Discrete Event Dynamic Systems (2020) 30:533–560
https://doi.org/10.1007/s10626-020-00315-z

Refinements of behavioural abstractions for the supervisory control of hybrid systems

Jung-Min Yang[1] · Thomas Moor[2] · Jörg Raisch[3]

Received: 7 December 2018 / Accepted: 4 March 2020 / Published online: 22 April 2020

Abstract

A common approach to controller synthesis for hybrid systems is to first establish a discrete-event abstraction and then to use methods from supervisory control theory to synthesise a controller. In this paper, we consider behavioural abstractions of hybrid systems with a prescribed discrete-event input/output interface. We discuss a family of abstractions based on so called experiments which consist of samples from the external behaviour of the hybrid system. The special feature of our setting is that the accuracy of the abstraction can be carefully adapted to suit the particular control problem at hand. Technically, this is implemented as an iteration in which we alternate trial control synthesis with abstraction refinement. While localising refinement to where it is intuitively needed, we can still formally establish that the overall iteration will solve the control problem, provided that an abstraction-based solution exists at all.

Keywords Hybrid systems · Behavioural abstractions · l-complete approximations · Local refinement · Supervisory control

This article belongs to the Topical Collection: *Topical Collection on Theory-2020*
Guest Editors: Francesco Basile, Jan Komenda, and Christoforos Hadjicostis

The research of J.–M. Yang was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF–2018R1D1A1A09082016), and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF–2018R1A5A1025137 and NRF–2015M3A9A7067220).

✉ Jörg Raisch
raisch@control.tu-berlin.de

Jung–Min Yang
jmyang@ee.knu.ac.kr

Thomas Moor
lrt@fau.de

[1] School of Electronics Engineering, Kyungpook National University, 80 Daehakro, Bukgu, Daegu, 41566, Republic of Korea

[2] Lehrstuhl für Regelungstechnik, Friedrich-Alexander-Universität Erlangen-Nürnberg, 91058, Erlangen, Germany

[3] Fachgebiet Regelungssysteme, Technische Universität Berlin, 10587, Berlin, Germany

1 Introduction

The analysis and the control of hybrid systems have become an important subject in modern control theory; see, e.g., Alur et al. (2000) and Tabuada (2009) and the references cited therein. A common approach is to construct a finite-state abstraction of the hybrid system under consideration and then to apply methods known from the domain of discrete-event systems, most notably model checking, reactive synthesis, or supervisory control.

A well established framework to obtain a finite-state abstraction is to strategically construct a finite partition or a finite cover on the continuous state space and to thereby define symbolic dynamics associated with the hybrid system; see e.g. Stiver et al. (1995), Tabuada (2009), Reissig et al. (2017), Pola and Tabuada (2009), Gol et al. (2014), Zamani et al. (2012), and Liu and Ozay (2016). For controller synthesis, this approach is particularly suited when the design of a discrete-event interface is considered part of the synthesis problem. In contrast, if the hybrid plant is equipped with a prescribed discrete-event interface, so called *behavioural abstractions* are an adequate alternative. In this approach, one seeks to derive a finite-state abstraction directly in terms of the external signals. This is the situation we study in the present paper. [1]

The behavioural abstractions proposed by Moor and Raisch (1999) and Moor et al. (2002) are based on the notion of l*-completeness* from Willems' behavioural systems theory; see, e.g., Willems (1991). By definition, a discrete-time system is l-complete if its infinite-time behaviour can be exactly recovered from all length l samples, $l \in \mathbb{N}_0$, taken from all infinite-length signals. For a system which does not exhibit this property, the *strongest* l*-complete approximation* is then introduced as the tightest behavioural over-approximation that is l-complete. In the following, whenever clear from the context, we simply refer to l-complete approximations when we mean strongest l-complete approximations. An l-complete approximation can be obtained by exhaustively taking samples of length l from the original behaviour and generating the abstraction by superposition of these samples. For the control of hybrid systems with discrete-valued input- and output-signals, l-complete approximations can be used to synthesise controllers that address inclusion-type specifications in these signals. In this situation, the finite external signal range of the hybrid system leads to a finite-state realisation of the l-complete approximation and a variant of Ramadge and Wonham's supervisory control theory (Ramadge and Wonham 1987, 1989) is subsequently applied to synthesise a supervisory controller. It is shown by Moor and Raisch (1999) and Moor et al. (2002) that if the supervisor suitably restricts the behaviour of the l-complete approximation, it also accomplishes the control objective for the underlying hybrid system. The applicability of this approach has been demonstrated with case studies in the area of process engineering, including the start-up of a distillation column (Moor and Raisch 2002). Recent methodological extensions have been reported by Schmuck and Raisch (2014), Park and Raisch (2015), and Moor and Götz (2018) to address time-variant systems and partial observation.

The existence of an appropriate supervisor depends on the approximation accuracy, namely, if the abstract model is too *coarse*, no supervisor may exist that meets the specification. This leads to an iteration of trial synthesis and abstraction refinement, until either a solution to the control problem is established or computational resources are exhausted. Considering l-complete approximations, the construction of a *finer* abstraction effectively amounts to incrementing the length l of samples taken from the original hybrid system.

[1] A technical comparison of abstractions by symbolic dynamics and behavioural abstractions is given in Schmuck et al. (2015).

This can be done uniformly for all samples as, e.g., proposed by Moor and Raisch (1999) or, more efficiently, in a non-uniform way tailored for the particular control problem at hand. For abstractions by symbolic dynamics, Clarke et al. (2003) introduce *counterexample guided refinement* for the verification of hybrid systems, with a further development to address synthesis by Stursberg (2006). For behavioural abstractions, Moor et al. (2006) introduce the notion of an *experiment* as a set of non-uniform length samples taken from the original behaviour, with a subsequent discussion that leads to abstractions obtained from experiments. Technically, the resulting abstractions are still l-complete and, hence, they can be safely utilised in an abstraction-based design.

In the present paper, we further develop abstraction-based synthesis by experiments on behaviours. While the study by Moor et al. (2006) is entirely set within Willems' behavioural systems theory, we now make use of explicit state machine realisations reported by Moor and Götz (2018). By a more detailed discussion, we gain some relevant benefits. First, we can literally refer to supervisory control of sequential behaviours with a synthesis algorithm given by Thistle and Wonham (1994a). As a consequence, we can address more general liveness specifications with eventual task completion as a prototypical example. This extends the results from Moor et al. (2006), which are restricted to l-complete safety specifications. Second, the technically involved temporal decomposition of the control problem used by Moor et al. (2006) to guided abstraction refinement can now be replaced by the *controllability prefix* introduced by Thistle and Wonham (1994b). The latter is an intermediate result of the synthesis procedure and characterises *winning states*, from which on the control objective can be accomplished. In the case that the synthesis procedure fails and, thus, a refinement of the abstraction is required, intuitively, such a refinement does not need to address any winning states. Likewise, we can identify *failing states*, from which on no supervisor can possibly satisfy the specification. Again intuitively, this status is retained under refinement and, thus, no refinement should address the behaviour after a failing state. For the iteration of trial controller synthesis and abstraction refinement, it is therefore proposed to refine the experiment only by addressing states that are neither winning nor failing. Since this iteration intentionally generates only specific experiments, it may fail to generate a particular experiment for which controller synthesis would succeed. Here, our main technical result, Theorem 17, guarantees that a successful experiment will be generated provided that such a one exists.

A predecessor of this paper has been presented at the Workshop on Discrete Event Systems; see Yang et al. (2018). The present version has been extended (a) to address iterative refinements in contrast to a single local refinement, (b) to allow for specifications with a Büchi acceptance condition, and (c) to include a formal proof of our main technical result. The remainder of the paper is organised as follows. After introducing elementary notation in Section 2, we summarise the concept of experiments and the behavioural abstractions obtained therefrom in Section 3. Realisations of the hybrid plant and its abstractions are discussed in Section 4. In Section 5 we present the control problem under consideration and derive an abstraction-based solution procedure. Finally, we discuss the proposed abstraction refinement scheme in Section 6. A simple example is used throughout the entire discussion, illustrating the suggested ideas and demonstrating the applicability of the proposed strategy.

2 Notation

We denote the positive, respectively non-negative, integers by $\mathbb{N}$, respectively $\mathbb{N}_0$. The cardinality of a finite set A is denoted by $|A| \in \mathbb{N}_0$.

Given a set W, referred to as a *signal range*, and $l \in \mathbb{N}$, we denote by $W^l := \{\langle w_1, \ldots, w_l\rangle | \forall k, 1 \leq k \leq l : w_k \in W\}$ the set of *sequences over W of length l* and we let $W^+ := \cup\{W^l | l \in \mathbb{N}\}$. Introducing the *empty sequence* $\epsilon \notin W$, we formally define $W^0 := \{\epsilon\}$ and write $W^* := \cup\{W^l | l \in \mathbb{N}_0\} = \{\epsilon\} \cup W^+$ for the set of finite sequences over W. For a sequence $s \in W^l \subseteq W^*$, its length l is denoted $|s|$. The set of all countably *infinite sequences over W is denoted* $W^{\mathbb{N}_0}$, with $\mathbf{w} \in W^{\mathbb{N}_0}$ commonly interpreted as a *discrete time signal* $\mathbf{w} : \mathbb{N}_0 \to W$. Subsets $S \subseteq W^*$ are referred to as $*$*-languages*, or *languages of finite words*, in contrast to ω*-languages* $\mathcal{B} \subseteq W^{\mathbb{N}_0}$, or *languages of infinite words*.

For two finite sequences $s = \langle w_1, \ldots, w_l\rangle \in W^l$ and $r = \langle u_1, \ldots, u_n\rangle \in W^n$, the *concatenation* is defined by $\langle s, r\rangle := \langle w_1, \ldots, w_l, u_1, \ldots, u_n\rangle \in W^{l+n}$. For the empty sequence, let $\langle s, \epsilon\rangle := s =: \langle \epsilon, s\rangle$. The concatenation of the finite sequence $s = \langle w_1, \ldots, w_l\rangle \in W^l$ with a signal $\mathbf{w} \in W^{\mathbb{N}_0}$ is denoted $\mathbf{v} = \langle s, \mathbf{w}\rangle \in W^{\mathbb{N}_0}$, with $\mathbf{v}(k) = w_{k+1}$ for $0 \leq k < l$ and $\mathbf{v}(k) = \mathbf{w}(k - l)$ for $k \geq l$. Again, for the empty sequence let $\langle \epsilon, \mathbf{w}\rangle := \mathbf{w}$. For notational convenience, we write $\langle \cdot, \cdot, \cdot\rangle$ for $\langle \cdot, \langle \cdot, \cdot\rangle\rangle$.

A sequence $r \in W^*$ is a *prefix* of $s \in W^*$ if there exists $t \in W^*$ such that $\langle r, t\rangle = s$; we then write $r \leq s$. If, in addition, $r \neq s$, we say that r is a *strict prefix* of s and write $r < s$. Likewise, $r \in W^*$ is a prefix of a signal $\mathbf{w} \in W^{\mathbb{N}_0}$, if there exits $\mathbf{v} \in W^{\mathbb{N}_0}$ such that $\langle r, \mathbf{v}\rangle = \mathbf{w}$; we then write $r < \mathbf{w}$. The set of all prefixes of a given sequence $s \in W^*$ or a given signal $\mathbf{w} \in W^{\mathbb{N}_0}$ is denoted $\operatorname{pre} s \subseteq W^*$ or $\operatorname{pre} \mathbf{w} \subseteq W^*$, respectively. A sequence $t \in W^*$ is a *suffix* of $s \in W^*$ if there exists $r \in W^*$ such that $\langle r, t\rangle = s$.

The *left-shift operator* σ^l, $l \in \mathbb{N}_0$, is defined for signals $\mathbf{w} \in W^{\mathbb{N}_0}$ by $\sigma^l \mathbf{w} \in W^{\mathbb{N}_0}$ with $(\sigma^l \mathbf{w})(k) := \mathbf{w}(k + l)$ for all $k \in \mathbb{N}_0$, and we let $\sigma := \sigma^1$. For a signal $\mathbf{w} \in W^{\mathbb{N}_0}$, the *restriction* to a finite integer interval $D \subseteq \mathbb{N}_0$ is denoted $\mathbf{w}|_D$ with, e.g., $D = [k_1, k_2) := \{k \in \mathbb{N}_0 | k_1 \leq k < k_2\}$ and left-open and/or right-closed intervals defined likewise. When taking restrictions, we drop absolute time and reinterpret $\mathbf{w}|_D$ as finite sequence, i.e., we identify $\mathbf{w}|_{[k_1,k_2)}$ with $\langle \mathbf{w}(k_1), \ldots, \mathbf{w}(k_2 - 1)\rangle \in W^{k_2-k_1}$.

Taking point-wise images, all operators and maps in this paper are identified with their respective extension to set-valued arguments; e.g., we write $\sigma\mathcal{B}$ for the image $\{\sigma\mathbf{w} | \mathbf{w} \in \mathcal{B}\}$ of the ω-language $\mathcal{B} \subseteq W^{\mathbb{N}_0}$ under the operator σ with domain $W^{\mathbb{N}_0}$, and, likewise, $\operatorname{pre}\mathcal{B}$ for the image $\{\operatorname{pre}\mathbf{w} | \mathbf{w} \in \mathcal{B}\}$ of $\mathcal{B}$ under the operator pre.

3 Behavioural abstractions

We present a general scheme of system abstraction that allows for a guided refinement, and we do so within Willems' behavioural systems theory. In this framework, a dynamical system is defined as a triple $\Sigma = (T, W, \mathcal{B})$, where T is the *time axis*, W is the *external signal range*, and $\mathcal{B} \subseteq W^T := \{\mathbf{w} | \mathbf{w} : T \to W\}$ is the *behaviour*, i.e., the set of all external signals that the system may generate. It is then proposed to discuss and to categorise dynamical systems in terms of their behaviours. For the present paper, we focus attention on time-invariant systems with an external discrete-event interface. Technically, we consider the discrete time axis $T = \mathbb{N}_0$ and the finite external signal range W, $|W| \in \mathbb{N}$, and we interpret ω-languages $\mathcal{B} \subseteq W^{\mathbb{N}_0}$ as behaviours. Regarding time invariance, we refer to the following definition from Willems (1991).

Definition 1 A behaviour $\mathcal{B} \subseteq W^{\mathbb{N}_0}$ is *time invariant* if $\sigma\mathcal{B} \subseteq \mathcal{B}$. A system $\Sigma = (\mathbb{N}_0, W, \mathcal{B})$ is *time invariant* if its behaviour $\mathcal{B}$ is time invariant.

Given a system $\Sigma = (\mathbb{N}_0, W, \mathcal{B})$, a *behavioural abstraction* is a system $\Sigma' = (\mathbb{N}_0, W, \mathcal{B}')$ with $\mathcal{B} \subseteq \mathcal{B}'$, i.e., provided that the original system Σ accounts for all possible trajectories the actual phenomenon modelled by Σ can generate, then so does the abstraction Σ'. Behavioural abstractions are commonly used for the verification and the synthesis of safety properties, with optional liveness properties being addressed by additional structural requirements. Considering a time invariant system, we ask for a time invariant system abstraction. We refer to Moor et al. (2006) for the following notion of an *experiment* which we will use to construct a rich family of time-invariant behavioural abstractions.

Definition 2 An *experiment over* W is a $*$-language $S \subseteq W^*$. If there exists a uniform upper bound on the length of all sequences in S, we say that S is *of bounded length*. Moreover, $S \subseteq W^*$ is an *experiment on a behaviour* $\mathcal{B} \subseteq W^{\mathbb{N}_0}$ if S accounts for each signal from $\mathcal{B}$ in terms of a prefix, i.e., if $(\text{pre}\,\mathbf{w}) \cap S \neq \emptyset$ for all $\mathbf{w} \in \mathcal{B}$.

Given an experiment S on some behaviour $\mathcal{B}$, Moor et al. (2006) discuss abstractions $\mathcal{B}_S$, $\mathcal{B} \subseteq \mathcal{B}_S$, that can be obtained exclusively from S. To this end, we consider the candidate

$$\mathcal{B}_S := \{\mathbf{w} : \mathbb{N}_0 \to W \,|\, \forall k \in \mathbb{N}_0 \exists l \in \mathbb{N}_0 : \mathbf{w}|_{[k,k+l)} \in S\}, \tag{1}$$

i.e., $\mathcal{B}_S$ consists of all those trajectories which at any instance of time continue to evolve on some finite future sequence that matches S; see Fig. 1.

It is immediate from the construction that $\mathcal{B}_S$ is time invariant and that S is an experiment on $\mathcal{B}_S$. Moreover, for any behaviour $\tilde{\mathcal{B}} \subseteq W^{\mathbb{N}_0}$ we have the following implication:

$$\text{if } \tilde{\mathcal{B}} \text{ is time-invariant and if } S \text{ is an experiment on } \tilde{\mathcal{B}} \text{ then } \tilde{\mathcal{B}} \subseteq \mathcal{B}_S. \tag{2}$$

It is shown in Moor et al. (2006), Proposition 7, that $\mathcal{B}_S$ is the unique smallest behaviour for which the above implication holds. Assuming that the original behaviour $\mathcal{B}$ is indeed time invariant, $\mathcal{B}_S$ is the smallest superset of $\mathcal{B}$ that can be characterised exclusively in terms of

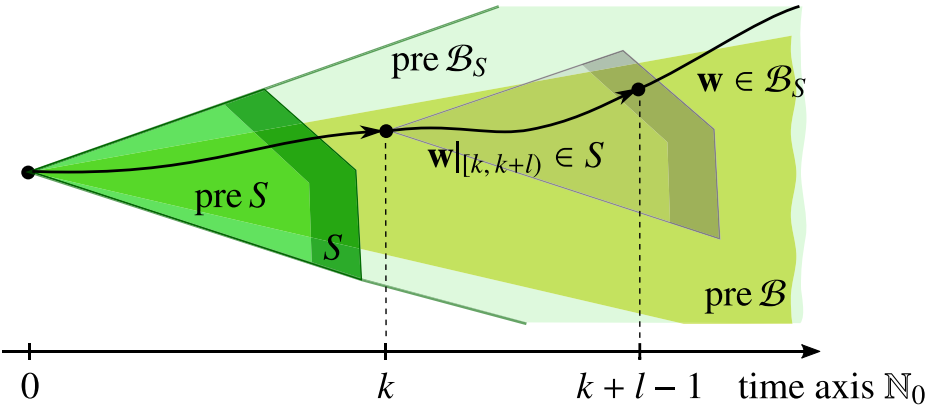

Fig. 1 Experiment S on $\mathcal{B}$ with associated abstraction $\mathcal{B}_S$. Here, the prefixes of behaviours are interpreted as infinite computational trees with the empty string as the root at the very left and progress of discrete time towards the right. In the sketch, the trees are abstractly shown as cones; i.e., pre $\mathcal{B}$ in dark yellow and pre $\mathcal{B}_S$ in light green. The bounded-length $*$-language S is shown as a dark green stripe. In particular, the sketch indicates that any signal from $\mathcal{B}$ must pass S, as required by Definition 3. Regarding the abstraction, the signal $\mathbf{w} \in \mathcal{B}_S$ shown in the sketch at every instance of time k must have a finite-length future that matches S; see Eq. 1. This is illustrated by the transparent grey copy of S shifted such that the root matches $\mathbf{w}(k)$ to indicate that $\mathbf{w}|_{[k,k+l)} = \mathbf{w}|_{[k,k+l-1]} \in S$

the experiment S. Therefore the system $\Sigma_S = (\mathbb{N}_0, W, \mathcal{B}_S)$ is referred to as *the behavioural abstraction obtained from S under the assumption of time invariance*, or in short as *the abstraction obtained from S*.

Moor et al. (2006) and Moor and Götz (2018) provide a comprehensive discussion regarding algebraic properties of experiments and the abstractions obtained therefrom. For the present paper, we pragmatically refer to Eq. 1 as the defining equation and point out some technical consequences relevant for the subsequent discussion. It is immediate from Eq. 1 that, regarding the associated abstraction, we may without loss of generality restrict our considerations to *prefix-free* experiments, i.e., to experiments that satisfy

$$\forall s, t \in S\colon s \leq t \;\;\Rightarrow\;\; s = t. \tag{3}$$

A prefix-free experiment S is commonly interpreted as a tree with *root* $\epsilon \in \operatorname{pre} S$, *nodes* $\operatorname{pre} S$ and *leaves* S. Likewise, it is not restrictive to assume that the experiment is *trim* in the sense that every sequence $s \in S$ contributes to the abstraction, i.e.,

$$\forall s \in S\, \exists k, l \in \mathbb{N}_0\colon s \in \mathcal{B}_S|_{[k,k+l)}, \tag{4}$$

or, equivalently, $S \subseteq \operatorname{pre} \mathcal{B}_S$. Thus, whenever convenient, we may restrict our discussion to trim and prefix-free experiments.

For the special case of the experiment $S = \mathcal{B}|_{[0,l]}$ with samples with uniform length $l+1$, $l \in \mathbb{N}_0$, the abstraction $\mathcal{B}_S$ obtained from S amounts to the *strongest l-complete approximation* proposed by Moor and Raisch (1999) and Moor et al. (2002). The approximation-refinement scheme known from l-complete approximations amounts to incrementing the sample length. This concept of refinement generalises to experiments as follows.

Definition 3 Given two experiments S and S' over W, we say that S' is a refinement of S if

$$(\forall s \in S\, \exists s' \in S'\colon s \leq s') \;\text{ and }\; (\forall s' \in S'\, \exists s \in S\colon s \leq s'). \tag{5}$$

We then write $S \leq S'$.

The first conjunct in Eq. 5 ensures that the refinement accounts for each sample $s \in S$ from the original experiment by an extended sample $s' \in S'$, $s \leq s'$, including the trivial case of $s = s'$. The second conjunct ensures that no other samples are in the refinement than those obtained by (possibly trivially) extending samples from the original experiment. Figure 2 illustrates a prefix-free experiment S on $\mathcal{B}$ where the leaves $s \in S$ form a "barrier" through which each trajectory from $\mathbf{w} \in \mathcal{B}$ must pass. In this view, a prefix-free refinement S' of S is obtained by pushing the "barrier" to the right. The figure also indicates that a refinement is expected to lead to a tighter abstraction in that it more accurately encodes which trajectories are not within $\mathcal{B}$, e.g, $\mathbf{v} \notin \mathcal{B}$ is possibly in $\mathcal{B}_S$ but cannot be in $\mathcal{B}_{S'}$. Technically, we obtain for two experiments S and S' on $\mathcal{B}$ with $S \leq S'$ that

$$\mathcal{B} \subseteq \mathcal{B}_{S'} \subseteq \mathcal{B}_S \tag{6}$$

as an immediate consequence of Eq. 1 and the second conjunct in Eq. 5; i.e., it is guaranteed that the abstraction does not become worse and we may optimistically expect it to become better.

For the systematic construction of refinements, we propose to nominate a set $R \subseteq S$ of *refinement candidates* and observe that

$$S' := \{s \in S | s \notin R\} \cup \{\langle s, w\rangle | s \in R, w \in W, \langle s, w\rangle \in \operatorname{pre} \mathcal{B}\} \tag{7}$$

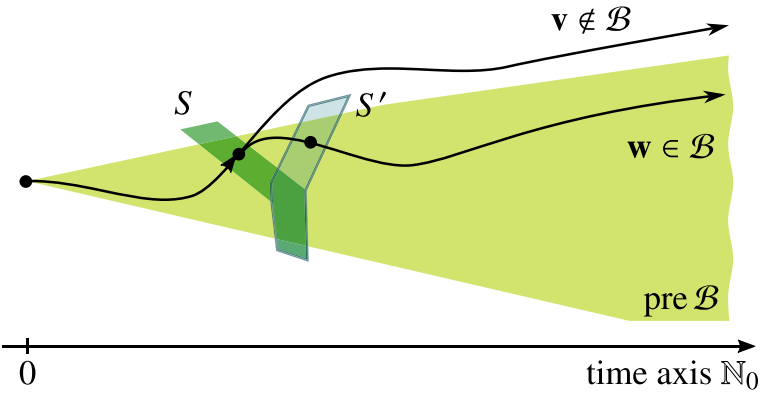

Fig. 2 Prefix-free experiment S on $\mathcal{B}$ with prefix-free refinement S', $S \leq S'$. As in Fig. 1, pre $\mathcal{B}$ is interpreted as an infinite computational tree with the empty string as the root at the very left, graphically represented as a dark yellow cone. Both bounded length experiments S and S' are shown as "barriers" and, as indicated in the sketch, any signal from $\mathcal{B}$ must pass both S and S'. Being a prefix-free refinement, the sequences in S' are obtained by extending specific sequences from S, i.e., pushing the boundary to the right. Referring exclusively to the experiment S, both signals $\mathbf{w}$ and $\mathbf{v}$ could possibly belong to some behaviour on which S was conducted. Hence, we may expect $\mathbf{v} \in \mathcal{B}_S$ although $\mathbf{v} \notin \mathcal{B}$. In contrast, we have $\mathbf{v}|_{[0,l)} \notin S'$ for all $l \in \mathbb{N}_0$ and, hence, $\mathbf{v} \notin \mathcal{B}_{S'}$

is indeed an experiment on $\mathcal{B}$ with $S \leq S'$. For the special case of $S = \mathcal{B}|_{[0,l]}$ and $R = S$ we obtain $S' = \mathcal{B}|_{[0,l+1]}$, which coincides with the refinement of l-complete approximations.

4 State machine realisations

We further elaborate the proposed scheme of behavioural abstractions in the context of state realisations. Here, we account for a class of transition systems referred to as *state machines*, which we will utilise for realisations of the plant model, finite state abstractions, and the specification in the control problem under consideration.

Definition 4 A state machine is a quadruple $P = (X, W, \delta, X_0)$, where X is the *state set*, W is the *external signal range*, $\delta \subseteq X \times W \times X$ is the *transition relation*, and $X_0 \subseteq X$ is the set of *initial states*. The state machine P is called a *finite state machine* if $|X| \in \mathbb{N}$.

We use the following terminology.

- Whenever convenient, we reinterpret the transition relation with a set-valued map, recursively defined for $(x, s) \in X \times W^*$ and $w \in W$ by (a) $\delta(x, \epsilon) := \{x\}$ and (b) $\delta(x, sw) := \{x'' | \exists x' \in \delta(x, s) \colon (x', w, x'') \in \delta\}$; i.e., $\delta(x, s)$ denotes the set of states reachable from x by taking $|s|$ transitions with labels as specified by s.
- Referring to the set-valued map interpretation of δ, the set $\delta(X_0, W^*)$ are the *reachable states*, and P is said to be *reachable* if $\delta(X_0, W^*) = X$. If $\delta(x, W)$ is non-empty for every reachable state $x \in \delta(X_0, W^*)$, then P is termed *deadlock-free*.
- The state machine P induces the *full behaviour*

$$\mathcal{B}_{\text{full}} := \{(\mathbf{w}, \mathbf{x}) | \forall k \in \mathbb{N}_0 \colon (\mathbf{x}(k), \mathbf{w}(k), \mathbf{x}(k+1)) \in \delta \text{ and } \mathbf{x}(0) \in X_0\} \tag{8}$$

and the state space system $\Sigma_{\text{full}} := (\mathbb{N}_0, W \times X, \mathcal{B}_{\text{full}})$.

- The *external behaviour* $\mathcal{B}_{\text{ex}}$ of Σ_{full} is the projection of $\mathcal{B}_{\text{full}}$ onto $W^{\mathbb{N}_0}$, i.e.,

$$\mathcal{B}_{\text{ex}} := \mathcal{P}_W \mathcal{B}_{\text{full}} := \{\mathbf{w} | \exists \mathbf{x} \colon (\mathbf{w}, \mathbf{x}) \in \mathcal{B}_{\text{full}}\}. \tag{9}$$

  If a state machine P' induces the external behaviour $\mathcal{B}'$ of a system Σ', P' is termed a realization of Σ', denoted $\Sigma' \cong P'$.
- If $|\delta(X_0, s)| \leq 1$ for every $s \in W^*$, then P is said to be *past-induced*; in automata theory, this is also referred to as *deterministic*. We then write $\delta_0(s) \in X$ for $s \in W^*$ with $|\delta(X_0, s)| = 1$ to denote the unique state reachable from X_0 via the external sequence s.

In our subsequent discussion of controller synthesis, we assume that the plant is given as a state machine $P = (X, W, \delta, X_0)$ with unrestricted initial conditions, i.e., $X_0 = X$, and we note that this assumption implies time invariance for the induced full behaviour as well as for the induced external behaviour. Moreover, we consider the product $W = U \times Y$ as external signal space, where U and Y denote the ranges of *input symbols* and *output symbols*, respectively, and where we assume that

$$\forall x \in X, u \in U \exists \in Y, x' \in X \colon (x, (u, y), x') \in \delta. \tag{10}$$

State machines with this property are called *I/S/- machines*. For the induced external behaviour, it can be seen that the *input is free* and the *output does not anticipate the input*, both technical terms defined within Willems' behavioural framework; see Proposition 24 in Moor and Raisch (1999), which refers to Definitions VIII.1 and VIII.4 in Willems (1991). In particular, we will consider supervisory controllers which at any instance of time disable specific input symbols and which in turn accept any output symbol. Technically, all external symbols are organised as pairs $w = (u, y) \in U \times Y = W$, with only the U-component considered controllable; this will be followed up in Section 5, including a formal definition of the corresponding control patterns (26). To this end, we address two remarks on how the requirement of a time-invariant I/S/- plant model can be relaxed by a preprocessing stage applicable in the context of controller synthesis with upper-bound behavioural-inclusion specifications.

Remark 5 Formally, Eq. 10 requires that every input symbol can be applied regardless of the current state of the plant. Nevertheless, if we are provided with a state machine $P = (X, U \times Y, \delta, X)$ which fails to satisfy (10), we consider the substitute model $P' = (X, U \times Y', \delta', X)$ with $Y' = Y \dot{\cup} \{\ddagger\}$ where we define the transition relation δ' to issue the distinguished output symbol $\ddagger \notin Y$ whenever an invalid input symbol was applied:

$$\delta' := \delta \cup \{(x, (u, \ddagger), x) | \forall x' \in X, y \in Y : (x, (u, y), x') \notin \delta\}. \tag{11}$$

Clearly, P' satisfies (10). The considered upper-bound behavioural-inclusion specification can then be used to prevent the distinguished output symbol $\ddagger \notin Y$ from occurring in the closed-loop configuration. Now assume that a controller that has been designed for the plant substitute P' such that $\ddagger$ does not indeed occur in any external closed-loop signal. Whenever P' attains a state $x \in X$ such that there exists a transition $(x, (u, \ddagger), x) \in \delta'$ for some $u \in U$, this transition will be prevented by the controller. In our specific setting of $W = U \times Y'$, the controller can only directly restrict the U-component of the external symbol. Hence, the controller must at least effectively disable the external symbols

$$\rho_x := \{(u, y) \in U \times Y' | u \in U, y \in Y' \text{ s.t. } (x, (u, \ddagger), x) \in \delta'\} \tag{12}$$

whenever the plant attains the state $x \in X$. For the remaining transitions, however, δ' matches the original transition relation δ; i.e.,

$$\{(x, (u, y), x') \in \delta' | (u, y) \notin \rho_x\} = \{(x, (u, y), x') \in \delta | (u, y) \notin \rho_x\}. \tag{13}$$

Therefore, the supervisor designed for the substitute P' will implement exactly the same closed-loop behaviour when applied to the actual plant P.

Remark 6 The situation of restricted initial states is addressed in a similar fashion. Given $P = (X, U \times Y, \delta, X_0)$ with $X_0 \subsetneq X$, we consider the substitute model $P' = (X, U' \times Y', \delta', X)$, $U' = U \dot{\cup} \{\dagger\}$, $Y' = Y \dot{\cup} \{\ddagger\}$. Here, the distinguished input symbol $\dagger$ is introduced to test whether the state is within the original set of initial states, and, if so, this is confirmed by the distinguished output symbol $\ddagger$. Technically, we define the transition relation by

$$\delta' := \delta \cup \{(x, (\dagger, \ddagger), x) | x \in X_0\} \cup \{(x, (\dagger, y), x) | x \in X - X_0, y \in Y\}. \tag{14}$$

We then design a controller for P' with a specification that requires (a) that $\dagger$ appears exclusively as the unique first input symbol and (b) that any further closed-loop requirements are only imposed conditionally, subject to $\ddagger$ being generated as the very first output symbol. In order to apply the resulting controller to the actual plant, we need an additional device that generates an adequate output symbol when the distinguished input symbol $\dagger$ is applied, i.e., we need to implement the additional transitions introduced by δ'. However, by clause (a), this is only necessary for the very first transition taken and, since P has the initial state restricted to X_0, the additional device is a-priori known to generate $\ddagger$ and it does so without affecting any state. From a practical perspective, this can be implemented by intercepting the closed-loop interconnection to hide the very first input symbol $\dagger$ from the plant and by injecting a fake response $\ddagger$ to the controller. For any subsequent transitions, the actual plant P matches the substitute P' as in our previous Remark 5. Hence, the supervisor will enforce the conditional specification (b) in the adapted closed-loop configuration with the actual plant.

For a discrete-time model of hybrid plant dynamics with a discrete external interface, we consider an I/S/- machine $P = (X, U \times Y, \delta, X)$ with a state set $X \subseteq D \times \mathbb{R}^n$, $|D| \in \mathbb{N}$, and with finite input- and output-ranges, i.e., $|U|, |Y| \in \mathbb{N}$. This is a rather general setting and, for practical applications, one needs to formally derive the transition relation δ from a more detailed model. Since the literature provides a rich variety of models for hybrid dynamics, we demonstrate this step by example.

Example 1 Consider a physical system with linear continuous dynamics and a finite number of linear controllers to implement individual modes of operation. Discretising time by a regular sampling period, we obtain the switched affine system

$$\mathbf{x}(k+1) = A_{\mathbf{d}(k)}\mathbf{x}(k) + B_{\mathbf{d}(k)}, \tag{15}$$

where $k \in \mathbb{N}_0$ denotes the discrete time; $\mathbf{x}\colon \mathbb{N}_0 \to \mathbb{R}^n$ is the sampled continuous state trajectory; the discrete signal $\mathbf{d}\colon \mathbb{N}_0 \to D$ selects the mode of operation $\mathbf{d}(k)$ at time k; and the square matrix $A_d \in \mathbb{R}^{n \times n}$ and the column vector $B_d \in \mathbb{R}^n$ are obtained by sampling the closed-loop configuration for mode of operation $d \in D$. We can either directly interpret $\mathbf{d}$ as our input signal, or encode additional discrete dynamics by

$$\mathbf{d}(k+1) = f(\mathbf{d}(k), \mathbf{u}(k)), \tag{16}$$

where $f : D \times U \to D$ is a complete transition function and $\mathbf{u}: \mathbb{N}_0 \to U$ is the discrete input signal. As discrete output, we propose a mode dependant finite partition of the continuous state space, i.e.,

$$\mathbf{y}(k) = g(\mathbf{d}(k), \mathbf{x}(k)), \tag{17}$$

with $g : D \times \mathbb{R}^n \to Y$. The transition relation δ is then formally defined by

$$\delta := \{((x, d), (u, y), (x', d'))|x' = A_d x + B_d, d' = f(d, u), y = g(d, x)\}. \tag{18}$$

With $W := U \times Y$, $X := D \times \mathbb{R}^n$ and unrestricted initial states $X_0 := X$, the latter completes the construction of an I/S/- machine $P = (X, W, \delta, X_0)$ with time-invariant induced external and internal behaviour, respectively.

Example 2 In a similar way to the above example, hybrid automata address the situation of a finite number of modes of operation, each with specific continuous dynamics. However, and in contrast to the above example, the generation of events is organised in dependency of the evolution of the continuous state and by referring to so called *mode invariants* and *guard relations*. A general and formal definition of hybrid automata semantics is quite involved and interested readers are referred to the literature; see, e.g., Henzinger (2000) or Chapter 7 in Tabuada (2009).

In the present study, we provide a simple practical example with hybrid automata semantics to which we will refer later in the context of abstraction based controller design. To this end, consider a vehicle which we shall navigate within a rectangular area $A := [0, w] \times [0, h] \subseteq \mathbb{R}^2$; see Fig. 3.

The vehicle is equipped with low-level continuous controllers which implement the modes of operation $U := \{\mathsf{u\_nw}, \mathsf{u\_ne}, \mathsf{u\_sw}, \mathsf{u\_se}\}$ to drive the vehicle in the respective direction, i.e., north-west, north-east, south-west or south-east. With each mode $u \in U$, we associate a differential inclusion $\frac{\mathrm{d}}{\mathrm{d}t}\varphi(t) \in F_u$, where the constant right-hand-side $F_u \subseteq \mathbb{R}^2$ is set up as the sum of the respective nominal velocity $(-v, v)$, (v, v), $(-v, -v)$ or $(v, -v) \in \mathbb{R}^2$ and a square with diameter $d \in \mathbb{R}$, $d > 0$, as a bounded additive disturbance. Each mode of operation is associated with the entire rectangular area $I_u := A$ as the *mode invariant*.

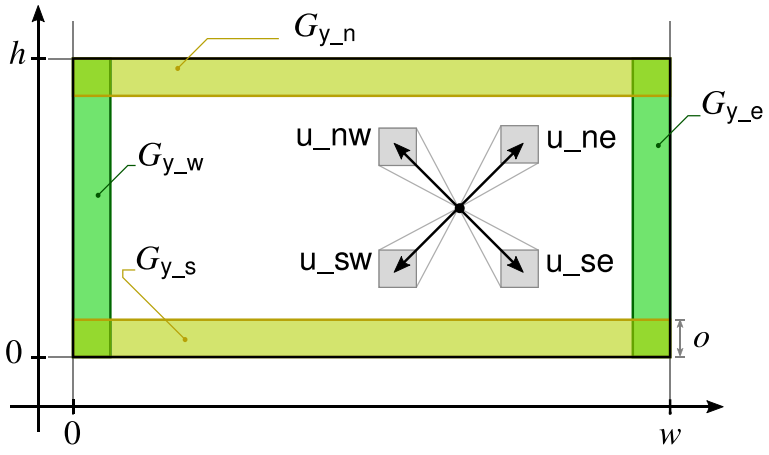

Fig. 3 Vehicle navigation example with rectangular area $A = [0, w] \times [0, h] \subseteq \mathbb{R}^2$

An output event $y \in Y := \{\mathsf{y\_n}, \mathsf{y\_s}, \mathsf{y\_w}, \mathsf{y\_e}\}$ will be generated while the vehicle is inside a *guard region*, denoted $G_{\mathsf{y\_n}}, G_{\mathsf{y\_s}}, G_{\mathsf{y\_w}}, G_{\mathsf{y\_e}} \subseteq \mathbb{R}^2$, respectively. To avoid trivial Zeno-behaviour, guards are enabled and disabled according to the mode of operation, e.g., when driving north-west, only the north guard $G_{\mathsf{y\_n}}$ and the west guard $G_{\mathsf{y\_w}}$ are enabled.

For a discrete-time model of the vehicle, we consider the overall state set $X := A$. On this state set, we define the transition relation $\delta \subseteq X \times (U \times Y) \times X$ by $(x, (u, y), x') \in \delta$, if and only if (a) there exists $\tau \geq 0$ and a continuous state trajectory $\varphi : [0, \tau] \to I_u$, differentiable on $(0, \tau)$ with $\frac{\mathrm{d}}{\mathrm{d}t}\varphi(t) \in F_u$ for all $t \in (0, \tau)$, with the initial state $\varphi(0) = x$ and the final state $x' = \varphi(\tau)$; (b) x' is within the guard G_y; and (c) the guard G_y is enabled by mode $u \in U$. To practically test whether a tuple $(x, (u, y), x')$ satisfies the above conditions (a)–(c), we observe that the relevant sets $I_u = A$, F_u and G_y are convex closed polyhedra. This implies that the positions reachable by some qualifying continuous trajectory φ amount to the convex closed polyhedron $V = \{x + \lambda v | v \in F_u, \lambda \geq 0\} \cap A$; see, e.g., Halbwachs et al. (1997). Provided that the guard G_y is enabled by mode u, we have that $(x, (u, y), x') \in \delta$ if and only if $x' \in V \cap G_y$. In this context, the transitions in δ are also referred to as *logic-time transitions* to contrast with the evolution of the continuous state with respect to physical time.

This completes the construction of the I/S/- machine $P = (X, W, \delta, X_0)$ with $W := U \times Y$ and $X_0 = X = A$.

Referring back to Definition 2, recall that an experiment S must account for all trajectories $\mathbf{w} \in \mathcal{B}$ by some finite prefix $s \in (\mathrm{pre}\,\mathbf{w}) \cap S$. Hence, the construction of an experiment practically amounts to the inspection of specific finite prefixes in $\mathrm{pre}\,\mathcal{B}$. For example, to set up an initial experiment by $S := \mathcal{B}|_{[0,l]} \subseteq \mathrm{pre}\,\mathcal{B}$ for some $l \in \mathbb{N}$ we need an implementable test for whether or not $s \in \mathrm{pre}\,\mathcal{B}$ for all finite sequences s of length $l + 1$. Likewise, referring to Eq. 7, a refinement of an experiment S w.r.t. the candidates $R \subseteq S$ requires us to test whether or not $\langle s, w \rangle \in \mathrm{pre}\,\mathcal{B}$ for all $s \in R$ and $w \in W$. Given an I/S/- machine $P = (X, W, \delta, X)$ with the external behaviour $\mathcal{B}$, Moor et al. (2002) propose to base the required test on the following recursively defined *sets of compatible states*:

$$\mathcal{X}(\epsilon) := X, \tag{19}$$

$$\mathcal{X}(\langle r, (u, y) \rangle) := \{x' | \exists x \in \mathcal{X}(r) \colon (x, (u, y), x') \in \delta\}, \tag{20}$$

where $r \in W^*$, $u \in U$, $y \in Y$, and the right hand side of Eq. 20 is a one-step forward reachability operator applied to $\mathcal{X}(r)$ with (u, y) as a constraint for the external symbols. By construction, $\mathcal{X}(s) \subseteq X$ consists of all states the I/S/- machine can attain after generating the finite sequence of external symbols $s \in W^*$. Since I/S/- machines do not deadlock, this implies that $s \in \mathrm{pre}\,\mathcal{B}$ if and only if $\mathcal{X}(s) \neq \emptyset$. Hence, for behaviours realised by I/S/- machines, setting up and/or refining experiments effectively amounts to a finite iteration of the one-step forward-reachability operator in Eq. 20. The latter type of reachability operator has been intensively investigated over the past two decades and the literature provides a variety of efficient computational methods addressing specific classes of hybrid systems; see, e.g., Alur et al. (2000), Alur et al. (1996), and Lafferriere et al. (2000) for the exact computation of sets of reachable states for a restricted class of continuous dynamics, or, e.g., Althoff et al. (2010), Chutinan and Krogh (1998), Frehse (2008), Henzinger et al. (2000), Maler and Dang (1998), Mitchell et al. (2005), and Reissig (2011) for safe over-approximations for richer classes of continuous dynamics.

Example 3 **(cont.)** For our vehicle navigation example, all relevant sets are convex closed polyhedra and the differential inclusions have a constant polyhedral right-hand-side. Sets of states reachable by one logic-time transition can hence be computed exactly, e.g., using the software Parma Polyhedra Library (PPL); see Bagnara et al. (2008). We outline the overall computational procedure that is used to construct an initial experiment and a refinement thereof.

Consider u_nw as the first input symbol being applied in the vehicle navigation example. Then the subsequent output symbol can be either y_n or y_w since the initial states are not restricted and since $G_{\mathsf{y\_n}}$ and $G_{\mathsf{y\_w}}$ are the only enabled guards. This implies $\mathcal{X}((\mathsf{u\_nw}, \mathsf{y\_s})) = \emptyset$ and $\mathcal{X}((\mathsf{u\_nw}, \mathsf{y\_e})) = \emptyset$. Under the additional hypothesis that we actually observe y_w, we conclude that the attained state must be within $G_{\mathsf{y\_w}}$. Hence, we obtain $\mathcal{X}((\mathsf{u\_nw}, \mathsf{y\_w})) = G_{\mathsf{y\_w}}$ as compatible states; see Fig. 4. Likewise, we obtain $\mathcal{X}((\mathsf{u\_nw}, \mathsf{y\_n})) = G_{\mathsf{y\_n}}$. Repeating the above reasoning for all possible first input symbols, we establish that

$$\begin{aligned}\mathcal{B}|_{[0,0]} = \{&(\mathsf{u\_ne}, \mathsf{y\_n}), (\mathsf{u\_ne}, \mathsf{y\_e}), (\mathsf{u\_nw}, \mathsf{y\_n}), (\mathsf{u\_nw}, \mathsf{y\_w}),\\ &(\mathsf{u\_se}, \mathsf{y\_s}), (\mathsf{u\_se}, \mathsf{y\_e}), (\mathsf{u\_sw}, \mathsf{y\_s}), (\mathsf{u\_sw}, \mathsf{y\_w})\},\end{aligned} \tag{21}$$

and obtain our initial experiment $S := \mathcal{B}|_{[0,0]}$. This experiment encodes the fact that for our vehicle navigation example, only relevant guards are enabled depending on the input symbol. This is expected to be insufficient for the design of a supervisor that solves typical navigation tasks like, e.g., to visit a specific guard region.

For illustration purposes, we consider $R := \{(\mathsf{u\_nw}, \mathsf{y\_w})\} \subseteq S$ as our refinement candidate. Since $\mathcal{X}((\mathsf{u\_nw}, \mathsf{y\_w})) = G_{\mathsf{y\_w}}$ from the foregoing discussion, we now need to compute $\mathcal{X}(\langle(\mathsf{u\_nw}, \mathsf{y\_w}), (u, y)\rangle)$ for all $u \in U$ and $y \in Y$ by applying Eq. 20. Consider the case of $u = \mathsf{u\_ne}$, i.e., we apply the input symbol u_ne when the vehicle position x is initially in $\mathcal{X}((\mathsf{u\_nw}, \mathsf{y\_w}))$. The continuous time motion of the vehicle is then modelled by a trajectory $\varphi : [0, \tau] \to I_{\mathsf{u\_ne}} = A$ with $\varphi(0) = x$ and $\frac{\mathrm{d}}{\mathrm{d}t}\varphi(t) \in F_{\mathsf{u\_ne}}$ for all $t \in (0, \tau)$. Recall that in our example, all relevant sets are convex closed polyhedra. This implies that all positions reachable by the vehicle are given by

$$V := \{x + \lambda v | x \in \mathcal{X}((\mathsf{u\_nw}, \mathsf{y\_w})), v \in F_{\mathsf{u\_ne}}, \lambda \geq 0\} \cap A; \tag{22}$$

see also Fig. 4. Compatible states are then obtained by $\mathcal{X}(\langle(\mathsf{u\_nw}, \mathsf{y\_w}), (\mathsf{u\_ne}, y)\rangle) = V \cap G_y$ with $y \in Y$ and are again convex closed polyhedra. More specifically, we have $\mathcal{X}(\langle(\mathsf{u\_nw}, \mathsf{y\_w}), (\mathsf{u\_ne}, \mathsf{y\_n})\rangle)$ as shown in Fig. 4 and $\mathcal{X}(\langle(\mathsf{u\_nw}, \mathsf{y\_w}), (\mathsf{u\_ne}, y)\rangle) = \emptyset$ for

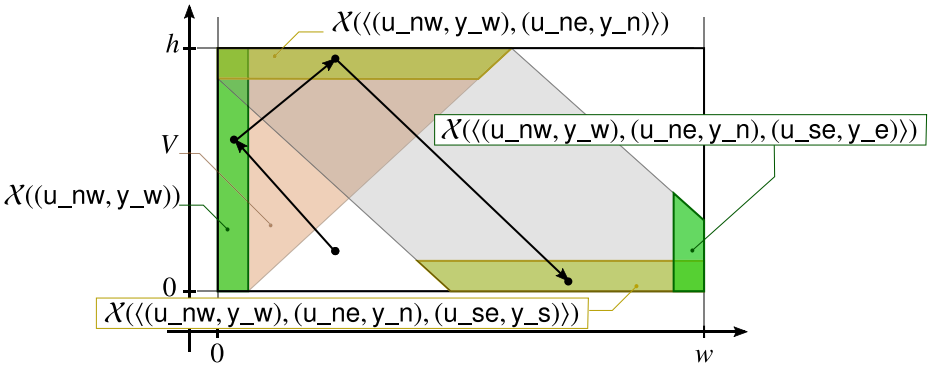

Fig. 4 Sets of compatible states for the navigation example

$y \neq$ y_n. This procedure can be applied repeatedly for subsequent input symbols. To this end, Fig. 4 shows the sets of compatible states for $\langle$(u_nw, y_w), (u_ne, y_n), (u_se, y_s)$\rangle$ and $\langle$(u_nw, y_w), (u_ne, y_n), (u_se, y_e)$\rangle$. Coming back to the refinement, we apply the same analysis to $\mathcal{X}$((u_nw, y_w)) as above, but now for the remaining choices of the input symbol, i.e., $u =$ u_ne, $u =$ u_sw and $u =$ u_se. As it turns out, the only extensions of our refinement candidate (u_nw, y_w) by one more pair of an input symbol and an output symbol with a non-empty set of compatible states are the following length-two sequences:

$$\begin{aligned}
&\langle(\text{u_nw}, \text{y_w}), (\text{u_ne}, \text{y_n})\rangle, \langle(\text{u_nw}, \text{y_w}), (\text{u_se}, \text{y_s})\rangle,\\
&\langle(\text{u_nw}, \text{y_w}), (\text{u_nw}, \text{y_n})\rangle, \langle(\text{u_nw}, \text{y_w}), (\text{u_nw}, \text{y_w})\rangle,\\
&\langle(\text{u_nw}, \text{y_w}), (\text{u_sw}, \text{y_s})\rangle, \langle(\text{u_nw}, \text{y_w}), (\text{u_sw}, \text{y_w})\rangle. \qquad (23)
\end{aligned}$$

Referring to Eq. 7, the refined experiment S' is obtained from $S := \mathcal{B}|_{[0,0]}$, Eq. 21, by removing the refinement candidate (u_nw, y_w) and by including the above six extensions; for a tree representation of S' see Fig. 5.

Once an experiment S on the external behaviour $\mathcal{B}$ of the plant P has been obtained, it can be used to set up a finite-state realisation of the corresponding abstraction Σ_S. Roughly speaking, the realisation tracks the longest suffix of the signal generated so far that matches some prefix within S.

Theorem 7 *(See Moor and Götz (2018), Lemma 14) Given a prefix-free trim experiment $S \subseteq W^*$, $\epsilon \notin S \neq \emptyset$, of bounded length, the abstraction $\Sigma_S = (\mathbb{N}_0, W, \mathcal{B}_S)$ obtained under the assumption of time invariance (see* Eq. 1*) is realised by the state machine $P_S = (Z_S, W, \delta_S, \{\epsilon\})$, where $Z_S :=$ pre$S$ consists of the prefixes of S and where the transition relation δ_S is defined as follows: $(z, w, z') \in Z_S \times W \times Z_S$ is in δ_S if and only if $z' = \langle z^\triangleleft, w\rangle$ with $z^\triangleleft$ the longest suffix of z that is in Z_S but not in S; i.e., if and only if $z^\triangleleft$ is the unique longest sequence in $\{r \in W^* | \exists t \in W^* : \langle t, r\rangle = z\} \cap \{r \in Z_S | r \notin S\}$. Moreover, P_S is reachable, deadlock-free and past-induced. Provided that W is a finite set, P_S is a finite state machine.*

Note that for the degenerated cases of $\epsilon \in S$ or $S = \emptyset$ we have $\mathcal{B}_S = W^{\mathbb{N}_0}$ or $\mathcal{B}_S = \emptyset$, respectively, with well known realisations. To provide some intuition regarding the transition relation given by the above theorem, we consider the initial state $z_0 := \epsilon$ and a signal

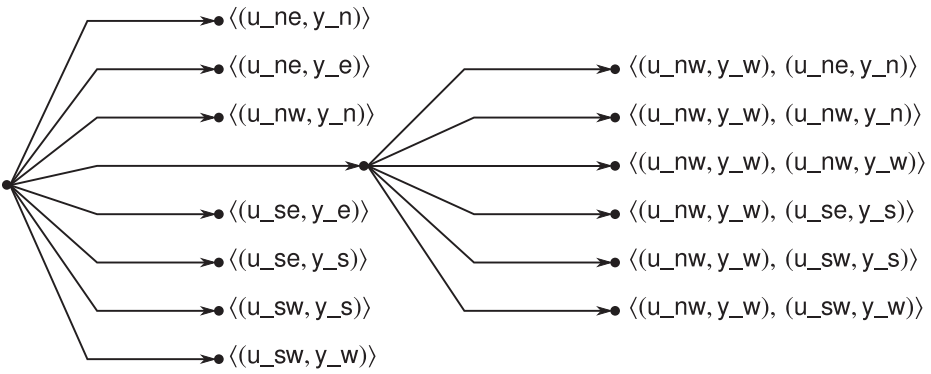

Fig. 5 Refined experiment S' on the external behaviour $\mathcal{B}$ for the navigation example

$\mathbf{w} \in \mathcal{B}_S$. In particular, there exists $l \in \mathbb{N}_0$ such that $w|_{[0,l)} \in S$. By $\epsilon \notin S$, this implies $l \geq 1$. Moreover, since S is prefix-free, we have $w|_{[0,k)} \notin S$ and $w|_{[0,k)} \in \operatorname{pre} S = Z_S$ for all $k < l$. Now let $z_k := w|_{[0,k)}$ for all $k \leq l$ and observe, for all $k < l$, that z_k is its own longest suffix $z^{\triangleleft}$ qualifying for $z^{\triangleleft} \notin S$ and $z^{\triangleleft} \in Z_S$, and, hence, $(z_k, \mathbf{w}(k), z_{k+1}) \in \delta_S$. In other words, the state records all past external symbols until $z_l \in S$. However, once in state $z_l \in S$, no additional external symbols can be recorded with the given state set unless one first drops a sufficient amount of symbols recorded earlier. Technically, we are asking for a suffix $z^{\triangleleft}$ of z_l such that $\langle z^{\triangleleft}, \mathbf{w}(l)\rangle \in Z_S$. To see that such a suffix exists, we consider the shortest suffix ϵ of z_l and we refer to time invariance to obtain $\langle \epsilon, \mathbf{w}(l)\rangle = \mathbf{w}(l) \in Z_S$. For the transition relation proposed in the theorem, we take the longest qualifying $z^{\triangleleft}$ to drop as little as possible of the recorded symbols and, as a conjecture, obtain $(z_l, \mathbf{w}(l), z_{l+1}) \in \delta_S$ with $z_{l+1} := \langle z^{\triangleleft}, \mathbf{w}(l)\rangle$. If we can continue this construction indefinitely and if the conjecture holds true at each stage, we obtain a state trajectory $\mathbf{z} : \mathbb{N}_0 \to Z_S$, $\mathbf{z}(k) := z_k$ for all k, such that $(\mathbf{w}, \mathbf{z})$ is in the full behaviour induced by P_S. Note that it is neither obvious that the construction actually can be continued indefinitely, nor, for the converse behavioural inclusion, that any trajectory generated by P_S is within $\mathcal{B}_S$. The cited reference in Theorem 7 provides technical proofs for both claims and the above theorem.

Example 4 **(cont.)** For the experiment S from Fig. 5, the realisation P_S is obtained by the following construction. The nodes $s \in \operatorname{pre} S$ in Fig. 5 become the states of P_S and the edges in Fig. 5 become transitions with an event label to match the most recent event of the respective target node; see Fig. 6, transitions given in black color. Then, per leaf $z \in S$, we: (a) drop the minimum prefix from the node label $z \in S$ to obtain $z^{\triangleleft} \in \{r \in Z_S | r \notin S\}$; and (b) for all $w \in W$ such that $z' := \langle z^{\triangleleft}, w\rangle \in Z_S$ insert a transition from z to z' with label w. For the node $z = \langle(\text{u\_nw}, \text{y\_w}), \text{u\_nw}, \text{y\_w})\rangle$, we have $z^{\triangleleft} = \langle(\text{u\_nw}, \text{y\_w})\rangle$ with out-going transitions indicated in green color in Fig. 6. For all other nodes $z \in S$, $z \neq \langle(\text{u\_nw}, \text{y\_w}), \text{u\_nw}, \text{y\_w})\rangle$, we obtain $z^{\triangleleft} = \epsilon$ and insert per $w \in W$ the transition $(z, w, w) \in \delta_S$. This amounts to $12 \times 8 = 96$ transitions, which are omitted in Fig. 6.

We observe for the vehicle navigation example that not only the actual plant P but also the realisation P_S of the abstraction is an I/S/- machine. This can always be achieved by suitable trimming, and we can without loss of generality restrict the discussion to experiments such that P_S is an I/S/- machine.

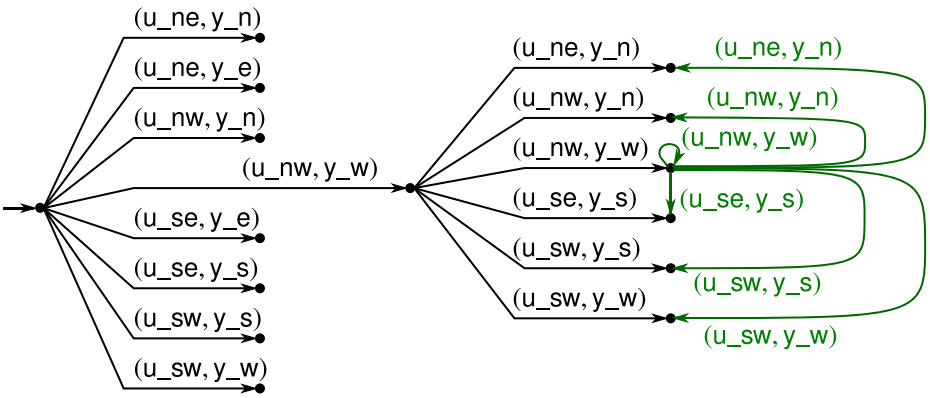

Fig. 6 Fragment of the realisation P_S of the experiment S from Fig. 5

Proposition 8 *Consider an experiment $S \subseteq W^*$, $W = U \times Y$, on an external behaviour $\mathcal{B} \subseteq W^{\mathbb{N}_0}$ realised by an I/S/- machine P with input range U and output range Y. With $P_S = (Z_S, W, \delta_S, \{\epsilon\})$ from Theorem 7, let*

$$Z_{\mathrm{io}} := \{z \in Z_S | \forall u \in U\, \exists y \in Y, z' \in Z_S \colon (z, (u, y), z') \in \delta_S\} \tag{24}$$

and trim S by

$$S' := \{s \in S | \operatorname{pre} s \subseteq Z_{\mathrm{io}}\}. \tag{25}$$

Then $S' \subseteq S$ is an experiment on $\mathcal{B}$.

Proof To show that S' is an experiment on $\mathcal{B}$, we pick an arbitrary $\mathbf{w} \in \mathcal{B}$. Since S is an experiment on $\mathcal{B}$, there exists $s \in S$ with $s < \mathbf{w}$. To show that $s \in S'$, we pick an arbitrary prefix $z \leq s$ and an arbitrary $u \in U$ and establish the existence of $y \in Y$ and $z' \in Z_S$ such that $(z, (u, y), z') \in \delta_S$. For our choice, we refer to any state trajectory $\mathbf{x}$ such that $(\mathbf{w}, \mathbf{x})$ is in the full behaviour induced by P. Since P is an I/S/- machine, there exists another trajectory $(\mathbf{w}', \mathbf{x}')$ from the full behaviour such that $(\mathbf{w}', \mathbf{x}')|_{[0,l)} = (\mathbf{w}, \mathbf{x})|_{[0,l)}$, $\mathbf{w}'(l) = (u, y)$ and $\mathbf{x}'(l) = \mathbf{x}(l)$ for $l = |z|$ and some $y \in Y$. By $\mathcal{B} \subseteq \mathcal{B}_S$ we obtain the unique state trajectory $\mathbf{z}'$ such that $(\mathbf{w}', \mathbf{z}')$ is in the full behaviour induced by P_S. By the definition of δ_S we have $\mathbf{z}'(k) = \mathbf{w}'|_{[0,k)}$ for all $k \leq l$. In particular, we have $\mathbf{z}'(l) = z$ and, hence, $(z, (u, y), \mathbf{z}'(l + 1)) \in \delta_S$. □

Applying the above trimming procedure repeatedly, it generates a monotonously decreasing sequence $S \supseteq S' \supseteq S'' \supseteq \cdots$ of experiments. For our situation of a finite external signal range and experiments of bounded length, S is a finite set. Hence, a fixpoint S_* is attained after finitely many stages of trimming. Then, the definition of Z_{io} implies that the S_* is indeed realised by an I/S/- machine. Moreover, the realisation by an I/S/- machine is retained under refinement by Eq. 7.

5 Supervisory Control

Given a plant model with discrete-event interface, we seek to design a controller that restricts the behaviour to satisfy a prescribed upper bound specification. This type of control problem is addressed by *supervisory control theory*, as introduced by Ramadge and Wonham (1987, 1989), however, using regular $*$-languages and finite automata realisations as base models. For the present paper, we refer to an adaption of supervisory control to ω-languages to address infinite-length signals as discussed by (Thistle and Wonham 1994a; 1994b), and we propose to substitute the actual plant by a finite state abstraction obtained from an experiment.

Formally, a supervisor is defined as a causal feedback map f with domain W^* that, at any instance of time $k \in \mathbb{N}_0$, maps the present prefix $s = \mathbf{w}|_{[0,k)}$ generated by the plant to a *control pattern* $\gamma = f(s) \subseteq W$, with the effect that the subsequently generated symbol must satisfy $\mathbf{w}(k) \in \gamma$, i.e., the plant is restricted to only generate symbols that match the respective control pattern. Most commonly the range Γ of all admissible control patterns is derived from partitioning W into controllable and uncontrollable events. However, to address I/S/- machines as plants, we refer to the product $W = U \times Y$ and define

$$\Gamma := \{\gamma \subseteq U \times Y | \emptyset \neq \gamma \text{ and } \forall (u, y) \in \gamma, y' \in Y \colon (u, y') \in \gamma\} \tag{26}$$

as the range of the supervisor, i.e., $f \colon W^* \to \Gamma$. By this choice, the supervisor imposes its restriction on the input symbol only and accepts any output symbol generated by the plant.

Note also that, by construction, Γ is closed under unions of control patterns and, thus, our setting here is formally covered by the relevant references (Thistle and Wonham 1994a, b).

In the more common setting with a partition into controllable and uncontrollable events, a supervisor could apply a control pattern such that the plant in its current state cannot generate any of the enabled symbols. This form of temporal blocking is undesired and, in general, needs to be addressed by the synthesis procedure. However, for the specific situation of I/S/- machines and our tailored choice of Γ in Eq. 26, temporal blocking is not an issue.

Definition 9 Given a deadlock-free state machine $P = (X, W, \delta, X_0)$, a supervisor $f\colon W^* \to \Gamma$ *preserves liveness* in closed-loop configuration with P if for all signals $(\mathbf{w}, \mathbf{x}) \in \mathcal{B}_{\text{full}}$ from the induced full behaviour that comply with f up to some time $k \in \mathbb{N}_0$, i.e., $\mathbf{w}(\kappa) \in f(\mathbf{w}|_{[0,\kappa)})$ for all $\kappa < k$, there exist $w \in f(\mathbf{w}_{[0,k)})$ and $(\mathbf{w}', \mathbf{x}') \in \mathcal{B}_{\text{full}}$ such that $\mathbf{x}'|_{[0,k]} = \mathbf{x}|_{[0,k]}$, $\mathbf{w}'|_{[0,k)} = \mathbf{w}|_{[0,k)}$ and $\mathbf{w}'(k) = w$.

Proposition 10 *Given an I/S/- machine $P = (X, W, \delta, X_0)$ where $W = U \times Y$ with input range U and output range Y, any supervisor $f\colon W^* \to \Gamma$ preserves liveness in closed-loop configuration with P.*

Proof Consider any $(\mathbf{w}, \mathbf{x}) \in \mathcal{B}_{\text{full}}$ compliant with f up to time $k \in \mathbb{N}_0$. Then at time $k \in \mathbb{N}_0$ the supervisor applies the control pattern $\gamma := f(\mathbf{w}_{[0,k)})$ and P is in state $x = \mathbf{x}(k)$. By Eq. 26, $\gamma \neq \emptyset$ and we can pick a symbol $w := (u, y) \in \gamma$. Since P is an I/S/- machine, there exist $y' \in Y$ and $x' \in X$ such that $(x, (u, y'), x') \in \delta$. Referring again to Eq. 26, we obtain $w' := (u, y') \in \gamma$. To construct the signals $\mathbf{x}'$ and $\mathbf{w}'$, let $\mathbf{x}'(\kappa) := \mathbf{x}(\kappa)$ for $0 \leq \kappa \leq k$, $\mathbf{x}'(k+1) = x'$, $\mathbf{w}'(\kappa) := \mathbf{w}(\kappa)$ for $0 \leq \kappa < k$ and $\mathbf{w}'(k) = w'$. Observe that $(\mathbf{x}'(\kappa), \mathbf{w}'(\kappa), \mathbf{x}'(\kappa+1)) \in \delta$ for $0 \leq \kappa \leq k$. Since P itself is deadlock-free, the signals can be extended to the entire time axis by taking arbitrary transitions from δ. We then obtain $(\mathbf{w}', \mathbf{x}') \in \mathcal{B}_{\text{full}}$ as required. □

Restricting consideration to I/S/- machines and the corresponding choice of Γ, Eq. 26, the problem of supervisory control is stated as follows.

Definition 11 Consider a plant $\Sigma = (\mathbb{N}_0, W, \mathcal{B})$ realised by an I/S/- machine $P = (X, W, \delta, X_0)$ with input range U and output range Y, and a specification $\Sigma_{\text{spec}} = (\mathbb{N}_0, W, \mathcal{B}_{\text{spec}})$. For a supervisor $f\colon W^* \to \Gamma$ with Γ from Eq. 26, the *closed-loop behaviour* is defined by

$$\mathcal{B}_f := \{\mathbf{w} \in \mathcal{B} | \forall k \in \mathbb{N}_0 \colon \mathbf{w}(k) \in f(\mathbf{w}|_{[0,k)})\}. \tag{27}$$

A supervisor $f\colon W^* \to \Gamma$ *solves the control problem* if it *enforces the specification*, i.e., if $\mathcal{B}_f \subseteq \mathcal{B}_{\text{spec}}$.

The provided references (Thistle and Wonham 1994a, b) present an algorithmic solution to the above problem for the case that the relevant behaviours are ω-regular and realised by past-induced finite state machines, extended by an acceptance condition. In the context of the present paper, we will substitute the actual plant by the abstraction Σ_S obtained from some experiment S with past-induced finite state realisation $P_S = (Z_S, W, \delta_S, \{\epsilon\})$. Regarding the specification, we account for past-induced finite realisations with *Büchi acceptance condition*, i.e., we consider a state machine $P_{\text{spec}} = (X_{\text{spec}}, W, \delta_{\text{spec}}, \{x_{\text{spec}0}\})$ with a set of

accepting states $X_{\text{specM}} \subseteq X_{\text{spec}}$ and require that signals in the full behaviour visit accepting states infinitely often. The specification $\Sigma_{\text{spec}} = (\mathbb{N}_0, W, \mathcal{B}_{\text{spec}})$ is then formally defined by

$$\begin{aligned}\mathcal{B}_{\text{spec}} := \quad & \{\mathbf{w} : \mathbb{N}_0 \to W | \exists \mathbf{x} : \mathbb{N}_0 \to X_{\text{spec}} : \\ & (\mathbf{w}, \mathbf{x}) \text{ is in the full behaviour of } P_{\text{spec}}, \\ & \text{and } \mathbf{x}(k) \in X_{\text{specM}} \text{ for infinitely many } k \in \mathbb{N}_0\}. \end{aligned} \tag{28}$$

As a technical consequence of introducing an acceptance condition, it is not restrictive to assume that the transition relation δ_{spec} is *full*, i.e., for all $\chi \in X_{\text{spec}}$ and all $w \in W$ there exists $\chi' \in X_{\text{spec}}$ such that $(\chi, w, \chi') \in \delta_{\text{spec}}$. For example, assume that we wish to exclude all closed loop trajectories that exhibit a certain string of symbols from W. We can encode this in a specification state machine with full transition relation where the occurrence of such a string leads into a "dump state", from where no other state can be reached. The assumption of a full transition relation is common in automata theory and it simplifies the subsequent discussion.

Supervisory controller synthesis is conducted in the following two steps. First, we extend the abstraction state set to also encode the specification state by the product composition $P_\times := P_S \times P_{\text{spec}} := (Q, W, \lambda, \{q_0\})$, where $Q = Z_S \times X_{\text{spec}}$, $q_0 = (\epsilon, x_{\text{spec}0})$, and where $\lambda \subseteq Q \times W \times Q$ is defined by $((z, \chi), w, (z', \chi')) \in \lambda$ if and only if $(z, w, z') \in \delta_S$ and $(\chi, w, \chi') \in \delta_{\text{spec}}$. Since δ_{spec} is full, the induced behaviours of $P_\times$ equal the respective behaviours induced by P_S. Moreover, past-inducedness of both components P_S and P_{spec} implies past-inducedness of the product $P_\times$. With $Q_{\text{M}} := Z_S \times X_{\text{specM}} \subseteq Q$ we lift the acceptance condition of the specification accordingly. Note also that, since δ_{spec} is full and since we assume P_S to be an I/S/- machine, $P_\times$ is also an I/S/- machine and, hence, is deadlock-free. Regarding the acceptance condition, however, there can be live-locks; i.e., reachable states with no execution path to attain a state in Q_{M} thereafter. This is addressed by the second step of the synthesis approach, where we refer to the iteration proposed by Thistle and Wonham (1994a, b) in order to identify states in $P_\times$ which can be controlled to eventually visit some accepting states of Q_{M} and to do so infinitely often. The resulting state set Q_{win} is referred to as the set of *winning states* — once the closed-loop has generated a prefix that corresponds to a winning state $q \in Q_{\text{win}}$, a supervisor can be employed to enforce the specification from then on. In particular, the supervisory control problem has a solution if and only if the initial state of $P_\times$ is a winning state. Addressing more general acceptance conditions for both the plant and the specification, Thistle and Wonham (1994a, b) obtain the set of winning states by a five-nested fixpoint iteration, which for the specific situation in the present paper collapses to the following simplified algorithm.

Algorithm 12 Winning states Q_{win} of $P_\times = (Q, W, \lambda, \{q_0\})$ w.r.t. the acceptance condition $Q_{\text{M}} \subseteq Q$ and control patterns Γ.

1) Initialise the target restriction with $D := Q$.
2) Initialise the winning states with $Q_{\text{win}} := \emptyset$.
3) Perform the following one-step controlled backward-reachability analysis:
   $B = \{q \in Q | \exists \gamma \in \Gamma : \emptyset \neq \lambda(q, \gamma) \subseteq (Q_{\text{M}} \cap D) \cup Q_{\text{win}}\}$.
4) If $B \not\subseteq Q_{\text{win}}$, then update the winning states by $Q_{\text{win}} := Q_{\text{win}} \cup B$ and proceed with Step 3. Else, proceed with Step 5.
5) If $Q_{\text{M}} \cap D \not\subseteq Q_{\text{win}}$, then update the restriction by $D := D \cap Q_{\text{win}}$ and proceed with Step 2. Else, terminate and report Q_{win} as the result.

We provide some intuition on the above algorithm; see also Fig. 7. The inner loop over Steps 2–4 begins with $Q_{\text{win}} = \emptyset$ to accumulate in Q_{win} states that can be controlled to reach $Q_{\text{M}} \cap D$ within a finite number of steps. Since during the inner loop Q_{win} grows monotonously and the reachability analysis in Step 3 is monotone in the iterate Q_{win}, finiteness of the state set Q implies that the termination condition $B \subseteq Q_{\text{win}}$ is satisfied after finitely many iterations. When proceeding with Step 5 for the first time, Q_{win} holds the states that can be controlled to reach Q_{M} at least once and, until then, to remain within Q_{win}. This is illustrated in Fig. 7 on the left, where the growth of Q_{win} occurs counter-clock-wise. In the figure it is assumed that the top-most transition which does not go to the target Q_{M} can be disabled by a suitable control pattern. As indicated in the figure, we cannot expect $Q_{\text{M}} \subseteq Q_{\text{win}}$, i.e., so far there may be states Q_{win} that can indeed only be controlled to reach Q_{M} once. Therefore, Step 5 restricts the effective target by letting $D = Q_{\text{win}}$. Repeating the inner loop again results in a set of winning states, but now they can all be controlled to reach Q_{M} at least twice. This is illustrated in Fig. 7 on the right, where the target has been restricted accordingly. Repeating the outer loop by monotonicity leads to a strictly decreasing restriction D and, by finiteness of Q, the termination condition must be satisfied after a finite number of iterations. At termination in Step 5, any state $q \in Q_{\text{win}}$ can be controlled to reach $Q_{\text{M}} \cap D$, and, by $Q_{\text{M}} \cap D \subseteq Q_{\text{win}}$, can be controlled to do so infinitely often.

A supervisor can be obtained from the above algorithm by recording for all $q \in Q_{\text{win}}$ an arbitrary successful control pattern from Step 3 of the last run of the inner loop. Technically, this defines a map $g : Q_{\text{win}} \to \Gamma$. By past-inducedness of $P_\times$, each prefix $s \in \text{pre}\,\mathcal{B}_S$ corresponds to exactly one state in $P_\times$ which we denote $\lambda_0(s) \in Q$. The supervisor f is then defined for $s \in W^*$ by $f(s) = g(\lambda_0(s))$ if $\lambda_0(s) \in Q_{\text{win}}$ and, else, $f(s) = \gamma_{\text{dummy}} \in \Gamma$ with $\gamma_{\text{dummy}} := \cup\{\gamma \in \Gamma\}$. By construction, this supervisor preserves liveness in closed-loop configuration with the abstraction P_S (even if it was not an I/S/- machine) it was designed for, and it conditionally enforces the specification once a prefix $s \in \text{pre}\,\mathcal{B}_S$ with $\lambda_0(s) \in Q_{\text{win}}$ has been generated; i.e., we have

$$\mathcal{B}_{S,f} \subseteq \{\mathbf{w} \in W^{\mathbb{N}_0} | (\exists s \in \text{pre}\,\mathbf{w} \colon \lambda_0(s) \in Q_{\text{win}}) \;\Rightarrow\; \mathbf{w} \in \mathcal{B}_{\text{spec}}\} \tag{29}$$

for the closed-loop behaviour $\mathcal{B}_{S,f}$; see Definition 11. Since the empty string $\epsilon \in \text{pre}\,\mathbf{w}$ corresponds to the initial state $q_0 = \lambda_0(\epsilon)$, the right-hand-side of the above inclusion collapses to $\mathcal{B}_{\text{spec}}$ if we have $q_0 \in Q_{\text{win}}$. In this case, we indeed obtain a solution of the control problem for the abstraction. This immediately carries over to the actual plant $P \cong \Sigma = (\mathbb{N}, W, \mathcal{B})$: the supervisor f preserves liveness in closed-loop configuration with P by Proposition 10 and we obtain $\mathcal{B}_f \subseteq \mathcal{B}_{S,f} \subseteq \mathcal{B}_{\text{spec}}$ as an immediate consequence of

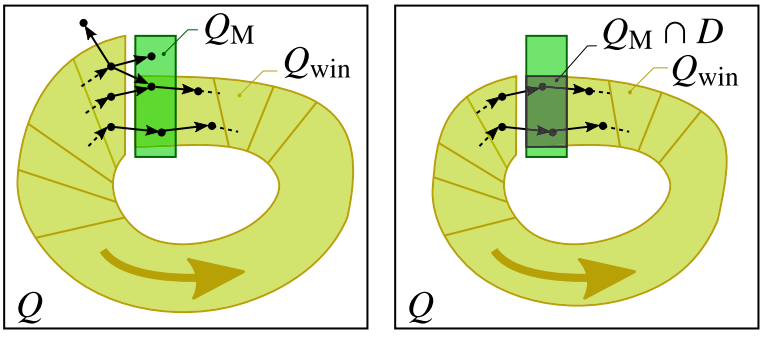

Fig. 7 Synthesis algorithm: first iteration of inner loop (left) and fixpoint of outer loop (right)

$\mathcal{B} \subseteq \mathcal{B}_S$ and Definition 11. If, on the other hand, $q_0 \notin Q_{\text{win}}$, it follows from the detailed study by Thistle and Wonham (1994a, b) that the control problem has no solution for the abstraction P_S at hand. In this case, we interpret Q_{win} as an intermediate result which in an overall synthesis approach can be used to guide a local refinement of the abstraction.

Example 5 **(cont.)** For the vehicle navigation example from the previous section, we consider the specification to navigate the vehicle eventually to the north guard $G_{\text{y\_n}}$. This can be expressed by a state machine P_{spec} with two states, where one is an accepting state and indicates that y_n has occurred at least once. For controller synthesis, we use the abstraction P_S obtained from the experiment S shown in Fig. 5. All states $(z, \chi) \in Q$, for which $z \in Z_S = \text{pre}\, S$ includes a y_n symbol, are immediately identified as winning states. Also, states (z, χ) with $z = \langle(\text{u\_nw}, \text{y\_w})\rangle$ turn out to be winning states, because one can apply the control pattern $\{(\text{u\_ne}, y) | y \in Y\}$ to enforce that the winning state $\langle(\text{u\_nw}, \text{y\_w}), (\text{u\_ne}, \text{y\_n})\rangle$ is attained by the next transition. Likewise, the initial state is a winning state: by applying the control pattern $\{(\text{u\_nw}, y) | y \in Y\}$ we either have $\langle(\text{u\_nw}, \text{y\_n})\rangle$ or $\langle(\text{u\_nw}, \text{y\_w})\rangle$, both known to be winning states by our previous observations. Thus, the supervisory control problem can be solved based on the abstraction. In contrast, if the control objective was to eventually visit the west guard $G_{\text{y\_w}}$, the provided abstraction is too coarse for a positive result — although intuition suggests that the actual plant can be very well controlled accordingly.

6 Guided refinements of experiments

We now consider the situation where abstraction-based synthesis of a supervisor as discussed in the previous section has failed, i.e., we are given a plant $\Sigma = (\mathbb{N}_0, W, \mathcal{B})$ realised by a I/S- machine P and an abstraction $\Sigma_S = (\mathbb{N}_0, W, \mathcal{B}_S)$ obtained from an experiment S on $\mathcal{B}$, but applying Algorithm 12 shows that $q_0 \notin Q_{\text{win}}$ for the winning states $Q_{\text{win}} \subseteq Q$ of the composed state machine $P_\times = (Q, W, \lambda, \{q_0\})$. Provided that we are optimistic about the control problem with the actual plant to exhibit a solution, it is proposed to refine the abstraction and to repeat the synthesis procedure. Referring to Moor and Raisch (1999) and Moor et al. (2002), where the abstraction used is an l-complete approximation, i.e., $S = \mathcal{B}|_{[0,l]}$ for some $l \in \mathbb{N}_0$, a refinement can be obtained by substituting l with $l + 1$. Effectively, this uniformly extends the sampled sequences in length by one more symbol. However, such an extension amounts to testing whether or not the extended sequence is in pre $\mathcal{B}$, and this test is implemented as a one-step reachability analysis conducted on the original system. Since this is considered computationally expensive, we seek to identify specific sequences $R \subseteq S$ that are worth the effort and use Eq. 7 to obtain a refinement of S tailored for the synthesis task at hand. The overall abstraction-based approach then becomes an iteration in which we alternate trial synthesis and abstraction refinement.

Algorithm 13 Iterative procedure to synthesise a supervisor for an $I/S/-$ machine $P = (X, W, \delta, X_0)$, $W = U \times Y$, $X = X_0$, and a past-induced specification $P_{\text{spec}} = (X_{\text{spec}}, W, \delta_{\text{spec}}, \{x_{\text{spec0}}\})$ with accepting states $X_{\text{specM}} \subseteq X_{\text{spec}}$.

1) Initialise the experiment $S \subseteq W^*$ by $S := \mathcal{B}|_{[0,0]}$, where $\mathcal{B}$ denotes the external behaviour induced by P.
2) Referring to Theorem 7, set up P_S to realise the abstraction obtained from S.
3) Run Algorithm 12 on the product $P_\times = P_S \times P_{\text{spec}}$ to obtain the winning states Q_{win}.

4) If the initial state q_0 of $P_\times$ is within Q_{win}, report the corresponding supervisor and terminate the iteration.
5) Choose refinement candidates $R \subseteq S$ to obtain a refinement by Eq. 7 to substitute S, and proceed with Step 2.

In the case that the procedure terminates at Step 4, we refer to the discussion of the previous section and recall that the supervisor not only solves the synthesis problem for the abstraction P_S but also for the actual plant P. Otherwise, the experiment is refined in Step 5 for the subsequent trial synthesis. The proposed iteration may fail to terminate regardless of the choice for the refinement in Step 5. This is to be expected: since the verification of language inclusion is known to be only semi-decidable even for restricted classes of hybrid systems, the synthesis problem cannot be decidable either. However, we will propose a refinement scheme for Step 5 that ensures termination under the hypothesis of the existence of some experiment for which synthesis succeeds. We are now left to set up sensible refinement candidates R to implement Step 5.

For our analysis, we inspect the composed system $P_\times = (Q, W, \lambda, \{q_0\}) := P_S \times P_{\text{spec}}$, with lifted marked states $Q_{\text{M}} \subseteq Q$ for the specification acceptance condition and winning states $Q_{\text{win}} \subseteq Q$ obtained by the synthesis algorithm. A refinement obtained by extending specific samples $s \in S$ then corresponds to extending the transition relation λ at states $q = (z, \chi) \in Z_S \times X_{\text{spec}} = Q$ with $z = s$. Hence, our inspection of $P_\times$ focuses attention on states in

$$Q_{\text{leaf}} := \{(z, \chi) \in Q | z \in S\}, \tag{30}$$

and we will identify two classes of states that in turn characterise sequences $s \in S$ that are not worth a refinement. For our formal argument, we consider two more experiments S' and S'' on $\mathcal{B}$ such that $S \leq S' \leq S''$. Here, we assume that S'' is a successful refinement of S in the sense that there exists a supervisor f'' such that the closed-loop $\mathcal{B}_{S'', f''}$ satisfies the specification. We then construct S' to refine S in the same way as S'' except for avoiding refinement at a specific sequence $s \in S$, see Fig. 8. Technically, we let

$$S' := \{s\} \cup \{r \in S'' | s \cap (\text{pre}\, r) = \emptyset\} \tag{31}$$

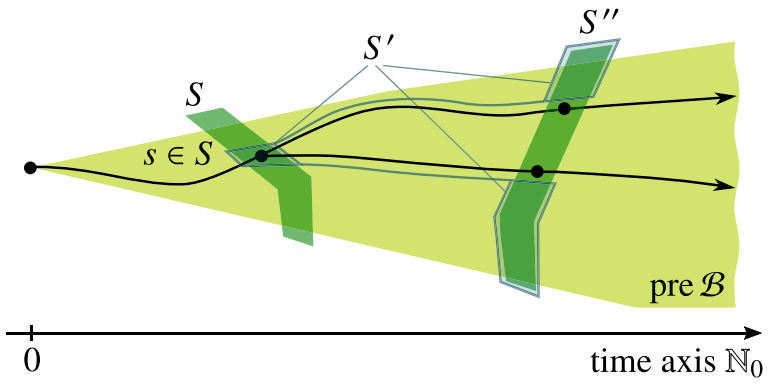

Fig. 8 Intermediate experiment S' with $S \leq S' \leq S''$ by not refining at $s \in S$. Both experiments S and S'' are shown as "barriers" (dark green) in the computational tree pre $\mathcal{B}$ (light green). The intermediate experiment S' lies between S and S'' and consists of (a) the sequence s and (b) all sequences $r \in S''$ except for extensions of s

to observe that S' is indeed an experiment on $\mathcal{B}$ and that $S \leq S''$ implies $S \leq S' \leq S''$. We then show that S' is also a successful refinement of S and thereby establish that the synthesis problem can be solved without refinements at the previously identified sequence $s \in S$.

For the remainder of this section, we refer to the synthesis problems based on the experiments S' and S'' by the same notational conventions as introduced for S, i.e., we denote the associated abstractions $\Sigma_{S'} = (\mathbb{N}_0, W, \mathcal{B}_{S'})$ and $\Sigma_{S''} = (\mathbb{N}_0, W, \mathcal{B}_{S''})$, the realisations thereof $P_{S'} = (Z_{S'}, W, \delta_{S'}, \{\epsilon\})$ and $P_{S''} = (Z_{S''}, W, \delta_{S''}, \{\epsilon\})$, the composed state machines $P'_\times = (Q', W, \lambda', \{q_0\})$ and $P''_\times = (Q'', W, \lambda'', \{q_0\})$, the lifted marked states Q'_{M} and Q''_{M}, and the winning states Q'_{win} and Q''_{win} as obtained by Algorithm 12, respectively.

6.1 Winning states

Once the abstraction Σ_S has generated a finite sequence s that drives the composed state machine $P_\times$ to a winning state, there exists a supervisor that enforces the specification from then on. Intuitively, for such states, no refinement is necessary.

For a formal argument, fix any $z = s \in S \subseteq Z_S$ such that

$$\{(z, \chi) | \chi \in X_{\mathrm{spec},s}\} \subseteq Q_{\mathrm{win}}, \tag{32}$$

where

$$X_{\mathrm{spec},s} := \{\chi \in X_{\mathrm{spec}} | \exists u \in W^* \colon \langle u, s\rangle \in \mathrm{pre}\, \mathcal{B}_S,\ \chi \in \delta_{\mathrm{spec}}(X_0, \langle u, s\rangle)\}. \tag{33}$$

Recall that we have synthesised a supervisor f that enforces the *conditional* specification (29) with $\mathcal{B}_S$ as the plant. Moreover, by hypothesis, there exists a supervisor f'' for the refined abstraction $\mathcal{B}_{S''}$ such that the closed loop satisfies $\mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{\mathrm{spec}}$. In order to establish the existence of a supervisor that enforces the specification for $\mathcal{B}_{S'}$ with the relaxed refinement S' in Eq. 31, we use the candidate $f' : W^* \to \Gamma$ defined by

$$f'(r) := \begin{cases} f''(r) & \text{if } r \neq \langle u, s, t\rangle \text{ for all } u,\ t \in W^*, \text{ or,} \\ f(\langle s, t\rangle) & \text{if } r = \langle u, s, t\rangle \text{ for some } u,\ t \in W^* \\ & \text{and } u \text{ chosen to be of minimum length,} \end{cases} \tag{34}$$

i.e., f' applies the same control patterns as f'' until the sequence s has been observed and, from then on, behaves as f in ignorance of any symbols generated before s. The intuition here is that if the closed loop formed by $\mathcal{B}_{S'}$ and f' happens to not generate s, then it evolves within $\mathcal{B}_{S''}$ and, hence, f'' enforces the specification. If, on the other hand, s is generated, this corresponds to a winning state of $P_\times$ and, hence, f enforces the specification. We obtain the following lemma.

Lemma 14 *Consider three experiments $S \leq S' \leq S''$ over the finite signal range $W = U \times Y$ with the respective associated abstractions $\Sigma_S = (\mathbb{N}_0, W, \mathcal{B}_S)$, $\Sigma_{S'} = (\mathbb{N}_0, W, \mathcal{B}_{S'})$ and $\Sigma_{S''} = (\mathbb{N}_0, W, \mathcal{B}_{S''})$, and with respective past-induced realisations P_S, $P_{S'}$ and $P_{S''}$ given by Theorem 7. Assume that S' relates to S and S'' as in* Eq. 31 *for some $s \in W^*$ that complies with* Eq. 32. *If there exists a supervisor f'' such that $\mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{\mathrm{spec}}$, then there also exists a supervisor f' such that $\mathcal{B}_{S',f'} \subseteq \mathcal{B}_{\mathrm{spec}}$.*

Proof We prove the existence by the candidate supervisor f' given in Eq. 34. To show $\mathcal{B}_{S',f'} \subseteq \mathcal{B}_{\mathrm{spec}}$, pick any $\mathbf{w} \in \mathcal{B}_{S',f'}$. We distinguish two cases.

First, assume that $\langle u, s\rangle \notin \mathrm{pre}\, \mathbf{w}$ for all $u \in W^*$. We then have $f(r) = f''(r)$ for all $r \in \mathrm{pre}\, \mathbf{w}$. Now pick arbitrary $k \in \mathbb{N}_0$ and refer to $\mathbf{w} \in \mathcal{B}_{S'}$ for the choice of $l \in \mathbb{N}_0$ such that $(\sigma^k \mathbf{w})|_{[0,l)} \in S'$. Here, the case hypothesis implies $(\sigma^k \mathbf{w})|_{[0,l)} \neq s$ and, hence $(\sigma^k \mathbf{w})|_{[0,l)} \in S''$. Since $k \in \mathbb{N}_0$ was arbitrary, we obtain $\mathbf{w} \in \mathcal{B}_{S''}$ to conclude with $\mathbf{w} \in \mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{\mathrm{spec}}$.

For the second case we pick the shortest sequence $u \in W^*$ such that $\langle u, s\rangle \in \operatorname{pre}\mathbf{w}$. We then have $(\sigma^{|u|}\mathbf{w})(k) = \mathbf{w}(|u|+k) \in f'(\mathbf{w}|_{[0,|u|+k)}) = f((\sigma^{|u|}\mathbf{w})|_{[0,k)})$ for all $k \geq |s|$. By $\mathbf{w} \in \mathcal{B}_{S',f'} \subseteq \mathcal{B}_S$ there uniquely exist state trajectories $\mathbf{z}\colon \mathbb{N}_0 \to Z_S$ and $\mathbf{x}\colon \mathbb{N}_0 \to X_{\text{spec}}$ such that $(\mathbf{w}, (\mathbf{z}, \mathbf{x}))$ is in the full behaviour induced by $P_\times$. In particular, we have that $\mathbf{x}(|\mathbf{u}| + |\mathbf{s}|) \in X_{\text{spec},s}$. By time invariance, we obtain $\mathbf{w}' := \sigma^k\mathbf{w} \in \sigma^k\mathcal{B}_S \subseteq \mathcal{B}_S$ and denote $\mathbf{z}'\colon \mathbb{N}_0 \to Z_S$ the unique state trajectory such that $(\mathbf{w}', \mathbf{z}')$ is in the full behaviour induced by P_S. Here, we observe that $\mathbf{z}'(|s|) = z = s$. With $\mathbf{x}' := \sigma^{|u|}\mathbf{x}$, we obtain a state trajectory $(\mathbf{z}', \mathbf{x}')$ with $((\mathbf{z}'(k), \mathbf{x}'(k)), \mathbf{w}'(k), (\mathbf{z}'(k+1), \mathbf{x}'(k+1)) \in \lambda$ for all $k \in \mathbb{N}_0$. By the choice of s in Eq. 32, we observe $(\mathbf{z}'(|s|), \mathbf{x}'(|s|)) \in Q_{\text{win}}$ and, referring to the case hypothesis, we also have $\mathbf{w}'(k) \in f(\mathbf{w}'|_{[0,k)})$ for all $k > |s|$. Therefore, there exist infinitely many $k > |s|$ such that $(\mathbf{z}'(k), \mathbf{x}'(k)) \in Q_{\text{M}}$ and, hence, $\mathbf{x}(|u|+k) = \mathbf{x}'(k) \in X_{\text{specM}}$. This implies $\mathbf{w} \in \mathcal{B}_{\text{spec}}$ and concludes the proof of $\mathcal{B}_{S',f'} \subseteq \mathcal{B}_{\text{spec}}$. □

6.2 Failing States

Denote $Q_{\text{fail}} \subseteq Q_{\text{leaf}}$ the set of *failing states*, i.e., states from which the accepting states Q_{M} are not reachable:

$$Q_{\text{fail}} := \{q \in Q_{\text{leaf}} | \lambda(q, W^+) \cap Q_{\text{M}} = \emptyset\}. \tag{35}$$

Obviously, a state $q \in Q_{\text{fail}}$ cannot be a winning state and, intuitively, it cannot become a winning state in any refinement. Therefore, a refinement at a failing state is not expected to be relevant for any solution of the control problem.

For our formal argument, fix any $z = s \in S \subseteq Z_S$ with

$$\{(z, \chi) | \chi \in X_{\text{spec},s}\} \subseteq Q_{\text{fail}} \tag{36}$$

where $X_{\text{spec},s}$ is defined in Eq. 33. By the following proposition, we associate with s a set of failing states in the composed state machine $P''_\times$ based on the refinement S''.

Proposition 15 *Consider two experiments $S \leq S''$ over $W = U \times Y$ with the respective associated abstractions Σ_S and $\Sigma_{S''}$, and with the respective past-induced realisations P_S and $P_{S''}$ given by Theorem 7. Referring to the composed state machine $P''_\times = P_{S''} \times P_{\text{spec}}$, let*

$$Q''_{\text{fail},s} := \{\xi \in Q'' | \exists t \in W^*, \chi \in X_{\text{spec}} \colon \xi = (\langle s, t\rangle, \chi)\} \subseteq Z_{S''} \times X_{\text{spec}} \tag{37}$$

for some $s \in W^*$ that complies with Eqs. 35 *and* 36*. Then any trajectory $(\mathbf{w}'', \mathbf{x}'')$ of the full behaviour induced by $P''_\times$ that passes $Q''_{\text{fail},s}$ does not satisfy the specification, i.e., if there exists $k \in \mathbb{N}_0$ with $\mathbf{x}(k) \in Q''_{\text{fail},s}$ then $\mathbf{w} \notin \mathcal{B}_{\text{spec}}$.*

Proof Consider a state $\xi \in Q''_{\text{fail},s}$, i.e., we have $\xi = (\langle s, t\rangle, \chi)$ for some $t \in W^*$ and some $\chi \in X_{\text{spec}}$. For a contradiction, assume that there exists a trajectory $(\mathbf{w}'', \mathbf{x}'')$ from the full behaviour induced by $P''_\times$ that first passes ξ and thereafter passes Q''_{M}. We can then find u, $v \in W^*$, $v \neq \epsilon$, such that $\langle u, s, t, v\rangle \in \operatorname{pre}\mathbf{w}''$, $\lambda''_0(\langle u, s, t\rangle) = \xi$, and $\lambda''_0(\langle u, s, t, v\rangle) \in Q''_{\text{M}}$. We denote the respective specification components of the state $\chi_u = \delta_{\text{spec},0}(u)$, $\chi_{us} = \delta_{\text{spec},0}(\langle u, s\rangle)$, $\chi_{ust} = \delta_{\text{spec},0}(\langle u, s, t\rangle) = \chi$ and $\chi_{ustv} = \delta_{\text{spec},0}(\langle u, s, t, v\rangle) \in X_{\text{specM}}$. By $\mathbf{w}'' \in \mathcal{B}_{S''} \subseteq \mathcal{B}_S$, we observe $\chi_{us} \in X_{\text{spec},s}$. By $\sigma^k\mathcal{B}_{S''} \subseteq \mathcal{B}_{S''} \subseteq \mathcal{B}_S$, for any $k \in \mathbb{N}_0$, we conclude that $\mathbf{w} := \sigma^{|u|}\mathbf{w}'' \in \mathcal{B}_S$. Hence, we can choose $\mathbf{z} : \mathbb{N}_0 \to Z_S$ such that $(\mathbf{w}, \mathbf{z})$ is in the full behaviour induced by P_S. Since the transition relation δ_{spec} is full, we can use $\mathbf{w}$ to generate a state trajectory for any initial state. In particular, there exists $\mathbf{x} : \mathbb{N}_0 \to X_{\text{spec}}$ such that $\mathbf{x}(0) = \chi_u$, $\mathbf{x}(|s|) = \chi_{us}$, $\mathbf{x}(|s| + |t|) = \chi_{ust}$, $\mathbf{x}(|s| + |t| + |v|) = \chi_{ustv}$ and $(\mathbf{x}(k), \mathbf{w}(k), \mathbf{x}(k+1)) \in \delta_{\text{spec}}$ for all $k \in \mathbb{N}_0$. Therefore, $(\mathbf{z}(|s|+|t|+|v|), \mathbf{x}(|s|+|t|+|v|))$ is

reachable from $(\mathbf{z}(|s|), \mathbf{x}(|s|))$ by transitions from λ. We observe $\mathbf{z}(|s|) = s = z$, and, hence $(\mathbf{z}(|s|), \mathbf{x}(|s|)) = (z, \chi_{us}) \in Q_{\text{fail}}$. This constitutes a contradiction with $\mathbf{x}(|s| + |t| + |v|) = \chi_{ustv} \in X_{\text{specM}}$. Therefore, no trajectory $(\mathbf{w}'', \mathbf{x}'')$ from the full behaviour induced by $P''_\times$ that passes ξ can pass Q''_{M} thereafter. □

Recall that, by hypothesis, there exists a supervisor f'' that when applied to $\Sigma_{S''}$ enforces the specification, i.e., $\mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{\text{spec}}$. Now consider for some $\mathbf{w}'' \in \mathcal{B}_{S'',f''}$ the unique state trajectory $\mathbf{x}''$ such that $(\mathbf{w}'', \mathbf{x}'')$ is in the full behaviour induced by $P''_\times$. We then conclude by the above proposition that $\mathbf{x}''$ does not pass $Q''_{\text{fail},s}$. Inspecting the definition of $Q''_{\text{fail},s}$ and the relaxed refinement S' in Eq. 31, this implies that f'' controls $P''_\times$ such that the set of reachable states is within $Z_{S'} \times X_{\text{spec}} \subset Z_{S''} \times X_{\text{spec}}$ This suggests that we may apply f'' to $\mathcal{B}_{S'}$ in order to obtain $\mathcal{B}_{S',f''} = \mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{\text{spec}}$. We provide a proof for this conjecture and obtain the following lemma.

Lemma 16 *Consider three experiments $S \leq S' \leq S''$ over the finite signal range $W = U \times Y$ with the respective associated abstractions $\Sigma_S = (\mathbb{N}_0, W, \mathcal{B}_S)$, $\Sigma_{S'} = (\mathbb{N}_0, W, \mathcal{B}_{S'})$ and $\Sigma_{S''} = (\mathbb{N}_0, W, \mathcal{B}_{S''})$, and with respective past-induced realisations P_S, $P_{S'}$ and $P_{S''}$ given by Theorem 7. Assume that S' relates to S and S'' as in* Eq. 31 *for some $s \in W^*$ that complies with* Eqs. 35 *and* 36*. Then $\mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{\text{spec}}$ for a supervisor f'' implies $\mathcal{B}_{S',f''} = \mathcal{B}_{S'',f''}$.*

Proof For a preliminary observation, denote $L_s := \{r \in W^* | s \notin \text{pre}\, r\}$ the set of all sequences that do not pass s. We then have $Q' \subset Q'' \subset Q''_{\text{fail},s} \dot{\cup} (L_s \times X_{\text{spec}})$. By Eq. 31, we further obtain $S' \cap L_s = S'' \cap L_s$, and, referring to the realisations by Theorem 7, $\delta_{S'} \cap (L_s \times W \times L_s) = \delta_{S''} \cap (L_s \times W \times L_s)$, and, hence, $\lambda' \cap ((L_s \times X_{\text{spec}}) \times W \times (L_s \times X_{\text{spec}})) = \lambda'' \cap ((L_s \times X_{\text{spec}}) \times W \times (L_s \times X_{\text{spec}}))$. In other words, the realisations $P'_\times$ and $P''_\times$ coincide when restricting the respective state set to $L_s \times X_{\text{spec}}$.

We always have $\mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{S',f''}$ directly from Definition 11. For the converse inclusion, pick any closed-loop trajectory $\mathbf{w}' \in \mathcal{B}_{S',f''}$, and let $\mathbf{x}'$ denote the unique corresponding state trajectory such that $(\mathbf{w}', \mathbf{x}')$ is in the full behaviour of $P'_\times$. For a contradiction, assume that there exists $k \in \mathbb{N}_0$ such that $\mathbf{x}'(k) \notin L_s \times X_{\text{spec}}$ and we pick the smallest such k. In particular, we have $\mathbf{x}'(k) \in Q''_{\text{fail},s}$. Referring to the input-output structure and the corresponding choice of control patterns, we can then construct a trajectory $(\mathbf{w}'', \mathbf{x}'')$ in the full behaviour of $P''_\times$ such that $\mathbf{x}''|_{[0,k]} = \mathbf{x}'|_{[0,k]}$ and $\mathbf{w}'' \in \mathcal{B}_{S'',f''}$. However, by Proposition 15, we have that $\mathbf{x}''(k) \in Q''_{\text{fail},s}$ implies $\mathbf{w}'' \notin \mathcal{B}_{\text{spec}}$ to constitute a contradiction to $\mathcal{B}_{S'',f''} \subseteq \mathcal{B}_{\text{spec}}$. Therefore, we have that $\mathbf{x}'(k) \in L_s \times X_{\text{spec}}$ for all $k \in \mathbb{N}_0$. Then, $(\mathbf{w}', \mathbf{x}')$ must be in the full behaviour of $P''_\times$ and, hence, $\mathbf{w}' \in \mathcal{B}_{S''}$. Together with $\mathbf{w}' \in \mathcal{B}_{S',f''}$ this implies $\mathbf{w}' \in \mathcal{B}_{S'',f''}$. By the arbitrary choice of $\mathbf{w}' \in \mathcal{B}_{S',f''}$, we conclude $\mathcal{B}_{S',f''} \subseteq \mathcal{B}_{S'',f''}$. □

6.3 Main result

Considering the two classes of states identified above, we propose the refinement candidates

$$R = \{s \in S | \exists \chi \in X_{\text{spec},s} \colon (s, \chi) \notin Q_{\text{fail}} \cap Q_{\text{win}}\} \tag{38}$$

for Step 5 of Algorithm 13 and state our main result.

Theorem 17 *Given a time invariant plant $\Sigma = (\mathbb{N}_0, W, \mathcal{B})$, $W = U \times Y$, realised by an I/S/- machine $P = (X, W, \delta, X)$ with finite input range U and finite output range*

Y, consider the supervisory controller synthesis problem for a specification $\Sigma_{\text{spec}} = (\mathbb{N}_0, W, \mathcal{B}_{\text{spec}})$ *realised by a past-induced finite state machine* $P_{\text{spec}} = (X_{\text{spec}}, W, \delta, X_{\text{spec0}})$ *with Büchi acceptance condition* $X_{\text{specM}} \subseteq X_{\text{spec}}$*; see* Eq. 28. *Assume that there exists an experiment* $S_* \subseteq W^*$ *on* $\mathcal{B}$ *with associated abstraction* $\Sigma_{S_*} = (\mathbb{N}_0, W, \mathcal{B}_{S_*})$ *and a supervisor* $f_* : W^* \to \Gamma$ *such that the closed-loop behaviour* $\mathcal{B}_{S_*, f_*}$ *obtained from* $\mathcal{B}_{S_*}$ *under supervision* f_* *satisfies the specification; i.e.,* $\mathcal{B}_{S_*, f_*} \subseteq \mathcal{B}_{\text{spec}}$*. Then Algorithm 13 with refinement candidates in* Eq. 38 *terminates with success after finitely many iterations.*

Proof For a proof by contradiction, assume the algorithm does not terminate. Denote $S_j \subseteq W^*$ the experiment in the j-th iteration with refinement candidates $R_j \subseteq S_j$ identified by Eq. 38. We then have that $S_j \leq S_{j+1}$ and $S_j \neq S_{j+1}$ for all $j \in \mathbb{N}_0$. This implies $\text{pre}\, S_j \subsetneq \text{pre}\, S_{j+1}$ for all $j \in \mathbb{N}_0$. Since the signal range is finite, we can choose a sufficiently large $j \in \mathbb{N}_0$ such that $(\text{pre}\, S_*) \cap (\text{pre}\, S_j) = (\text{pre}\, S_*) \cap (\text{pre}\, S_{j+1})$. This implies $(\text{pre}\, S_*) \cap S_j = (\text{pre}\, S_*) \cap S_{j+1}$. Referring to the general scheme of refinement Eq. 7, we obtain that $(\text{pre}\, S_*) \cap R_j = \emptyset$. We now construct one more experiment on $\mathcal{B}$:

$$S'' := \{s \in S_* | (\text{pre}\, s) \cap S_j \neq \emptyset\} \cup \{s \in S_j | (\text{pre}\, s) \cap S_* \neq \emptyset\}, \tag{39}$$

such that $S_* \leq S''$, $S_j \leq S''$ and $R_j \subseteq S''$; see Fig. 9.

We denote the associated behaviour $\mathcal{B}_{S''}$, where $S_* \leq S''$ implies $\mathcal{B}_{S''} \subseteq \mathcal{B}_{S_*}$. Thus, we can formally interpret $\mathcal{B}_{S_*}$ as a behavioural abstraction of $\mathcal{B}_{S''}$. In particular, the existence of a solution to the control problem for Σ_{S_*} implies the existence of a solution for $\Sigma_{S''} = (\mathbb{N}_0, W, \mathcal{B}_{S''})$. We now turn to the ordering $S_j \leq S''$. Since we have identified S'' as a successful experiment, Lemmata 14 and 16 grant the existence of a third experiment S' such that (a) $S_j \leq S' \leq S''$, (b) $S_j - S' \subseteq R_j$ and (c) the control problem exhibits a solution for the abstraction $\Sigma_{S'} = (\mathbb{N}_0, W, \mathcal{B}_{S'})$ obtained from S'. By clause (b) and the construction of S'' we conclude $S_j = S'$. Hence, S_j itself must be a successful experiment. This constitutes a contradiction and concludes the proof. □

Note that the above theorem refers to the existence of a successful experiment S_*. However, S_* does not need to be known explicitly in order to run Algorithm 13. In other words,

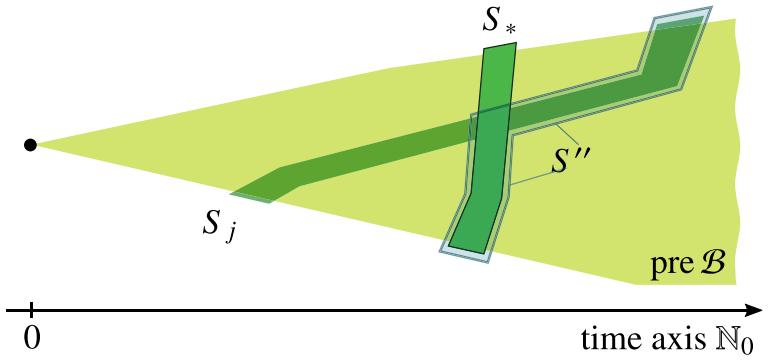

Fig. 9 Experiment S'' with $S_* \leq S''$ and $R_j \subseteq S''$. All three experiments are again shown as "barriers" in the computational tree pre$\mathcal{B}$. By construction, S'' consists of (a) sequences from S_* and (b) extensions of sequences from S_* within S_j. In particular, S'' is a refinement of S_*. By our choice of a sufficiently large $j \in \mathbb{N}_0$, all refinement candidates $R_j \subseteq S_j$ must be on the right from S_* and therefore, by construction, within S''

we still have the expected situation of semi-decidability, but we also have the guarantee that the algorithm terminates with success as long as a successful experiment exists.

Example 6 **(cont.)** Recall that for the vehicle navigation example the abstraction obtained from S in Fig. 5 is too coarse for controller synthesis when the objective is to eventually visit the west guard $G_{\mathsf{y\_w}}$. The worst case for this control objective is an initial state at the very east of the area A which requires a number of successive u_nw and u_sw control symbols, where the exact number depends on the with-to-height ratio of the rectangular area. Using a width of $w = 30$ units versus a height of $h = 10$ units, a thickness of $o = 1$ unit for the guards, a nominal velocity of $v = 10$ and a disturbance of diameter $d = 2$, a numerical simulation suggests that it can take up to 6 control inputs to reach the guard $G_{\mathsf{y\_w}}$. Thus, for a successful experiment we expect the longest sequences to be of length 6. On the other hand, any sequence that contains a y_w symbol corresponds to a set of winning states in $P_\times$ and, thus, needs no more refinement.

Running Algorithm 13 with refinement candidates (38) yields a successful experiment S with 33262 sequences of length ranging from 1 to 6. We can also use the a-priori knowledge that it makes no sense to apply the same input symbol twice in row and encode this in the specification automation P_{spec}. Algorithm 13 then also encounters non-trivial sets of failing states and constructs a successful experiment S with only 9020 sequences. Since a winning state must be a predecessor of a winning state, we can prioritise our refinement candidates (38) accordingly, i.e., we prefer to refine sequences in the experiment that correspond to a state $q = (z, \chi) \in Q$ with a successor state in Q_{win}. This again reduces the number of sequences constructed by Algorithm 13 to 164. All three figures compare favorable to the strongest 5-complete approximation, which is constructed from the 59304 sequences found in $\mathcal{B}_{|[0,5]}$.

7 Conclusion

We have addressed a question arising in abstraction-based control of hybrid systems. More specifically, the l-complete approximation scheme (Moor and Raisch 1999; Moor et al. 2002) provides a sequence of finite state abstractions P_l, $l = 1, 2, \ldots$, for a (possibly infinite state) system with discrete-valued external signals. Here, the parameter l corresponds to the uniform length of finite sequences $S \subseteq W^{l+1}$ sampled from the external behaviour of the hybrid plant when constructing the finite state abstraction. In this scheme, a refinement amounts to incrementing the parameter l. Although the required number of samples $|S|$ in most applications is by orders of magnitude less than the worst case $|W|^{l+1}$, it is still exponential in l. In this paper, we have further developed the approach proposed by Moor et al. (2006) in that we drop the requirement of a uniform length and consider so called experiments $S \subseteq W^*$, from which we construct the abstraction finite state machine P_S. In particular, this approach allows us to increase the length of the samples locally as required for the specific synthesis task at hand. In order to identify the sequences that are worth the effort of a refinement, we first apply the synthesis procedure provided by Thistle and Wonham (1994a, b) on the product $P_\times := P_S \times P_{\text{spec}}$, where P_{spec} realises the specification which the supervisor is meant to enforce. If the procedure succeeds and returns a supervisor to control P_S, this supervisor is known to also enforce the specification when applied to the hybrid plant. If, on the other hand, the control problem cannot be completely solved for the abstraction P_S, the procedure still establishes a conditional solution, i.e., a supervisor that enforces the specification provided that the state trajectory passes through the set of

so called winning states Q_{win}. Intuitively, sequences in S that correspond to winning states do not need to be increased in length. In contrast to the winning states, we also identify a set of failing states Q_{fail}, i.e., states such that any trajectory that passes through is known to violate the specification under any supervisor. Again, sequences in S that correspond to failing states do not need to be increased in length. To this end, we refine the abstraction S by increasing the sample length only for sequences that correspond neither to winning states nor to failing states to obtain the locally refined experiment S' with associated abstraction $P_{S'}$. Iterating abstraction refinement and trial synthesis, we obtain a sequence of experiments S_j, $j = 1, 2, \ldots$. By our main technical result, this iteration will not miss out on solutions to the control problem: if there exists an experiment such that the control problem can be solved based on the associated abstraction, then the iteration will also produce a successful experiment S_j for some $j \in \mathbb{N}$.

Funding Information Open Access funding provided by Projekt DEAL.

References

Althoff M, Stursberg O, Buss M (2010) Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes. Nonlinear Analysis: Hybrid Systems 4(2):233–249

Alur R, Henzinger TA, Ho P-H (1996) Automatic symbolic verification of embedded systems. IEEE Trans Softw Eng 22(3):181–201

Alur R, Henzinger T, Lafferriere G, Pappas G (2000) Discrete abstractions of hybrid systems. Proc IEEE 88(7):971–984

Bagnara R, Hill PM, Zaffanella E (2008) The Parma polyhedra library: toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems. Sci Comput Program 72(1–2):3–21

Chutinan A, Krogh BH (1998) Computing polyhedral approximations to flow pipes for dynamic systems. In: IEEE 37th international conference on decision and control, pp 2089–2094

Clarke E, Fehnker A, Han Z, Krogh B, Ouaknine J, Stursberg O, Theobald M (2003) Abstraction and counterexample-guided refinement in model checking of hybrid systems. Int J Found Comput Sci 14(4):583–604

Frehse G (2008) PHAVEr: algorithmic verification of hybrid systems past HyTech. Int J Softw Tools Technol Transfer 10(3):263–279

Gol EA, Ding X, Lazar M, Belta C (2014) Finite bisimulations for switched linear systems. IEEE Trans Autom Control 59(12):3122–3134

Halbwachs N, Proy Y-E, Roumanoff P (1997) Verification of real-time systems using linear relation analysis. Formal Methods in System Design 11:157–185

Henzinger TA (2000) The theory of hybrid automata. In: Inan MK, Kurshan RP (eds) Verification of digital and hybrid systems. NATO ASI Series (Series F: Computer and Systems Sciences), vol 170. Springer

Henzinger TA, Horowitz B, Majumdar R, Wong-Toi H (2000) Beyond HyTech: Hybrid systems analysis using interval numerical methods. Hybrid Systems: Computation and Control. LNCS 1790:130–144

Lafferriere G, Pappas GJ, Sastry S (2000) O-minimal hybrid systems. Mathematics of Control, Signals and Systems 13(1):1–21
Liu J, Ozay N (2016) Finite abstractions with robustness margins for temporal logic-based control synthesis. Nonlinear Analysis: Hybrid Systems 22:1–15
Maler O, Dang T (1998) Reachability analysis via face lifting. Hybrid Systems: Computation and Control. LNCS 1386:96–109
Mitchell IM, Bayen AM, Tomlin CJ (2005) A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. IEEE Trans Autom Control 50(7):947–957
Moor T, Davoren JM, Raisch J (2006) Learning by doing: systematic abstraction refinement for hybrid control synthesis. IEE Proceedings: Control Theory and Applications 153(5):591–599
Moor T, Götz S (2018) Deterministic finite-automata abstractions of time-variant sequential behaviours. In: 14th Workshop on discrete event systems (WODES'18), p 399
Moor T, Raisch J (1999) Supervisory control of hybrid systems within a behavioural framework. Syst Control Lett 38(3):157–166
Moor T, Raisch J (2002) Abstraction based supervisory controller synthesis for high order monotone continuous systems. Modelling, Analysis, and Design of Hybrid Systems, LNCIS 279, pp 247–265
Moor T, Raisch J, O'Young S (2002) Discrete supervisory control of hybrid systems based on l-complete approximations. Discrete Event Dynamic Systems: Theory and Applications 12(1):83–107
Park SJ, Raisch J (2015) Supervisory control of hybrid systems under partial observation based on l-complete approximations. IEEE Trans Autom Control 60(5):1404–1409
Pola G, Tabuada P (2009) Symbolic models for nonlinear control systems: alternating approximate bisimulations. SIAM J Control Optim 48(2):719–733
Ramadge PJ, Wonham WM (1987) Supervisory control of a class of discrete event systems. SIAM J Control Optim 25(1):206–230
Ramadge PJ, Wonham WM (1989) The control of discrete event systems. Proc IEEE 77(1):81–98
Reissig G (2011) Computing abstractions of nonlinear systems. IEEE Trans Autom Control 56(11):2583–2598
Reissig G, Weber A, Rungger M (2017) Feedback refinement relations for the synthesis of symbolic controllers. IEEE Trans Autom Control 62(4):1781–1796
Schmuck AK, Raisch J (2014) Asynchronous l-complete approximations. Syst Control Lett 73:67–75
Schmuck AK, Tabuada P, Raisch J (2015) Comparing asynchronous l-complete approximations and quotient based abstractions. In: IEEE 54th international conference on decision and control, pp 6823–6829
Stiver JA, Antsaklis PJ, Lemmon MD (1995) Interface and controller design for hybrid systems. Hybrid Systems II. LNCS 999:462–492
Stursberg O (2006) Supervisory control of hybrid systems based on model abstraction and guided search. Nonlinear Anal Theory Methods Appl 65(6):1168–1187
Tabuada P (2009) Verification and control of hybrid systems: a symbolic approach. Springer, New York
Thistle JG, Wonham WM (1994a) Control of infinite behavior of finite automata. SIAM J Control Opt 32(4):1075–1097
Thistle JG, Wonham WM (1994b) Supervision of infinite behavior of discrete event systems. SIAM J Control Optim 32(4):1098–1113
Willems JC (1991) Paradigms and puzzles in the theory of dynamic systems. IEEE Trans Autom Control 36(3):258–294
Yang JM, Moor T, Raisch J (2018) Local refinement of l-complete approximations for supervisory control of hybrid systems. In: 14th Workshop on discrete event systems (WODES'18), p 496
Zamani M, Pola G, Mazo M, Tabuada P (2012) Symbolic models for nonlinear control systems without stability assumptions. IEEE Trans Autom Control 57(7):1804–1809

Jung-Min Yang received his B.S., M.S., and Ph.D. degrees in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1993, 1995, and 1999, respectively. Since September 2013, he has been with the School of Electronics Engineering, Kyungpook National University, Korea, where he is currently a Professor. His research interests include control of asynchronous sequential machines, systems biology, and control of hybrid systems.

Thomas Moor received his PhD degree (Dr.-Ing.) in 1999 from the University of the Federal Armed Forces Hamburg. From 2000 to 2003 he was a research fellow with the Research School of Information Sciences and Engineering at the Australian National University. Since 2003, he holds a professorship at the Lehrstuhl für Regelungstechnik, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany. His research interests include the control of discrete-event systems and hybrid systems, hierarchical and/or modular control systems, control system abstraction and faulttolerant control. He serves on the Editorial Board of the Journal of Discrete Event Dynamic Systems and Nonlinear Analysis: Hybrid Systems. He is maintainer and principle developer of the discrete-event systems software library libFAUDES, with a particular focus on supervisory control in an industrial application context.

Jörg Raisch studied Engineering Cybernetics at Stuttgart University and Control Systems at UMIST, Manchester. He received a PhD and a Habilitation degree, both from Stuttgart University. He holds the chair for Control Systems in the EECS Department at TU Berlin, and he is also an external scientific member of the Max Planck Institute for Dynamics of Complex Technical Systems. His main research interests are hybrid and hierarchical control, distributed cooperative control, and control of timed discrete event systems in tropical algebras, with applications in chemical, medical, and power systems engineering. He was on the editorial boards of the European Journal of Control, the IEEE Transactions on Control Systems Technology, and Automatica. He serves on the editorial boards of Discrete Event Dynamic Systems and Foundations and Trends in Systems and Control.