How are we to formulate abstractions, so that they allow for a rich variety of refinements? How are we to link the alternation of controller synthesis and abstraction refinement, so that we gain relevant benefits from the increased flexibility in abstraction refinement? In addressing these questions, we propose a temporal decomposition of the supervisory control problem in order to diagnose why a particular abstraction fails to yield a solution of the synthesis problem. Our refinement procedure then focuses its efforts on those aspects of the abstraction that have 'caused' the failure in synthesis. Rather than an unfocused global refinement, our new iteration 'learns from failure' in the synthesis step and implements a refinement that is tailored to the particular hybrid plant and specification at hand.

This idea of systematic refinements for abstraction-based supervisory controller synthesis was first proposed by Moor *et al.* [16]. For abstraction-based verification of temporal properties, related strategies for refinements have been independently developed in the work of Stursberg *et al.* [17] and Clarke *et al.* [18]. There, the refinement was guided by counter-examples detected by model-checking algorithms. Our refinement scheme is technically based on so called experiments that are related to but distinct from positive examples used for inductive inference [19].

Notation and preliminaries

Let $\mathbb{N}$ denote the positive integers and $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. We write $|A|$ for the cardinality of a set A and $|A| \in \mathbb{N}_0$ to indicate that A is a finite set. The set of all maps from $\mathbb{N}_0$ to W is denoted $W^{\mathbb{N}_0} := \{w\colon \mathbb{N}_0 \to W\}$, and subsets $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ are called behaviours over the signal space W [6].

The shift operators $\sigma^k\colon W^{\mathbb{N}_0} \to W^{\mathbb{N}_0}$, for $k \in \mathbb{N}_0$, are defined by $(\sigma^k w)(\kappa) := w(\kappa + k)$ for $\kappa,\ k \in \mathbb{N}_0$, and $\sigma := \sigma^1$. A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ is time invariant if $\sigma\mathfrak{B} \subseteq \mathfrak{B}$. The restriction operator $(\cdot)|_{[k_1,k_2)}\colon W^{\mathbb{N}_0} \to W^{k_2-k_1}$, maps sequences $w \in W^{\mathbb{N}_0}$ to finite strings $w|_{[k_1,k_2)} := w(k_1)w(k_1+1)\cdots w(k_2-1) \in W^{k_2-k_1}$ where $k_1,\ k_2 \in \mathbb{N}_0$, $k_1 \le k_2$. For the case $k_1 = k_2$, let $W^0 := \{\epsilon\}$, where ϵ is the empty string. Note that, by the natural extension of $(\cdot)|_{[k_1,k_2)}$ to sets, $\mathfrak{B}|_{[0,0)} = \{\epsilon\}$ for $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$, $\mathfrak{B} \neq \emptyset$, but $\emptyset|_{[0,0)} = \emptyset$. For closed intervals, the operator $(\cdot)|_{[k_1,k_2]}$ is defined accordingly. A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ is complete if $(\forall k_1,\ k_2 \in \mathbb{N}_0,\ k_2 \ge k_1\colon\ w|_{[k_1,k_2)} \in \mathfrak{B}|_{[k_1,k_2)})$ implies $w \in \mathfrak{B}$. For $l \in \mathbb{N}_0$, a behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ is l-complete if $(\forall k \in \mathbb{N}_0\colon\ \sigma^k w|_{[0,l)} \in \mathfrak{B}|_{[0,l)})$ implies $w \in \mathfrak{B}$. Note that l-completeness implies time invariance and completeness.

For $W = U \times Y$, we denote $\mathcal{P}_U$ and $\mathcal{P}_Y$ the natural projection operators to the respective component, that is, $\mathcal{P}_U w = u$ and $\mathcal{P}_Y w = y$ for $w = (u,\ y) \in W^{\mathbb{N}_0},\ u \in U^{\mathbb{N}_0},\ y \in Y^{\mathbb{N}_0}$. A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$, $W = U \times Y$, is an I/- behaviour if (i) the input is free, that is, $\mathcal{P}_U\mathfrak{B} = U^{\mathbb{N}_0}$ and (ii) the output does not anticipate the input, that is, for all $k \in \mathbb{N}_0$, $\tilde{w}$, $\hat{w} \in \mathfrak{B}$, if $\mathcal{P}_U\tilde{w}|_{[0,k)} = \mathcal{P}_U\hat{w}|_{[0,k)}$ then there exists $w \in \mathfrak{B}$ such that $\mathcal{P}_Y w|_{[0,k)} = \mathcal{P}_Y\tilde{w}|_{[0,k)}$ and $\mathcal{P}_U w = \mathcal{P}_U\hat{w}$ [6, 7].

The set of all finite strings over W is denoted W^*; that is, $W^* := \bigcup_{l\in\mathbb{N}_0} W^l$. We write $|s| \in \mathbb{N}_0$ for the length of a string $s \in W^*$, that is, $|s| = l$ for all $s \in W^l$. For two strings $a,\ b \in W^*$, we say that a is a prefix of b (and write $a \preccurlyeq b$) if and only if there exists $c \in W^*$ such that $b = ac$. We say a is a strict prefix of b (and write $a \prec b$) if and only if $a \preccurlyeq b$ and $a \neq b$. Note that $\preccurlyeq$ is a partial order on W^*. A set $S \subseteq W^*$ is prefix-closed if and only if $(s \in S \wedge \tilde{s} \prec s)$ implies $\tilde{s} \in S$. We say S is prefix-free if and only if $(s \in S \wedge \tilde{s} \prec s)$ implies $\tilde{s} \notin S$.

2 Abstraction-based supervisory control

We summarise key results from earlier work [7, 8], where we make use of Willems' behavioural framework to address supervisory controller synthesis for hybrid systems according to Fig. 1.

For our plant model, we choose $W := U \times Y$ as a signal space, where U and Y are finite alphabets of input and output events, respectively. The external plant behaviour $\mathfrak{B}_p$ is then defined as a set of all event sequences $w = (u,\ y) \in (U \times Y)^{\mathbb{N}_0}$ on which the hybrid plant can evolve in open loop. For our switched-server example in Section 3, it is readily verified that $\mathfrak{B}_p$ is an I/- behaviour. We derive the I/- property of the external plant behaviour for a general class of hybrid systems [7, 9].

The closed-loop behaviour under supervision $\mathfrak{B}_{sup} \subseteq W^{\mathbb{N}_0}$ is obtained by $\mathfrak{B}_{cl} := \mathfrak{B}_p \cap \mathfrak{B}_{sup}$. In compliance with the notion of I/- behaviours, we identify two admissibility criteria that address controllability and liveness.

Definition 1 [7, 20]: Let $\mathfrak{B}_p,\ \mathfrak{B}_{sup} \subseteq W^{\mathbb{N}_0}$, $W = U \times Y$. Then

(i) $\mathfrak{B}_{sup}$ is generically implementable if $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_{sup}|_{[0,k]}$, $\tilde{w}|_{[0,k]} \in W^{k+1}$, $\mathcal{P}_U\tilde{w}_{[0,k]} = \mathcal{P}_U\tilde{w}|_{[0,k]}$, $\mathcal{P}_Y\tilde{w}|_{[0,k)} = \mathcal{P}_Y w|_{[0,k)}$ implies $\tilde{w}|_{[0,k]} \in \mathfrak{B}_{sup}|_{[0,k]}$.
(ii) $\mathfrak{B}_p$ and $\mathfrak{B}_{sup}$ are non-conflicting if $\mathfrak{B}_p|_{[0,k]} \cap \mathfrak{B}_{sup}|_{[0,k]} = (\mathfrak{B}_p \cap \mathfrak{B}_{sup})|_{[0,k]}$ for all $k \in \mathbb{N}_0$.

Adopting the concepts of Ramadge and Wonham's [10, 11] supervisory control theory for DESs, the control objective is given by a specification $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$, which consists of all acceptable closed-loop trajectories. This leads to the following formulation of the problem of supervisory control.

Definition 2 [7, 20]: Given a plant $\mathfrak{B}_p \subseteq W^{\mathbb{N}_0}$, $W = U \times Y$, and a specification $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$, the pair $(\mathfrak{B}_p,\ \mathfrak{B}_{spec})_{cp}$ is a supervisory control problem. A supervisor $\mathfrak{B}_{sup} \subseteq W^{\mathbb{N}_0}$ is admissible to the plant $\mathfrak{B}_p$, if $\mathfrak{B}_p$ and $\mathfrak{B}_{sup}$ are non-conflicting and $\mathfrak{B}_{sup}$ is generically implementable. A supervisor $\mathfrak{B}_{sup} \subseteq W^{\mathbb{N}_0}$ enforces the specification $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$, if $\mathfrak{B}_{cl} := \mathfrak{B}_p \cap \mathfrak{B}_{sup} \subseteq \mathfrak{B}_{spec}$. A supervisor $\mathfrak{B}_{sup}$ that is admissible to $\mathfrak{B}_p$ and that enforces $\mathfrak{B}_{spec}$ is said to be a solution of $(\mathfrak{B}_p,\ \mathfrak{B}_{spec})_{cp}$. A solution $\mathfrak{B}_{sup}$ is non-trivial if it imposes a non-trivial closed-loop behaviour $\mathfrak{B}_{cl} := \mathfrak{B}_p \cap \mathfrak{B}_{sup} \neq \emptyset$.

It can be seen that a solution $\mathfrak{B}_{sup}$ is trivial if and only if $\mathfrak{B}_{sup} = \emptyset$ [20]. In the spirit of the work of Ramadge and Wonham [10, 11], we obtain the unique existence of a least restrictive solution by a set-theoretic lattice argument.

Corollary 3 [7, 9]: The set of all solutions of a supervisory control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$ is a complete upper semi-lattice with the usual set-theoretic operator $\cup$ and the partial order $\subseteq$. The supremal element $\mathfrak{B}^{\uparrow}_{sup}$ of this lattice is referred to as least restrictive solution of $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$.

If both $\mathfrak{B}_p$ and $\mathfrak{B}_{spec}$ were realised by finite automata, the least restrictive solution of the control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$ could be readily computed with a slight modification of existing DES tools. Although a finite automaton realisation for $\mathfrak{B}_{spec}$ is a modest requirement, the hybrid plant in general is not realisable on a finite state space. We approach the problem by replacing $\mathfrak{B}_p$ with an abstraction $\mathfrak{B}_{ca}$, $\mathfrak{B}_p \subseteq \mathfrak{B}_{ca}$, which is realised by a finite automaton. Clearly, a supervisor $\mathfrak{B}_{sup}$ that enforces a specification for a plant abstraction also enforces the specification for the original plant. Under modest technical conditions, Moor and Raisch [7] showed that solutions to $(\mathfrak{B}_{ca}, \mathfrak{B}_{spec})_{cp}$ are also admissible to the original plant $\mathfrak{B}_p$. Hence, the solutions of $(\mathfrak{B}_{ca}, \mathfrak{B}_{spec})_{cp}$ are seen to be solutions of $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$ and we can approach $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$ by applying finite automata methods to $(\mathfrak{B}_{ca}, \mathfrak{B}_{spec})_{cp}$. We verify the technical requirements directly from structural properties of an underlying hybrid system [8, 9]. In the present paper, we avoid the discussion of a detailed model and give preference to the argument in the work of Moor and Raisch [7], which is based on the completeness property.

Theorem 4 [7, 20]: Let $\mathfrak{B}_{ca} \subseteq W^{\mathbb{N}_0}$, $W = U \times Y$, be an abstraction of an I/- behaviour $\mathfrak{B}_p \subseteq W^{\mathbb{N}_0}$, let $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$ and let $\mathfrak{B}_{sup} \subseteq W^{\mathbb{N}_0}$ be a non-trivial solution to the supervisory control problem $(\mathfrak{B}_{ca}, \mathfrak{B}_{spec})_{cp}$. If $\mathfrak{B}_p$ and $\mathfrak{B}_{sup}$ are complete then $\mathfrak{B}_{sup}$ is a non-trivial solution of $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$.

Note that while Theorem 4 does not require $\mathfrak{B}_{spec}$ to be complete, in the case that it is, the least restrictive supervisor is also complete [20].

We outline a generic procedure for abstraction-based supervisory controller synthesis.

(S1) Let $j := 0$ and find a plant abstraction $\mathfrak{B}_0$, $\mathfrak{B}_p \subseteq \mathfrak{B}_0$.
(S2) Compute the least restrictive solution $\mathfrak{B}^j_{sup}$ of $(\mathfrak{B}_j, \mathfrak{B}_{spec})_{cp}$.
(S3) If $\mathfrak{B}_j \cap \mathfrak{B}^j_{sup} \neq \emptyset$, then terminate this iteration and return the solution $\mathfrak{B}^j_{sup}$.
(S4) Compute a refinement $\mathfrak{B}_{j+1}$, $\mathfrak{B}_p \subseteq \mathfrak{B}_{j+1} \subseteq \mathfrak{B}_j$, and proceed with (S2) for $j := j + 1$.

Note that the refinement step (S4) necessarily refers to the original plant $\mathfrak{B}_p$ and, for hybrid systems, involves computationally expensive reachability analysis over the underlying continuous state space. In general, the above procedure may fail to terminate or computational resources may be exhausted prior to termination. However, if the procedure terminates faithfully, then the last supervisor $\mathfrak{B}^j_{sup}$ solves the control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$; for an application case study from process control, refer to the work of Moor and Raisch [21]. Other abstraction-based approaches incorporate a similar iteration; Krogh and Chutinan [13] give a flow diagram that very much matches the structure of (S1)–(S4).

3 Example

The switched-server system (Fig. 2) consists of two tanks and a supply that can be either directed to one of the tanks or turned off. Control events reconfigure the server and measurement events indicate that a tank level crosses

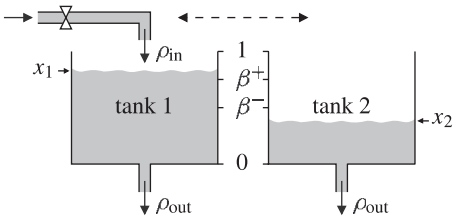

Fig. 2 *Switched-server system*

certain thresholds. The task of the supervisor is to eventually maintain certain tank levels.

We represent the levels of the tanks by a continuous state variable $x = (x_1, x_2)$ that evolves within the continuous state space $X := [0, 1]^2 \subset \mathbb{R}^2$. For given parameters $\rho_{in} > 0$ and $\rho_{out} > 0$, we assume that the supply rate of the active server remains within $[\rho_{in}, 2\rho_{in}] \subset \mathbb{R}$ and the drainage for each tank remains within $[(1/2)\rho_{out}, \rho_{out}] \subset \mathbb{R}$. Output events will be based on the threshold levels 0, β^-, β^+ and 1, where $0 < \beta^- < \beta^+ < 1$. The control objectives are that

(a) eventually, all tank levels are kept above or equal to β^-;
(b) serving a tank only commences at a level below or equal to β^+ and continues until the respective tank is full.

Our technical device to model the effect of control events and the generation of measurement events is a variation of the hybrid automata model [3]. With each discrete mode $\mu \in U$ there are associated continuous dynamics $(d/dt)x(t) \in F_\mu(x(t))$ on the state space X and a mode invariant $D_\mu \subseteq X$. The mode invariants function as a constraint on the continuous state. When the plant is in mode $\mu \in U$ and the continuous state is about to violate the constraint D_μ, an output event is triggered to give a quantised version of the current state. Quantisation is modelled by finitely many transition guards $G_{\mu,\nu} \subseteq D_\mu$, $\nu \in Y$, $|Y| \in \mathbb{N}_0$, where the ν-index is reported as the measurement of the event.

In our particular example, discrete modes correspond to server configurations 'serve tank 1', 'serve tank 2' and 'server off', respectively, encoded by the control alphabet $U = \{1, 2, 3\}$. Mode invariants and transition guards are chosen such that the hybrid automaton generates relevant measurement events $Y = \{1, 2, 3\}$. The associated continuous dynamics turn out as piecewise constant rectangular differential inclusions (Fig. 3).

In our detailed model, plant trajectories are built from segments $\gamma = (\mu, x, \nu)$ that consist of an input event μ to select the mode, an absolutely continuous state trajectory x: $[0, T] \to X$, $T > 0$, which satisfies $(d/dt)x(t) \in F_\mu$ for almost all $t \in [0, T]$ and an output event $\nu \in Y$. We distinguish two cases. For a segment γ to be regular, we require that $x(t)$ lies within D_μ for all $t \in [0, T]$, and that $x(T) \in G_{\mu,\nu}$. However, if the supervisor applies a control event $\mu \in U$ with $x(0) \notin D_\mu$, the continuous state trajectory is doomed to violate the constraint imposed by the mode

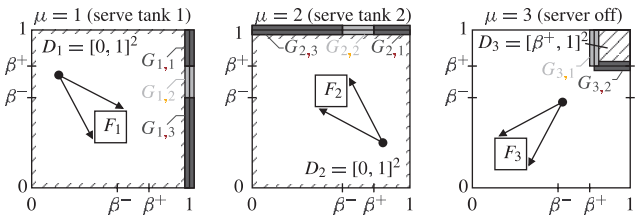

Fig. 3 *Mode invariants D_μ, guards $G_{\mu\nu}$ and continuous dynamics $(d/dt)x(t) \in F_\mu$*

invariant (other pathological situations can arise when the continuous state trajectory leaves the mode invariant without passing a guard, or, if it can remain within the mode invariant forever). We model this undesired situation by irregular segments $\gamma = (\mu, x, \ddagger)$, where $x(0) \notin D_\mu$ and $\ddagger$ is a distinguished error event. Naturally, it will be the supervisor's task to prevent irregular segments from occurring in the closed-loop system. We denote the set of all segments by Γ.

We can now formally characterise the external plant behaviour $\mathfrak{B}_p$. A labelled transition system with continuous state space $X = [0, 1]^2 \subset \mathbb{R}^2$ and labels $W = U \times \tilde{Y}$, $\tilde{Y} = Y \cup \{\ddagger\}$, is utilised to enforce the initial state of each segment to match the terminal state of the preceding segment. Let δ be the set of all transition steps $(\xi, (\mu, \nu), \xi') \in X \times (U \times \tilde{Y}) \times X$ such that there exists $T > 0$ and $x: [0, T] \to X$ with $(\mu, x, \nu) \in \Gamma$, $\xi = x(0)$ and $\xi' = x(T)$. Then

$$\mathfrak{B}_p := \{(u, y) \in (U \times \tilde{Y})^{\mathbb{N}_0} | \exists (\xi_k)_{k \in \mathbb{N}_0} \forall k \in \mathbb{N}_0 : (\xi_k, (u(k), y(k)), \xi_{k+1}) \in \delta\} \quad (1)$$

Note that, while $\mathfrak{B}_p$ is defined over the finite alphabet W, it does crucially depend on the underlying continuous dynamics. In particular, we must not expect the existence of a finite automaton realisation of $\mathfrak{B}_p$ [22].

For the statement of our control problem in terms of behaviours, we choose $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$ such that it represents our control objectives (a) and (b). Let $\mathfrak{B}_{spec}$ be the set of all $(u, y) \in (U \times \tilde{Y})^{\mathbb{N}_0}$ that satisfy the following conditions:

(a′) [all segments are regular] $\forall k \in \mathbb{N}_0$: $y(k) \neq \ddagger$.
(a″) [eventually no level drops below β^-] $\exists K \in \mathbb{N}_0$ $\forall k \geq K$: $(u(k), y(k)) \notin \{(1, 3), (2, 3)\}$;
(b′) [serving of tank 1 commences only when its level is not above β^+] $\forall k \in \mathbb{N}_0$: $u(k+1) = 1 \Rightarrow (u(k), y(k)) \notin \{(2, 1), (3, 2)\}$;
(b″) [serving of tank 2 commences only when its level is not above β^+] $\forall k \in \mathbb{N}_0$: $u(k+1) = 2 \Rightarrow (u(k), y(k)) \notin \{(1, 1), (3, 1)\}$.

Conditions (a′), (b′) and (b″) can be individually encoded as finite state machines, and, hence, so can their conjunct. For a finite automata representation of (a″) one needs to introduce marked states to encode the eventuality construct. Note that by virtue of (a″), $\mathfrak{B}_{spec}$ fails to be complete.

4 Experiments on behaviours

We develop an abstract notion of experiments on behaviours to provide a tool for generating plant abstractions. Technically, we define an experiment to be a set of strings bounded in length that relates to the underlying behaviour by two complementary conditions: (i) any string in the experiment must be a prefix of some trajectory in the behaviour; and (ii) any trajectory in the behaviour must exhibit a prefix that is in the experiment.

Definition 5: A set of finite strings $S \subseteq W^*$ is an experiment over W, if there exists $k \in \mathbb{N}_0$ such that $|s| \leq k$ for all $s \in S$. Given a behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ and an experiment $S \subseteq W^*$, we say that S is an experiment on $\mathfrak{B}$ if the following conditions hold for all $s \in W^*$ and all $w \in W^{\mathbb{N}_0}$:

(i) $s \in S \Rightarrow s \in \mathfrak{B}|_{[0, |s|)}$;
(ii) $w \in \mathfrak{B} \Rightarrow \exists l \in \mathbb{N}_0$: $w|_{[0, l)} \in S$.

Fig. 4 illustrates the experiment $S = \{1, 00, 010\}$ over $W = \{0, 1\}$ that has been conducted on a behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$. The tree diagram shows the first three external signal values of all possible trajectories over W. The solid lines indicate evolution that conforms with $\mathfrak{B}$, the black nodes mark strings that belong to S.

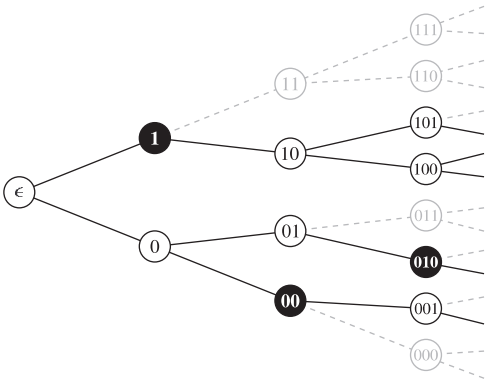

Fig. 4 *Experiment S (black nodes), S := {1, 00, 010} ⊂ {0, 1}**

4.1 Strongest model from an experiment

Suppose we are provided an experiment S on some behaviour $\mathfrak{B}$. Our objective is then to recover a model $\mathfrak{B}_S$ from S that is guaranteed to be an abstraction of $\mathfrak{B}$; that is, $\mathfrak{B} \subseteq \mathfrak{B}_S$. Clearly, in the recovery process, one wants to take into account any structural knowledge of the underlying $\mathfrak{B}$. In the subsequent argument, we focus attention on time invariant behaviours.

Definition 6: Let $\mathfrak{M}_W$ be the set of all time invariant behaviours over W. An experiment $S \subseteq W^*$ is consistent with time invariance, if there exists a $\mathfrak{B} \in \mathfrak{M}_W$ such that S is an experiment on $\mathfrak{B}$. Let $\mathfrak{E}_W$ be the set of all experiments over W, which are consistent with time invariance. For any $S \in \mathfrak{E}_W$, we say $\mathfrak{B}_S \subseteq \mathfrak{M}_W$ is a model from S under the assumption of time invariance, if

$$\forall \mathfrak{B} \in \mathfrak{M}_W : (S \text{ is an experiment on } \mathfrak{B} \Rightarrow \mathfrak{B} \subseteq \mathfrak{B}_S) \quad (2)$$

Given an experiment $S \in \mathfrak{E}_W$, the strongest (i.e. smallest with respect to $\subseteq$) model from S [under the assumption of time invariance] is obtained by

$$\mathcal{M}^\downarrow(S) := \cup \{\mathfrak{B} \in \mathfrak{M}_W | S \text{ is an experiment on } \mathfrak{B}\} \quad (3)$$

The following proposition gives an explicit characterisation in terms of S.

Proposition 7: For any $S \in \mathfrak{E}_W$, the following properties hold:

(i) $\mathcal{M}^\downarrow(S) \in \mathfrak{M}_W$ is a model from S under the assumption of time invariance.
(ii) S is an experiment on $\mathcal{M}^\downarrow(S)$.
(iii) $\mathcal{M}^\downarrow(S) = \{w \in W^{\mathbb{N}_0} | \forall k \in \mathbb{N}_0 \exists l \in \mathbb{N}_0$: $\sigma^k w|_{[0, l)} \in S\}$.

Note that, from (iii), we can without loss of generality restrict our subsequent discussion to prefix-free experiments. In particular, either $S = \{\epsilon\}$ or $\epsilon \notin S$.

4.2 Minimal experiments

The map $\mathcal{M}^\downarrow$ is interpreted as a parametrisation of the class of behaviours $\mathcal{M}^\downarrow(\mathfrak{E}_W) \subseteq \mathfrak{M}_W$, and we say that the behaviour $\mathfrak{B} = \mathcal{M}^\downarrow(S)$ is realised by the experiment S. Note that, if $|W| \in \mathbb{N}_0$ then the behaviour $\mathfrak{B} = \mathcal{M}^\downarrow(S)$ can, in

fact, be realised by a finite automaton. It can also be seen that $\mathfrak{B} \in \mathfrak{M}^{\downarrow}(\mathfrak{E}_W)$ if and only if $\mathfrak{B}$ is l-complete for some $l \in \mathbb{N}_0$. Another question from this perspective is how to obtain canonical realisations, that is how to choose a representative from the equivalence class $[S] := \{\tilde{S} \in \mathfrak{E}_W | \mathcal{M}^{\downarrow}(\tilde{S}) = \mathcal{M}^{\downarrow}(S)\}$, $S \in \mathfrak{E}_W$. Naturally, such a choice depends on the intended purpose of the canonical realisation. In the context of hybrid systems, a reasonable objective is to keep the number of strings small and the length of strings short, as this will reduce the computational effort when conducting the experiment. We therefore extend the partial order on strings to experiments

$$S_1 \preccurlyeq S_2 :\Leftrightarrow (\forall s_2 \in S_2\, \exists s_1 \in S_1 : s_1 \preccurlyeq s_2) \quad \text{and} \quad (\forall s_1 \in S_1\, \exists s_2 \in S_2 : s_1 \preccurlyeq s_2) \tag{4}$$

Indeed, minimality with respect to the partial order (4) does lead to a canonical form.

Theorem 8: Let $S \in \mathfrak{E}_W$. Then, there uniquely exists a minimal experiment $S_{\min} \in [S]$; that is, $S_{\min} \preccurlyeq \tilde{S}$ for all $\tilde{S} \in [S]$. Furthermore, $S_{\min}$ is prefix-free.

Proof (outline): Uniqueness is a direct consequence of minimality. Existence is witnessed by

$$S_{\min} := \{s \in W^* | (\exists \tilde{S} \in [S] : s \in \tilde{S}) \quad \text{and} \quad (\forall \tilde{S} \in [S], \forall \tilde{s} \in W^{\mathbb{N}_0} : \tilde{s} \prec s \Rightarrow \tilde{s} \notin \tilde{S})\} \tag{5}$$

□

4.3 Refinement of experiments

Observe that for two experiments S_1, $S_2 \in \mathfrak{E}_W$ with $S_1 \preccurlyeq S_2$, we obtain from proposition 7, (iii), that $\mathcal{M}^{\downarrow}(S_2) \subseteq \mathcal{M}^{\downarrow}(S_1)$. This suggests that a refinement S_2 can be generated by replacing strings $s_1 \in S_1$ by longer strings s_2 such that $s_1 \preccurlyeq s_2$. Note, however, that care must be taken to cover all possible future evolution from a string s_1 in the refinement S_2. Formally, we say that $S_2 \in \mathfrak{E}_W$ is a refinement of S_1 if $\mathcal{M}^{\downarrow}(S_2) \subseteq \mathcal{M}^{\downarrow}(S_1)$. Hence, the extension ordering $S_1 \preccurlyeq S_2$ is sufficient for S_2 to be a refinement of S_1, but it is not necessary.

We give a simple example of a sequence of refined experiments. For a behaviour $\mathfrak{B} \in \mathfrak{M}_W$, we start with $S_0 := \{\epsilon\} \in \mathfrak{E}_W$, and iteratively define a sequence of experiments S_j on $\mathfrak{B}$ as follows

$$S_{j+1} := \{s \in W^* | \exists \tilde{s} \in S_j : \tilde{s} \prec s \quad \text{and} \quad |s| = |\tilde{s}| + 1 \text{ and } s \in \mathfrak{B}|_{[0,|s|)}\} \tag{6}$$

Observe that, for each j, $S_j = \mathfrak{B}|_{[0,j)}$ and that $S_j \in \mathfrak{E}_W$ is indeed an experiment on $\mathfrak{B}$. Obviously, $S_j \preccurlyeq S_{j+1}$, and hence $\mathfrak{B} \subseteq \mathcal{M}^{\downarrow}(S_{j+1}) \subseteq \mathcal{M}^{\downarrow}(S_j)$, for all $j \in \mathbb{N}_0$. In fact, the generated sequence of models is identical to the so-called strongest l-complete approximation [7].

Consider the sequence of refinements defined by (6) in the context of the iterative synthesis procedure (S1)–(S4), Section 2, applied to the switched-server control problem, Section 3. With numerical values $\rho_{\text{out}} = 0.5$, $\rho_{\text{in}} = 1.125$, $\beta^- = 0.8$, $\beta^+ = 0.9$, the iteration terminates with success at $j = 9$, where the last experiment consists of 3330 relevant strings (not counting those that are in immediate conflict with the specification). Fig. 5 shows a closed-loop simulation for the initial state $x_0 = (0, 0)$.

From the simulation, it can be observed that it takes at most nine control events to drive a state into the region $[\beta^-, 1]^2$. Therefore, the fact that it is possible to drive any state to $[\beta^-, 1]^2$ will show in the abstraction only for $j \geq 9$. As this aspect of the actual plant dynamics is crucial for the synthesis of a supervisor that enforces the specification at hand, abstraction-based synthesis with a refinement scheme (6) cannot be successful for $j < 9$. The simulation also suggests that it takes only two control events to drive the state from within $[\beta^-, 1]^2$ to the region $[\beta^+, 1]^2$, and that the state, henceforth, can be kept within $[\beta^-, 1]^2$. The example demonstrates that the specification imposes different requirements on the accuracy of the abstraction for various aspects of plant dynamics. The refinement scheme (6), however, generates experiments with a uniform length of strings. Thus, in procedure (S1)–(S4) every string is extended to the length required in the worst case and, in this sense, the refinement is unfocused and global.

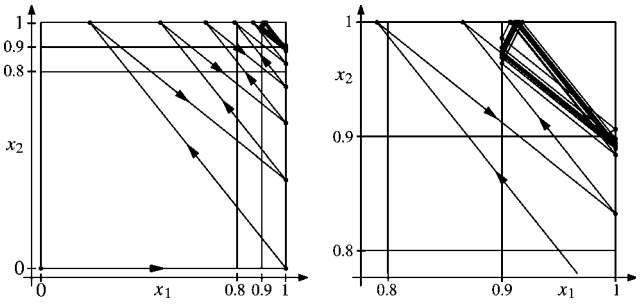

Fig. 5 *Closed-loop simulation*

5 Temporal decomposition of control tasks

We decompose the supervisory control task into two subproblems: a start-up control problem and a long-term control problem. The proposed decomposition is temporal in the sense that the start-up control problem imposes a specification only before a certain condition becomes true, whereas the long-term control problem imposes a specification only after a condition becomes true.

5.1 Start-up and long-term control problems

A start-up control problem asks for a controller that drives the plant into a certain mode of operation. Formally, we define such a problem as a specific supervisory control problem that requires closed-loop trajectories to evolve into a target set of finite strings. Once the closed loop has generated a string in the target set, a start-up control problem imposes no further requirements on the closed-loop trajectory.

Definition 9: Given a plant $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$, a specification $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$ and a target $T \subseteq W^*$, let

$$\mathfrak{B}^{\text{stc}}_{\text{spec}} := \{w \in W^{\mathbb{N}_0} | \exists l \in \mathbb{N}_0 : w|_{[0,l)} \in T \cap \mathfrak{B}_{\text{spec}}|_{[0,l)}\} \tag{7}$$

A supervisor $\mathfrak{B}^{\text{stc}}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$ is said to be a solution of the start-up control problem $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}}, T)_{\text{sp}}$, if $\mathfrak{B}^{\text{stc}}_{\text{sup}}$ solves the control problem $(\mathfrak{B}_{\text{p}}, \mathfrak{B}^{\text{stc}}_{\text{spec}})_{\text{cp}}$.

Viewed as temporal properties [23], $\mathfrak{B}^{\text{stc}}_{\text{spec}}$ is seen to express an eventuality or guarantee closed-loop property. Fig. 6 illustrates (in the context of control problems, we consider a signal space $W = U \times Y$ and expect $|W| \geq 4$. However, we continue our illustrations for $W = \{0, 1\}$, because larger alphabets cannot be given as explicit graphical representations) the start-up specification for the target $T = \{1, 01\}$, assuming that the two strings conform with $\mathfrak{B}_{\text{spec}}$. For the switched-server example, a start-up control problem may require the supervisor to drive the continuous

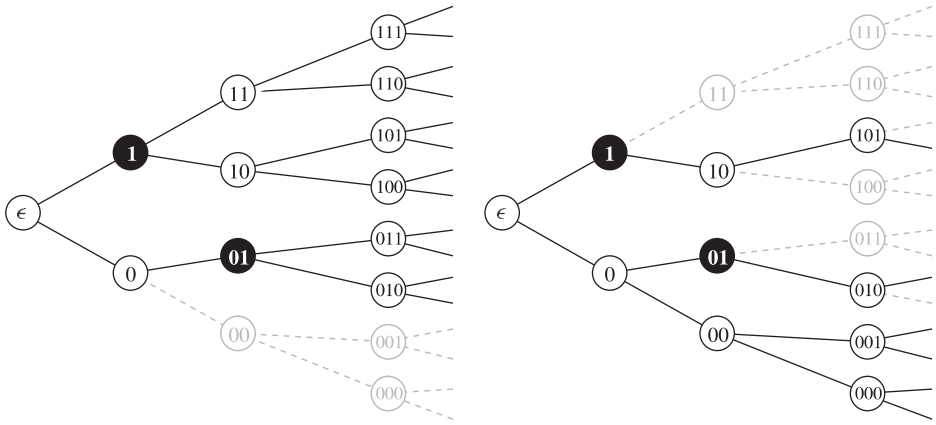

Fig. 6 *Specification $\mathfrak{B}^{stc}_{spec}$ and $\mathfrak{B}^{ltc}_{spec}$ for target T and domain D, respectively (black nodes)*

state into the region $[\beta^+, 1]^2$. This can be ensured by choosing $T = \{s(\mu, 1)|s \in W^*, \mu \in \{1, 2\}\}$.

The unique existence of a least restrictive solution to the start-up control problem follows from corollary 3. If T is bounded in length of its elements, then $\mathfrak{B}^{stc}_{spec}$ is complete. This case corresponds to targets T, which are experiments, and it implies that the least restrictive solution of $(\mathfrak{B}_p, \mathfrak{B}_{spec}, T)_{sp}$ is complete.

A long-term control problem asks for a controller that, once the closed loop has evolved into a specified domain, restricts all future evolution to a language specification. We define such a problem in terms of a specific supervisory control problem that builds on a conditional performance criterion. In case the closed loop does not evolve into the domain, a long-term control problem imposes no requirements on the closed-loop trajectory at all.

Definition 10: Given a plant $\mathfrak{B}_p \subseteq W^{\mathbb{N}_0}$, a specification $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$ and a domain $D \subseteq W^*$, let

$$\mathfrak{B}^{ltc}_{spec} := \{w \in W^{\mathbb{N}_0} | \forall l \in \mathbb{N}_0 : (w|_{[0,l)} \in D \Rightarrow w \in \mathfrak{B}_{spec})\} \quad (8)$$

A supervisor $\mathfrak{B}^{ltc}_{sup} \subseteq W^{\mathbb{N}_0}$ is said to be a solution of the long-term control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec}, D)_{lp}$ if $\mathfrak{B}^{ltc}_{sup}$ solves the control problem $(\mathfrak{B}_p, \mathfrak{B}^{ltc}_{spec})_{cp}$ and if $d \in (\mathfrak{B}_p \cap \mathfrak{B}^{ltc}_{sup})|_{[0,|d|)}$ for all $d \in D$.

Viewed as temporal properties [23], $\mathfrak{B}^{ltc}_{spec}$ is the union of the original $\mathfrak{B}_{spec}$ with a safety property (with respect to the complement of D). Fig. 6 illustrates the long-term specification for the domain $D = \{1, 01\}$, where it is assumed that $\mathfrak{B}_{spec}$ accepts precisely those trajectories in which the symbols '0' and '1' alternate. Note that any signal with prefix '00' does not evolve into D and, hence, will be in $\mathfrak{B}^{ltc}_{spec}$. For the switched-server example, a long-term control problem may require the supervisor to keep the continuous state within the region $[\beta^-, 1]^2$, once it has reached $[\beta^+, 1]^2$. This can be expressed by $D = \{s(\mu, 1)|s \in W^*, \mu \in \{1, 2\}\}$, $\mathfrak{B}_{spec} = \{(u, y)|\forall k: (1, 3) \neq (u(k), y(k)) \neq (2, 3)\}$.

The uniqueness of a least restrictive solution to any long-term control problem follows from two observations: (i) the solutions of $(\mathfrak{B}_p, \mathfrak{B}^{ltc}_{spec})_{cp}$ form a complete upper semilattice (i.e. closed under arbitrary unions); and (ii) the extra condition $d \in (\mathfrak{B}_p \cap \mathfrak{B}^{ltc}_{sup})|_{[0,|d|)}$ is retained under unions of long-term controllers. The latter non-triviality condition rules out the trivial solution $\mathfrak{B}^{ltc}_{sup} = \emptyset$ whenever the domain D is non-empty. Hence, for a long-term control problem, there is no general guarantee that a solution will exist. If a solution does exist, then the least restrictive solution of $(\mathfrak{B}_p, \mathfrak{B}^{ltc}_{spec})_{cp}$ is the least restrictive solution of $(\mathfrak{B}_p, \mathfrak{B}_{spec}, D)_{lp}$. As with the start-up problem, observe that $\mathfrak{B}^{ltc}_{spec}$ is complete whenever $\mathfrak{B}_{spec}$ is complete and D is an experiment. Consequently, we obtain completeness of the least restrictive solution of $(\mathfrak{B}_p, \mathfrak{B}_{spec}, D)_{lp}$.

The possibility of no solutions of the problem $(\mathfrak{B}_p, \mathfrak{B}_{spec}, D)_{lp}$ raises the question as to whether we can at least find a supremal subset $D_{max} \subseteq D$ on which a solution to $(\mathfrak{B}_p, \mathfrak{B}_{spec}, D_{max})_{lp}$ does exist. The answer is yes.

Proposition 11: Given a complete I/- behaviour $\mathfrak{B}_p \subseteq W^{\mathbb{N}_0}$, a complete specification $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$, and a prefix-free candidate domain $D \subseteq W^*$ that is an experiment. Then there exists a supremal (with respect to $\subseteq$) subset $D_{max} \subseteq D$ such that the long-term control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec}, D_{max})_{lp}$ exhibits a solution. We refer to a D_{max} as the maximal domain of $(\mathfrak{B}_p, \mathfrak{B}_{spec}, D)_{lp}$.

5.2 Temporal decomposition and composition of supervisors

A non-trivial solution to a supervisory control problem can be decomposed into start-up and long-term components where the target and the domain are an arbitrary experiment on the closed-loop behaviour:

Proposition 12: Given a plant $\mathfrak{B}_p \subseteq W^{\mathbb{N}_0}$ and a specification $\mathfrak{B}_{spec} \subseteq W^{\mathbb{N}_0}$, let $\mathfrak{B}_{sup}$ denote a non-trivial solution to the control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec})_{cp}$ with closed loop $\mathfrak{B}_{cl} = \mathfrak{B}_p \cap \mathfrak{B}_{sup}$. Let $S \subseteq W^*$ be an experiment on $\mathfrak{B}_{cl}$. Then $\mathfrak{B}_{sup}$ is a non-trivial solution of both the start-up control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec}, S)_{sp}$ and the long-term control problem $(\mathfrak{B}_p, \mathfrak{B}_{spec}, S)_{lp}$.

We now turn to the question of how to compose a start-up and a long-term controller to form an overall solution to a control problem. For the switched-server example, we expect that if a start-up controller can drive the closed-loop state trajectories to the region $[\beta^+, 1]^2$ and if a long-term controller can take care once $[\beta^+, 1]^2$ has been reached, we can combine the two controllers to form an overall solution to the control problem. To generalise this line of thought, we define two versions of an operator for the temporal composition of two controllers. In both cases, the switch from one controller to the other is triggered by the trajectory evolving into a certain switching set. For the first operator, the switching condition is tied to the time axis and must be satisfied from time $k = 0$ onwards.

Definition 13: Given two behaviours $\mathfrak{B}^{stc}_{sup}$, $\mathfrak{B}^{ltc}_{sup} \subseteq W^{\mathbb{N}_0}$ and a switching set $C \subseteq W^*$, the combined behaviour

$\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \wedge^{\circ}_{C} \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ is defined to be the set of all $w \in W^{\mathbb{N}_0}$ with either (i) or (ii):

(i) there exists an $l \in \mathbb{N}_0$ such that $w|_{[0,l)} \in \mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}}|_{[0,l)} \cap C$, and $w \in \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}$;
(ii) $w|_{[0,l)} \notin C$, for all $l \in \mathbb{N}_0$, and $w \in \mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}}$.

In contrast, our second operator resets time for the second controller when switching takes place, and here the switching condition does not depend on absolute time.

Definition 14: Given two behaviours $\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}}$, $\mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ and a switching set $C \subseteq W^*$, the combined behaviour $\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \wedge_{C} \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ is defined to be the set of all $w \in W^{\mathbb{N}_0}$ with either (i) or (ii):

(i) there exist $k, l \in \mathbb{N}_0$ such that $w|_{[0,k+l)} \in \mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}}|_{[0,k+l)}$, $\sigma^k w \in \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}$, $\sigma^k w|_{[0,l)} \in C$ and $\sigma^{\kappa} w|_{[0,\lambda)} \notin C$, for all κ, $\lambda \in \mathbb{N}_0$ where $\kappa + \lambda < k + l$;
(ii) $w \in \mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}}$ and $\sigma^{\kappa} w|_{[0,\lambda)} \notin C$, for all κ, $\lambda \in \mathbb{N}_0$.

By the following lemma, generic implementability and completeness are retained under our temporal compositions of supervisors. Consequently, the composition of two generically implementable complete supervisors is admissible to any complete I/- plant.

Lemma 15: Given $\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}}$ and $\mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}$, let $\mathfrak{B}_{\mathrm{sup}} := \mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \wedge_{C} \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}$ and $\mathfrak{B}^{\circ}_{\mathrm{sup}} := \mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \wedge^{\circ}_{C} \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}$. Let $C \subseteq W^*$. If $\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ and $\mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ are complete, then so are $\mathfrak{B}_{\mathrm{sup}}$ and $\mathfrak{B}^{\circ}_{\mathrm{sup}}$. If $\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$, $W = U \times Y$ and $\mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ are generically implementable, and it is the case that $c \in \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}|_{[0,|c|)}$ for all $c \in C$, then both $\mathfrak{B}_{\mathrm{sup}}$ and $\mathfrak{B}^{\circ}_{\mathrm{sup}}$ are generically implementable.

The composition operator $\wedge^{\circ}_{C}$ matches our definitions of the start-up and long-term control problems in that it allows for the composition of an overall solution, provided the target set of the start-up controller lies within the domain of the long-term controller.

Theorem 16: Given a plant $\mathfrak{B}_{\mathrm{p}} \subseteq W^{\mathbb{N}_0}$ and a specification $\mathfrak{B}_{\mathrm{spec}} \subseteq W^{\mathbb{N}_0}$, let $\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ be a non-trivial solution to the start-up control problem $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}}, T)_{\mathrm{sp}}$ for a target $T \subseteq W^*$. Furthermore, let $\mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}} \subseteq W^{\mathbb{N}_0}$ denote a solution to the long-term control problem $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}}, D)_{\mathrm{lp}}$ on a domain $D \subseteq W^*$. If $\mathfrak{B}_{\mathrm{p}}$ is a complete I/- behaviour, and if $\mathfrak{B}_{\mathrm{spec}}$, $\mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}}$ and $\mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}$ are all complete, and if $T \subseteq D$, then $\mathfrak{B}_{\mathrm{sup}} := \mathfrak{B}^{\mathrm{stc}}_{\mathrm{sup}} \wedge^{\circ}_{T} \mathfrak{B}^{\mathrm{ltc}}_{\mathrm{sup}}$ is a non-trivial solution to the control problem $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})_{\mathrm{cp}}$.

6 Strategic experiments

In abstraction-based supervisory controller synthesis, one alternates trial controller synthesis performed on an abstraction with abstraction refinement, until either synthesis succeeds or computational resources are exhausted; see Section 2, (S1)–(S4). We present an advanced version of the refinement step (S4), which uses diagnostic information from synthesis failure as obtained from a temporal decomposition of the synthesis problem. The abstraction procedure is based on experiments, and systematic refinement focuses on the complement of the maximum domain of the long-term control problem.

Consider the supervisory control problem $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})_{\mathrm{cp}}$. For the first part of our discussion, we suppose that we are given two experiments $\tilde{S} \preccurlyeq S$ on $\mathfrak{B}_{\mathrm{p}}$, where supervisory controller synthesis fails on the abstraction $\mathcal{M}^{\downarrow}(\tilde{S})$ but succeeds on $\mathcal{M}^{\downarrow}(S)$. We will construct a third experiment $\hat{S}$, with $\tilde{S} \preccurlyeq \hat{S} \preccurlyeq S$, that lies between the two given experiments and still leads to a model strong enough for successful synthesis. In the second part, we derive an algorithm for iterative refinement, which does not require knowledge of a successful experiment beforehand. Our argument is based on the following assumptions:

(A1) The plant $\mathfrak{B}_{\mathrm{p}} \in \mathfrak{N}_W$ is a complete I/- behaviour and the specification $\mathfrak{B}_{\mathrm{spec}} \subseteq W^{\mathbb{N}_0}$ is complete. This is merely a technical requirement to build our result on the previous propositions.
(A2) There exists an experiment $S \in \mathfrak{E}_W$ on $\mathfrak{B}_{\mathrm{p}}$ that is successful in the sense that $(\mathcal{M}^{\downarrow}(S), \mathfrak{B}_{\mathrm{spec}})_{\mathrm{cp}}$ exhibits a non-trivial solution. Implicitly, this requires the existence of a non-trivial solution of the original problem $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})_{\mathrm{cp}}$; see Theorem 4. Without further loss of generality, we may additionally assume $S = \mathfrak{B}_{\mathrm{p}}|_{[0,l)}$ for some $l \in \mathbb{N}_0$.
(A3) There exists an $l_{\mathrm{spec}} \in \mathbb{N}_0$, such that $w \in W^{\mathbb{N}_0}$, $k \in \mathbb{N}_0$, $w|_{[0,k+l_{\mathrm{spec}})} \in \mathfrak{B}_{\mathrm{spec}}|_{[0,k+l_{\mathrm{spec}})}$ and $\sigma^k w \in \mathfrak{B}_{\mathrm{spec}}$, implies that $w \in \mathfrak{B}_{\mathrm{spec}}$. Note that this condition is readily observed to hold for all l_{spec}-complete specifications $\mathfrak{B}_{\mathrm{spec}}$.
(A4) We are given a further experiment $\tilde{S} \in \mathfrak{E}_W$ on $\mathfrak{B}_{\mathrm{p}}$ such that $\tilde{S} \preccurlyeq S$, and there exists only the trivial solution to the control problem $(\mathcal{M}^{\downarrow}(\tilde{S}), \mathfrak{B}_{\mathrm{spec}})_{\mathrm{cp}}$. Furthermore, all strings $\tilde{s}$ are at least of length l_{spec}; that is, $l_{\mathrm{spec}} \leq \min\{|\tilde{s}| \mid \tilde{s} \in \tilde{S}\}$. Without loss of generality, we may additionally assume that $\tilde{S}$ is prefix-free, and that the shortest string in $\tilde{S}$ is of length l_{spec}.

We apply the temporal decomposition of Section 5 to the control problem, and choose $\tilde{S}$ both as target and domain set. In particular, the maximum domain $\tilde{D}_{\max} \subset \tilde{S}$ on which the long-term control problem $(\mathcal{M}^{\downarrow}(\tilde{S}), \mathfrak{B}_{\mathrm{spec}}, \tilde{S})_{\mathrm{lp}}$ has a solution must be a proper subset of $\tilde{S}$. The mismatch between $\tilde{S}$ and $\tilde{D}_{\max}$ quantifies that portion of the model $\mathcal{M}^{\downarrow}(\tilde{S})$, which requires refinement in order to solve the original control problem $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})_{\mathrm{cp}}$. Thus, we refine $\tilde{S}$ on the complement of $\tilde{D}_{\max}$

$$\hat{S} := \{s \in S | \exists \tilde{s} \in \tilde{S} \setminus \tilde{D}_{\max} : \ \tilde{s} \preccurlyeq s\} \cup \tilde{D}_{\max} \qquad (9)$$

see also Fig. 7. By the following lemma, $\hat{S}$ is indeed an experiment on $\mathfrak{B}_{\mathrm{p}}$, which leads to an abstraction between $\mathcal{M}^{\downarrow}(\tilde{S})$ and $\mathcal{M}^{\downarrow}(S)$.

Lemma 17: The above $\hat{S}$ is an experiment on $\mathfrak{B}_{\mathrm{p}}$. The strongest models $\mathcal{M}^{\downarrow}(S)$, $\mathcal{M}^{\downarrow}(\hat{S})$ and $\mathcal{M}^{\downarrow}(\tilde{S})$ are complete I/- behaviours, and $\mathcal{M}^{\downarrow}(S) \subseteq \mathcal{M}^{\downarrow}(\hat{S}) \subseteq \mathcal{M}^{\downarrow}(\tilde{S})$.

Theorem 18: Under assumptions (A1)–(A4), there exists a non-trivial solution for $(\mathcal{M}^{\downarrow}(\hat{S}), \mathfrak{B}_{\mathrm{spec}})_{\mathrm{cp}}$.

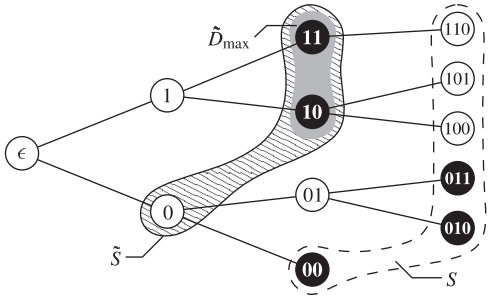

Fig. 7 *For given S, $\tilde{S}$ and $\tilde{D}_{max}$, the refinement $\hat{S}$ (black nodes) of $\tilde{S}$*

Proof (outline): Let D denote the set of strings $d \in S$ that survive in the closed-loop $\mathcal{M}^{\downarrow}(S) \cap \mathfrak{B}_{\text{sup}}$, that is, $D := \{d \in S | d \in \mathfrak{B}_{\text{sup}}|_{[0,|d|)}\}$. Consider $\hat{T} := \tilde{D}_{\max} \cup (D \cap \hat{S})$ as a target set. The claim can be established by showing that: (i) the start-up control problem $(\mathcal{M}^{\downarrow}(\hat{S}), \mathfrak{B}_{\text{spec}}, \hat{T})_{\text{sp}}$ exhibits a non-trivial solution; and (ii) the long-term control problem $(\mathcal{M}^{\downarrow}(\hat{S}), \mathfrak{B}_{\text{spec}}, \hat{T})_{\text{lp}}$ has a non-trivial solution. One then appeals to theorem 16. □

The definition of $\hat{S}$ in (9) is not constructive, as we cannot assume S to be known. However, the above theorem suggests the following abstraction-based synthesis procedure to solve the supervisory control problem $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})_{\text{cp}}$, subject to assumptions (A1)–(A3).

(R1) Let $j := l_{\text{spec}}$ and $S_j := \mathfrak{B}_{\text{p}}|_{[0,j)}$.
(R2) Compute the least restrictive solution $\mathfrak{B}^j_{\text{sup}}$ of $(\mathcal{M}^{\downarrow}(S_j), \mathfrak{B}_{\text{spec}})_{\text{cp}}$.
(R3) If $\mathcal{M}^{\downarrow}(S_j) \cap \mathfrak{B}^j_{\text{sup}} \neq \emptyset$, then terminate this iteration and return the solution $\mathfrak{B}^j_{\text{sup}}$.
(R4) Compute the maximum domain $\tilde{D}^j_{\max}$ of $(\mathcal{M}^{\downarrow}(S_j), \mathfrak{B}_{\text{spec}}, (S_j)_{\text{lp}}$ and refine the experiment S_j

$$S_{j+1} := \{s \in W^* | \exists \tilde{s} \in S_j \setminus \tilde{D}^j_{\max} : \tilde{s} \prec s, s \in \mathfrak{B}_{\text{p}}|_{[0,|\tilde{s}|+1)}\} \cup \tilde{D}^j_{\max} \quad (10)$$

then proceed with step (R2) for $j := j + 1$.

Note that the synthesis problem in (R2) and the maximum domain in (R4) are not stated for the underlying hybrid plant $\mathfrak{B}_{\text{p}}$, but rather for models from experiments. The latter, in the case of $|W| \in \mathbb{N}$, can be realised by finite automata, and we can perform the computations by highly efficient algorithms from DES theory. It is readily seen that all S_j are indeed experiments on $\mathfrak{B}_{\text{p}}$. Obviously, S_{j+1} is a refinement of S_j, where (10) points out the strategy of refinement, namely to focus on strings that do not lie in the maximal domain. Note, however, that the refinement of the experiment in (10) refers to the underlying hybrid plant $\mathfrak{B}_{\text{p}}$, and thus requires a reachability analysis over a continuous state space.

The procedure (R1)–(R4) systematically explores those aspects of the plant behaviour, which could not be resolved in the trial synthesis; that is, do not lie in the current $\tilde{D}^j_{\max}$. In Fig. 8, we illustrate the growing area of success $\tilde{D}^j_{\max}$ as well as the fact that no further experiments need to be conducted regarding the strings that already have a prefix in $\tilde{D}^j_{\max}$. By (A2), we have assumed that the control problem can be solved via a model based on some experiment. Therefore we expect that our iteration finds such a model. This is indeed the case.

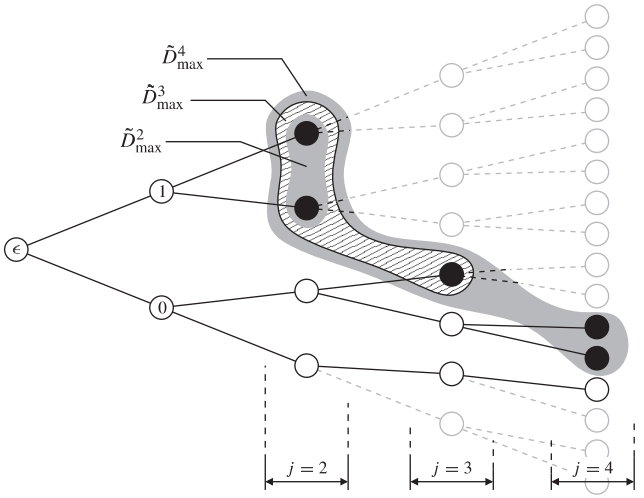

Fig. 8 *Procedure (R1)–(R4) for $j \in \{2, 3, 4\}$*

Theorem 19: Under assumptions (A1)–(A3), the iteration (R1)–(R4) terminates after a finite number of steps.

Proof: Suppose the procedure fails to terminate in a finite number of steps. Let $\tilde{S} := S_{l_{\text{spec}}}$ and observe that this choice satisfies assumption (A4). We construct $\hat{S}$ according to (9). By theorem 18, there exists a solution to $(\mathcal{M}^{\downarrow}(\hat{S}), \mathfrak{B}_{\text{spec}})_{\text{cp}}$. From $S_{j+1} \supseteq \tilde{D}^j_{\max}$ and $\mathcal{M}^{\downarrow}(S_{j+1}) \subseteq \mathcal{M}^{\downarrow}(S_j)$, we obtain $\tilde{D}^{j+1}_{\max} \supseteq \tilde{D}^j_{\max}$, and the minimum length of strings in $S_j \backslash \tilde{D}^j_{\max}$ is strictly increasing as a function of j. Hence, for some j and all $\tilde{s} \in S_j \backslash \tilde{D}^j_{\max}$ there exists an $s \in S$ with $s \preccurlyeq \tilde{s}$. In particular, S_j is a refinement of $\hat{S}$. Hence by theorem 4, the least restrictive solution of $(\mathcal{M}^{\downarrow}(S_j), \mathfrak{B}_{\text{spec}})_{\text{cp}}$ is non-trivial. This contradicts the assumption. Thus, the iteration must terminate after a finite number of steps. □

For the switched-server example from Section 3, the proposed refinement procedure (R1)–(R4) terminates successfully at $j = 9$, where we use the same parameters as in Section 4. In contrast to the 3330 strings when using the uniform refinement in Section 4, here the successful experiment consists of only 158 strings.

Although conditions (A2) and (A3) hold for the example, we have already noted that neither the external plant $\mathfrak{B}_{\text{p}}$ nor the specification $\mathfrak{B}_{\text{spec}}$ are complete. So is it by luck that the algorithm terminates with success? Dealing first with the incompleteness of the plant, we formally substitute $\mathfrak{B}_{\text{p}}$ with its completion $\mathfrak{B}'_{\text{p}} := \{w \in W^{\mathbb{N}_0} | \forall k \in \mathbb{N}_0 : w|_{[0,k)} \in \mathfrak{B}_{\text{p}}|_{[0,k)}\} \supseteq \mathfrak{B}_{\text{p}}$. Note that all possible experiments on $\mathfrak{B}'_{\text{p}}$ are the same as those on $\mathfrak{B}_{\text{p}}$, and, hence, the substitution has no effect on the actual procedure (R1)–(R4). In the case of successful termination, we are provided a solution to $(\mathcal{M}^{\downarrow}(S_j), \mathfrak{B}_{\text{spec}})_{\text{cp}}$. To see that this solution carries over not only to $\mathfrak{B}'_{\text{p}}$ but also to $\mathfrak{B}_{\text{p}}$, we refer to a variant of theorem 4 for state machines [8]. Regarding incompleteness of $\mathfrak{B}_{\text{spec}}$, we argue that the existence of a complete solution $\mathfrak{B}_{\text{sup}}$ to $(\mathfrak{B}'_{\text{p}}, \mathfrak{B}_{\text{spec}})_{\text{cp}}$ can serve as a formally weaker sufficient condition: in that case, the closed loop $\mathfrak{B}'_{\text{p}} \cap \mathfrak{B}_{\text{sup}}$ can serve as a more restrictive, but complete, specification $\mathfrak{B}'_{\text{spec}}$. Thus, we can invoke theorem 18 for the control problem $(\mathfrak{B}'_{\text{p}}, \mathfrak{B}'_{\text{spec}})_{\text{cp}}$ and follow the argument in the proof of theorem 19 to guarantee successful termination of (R1)–(R4).

7 Conclusions

In most abstraction-based approaches to supervisory controller synthesis for hybrid systems, the bulk of the computational effort lies in reachability analysis for restricted flow relations, which is used to construct an abstraction, or to refine a given abstraction. Compared with the computational cost of these experiments conducted on a hybrid system, the effort required for the actual synthesis is negligible. We decompose the control problem into a start-up aspect and a long-term aspect, and thereby extract reasons for a failure in the controller synthesis for an individual abstraction. From this diagnostic information, we systematically avoid expensive experiments that are irrelevant to the particular synthesis task at hand. By theorem 19, we show that there is no loss in our divide-and-conquer strategy: if the synthesis of a non-trivial solution supervisor can be

based on some experiment, our method will detect such an experiment.

8 References

1 Cury, J.E.R., Krogh, B.A., and Niinomi, T.: 'Synthesis of supervisory controllers for hybrid systems based on approximating automata', *IEEE Trans. Autom. Control*. Special issue on Hybrid Systems, 1998, vol. 43, pp. 564–568

2 Koutsoukos, X., Antsaklis, P.J., Stiver, J.A., and Lemmon, M.D.: 'Supervisory control of hybrid systems', *Proc. IEEE*, 2000, **88**, pp. 1026–1049

3 Alur, R., Henzinger, T.A., Lafferriere, G., and Pappas, G.: 'Discrete abstractions of hybrid systems', *Proc. IEEE*, 2000, **88**, pp. 971–984

4 Henzinger, T.A.: 'The theory of hybrid automata'. Proc. 11th Annual IEEE Symp. (LICS'96), (IEEE Computer Society Press, 1996), pp. 278–292

5 Willems, J.C.: 'Models for dynamics', *Dyn. Rep.*, 1989, **2**, pp. 172–269

6 Willems, J.C.: 'Paradigms and puzzles in the theory of dynamic systems', *IEEE Trans. Autom. Control*, 1991, **36**, pp. 258–294

7 Moor, T., and Raisch, J.: 'Supervisory control of hybrid systems within a behavioural framework', *Syst. Control Lett.*, 1999, **38**, pp. 157–166

8 Moor, T., Raisch, J., and O'Young, S.D.: 'Discrete supervisory control of hybrid systems based on *l*-complete approximations', *J. Discrete Event Dyn. Syst.*, 2002, **12**, pp. 83–107

9 Moor, T., Davoren, J.M., and Anderson, B.D.O.: 'Robust hybrid control from a behavioural perspective'. IEEE Proc. 41st Int. Conf. on Decision and Control, 2002, pp. 1169–1174

10 Ramadge, P.J., and Wonham, W.M.: 'Supervisory control of a class of discrete event systems', *SIAM J. Control Optim.*, 1987, **25**, pp. 206–230

11 Ramadge, P.J., and Wonham, W.M.: 'The control of discrete event systems', *Proc. IEEE*, 1989, **77**, pp. 81–98

12 Davoren, J.M., Moor, T., and Nerode, A.: 'Hybrid control loops, a/d maps, and dynamic specifications' in Tomlin, C.J., and Greenstreet, M.R. (Eds.): 'Hybrid systems: computation and control (HSCC'02)' (Springer-Verlag, 2002, LNCS 2289), pp. 149–163

13 Krogh, B.A., and Chutinan, A.: 'Hybrid systems: modeling and supervisory control' in Frank, P.M. (Ed.): 'Advances in control, highlights of ECC'99' (Springer-Verlag, 1999), pp. 228–246

14 Lunze, J., Nixdorf, B., and Richter, H.: 'Hybrid modelling of continuous-variable systems with application to supervisory control'. Proc. European Control Conf., 1997

15 Philips, P., Weiss, M., and Preisig, H.A.: 'Control based on discrete-event models of continuous systems'. Proc. European Control Conf., 1999

16 Moor, T., Davoren, J.M., and Raisch, J.: 'Strategic refinements in abstraction based supervisory control of hybrid systems'. Proc. 6th Int. Workshop on Discrete Event Systems (WODES), Zaragoza, Spain, 2002, pp. 329–334

17 Stursberg, O., Fehnker, A., Han, Z., and Krogh, B.H.: 'Specification-guided analysis of hybrid systems using a hierarchy of validation methods'. Proc. IFAC Conf. on the Analysis and Design of Hybrid Systems (ADHS'03), 2003

18 Clarke, E.M., Fehnker, A., Han, Z., Krogh, B.J., Ouaknine, J., Stursberg, O., and Theobald, M.: 'Abstraction and counterexample-guided refinement of hybrid systems', *Int. J. Found. Comput. Sci.*, 2003, **14**, pp. 583–604

19 Angulin, A., and Smith, C.H.: 'Inductive inference: theory and methods', *Comput. Surv.*, 1983, **15**, (3), pp. 237–269

20 Moor, T., and Raisch, J.: 'Think continuous, act discrete: DES techniques for continuous systems'. Proc. 10th Mediterranean Conf. on Control and Automation, 2002

21 Moor, T., and Raisch, J.: 'Abstraction based supervisory controller synthesis for high order monotone continuous systems' in Engell, S., Frehse, G., and Schnieder, E. (Eds.): 'Modelling, analysis and design of hybrid systems' (Springer, 2002, LNCIS 279), pp. 247–265

22 Chase, C., Serano, J., and Ramadge, P.: 'Periodicity and chaos from switched flow systems: contrasting examples of discretely controlled continuous systems', *IEEE Trans. Autom. Control*, 1993, **38**, pp. 70–83

23 Baier, C., and Kwiatkowska, M.Z.: 'On topological hierarchies of temporal properties', *Fundam. Inform.*, 2000, **41**, pp. 259–294