Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

An Efficient Algorithm for the Computation of the Controllability Prefix of $*$-Languages

Thomas Moor [*] **Klaus Werner Schmidt** [**] **Anne-Kathrin Schmuck** [***]

[*] *Lehrstuhl für Regelungstechnik,*
Friedrich-Alexander Universität Erlangen-Nürnberg, Germany
(e-mail: lrt@fau.de)
[**] *Department of Electrical and Electronics Engineering,*
Middle East Technical University, Ankara, Turkey
(e-mail: schmidt@metu.edu.tr)
[***] *Max Planck Institute for Software Systems, Kaiserslautern, Germany*
(e-mail: akschmuck@mpi-sws.org)

Abstract: Given a plant and a specification, both represented as formal languages, the *controllability prefix* is defined as the set of event sequences from which on a supervisor can control the plant according to the specification. The controllability prefix was first introduced in the context of ω-languages, where it plays a crucial role in the solution of the supervisory controller synthesis problem. In the present paper, we address the controllability prefix for $*$-languages. In our discussion, we (a) present a novel characterisation of the supremal controllable and relatively closed sublanguage in terms of the controllability prefix; we (b) derive a fixpoint characterisation of *winning states* from a game theoretic interpretation of a specific state feedback synthesis problem; and (c) we establish a one-to-one correspondence between winning states and the controllability prefix. In summary, we obtain an efficient algorithm for the computation of the controllability prefix.

Keywords: Discrete-event systems, supervisory control, supremal controllable sublanguage, controllability prefix.

INTRODUCTION

A discrete-event system in closed-loop configuration, as originally introduced by Ramadge and Wonham (1987, 1989), can be interpreted as a two player game. From this perspective, each turn consists of two moves in which the *supervisor* (player 0) applies a *control pattern* to specify the set of enabled events and the *plant* (player 1) subsequently generates one such event. The objective of the supervisor is to enforce a safety specification while avoiding livelocks and deadlocks. A *player-0-winning configuration* is a sequence of past events such that the supervisor can henceforth organise its moves to meet its objective regardless the subsequent moves of the plant. Following Thistle and Wonham (1994b), we refer to the set of all player-0-winning configurations as the *controllability prefix* of a specification w.r.t. a given plant.

Taking a game theoretic perspective is common in the field of *reactive synthesis*, which specifically addresses not only safety specifications but also general liveness specifications representable as ω*-languages*, i.e., languages of infinite-length words; see Finkbeiner (2016). Regarding supervisory control, ω-languages are addressed by Thistle and Wonham (1994b). In this setting, a supremal achievable closed-loop behaviour does in general not exist, however, a tight upper bound can be stated in terms of the controllability prefix. Thus, the controllability prefix plays a key role in the solution of the synthesis problem for supervisory control of ω-languages. Notably, this branch of supervisory control is closely related to reactive synthesis; see (Ehlers et al., 2017; Schmuck et al., 2020).

In this paper, we address supervisory control in the original setting introduced by Ramadge and Wonham (1987, 1989), where the plant and the specification are representable as $*$-languages, i.e., languages of finite-length words. Here, the supremal achievable closed-loop behaviour does exist and, for regular problem parameters, it can be computed by well established procedures. If the supremal closed-loop behaviour turns out non-empty, a suitable supervisor can be extracted and the problem is solved. If, on the other hand, the supremal closed-loop behaviour turns out empty, no solution exists. The latter case motives our study of the controllability prefix in the context of $*$-languages: while the "empty-set result" provides no further insight in how the problem parameters can be modified in order to obtain a solvable synthesis problem, such guidance can be extracted from the controllability prefix. This is of interest, e.g., in the context of abstraction based controller design. In this approach, the plant is substituted by an abstraction and, if synthesis fails for the abstraction, the question arises in which regard the abstraction should be refined in order to obtain a solvable synthesis problem; see e.g. (Yang et al., 2020) for abstraction based controller synthesis for hybrid systems with abstraction refinement based on the controllability prefix. Another option in the situation of failed supervisory controller synthesis is to relax the specification. This is of interest, e.g., in the context of fault-tolerant control. The question here is, when a nominal specification can not be enforced under consideration of a specific fault, how this specification can be strategically relaxed in order to obtain a solvable synthesis problem; see (Moor and Schmidt, 2015, 2017) for a discussion of this situation which utilises the controllability prefix.

The paper is organised as follows. After introducing relevant notation, Section 1, we recall the formal definition of the controllability prefix and present an alternative characterisation of the supremal achievable closed-loop behaviour in Section 2. As a trivial consequence of this discussion, we obtain a first procedure for the computation of the controllability prefix, however, with cubic complexity. The language based perspective is complemented by Section 3 in which we discuss the synthesis of non-blocking state feedback and, taking a game theoretic perspective, characterise the set of *winning states* by a two-nested fixpoint formula. Finally, in Section 4, we establish a one-to-one correspondence of winning states and the controllability prefix. Thus, the fixpoint identified in Section 3 can be used to effectively compute the controllability prefix with quadratic complexity. This constitutes our main result.

1. PRELIMINARIES AND NOTATION

We largely follow the same notational conventions as Cassandras and Lafortune (2008); Wonham and Cai (2019).

Let Σ be a *finite alphabet*, i.e., a finite set of symbols $\sigma \in \Sigma$. The *Kleene-closure* Σ^* is the set of finite strings $s = \sigma_1\sigma_2\cdots\sigma_n$, $n \in \mathbb{N}$, $\sigma_i \in \Sigma$, and the *empty string* $\epsilon \in \Sigma^*$, $\epsilon \notin \Sigma$. If, for two strings $s, r \in \Sigma^*$, there exists $t \in \Sigma^*$ such that $s = rt$, we say r is a *prefix* of s, and write $r \leq s$; if in addition $r \neq s$, we say r is a *strict prefix* of s and write $r < s$. Given two equivalence relations $\equiv_1$ and $\equiv_2$ on Σ^*, we say that $\equiv_1$ is *at least as fine as* $\equiv_2$ if, for all s' $s'' \in \Sigma^*$, $s' \equiv_1 s''$ implies that $s' \equiv_2 s''$.

A $*$-*language* (or short a *language*) over Σ is a subset $L \subseteq \Sigma^*$. The *prefix* of a language $L \subseteq \Sigma^*$ is defined by $\mathrm{pfx}\, L := \{r \in \Sigma^* \mid \exists\, s \in L : r \leq s\}$. The prefix operator is also referred to as the *prefix-closure*, and, a language L is *closed* if $L = \mathrm{pfx}\, L$. A language K is *relatively closed w.r.t. L* if $K = (\mathrm{pfx}\, K) \cap L$. The prefix operator distributes over arbitrary unions of languages. However, for the intersection of two languages L and M, we have $\mathrm{pfx}\,(L \cap M) \subseteq (\mathrm{pfx}\, L) \cap (\mathrm{pfx}\, M)$. If equality holds, L and M are said to be *non-conflicting*. Given a language $L \subseteq \Sigma^*$, the *Nerode equivalence* $\equiv_L$ is an equivalence relation on Σ^* defined by $s' \equiv_L s''$ if and only if $(\forall\ t \in \Sigma^*)[\, s't \in L \leftrightarrow s''t \in L\,]$. Given two languages $K \subseteq L \subseteq \Sigma^*$, and a set of *uncontrollable events* $\Sigma_{\mathrm{uc}} \subseteq \Sigma$, we say K is *controllable w.r.t. L*, if $(\mathrm{pfx}\, K)\Sigma_{\mathrm{uc}} \cap (\mathrm{pfx}\, L) \subseteq \mathrm{pfx}\, K$. Controllability, closedness and relative closedness are retained under arbitrary union.

An *automaton* is a tuple $G = (Q, \Sigma, \delta, q_{\mathrm{o}}, Q_{\mathrm{m}})$, with *state set Q*, *initial state $q_{\mathrm{o}} \in Q$*, *marked states $Q_{\mathrm{m}} \subseteq Q$*, and the partial *transition function $\delta : Q \times \Sigma \to Q$*. We write $\delta(q, s)!$ to indicate that δ is defined for the specified arguments $q \in Q$ and $s \in \Sigma^*$. The automaton is *full* if $\delta(q, \sigma)!$ for all $q \in Q$ and $\sigma \in \Sigma$, i.e., if δ is a proper function. We identify δ with its common extension to the domain $Q \times \Sigma^*$; i.e., for $q \in Q$, we have $\delta(q, \epsilon) = q$ and, for $s \in \Sigma^*$ and $\sigma \in \Sigma$, we have $\delta(q, s\sigma)!$ with $\delta(q, s\sigma) = \delta(\delta(q, s), \sigma))$ if and only if $\delta(q, s)!$ and $\delta(\delta(q, s), \sigma))!$. For set-valued arguments we identify δ with the associated image operator; e.g., with $\gamma \subseteq \Sigma$ we use the notation $\delta(q, \gamma) = \{\, q' \in Q \mid \exists\, \sigma \in \gamma \,.\, q' = \delta(q, \gamma)\,\}$ and, with $L \subseteq \Sigma^*$, $\delta(q, L) = \{\, q' \in Q \mid \exists\, s \in L \,.\, q' = \delta(q, s)\,\}$. A state $q \in Q$ is *reachable* if $q \in \delta(q_{\mathrm{o}}, \Sigma^*)$. A state $q \in Q$ is *co-reachable* if $\delta(q, \Sigma^*) \cap Q_{\mathrm{m}} \neq \emptyset$. If all reachable states in G are co-reachable, then G is *non-blocking*. Moreover, G is called *reachable* (*co-reachable*) if all states are reachable (co-reachable), and G is called *trim* if it is reachable and co-reachable.

With the automaton $G = (Q, \Sigma, \delta, q_{\mathrm{o}}, Q_{\mathrm{m}})$, we denote the *generated language* $\mathrm{L}(G) := \{\, s \in \Sigma^* \mid \delta(q_{\mathrm{o}}, s)!\,\}$ and the *marked language* $\mathrm{L}_{\mathrm{m}}(G) := \{\, s \in \Sigma^* \mid \delta(q_{\mathrm{o}}, s) \in Q_{\mathrm{m}}\,\}$. Moreover, we associate with G the equivalence relation $\equiv_G$ on Σ^*, defined for $s', s'' \in \Sigma^*$ by $s' \equiv_G s''$ if and only if $\delta(q_{\mathrm{o}}, s') = \delta(q_{\mathrm{o}}, s'')$.

Let Q denote a finite set and consider a *monotone operator* f, i.e., $f(P') \subseteq f(P'') \subseteq Q$ for all $P' \subseteq P'' \subseteq Q$. Then the least fixpoint of f is given by $\cup\{\, f^i(\emptyset) \mid i \in \mathbb{N}_0\,\}$ and can be obtained by the iteration $P_0 := \emptyset$, $P_{i+1} := P_i \cup f(P_i)$. In particular, the fixpoint is attained for some finite $i \in \mathbb{N}_0$. Likewise, the iteration $P_0 := Q$, $P_{i+1} := P_i \cap f(P_i)$ can be used to obtain the greatest fixpoint. As a μ-calculus formula, the least fixpoint of f is denoted $\mu P \,.\, f(P)$, whereas the greatest fixpoint is denoted $\nu P \,.\, f(P)$. Now consider an operator g that depends on multiple set-valued parameters, e.g., $g(P', P'') \subseteq Q$ for $P', P'' \subseteq Q$. Assuming that g is monotone in both arguments, the fixpoints $\mu P' \,.\, g(P', P'')$ and $\nu P'.g(P', P'')$ are monotone in P''. In this case, nested μ-calculus formulae are well defined, e.g. $\nu P'' \,.\mu P' \,.\, g(P', P'')$ evaluates to the greatest fixpoint of $\mu P' \,.\, g(P', P'')$, interpreted as an expression in terms of P''.

2. THE CONTROLLABILITY PREFIX

Taking the perspective proposed by Thistle and Wonham (1994b,a), we adapt the notion of the *controllability prefix* to the common setting of supervisory control of $*$-languages; see also (Moor and Schmidt, 2015, 2017). More specifically, we consider the following closed-loop configuration of a plant, represented as a $*$-language, and a supervisor.

Definition 1. Given an alphabet Σ partitioned in controllable and uncontrollable events, $\Sigma = \Sigma_{\mathrm{c}} \dot\cup \Sigma_{\mathrm{uc}}$, the set of *control patterns* is denoted $\Gamma := \{\, \gamma \subseteq \Sigma \mid \Sigma_{\mathrm{uc}} \subseteq \gamma\,\}$. For a plant $L \subseteq \Sigma^*$, a *supervisor* is a map $f : \mathrm{pfx}\, L \to \Gamma$. The *local closed-loop behaviour* and the *accepted closed-loop behaviour* are then defined by

$$L_f := \{\, s \in \mathrm{pfx}\, L \mid \forall\ t \in \Sigma^*,\ \sigma \in \Sigma \,.\, t\sigma \in \mathrm{pfx}\, s \to \sigma \in f(t)\,\}, \quad (1)$$

and

$$K_f := L_f \cap L, \quad (2)$$

respectively. The supervisor is *non-blocking* if L_f and L are non-conflicting; i.e., if for all $s \in L_f$ there exists $t \in \Sigma^*$ such that $st \in K_f$. □

Although we avoid an explicit reference to an automata representation and thereby deviate from the original literature (Ramadge and Wonham, 1987, 1989), the well known characterisation of achievable closed-loop behaviours immediately carries over to our language based setting.

Lemma 2. Consider a plant $L \subseteq \Sigma^*$ and a non-empty closed-loop candidate $K \subseteq \Sigma^*$, $\emptyset \neq K \subseteq L$. Then there exists a non-blocking supervisor $f : \mathrm{pfx}\, L \to \Gamma$ with accepted closed-loop behaviour K_f such that $K_f = K$ if and only if (a) K is relatively closed w.r.t. L and (b) K is controllable w.r.t. L.

Proof. See Theorem 6.1 in (Ramadge and Wonham, 1987) □

Given a plant $L \subseteq \Sigma^*$ and an upper-bound specification E, $\emptyset \neq E \subseteq L$, the most basic problem studied in supervisory control is the synthesis of a non-blocking supervisor $f : \mathrm{pfx}\, L \to \Gamma$ such that the accepted closed-loop behaviour K_f satisfies the upper bound E; i.e., we require that $K_f \subseteq E$. Referring to Lemma 2, we recall that the two properties (a) relative closedness and (b) controllability are retained under arbitrary union; see Proposition 7.1 in (Ramadge and Wonham, 1987). Hence, the supremum

$$K^{\uparrow} := \sup \mathcal{CF}(L,\, E) := \cup\{\, K \subseteq E \mid \\ K \text{ is relatively closed and controllable w.r.t. } L \,\} \quad (3)$$

itself is relatively closed and controllable. Consequently, the control problem has a solution if and only if $K^{\uparrow}$ is non-empty. In this case, $K^{\uparrow}$ is referred to as the *closed-loop behaviour under maximally permissive supervisory control.* As a technical comment, note that $\sup \mathcal{CF}(L,\, E)$ is not affected if we substitute E by its supremal relatively closed sublanguage $E^{\uparrow} \subseteq E$; i.e., we have $K^{\uparrow} = \sup \mathcal{CF}(L,\, E^{\uparrow})$. An explicit formula to obtain $E^{\uparrow}$ is given by Ziller and Cury (1994); see also (Yari and Hashtrudi-Zad, 2016) for an automata-based algorithm. We may henceforth assume E to be relatively closed whenever convenient.

For practical solutions to the supervisory control problem, L and E are considered regular and, for this case, the literature provides algorithms for the computation of a finite automaton realisation of $K^{\uparrow}$; see (Wonham and Ramadge, 1987). Provided that $K^{\uparrow}$ turns out non-empty, a Moore automaton realisation of a respective supervisor f can be derived from the realisation of $K^{\uparrow}$. However, if $K^{\uparrow}$ turns out empty, nothing can be concluded except that the synthesis problem at hand exhibits no solution. In the following, we address this situation by investigating relaxed variants of the original synthesis problem. More specifically, we impose the hypothesis, that the plant — for whatever reasons — starts its operation by generating a certain event sequence $s \in \operatorname{pfx} E$, and we then ask whether the problem can be solved under this additional hypothesis. The set of all strings $s \in \operatorname{pfx} E$ that allow for an affirmative answer is referred to as the *controllability prefix of E w.r.t. L.* The controllability prefix was originally introduced by Thistle and Wonham (1994b) in the context of supervisory control for ω-languages. For the situation of $*$-languages, we recall the formal definition from Moor and Schmidt (2015, 2017).

Definition 3. Given a plant $L \subseteq \Sigma^*$, a specification E, $\emptyset \neq E \subseteq L$, and a string $s \in \Sigma^*$, we use the convenient notation

$$L_s := L \cap (s\Sigma^*)\,, \quad E_s := E \cap (s\Sigma^*)\,. \quad (4)$$

The *controllability prefix* $\operatorname{cfx} E \subseteq \Sigma^*$ of E is then defined as the set of all strings $s \in \Sigma^*$ such that $\sup \mathcal{CF}(L_s,\, E_s) \neq \emptyset$. □

Note that $\operatorname{cfx} E \subseteq \operatorname{pfx} E$ and that, in general, $\operatorname{cfx} E$ is not closed. It is immediate from the above definition that $\epsilon \in \operatorname{cfx} E$ if and only if $\sup \mathcal{CF}(L,\, E) \neq \emptyset$; i.e., the control problem under consideration exhibits a solution if and only if the empty string is within the controllability prefix. The following alternative characterisation of the controllability prefix turns our useful for our subsequent discussion.

Proposition 4. Consider a plant $L \subseteq \Sigma^*$ and a relatively closed specification E, $\emptyset \neq E = (\operatorname{pfx} E) \cap L$. Then a string $s \in \Sigma^*$ is in $\operatorname{cfx} E$ if and only if there exists a supervisor $f : \operatorname{pfx} L \to \Gamma$ for the plant L such that the associated local closed-loop behaviour L_f exhibits the following three properties:

(a) $s \in L_f$,

(b) $\forall\, t \in \Sigma^*\, .\, st \in L_f \;\to\; st \in \operatorname{pfx} E$, and

(c) $\forall\, t \in \Sigma^*\; \exists\, r \in \Sigma^*\, .\, st \in L_f \;\to\; str \in L_f \cap L$.

Note that we do require the supervisor f to be non-blocking.

Proof. Given $s \in \Sigma^*$, we prove both implications separately.

Assume that there exists a supervisor $f : \operatorname{pfx} L \to \Gamma$ with the above closed-loop properties (a) – (c). We then formally define a supervisor h for the plant $L_s := L \cap (s\Sigma^*)$ by restricting the domain of f accordingly; i.e., $g : \operatorname{pfx} L_s \to \Gamma$ and $g(s) = f(s)$ for all $s \in \operatorname{pfx} L_s$. Denote $L_{s,g}$ and $K_{s,g} := L_{s,g} \cap L_s$ the corresponding local and accepted closed-loop behaviours, respectively. Note that $L_{s,g} = L_f \cap \operatorname{pfx}(s\Sigma^*)$, $K_{s,g} = L_{s,g} \cap L_s = L_f \cap L \cap (s\Sigma^*)$ and, by (a), $s \in L_{s,g}$. To observe that g is a non-blocking supervisor for L_s, pick an arbitrary $u \in L_{s,g}$ and distinguish two cases. If $u \in \operatorname{pfx} s$, we pick $v \in \Sigma^*$ such that $uv = s \in L_{s,g} \subseteq L_f$. For this case, we refer to (c) with $t = \epsilon$ to conclude the existence of $r \in \Sigma^*$ such that $uvr = str \in L_f \cap L \cap (s\Sigma^*) = K_{s,g}$. For our second case, $u \notin \operatorname{pfx} s$, we infer that $s < u$ and, hence, can pick $t \in \Sigma^*$ such that $st = u \in L_{s,g} \subseteq L_f$. We again refer to (c) and conclude the existence of $r \in \Sigma^*$ such that $ur = str \in L_f \cap L \cap (s\Sigma^*) = K_{s,g}$. In both cases, we have extended $u \in L_{s,g}$ to a string in $K_{s,g}$. Hence g is a non-blocking supervisor for L_s, and, in particular, $s \in L_{s,g} \neq \emptyset$ implies $K_{s,g} \neq \emptyset$. To see that g enforces the specification $E_s := E \cap (s\Sigma^*)$, pick any $u \in L_{s,g} \cap L_s$. In particular, we have $s \leq u$ and, hence, can pick $t \in \Sigma^*$ such that $st = u \in L_{s,g} \subseteq L_f$. We refer to (b) to conclude $u = st \in \operatorname{pfx} E$, and obtain $u \in E$ by relative closedness of E w.r.t. L. In summary, g is a non-blocking supervisor for L_s which enforces the specification E_s. By Lemma 2 we then obtain $\emptyset \neq K_{s,g} \subseteq \sup \mathcal{CF}(L_s,\, E_s)$ and, by Definition 3, $s \in \operatorname{cfx} E$.

For the converse implication, assume that $s \in \operatorname{cfx} E$. By Definition 3 and Lemma 2 there exists a non-blocking supervisor g for the plant $L_s := L \cap (s\Sigma^*)$ that enforces the specification $E_s := E \cap (s\Sigma^*)$. Note that non-blockingness of g implies $s \in L_{s,g}$, where $L_{s,g}$ again denotes the associated local closed-loop behaviour. We extend the domain of h to obtain a supervisor $f : \operatorname{pfx} L \to \Gamma$ by $f(st) := g(st)$ for all $t \in \Sigma^*$ with $st \in \operatorname{pfx} L$ and $f(u) := \Sigma \in \Gamma$ for all $u \in \operatorname{pfx} L$ with $s \notin \operatorname{pfx} u$; i.e., f imposes the dummy control pattern Σ until the event sequence s has been generated and then switches to mimic g. Denote L_f the local closed-loop behaviour of L under supervision f and observe that $L_{s,g} = L_f \cap (s\Sigma^*) \subseteq L_f$. Then (a) $s \in L_f$ is immediate by $s \in L_{s,g}$. Regarding (c), pick any $t \in \Sigma^*$ such that $st \in L_f$. This implies $st \in L_{s,g}$ and, by non-blockingness of g, the existence of $r \in \Sigma^*$ such that $str \in L_{s,g} \cap L_s \subseteq L_f \cap L$. Regarding (b), pick again $t \in \Sigma^*$ such that $st \in L_f$. Referring to (c), we extend by $r \in \Sigma^*$ such that $str \in L_f \cap L \cap (s\Sigma^*) = L_{s,g} \cap L_s \subseteq E_s \subseteq E$. In particular, we have that $s \in \operatorname{pfx} E$. In summary, we have constructed a supervisor f for L with properties (a) – (c). □

As an immediate consequence of the above proposition, for any $s \in \operatorname{cfx} E$ and any supervisor $f : \operatorname{pfx} L \to \Gamma$ with the above properties (a) – (c), the local closed loop L_f satisfies

$$\forall\, t \in \Sigma^*\, .\, st \in L_f \;\to\; st \in \operatorname{cfx} E\,; \quad (5)$$

i.e., f constraints the future of s to remain within $\operatorname{cfx} E$ indefinitely; see also Proposition 3 in (Moor and Schmidt, 2015).

The closed-loop behaviour under maximally permissive supervision can be stated in terms of the controllability prefix.

Lemma 5. Consider a plant $L \subseteq \Sigma^*$, and a relatively closed specification E, $\emptyset \neq E = (\operatorname{pfx} E) \cap L$. Then

$$\sup \mathcal{CF}(L,\, E) = \sup \mathcal{P}(\operatorname{cfx} E) \cap L\,, \quad (6)$$

where $\sup \mathcal{P}(M) := \cup\{\, K \subseteq M \mid K = \operatorname{pfx} K \,\}$ denotes the supremal closed sublanguage of a language $M \subseteq \Sigma^*$.

Proof. Denote $K^{\uparrow} := \sup \mathcal{CF}(L,\, E)$ and $K := \sup \mathcal{P}(\operatorname{cfx} E) \cap L$.

For $K \subseteq K^{\uparrow}$, we first establish that K is controllable w.r.t. L. Pick any $s \in \operatorname{pfx} K$ and $\sigma \in \Sigma_{\mathrm{uc}}$ such that $s\sigma \in \operatorname{pfx} L$. Observe that

$$s \in \mathrm{pfx}\, s \subseteq \mathrm{pfx}\, K \subseteq \mathrm{pfx}\, \sup \mathcal{P}(\mathrm{cfx}\, E) = \sup \mathcal{P}(\mathrm{cfx}\, E) \subseteq \mathrm{cfx}\, E \subseteq \mathrm{pfx}\, E\,. \qquad (7)$$

In particular, $s \in \mathrm{cfx}\, E$. Denote $f : \mathrm{pfx}\, L \to \Gamma$ a supervisor such that the local closed loop L_f exhibits properties (a) – (c) as granted by Proposition 4. Then $\sigma \in f(s)$ and, hence $s\sigma \in L_f$. Referring to property (c), we obtain $s\sigma t \in L_f \cap L$ for a suitable choice of $t \in \Sigma^*$, and, by Eq. (5), $s\,\mathrm{pfx}\,(\sigma t) \subseteq \mathrm{cfx}\, E$. Together with $\mathrm{pfx}\, s \subseteq \mathrm{cfx}\, E$ we infer that $\mathrm{pfx}\,(s\sigma t) \subseteq \mathrm{cfx}\, E$. Hence, $s\sigma t \in \mathrm{pfx}\, s\sigma t \subseteq \sup \mathcal{P}(\mathrm{cfx}\, E)$. Recalling that $s\sigma t \in L$ we obtain $s\sigma t \in K$ and, thus, $s\sigma \in \mathrm{pfx}\, K$. This establishes controllability of K. Since K is the intersection of a closed language with L, we also have that K is relatively closed w.r.t. L. Finally, we have that $K \subseteq (\mathrm{pfx}\, E) \cap L = E$ and, hence, K is a controllable and relatively closed subset of E. This implies $K \subseteq K^{\uparrow}$.

For the converse inclusion, $K^{\uparrow} \subseteq K$, we may assume that $K^{\uparrow} \neq \emptyset$ and denote f a non-blocking supervisor with accepted closed-loop behaviour $K_f = K^{\uparrow}$; see Lemma 2. This supervisor exhibits properties (a) – (c), as required by Proposition 4, for any $s \in \mathrm{pfx}\, K^{\uparrow} = L_f$ and, hence, $\mathrm{pfx}\, K^{\uparrow} \subseteq \mathrm{cfx}\, E$. Obviously, $\mathrm{pfx}\, K^{\uparrow}$ is closed and we obtain $K^{\uparrow} \subseteq \mathrm{pfx}\, K^{\uparrow} \subseteq \sup \mathcal{P}(\mathrm{cfx}\, E)$. By $K^{\uparrow} \subseteq E \subseteq L$ this implies $K^{\uparrow} \subseteq K$. □

The controllability prefix $\mathrm{cfx}\, E \subseteq \mathrm{pfx}\, E$, as a formal language, can be realised by an automaton G with suitable marking, provided that $\equiv_G$ is at least as fine as $\equiv_L$ and $\equiv_E$.

Proposition 6. Consider a plant $L \subseteq \Sigma^*$, a relatively closed specification E, $\emptyset \neq E = (\mathrm{pfx}\, E) \cap L$, and two strings $s,\ s' \in \mathrm{pfx}\, E$ such that $s \equiv_L s'$ and $s \equiv_E s'$. Then $s \in \mathrm{cfx}\, E$ if and only if $s' \in \mathrm{cfx}\, E$.

Proof. Assume that $s \in \mathrm{cfx}\, E$ and refer to Proposition 4 for a supervisor $f : \mathrm{pfx}\, L \to \Gamma$ with properties (a) – (c). We then define an alternative supervisor $f' : \mathrm{pfx}\, L \to \Gamma$ by $f'(s't) := f(st)$ for all $t \in \Sigma^*$ with $st \in \mathrm{pfx}\, L$ and $f'(u) = \Sigma \in \Gamma$ for $u \in \mathrm{pfx}\, L$ with $s' \notin \mathrm{pfx}\, u$. Denote $L_{f'}$ the generated closed-loop behaviour with the plant L. By construction, we have that $s' \in L_{f'}$; i.e., the candidate f' qualifies for property (a) of Proposition 4 for the candidate string s'. To discuss properties (b) and (c) consider any $t \in \Sigma^*$ such that $s't \in L_{f'}$. By $s \equiv_L s'$, we have that $st \in \mathrm{pfx}\, L$ and, by the construction of f', we obtain $st \in L_f$. Referring to property (b) for f and $s \in \mathrm{cfx}\, E$, this implies $st \in \mathrm{pfx}\, E$ and, by $s \equiv_E s'$, in turn $s't \in \mathrm{pfx}\, E$. This constitutes property (b) for the string s' and the supervisor f'. Likewise we refer to property (c) for f and $s \in \mathrm{cfx}\, E$, to obtain the existence of $r \in \Sigma^*$ such that $str \in L_f \cap L$. By the construction of f', this implies $s'tr \in L_{f'}$ and, by $s \equiv_L s'$, $s'rt \in L$. In summary, we have constructed a supervisor f' that exhibits properties (a) – (c) for the candidate sequence s'. We then conclude with Proposition 4 that $s' \in \mathrm{cfx}\, E$. Note that is sufficient to establish the "if part" since the "only-if part" then follows by uniform substitution. □

Given regular parameters $L \subseteq \Sigma^*$ and $E \subseteq \Sigma^*$, $\emptyset \neq E = (\mathrm{pfx}\, E) \cap L$, with finite automata realisations, a realisation of the controllability prefix can be computed as follows:

(A1) obtain a trim realisation for L and a full realisation of E; the latter is constructed by introducing an unmarked *dump state* as a dummy successor for states and events with otherwise undefined transition function;

(A2) use the product composition to obtain an automaton $G = (Q, \Sigma, \delta, q_o, Q_m)$ with $\mathrm{L}(G) = \mathrm{pfx}\, L$, $\mathrm{L_m}(G) = E$, and such that $\equiv_G$ is at least as fine as $\equiv_L$ and $\equiv_E$;

(A3) iterate over all states $q \in Q$ of G and derive realisations of L_s and E_s in order to compute $\sup \mathcal{CF}(L_s, E_s)$ for some minimum length $s \in \Sigma^*$ with $\delta(q_o, s) = q$;

(A4) substitute the marking of G by the set Q_{cfx} of all states $q \in Q$ for which $\sup \mathcal{CF}(L_s, E_s) \neq \emptyset$ as identified at step (A3) and report $G_{\mathrm{cfx}} := (Q, \Sigma, \delta, q_o, Q_{\mathrm{cfx}})$ as the result.

By Proposition 6, the choice of $s \in \Sigma^*$, $\delta(q_o, s) = q$, in step (A3) does not effect the result in step (A4) and we indeed have $\mathrm{cfx}\, E = \{\, s \in \Sigma^* \,|\, \delta(q_o, s) \in Q_{\mathrm{cfx}} \,\} = \mathrm{L_m}(G_{\mathrm{cfx}})$. Since the complexity of computing $\sup \mathcal{CF}(L_s, E_s)$ is $O(n^2)$ with $n \in \mathbb{N}$ the size of Q, the proposed procedure has an overall complexity of $O(n^3)$. In the subsequent discussion, we will derive a more elegant procedure to realise the controllability prefix with quadratic complexity.

3. NON-BLOCKING STATE FEEDBACK

Complementing the language based discussion in the previous section, we now focus attention on automata realisations. A generic synthesis problem here is the construction of a state feedback for a possibly blocking automaton such that the closed-loop is non-blocking. In this section, we briefly outline a well known algorithmic solution and derive an alternative which relates to the controllability prefix. As we shall see in the subsequent section, this algorithm can be used to solve the common synthesis problem as discussed in Section 2.

Definition 7. Given an alphabet Σ with the common partition $\Sigma := \Sigma_c \dot{\cup} \Sigma_{uc}$, denote Γ the set of control patterns as in Definition 1, and consider a plant realised by a not necessarily non-blocking automaton $G = (Q, \Sigma, \delta, q_o, Q_m)$. Then a *state feedback* for G is a map $h : Q \to \Gamma$ and the associated *closed-loop system* is the automaton $G_h := (Q, \Sigma, \delta_h, q_o, Q_m)$, with $\delta_h(q, \sigma) := \delta(q, \sigma)$ for $q \in Q$, $\sigma \in h(q)$ with $\delta(q, \sigma)!$, and otherwise undefined. If G_h is non-blocking, we refer to h as a *non-blocking state feedback* for G. □

A common approach to synthesise a non-blocking state feedback is to iteratively remove critical states and associated transitions from the automaton G, and to thereby construct a sequence of trim automata G_i, $i \in \mathbb{N}_0$. Here, a state and all associated transitions are critical (a) if it is not reachable or not co-reachable or (b) if it misses uncontrollable transitions when compared with the original automaton G. Since all relevant sets are finite, the procedure attains a fixpoint G_∞ after finitely many iterations. If the initial state turns out non-critical, a non-blocking state feedback h for G can be extracted from G_∞.

In our development of an alternative procedure we take a game-theoretic perspective common in the field of reactive synthesis; see (Finkbeiner, 2016) for an overview. More specifically, we consider the closed-loop as a two-player game in which a move of the supervisor (player 0) amounts to the choice of a control pattern and in which the subsequent move of the plant (player 1) amounts to the choice of an enabled event and the execution of the respective transition. The supervisor wins the game if all visited states are co-reachable. A state from which on the supervisor can strategically organise its future moves such that regardless the moves of the plant all visited states will be co-reachable is hence referred to as a *player-0-winning state*, or concisely as a *winning state*. Note that in order to actually attain a marked state, the plant needs to cooperate with the supervisor by taking appropriate moves. Technically, the set of winning states $W \subseteq Q$ is defined as

$$W := \{\, q \in Q \,|\, \exists\, h : Q \to \Gamma \\ \forall\, q' \in \delta_h(q, \Sigma^*) \,.\, \delta_h(q', \Sigma^*) \cap Q_m \neq \emptyset \,\}\,. \quad (8)$$

In analogy to the situation with the controllability prefix, it is an immediate consequence that a non-blocking state feedback exists if and only if the initial state is a winning state. As it turns out, the set of all winning states can be conveniently characterised by a two-nested fixpoint formula in terms of two strategically defined pre-image operators.

Given a target set $X \subseteq Q$, the *existential pre-image* $\mathrm{pre}_\exists(X)$ is defined by

$$\mathrm{pre}_\exists(X) := \{\, q \in Q \,|\, \delta(q, \Sigma) \cap X \neq \emptyset \,\}\,, \quad (9)$$

i.e., $\mathrm{pre}_\exists(X)$ consists of states with some one-step successor in X. Clearly, $\mathrm{pre}_\exists(X)$ is monotone in X, i.e., given $X' \subseteq X'' \subseteq Q$ we have $\mathrm{pre}_\exists(X') \subseteq \mathrm{pre}_\exists(X'')$. Moreover, we have that $\mathrm{pre}_\exists(\emptyset) = \emptyset$. Repeatedly applying this operator to a given target set, one obtains the set of states from which the target can be reached via some finite event sequence. Technically, we consider the monotone sequence $(X_i)_{i\in\mathbb{N}_0}$, $X_i \subseteq X_{i+1} \subseteq Q$ for all $i \in \mathbb{N}_0$, generated by the iteration

$$X_0 := \emptyset\,, \quad X_{i+1} := X_i \cup \mathrm{pre}_\exists(X_i) \cup T\,, \quad (10)$$

with the target $T \subseteq Q$ as a parameter. We then let $X_\infty := \cup\{\, X_i \,|\, i \in \mathbb{N}_0 \,\}$ and observe that $X_\infty = \{q \in Q \,|\, \delta(q, \Sigma^*) \cap T \neq \emptyset\}$. Note that, for the special case of $T = Q_m$ the limit X_∞ amounts to the set of all co-reachable states. Since Q is a finite set, X_∞ is attained after some finite number of iterations; i.e., there exists $k \in \mathbb{N}_0$ such that $X_i = X_\infty$ for all $i \geq k$. It is well known that X_∞ is the smallest fixpoint of the monotone expression $\mathrm{pre}_\exists(\,\cdot\,) \cup T$. Thus, the μ-calculus formula

$$\mu X \,.\, \mathrm{pre}_\exists(X) \cup T \quad (11)$$

evaluates to the limit X_∞ and therefore characterises the set of states from which T is reachable.

The existential pre-image is complemented by the *controlled universal pre-image*, defined by

$$\mathrm{pre}_\forall(Y) := \{\, q \in Q \,|\, \delta(q, \Sigma_{uc}) \subseteq Y \,\}\,, \quad (12)$$

for the domain $Y \subseteq Q$; i.e., $\mathrm{pre}_\forall(Y)$ consists of those states that can be controlled such that all one-step successor states are within Y. Clearly, $\mathrm{pre}_\forall(Y)$ is monotone in Y, i.e., given $Y' \subseteq Y'' \subseteq Q$ we have $\mathrm{pre}_\forall(Y') \subseteq \mathrm{pre}_\forall(Y'')$. Moreover, we have that $\mathrm{pre}_\forall(Q) = Q$. If a domain $Y \subseteq Q$ turns out a fixpoint of the controlled universal pre-image, $\mathrm{pre}_\forall(Y) = Y$, states $q \in Y$ can be controlled by suitable state-feedback to remain within Y indefinitely. Given a domain $D \subseteq Q$, the μ-calculus formula

$$\nu Y \,.\, \mathrm{pre}_\forall(Y) \cap D \quad (13)$$

evaluates to the greatest fixpoint Y_∞ of the monotone expression $\mathrm{pre}_\forall(\,\cdot\,) \cap D$, and, hence, characterises the largest controlled invariant subset of D. Since Q is finite, Y_∞ can be computed by

$$Y_0 := Q\,, \quad Y_{i+1} := Y_i \cap \mathrm{pre}_\forall(Y_i) \cap D\,, \quad (14)$$

with $Y_\infty := \cap\{\, Y_i \,|\, i \in \mathbb{N}_0 \,\}$. In particular, there exists $k \in \mathbb{N}_0$ such that $Y_\infty = Y_i$ for all $i \geq k$.

Our conjecture here is that the strategic combination

$$Z := \nu Y \,.\, \mu X \,.\, (\mathrm{pre}_\exists(X) \cup Q_m) \cap \mathrm{pre}_\forall(Y) \quad (15)$$

of both fixpoints (11) and (13) evaluates to the set of winning states; i.e., $Z = W$. Note that the expression $(\mathrm{pre}_\exists(\,\cdot\,) \cup Q_m) \cap \mathrm{pre}_\forall(\,\cdot\,)$ is monotone in both arguments, and, hence, the two-nested fixpoint (15) is well defined. In support of the subsequent analysis, we refer to one entire run of the inner μ-iteration at a stage where the outer ν-iteration has attained its fixpoint, i.e., we consider the monotone sequence $(X_i)_{i\in\mathbb{N}_0}$ generated by

$$X_0 := \emptyset\,, \quad X_{i+1} := X_i \cup (\,(\mathrm{pre}_\exists(X_i) \cup Q_m) \cap \mathrm{pre}_\forall(Z)\,)\,. \quad (16)$$

In particular, we have that $Z = \cup\{\, X_i \,|\, i \in \mathbb{N}_0 \,\}$, and we can introduce the following ranking of states $q \in Z$:

$$\mathrm{rank}(q) := i \in \mathbb{N}_0 \quad \text{s.t.} \quad q \in X_i,\ q \notin X_{i-1}\,. \quad (17)$$

Note that $\mathrm{rank}(q) \geq 1$ for all $q \in Z$ since $X_0 = \emptyset$.

Example. We illustrate the fixpoint (15) by the automaton G with events $\Sigma = \{\alpha, \beta\}$, $\Sigma_c = \{\alpha\}$, $\Sigma_{uc} = \{\beta\}$, and with states and transitions as shown in Figure 1. There, the marked states are represented by full nodes, and transitions via the controllable event α are shown with a middle tick. Beginning with $Y_0 := Q$, we obtain after one inner μ-iteration in X the set of all co-reachable states which is assigned to Y_1. The state at the top right, however, can not be controlled such that its successors reside in Y_1. Hence, after the second inner iteration in X we obtain Y_2 as indicated. In the subsequent iterations, two more states are removed from the iterate Y and we finally obtain the fixpoint $Z = Y_4$. Considering the inner μ-iteration at this stage, Eq. (16), we obtain the ranking as indicated. A state feedback that renders Z a co-reachable invariant is derived by disabling the highlighted (dashed) transitions. However, since $q_o \notin Z$ and modulo our conjecture $Z = W$, we do not expect a non-blocking state feedback for G to exists. □

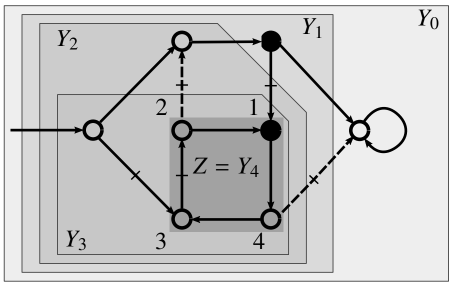

Fig. 1. Automaton G with iterates Y_0 to Y_4 and fixpoint Z

Example. For a slightly more involved example, consider the automaton G given by Figure 2. The states $Y_{mns} := Q \setminus Z$ are removed from the Y-iterate in the outer ν-iteration one by one from the right to the left. In particular, we have $q_o \in Z$ and a non-blocking state feedback for G is obtained by disabling the highlighted (dashed) transitions. □

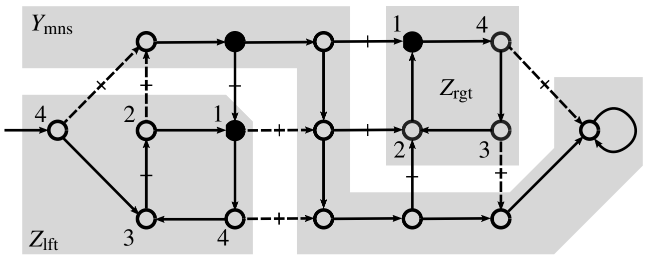

Fig. 2. Automaton G with fixpoint $Z = Z_{lft} \cup Z_{rgt}$

For a formal proof of our conjecture, $Z = W$, we consider the following candidate state feedback $h : Q \to \Gamma$,

$$h(q) := \begin{cases} \Sigma_{uc} \cup \{\sigma \in \Sigma \,|\, \delta(q, \sigma) \in Z\} & \text{if } q \in Z\,, \quad (18a) \\ \Sigma & \text{if } q \notin Z\,, \quad (18b) \end{cases}$$

for $q \in Q$. With this candidate, we now establish that once a state in the fixpoint Z is attained, state feedback can be applied such that all successor states are co-reachable.

Lemma 8. Given an automaton $G = (Q, \Sigma, \delta, q_o, Q_m)$, apply the state feedback $h : Q \to \Gamma$ derived by Eqs. (18a/b) from the fixpoint Z specified in Eq. (15). Denote the closed-loop system $G_h = (Q, \Sigma, \delta_h, q_o, Q_m)$. Then

(a) Z is an invariant set of states in G_h, i.e., $\delta_h(Z, \Sigma^*) \subseteq Z$;

(b) all states in Z are co-reachable in G_h.

In particular, we have that $Z \subseteq W$ with W the winning states defined by Eq. (8).

Proof. We again refer to $Z = \cup\{ X_i \,|\, i \in \mathbb{N}_0 \}$ for the monotone sequence $(X_i)_{i\in\mathbb{N}_0}$ generated by iteration (16).

Ad (a). Note that $X_i \subseteq \mathrm{pre}_\forall(Z)$ for all $i \in \mathbb{N}_0$ as an inductive consequence of Eq. (16). Hence, $Z = X_\infty := \cup\{ X_i \,|\, i \in \mathbb{N}_0 \} \subseteq \mathrm{pre}_\forall(Z)$. Now pick an arbitrary $q \in Z$. By the definition of $\mathrm{pre}_\forall(\cdot)$, Eq. (12), we have that $\delta(q, \Sigma_{\mathrm{uc}}) \subseteq Z$. With $\sigma \in h(q)$ and referring to Eq. (18a), this implies $\delta(q, \sigma) \in Z$ whenever $\delta(q, \sigma)!$. Hence, $\delta_h(q, \Sigma) = \delta(q, h(q)) \subseteq Z$. By induction over the length of sequences $s \in \Sigma^*$ we obtain $\delta_h(q, \Sigma^*) \subseteq Z$.

Ad (b). Pick an arbitrary $q \in Z$. For the special case that $\mathrm{rank}(q) = 1$, we refer to $q \in X_1 = Q_m \cap \mathrm{pre}_\forall(Z) \subseteq Q_m$ to observe that q is co-reachable. For the general case $i := \mathrm{rank}(q) \geq 1$, we provide an inductive argument. For the base case $i = 1$ we have already established co-reachability. Now assume that for some $i \in \mathbb{N}$ all states with rank i are co-reachable in G_h and consider a state q with $\mathrm{rank}(q) = i + 1 > 1$. Observe by monotonicity of $(X_i)_{i\in\mathbb{N}_0}$ that $q \in X_{i+1}$, $q \notin X_i$, implies $q \notin X_1 = Q_m \cap \mathrm{pre}_\forall(Z)$, and, hence, $q \in \mathrm{pre}_\exists(X_i) \cap \mathrm{pre}_\forall(Z)$. By the definition of $\mathrm{pre}_\exists(X_i)$, Eq. (9), this implies the existence of $\sigma \in \Sigma$ such that $q' := \delta(q, \sigma) \in X_i \subseteq Z$. In particular, we have that $\mathrm{rank}(q') \leq i$ and, by Eq. (18a), $\sigma \in h(q)$; i.e., $q' = \delta_h(q, \sigma)$. By the induction hypothesis, q' is co-reachable in G_h and we can pick $s \in \Sigma^*$ such that $\delta_h(q', s) \in Q_m$. Co-reachability of q is then verified by $\delta_h(q, \sigma s) = \delta_h(\delta_h(q, \sigma), s) = \delta_h(q', s) \in Q_m$. This concludes the inductive argument and we have established that all $q \in Z$ are co-reachable in G_h.

Ad $Z \subseteq W$. Pick $q \in Z$ and an arbitrary $q' \in \delta_h(q, \Sigma^*)$. We refer to (a) to obtain $q' \in Z$ and subsequently to (b) to obtain $\delta_h(q', \Sigma^*) \cap Q_m \neq \emptyset$. Thus, $q \in W$ by definition; see Eq. (8) □

The following lemma addresses the converse implication and establishes that all winning states are within the fixpoint Z.

Lemma 9. Given an automaton $G = (Q, \Sigma, \delta, q_o, Q_m)$, consider a state $q \in Q$. Assume that there exists a state feedback $h : Q \to \Gamma$ such that in the closed-loop system $G_h = (Q, \Sigma, \delta_h, q_o, Q_m)$ all states $V := \delta_h(q, \Sigma^*)$ that are reachable states from q are co-reachable. Then $V \subseteq Z$ with the fixpoint Z from Eq. (15). In particular, $W \subseteq Z$ for the winning states W defined by Eq. (8).

Proof. Note that $\delta(p, \Sigma_{\mathrm{uc}}) \subseteq \delta_h(p, \Sigma^*) \subseteq V$ for all $p \in V$. In particular, we have that $V \subseteq \mathrm{pre}_\forall(V)$ as a preliminary observation. The remainder of the proof is organised in three separate parts.

First, we refer to the sequence $(X_i)_{i\in\mathbb{N}_0}$ generated by the inner μ-iteration with parameter $Y = V$, i.e.,

$$X_0 := \emptyset\,,\;\; X_{i+1} := X_i \cup (\,\mathrm{pre}_\exists(X_i) \cup Q_m\,) \cap \mathrm{pre}_\forall(V)\,)\,, \qquad (19)$$

and we will establish $V \subseteq X_\infty := \cup\{ X_i \,|\, i \in \mathbb{N}_0 \}$. Pick any $p \in V$. If $p \in Q_m$, we refer to $V \subseteq \mathrm{pre}_\forall(V)$, and hence $p \in X_1$. If $p \notin Q_m$, then there exists a sequence $s \in \Sigma^*$, $s \neq \epsilon$ such that $\delta_h(p, s) \in Q_m$ and $\delta_h(p, \mathrm{pfx}\, s) \subseteq \delta_h(p, \Sigma^*) \subseteq V \subseteq \mathrm{pre}_\forall(V)$. As in the first case, we conclude $\delta_h(p, s) \in X_1$. For an inductive argument, we decompose s by $s = r\sigma t$ with $r \in \Sigma^*$, $\sigma \in \Sigma$, and $t \in \Sigma^*$. Assume that $p'' := \delta_h(p, r\sigma) \in X_i$ for some $i \in \mathbb{N}$ and consider $p' := \delta_h(p, r)$. By $p'' = \delta_h(p, r\sigma) \in X_i$, we have $\delta(p', h(p')) \cap X_i \neq \emptyset$. By $\delta_h(p, \Sigma^*) \subseteq V$ we have that $\delta(p', \Sigma_{\mathrm{uc}}) \subseteq \delta(p', h(p')) \subseteq V$. Thus, we have established $p' \in \mathrm{pre}_\exists(X_i) \cap \mathrm{pre}_\forall(V) \subseteq X_{i+1}$. By beginning with $t = \epsilon$ and subsequently incrementing the length of t we obtain $\delta_h(p, \mathrm{pfx}\, s) \subseteq X_\infty$, and, hence $p = \delta_h(p, \epsilon) \in X_\infty$. Since $p \in V$ was chosen arbitrarily, we obtain as an intermediate result

$$V \subseteq \mu X\,.\,(\,\mathrm{pre}_\exists(X) \cup Q_m\,) \cap \mathrm{pre}_\forall(V)\,. \qquad (20)$$

For the second part of the proof we refer to the sequence $(Y_i)_{i\in\mathbb{N}_0}$ generated by the outer ν-iteration, i.e.,

$$Y_0 := Q\,,\;\; Y_{i+1} := Y_i \cap \Phi(Y_i) \qquad (21)$$

with

$$\Phi(Y) := \mu X\,.\,(\,\mathrm{pre}_\exists(X) \cup Q_m\,) \cap \mathrm{pre}_\forall(Y)\,, \qquad (22)$$

for $Y \subseteq Q$. Clearly, we have $V \subseteq Y_0$. Now assume that $V \subseteq Y_i$ for some $i \in \mathbb{N}_0$. By the monotonicity of the fixpoint $\Phi(Y)$ w.r.t. the parameter Y and the assumption of $V \subseteq Y_i$ we obtain $\Phi(V) \subseteq \Phi(Y_i)$. With Eq. (20), i.e., $V \subseteq \Phi(V)$, this implies $V \subseteq V \cap \Phi(V) \subseteq Y_i \cap \Phi(Y_i) = Y_{i+1}$. Hence, $V \subseteq Y_i$ for all $i \in \mathbb{N}_0$ and, thus, $V \subseteq \nu Y\,.\,\Phi(Y) = Z$.

Regarding $W \subseteq Z$, pick an arbitrary $q \in W$. By the definition of W there exists a state feedback $h : Q \to \Gamma$ that satisfies the conditions imposed by the first part in this Lemma. We conclude that $q \in \delta_h(q, \Sigma^*) = V \subseteq Z$ and thus $W \subseteq Z$. □

As an immediate consequence of Lemmata 8 and 9 we obtain the following theorem.

Theorem 10. Consider an automaton $G = (Q, \Sigma, \delta, q_o, Q_m)$ and a state $q \in Q$. The following two statements are equivalent:

(a) $q \in Z$ with Z the fixpoint given in Eq (15);

(b) there exists a state feedback $h : Q \to \Gamma$ with closed-loop system denoted $G_h = (Q, \Sigma, \delta, q_o, Q_m)$ such that all states $\delta_h(q, \Sigma^*)$ reachable from q are co-reachable in G_h.

In particular, the fixpoint Eq. (15) evaluates as the set of winning states specified by Eq. (8), i.e., we have $W = Z$. □

4. SUPERVISORY CONTROLLER SYNTHESIS

Intuitively, the concept of winning states from the previous section is closely related to the controllability prefix: both notions address a conditional solution to a synthesis problem where the condition is a hypothesis regarding the initial evolution of the system; i.e., either by assuming that somehow a winning state is attained or in that the system for some reason initially generates an event sequence from the controllability prefix.

Consider a plant $L \subseteq \Sigma^*$ and a relatively closed specification E, $\emptyset \neq E = \mathrm{pfx}\, E \cap L$, both regular. Recall from Proposition 6 and the procedure (A1) – (A4) given in Section 2, that we can strategically construct an automaton $G = (Q, \Sigma, \delta, q_o, Q_m)$ such that the controllability prefix can be represented as $\mathrm{cfx}\, E = \{ s \in \Sigma^* \,|\, \delta(q_o, s) \in Q_{\mathrm{cfx}} \}$ with some suitable marking $Q_{\mathrm{cfx}} \subseteq Q$. In this section, we establish that Q_{cfx} matches the set of winning states W defined by Eq. (8). Thus, the fixpoint iteration Eq. (15) can be used to effectively compute a representation of the controllability prefix. Note that, in contrast to the procedure (A1) – (A4), the iteration associated with the two-nested fixpoint is of complexity $O(n^2)$ with $n \in \mathbb{N}$ the size of Q.

We first show that if a winning state is reached by some string, then this string must be within the controllability prefix.

Lemma 11. Given a plant $L \subseteq \Sigma^*$, and a relatively closed specification E, $\emptyset \neq E = (\mathrm{pfx}\, E) \cap L$, both regular, consider a realisation $G = (Q, \Sigma, \delta, q_o, Q_m)$, with $L(G) = L$, $L_m(G) = E$. Let $W \subseteq Q$ denote the winning states defined by Eq. (8). Then

$$\forall\, s \in \Sigma^* . \delta(q_o, s) \in W \rightarrow s \in \operatorname{cfx} E . \tag{23}$$

Proof. Given $s \in \Sigma^*$ such that $q := \delta(q_o, s) \in W$, let h denote a state feedback that qualifies for the existential quantification in Eq. (8) and $G_h = (Q, \Sigma, \delta_h, q_o, Q_m)$ the corresponding closed-loop system. We then refer to Proposition 4 and consider the candidate supervisor $f : \operatorname{pfx} L \rightarrow \Gamma$ defined by $f(st) := h(\delta(q, t))$ for $t \in \Sigma^*$ with $st \in \operatorname{pfx} L$ and $f(r) := \Sigma$ for $r \in \operatorname{pfx} L$ with $s \notin \operatorname{pfx} r$. Denote L_f the closed-loop behaviour of L under supervision f. Since $\delta(q_o, s)!$ implies $s \in \operatorname{pfx} L$, the clause $f(r) = \Sigma$ for $s \notin \operatorname{pfx} r$ implies $s \in L_f$, and this amounts to property (a) in Proposition 4. We now refer to the second clause $f(st) = h(\delta(q, t))$ in the construction of f and, for all $t \in \Sigma^*$, note that $st \in L_f$ is equivalent to $\delta_h(q, t)!$. Pick any $t \in \Sigma^*$ such that $st \in L_f$ and denote $q' := \delta_h(q, t)$. Then the definition of W, Eq. (8), implies $\delta_h(q, t\Sigma^*) \cap Q_m \neq \emptyset$, and, in turn, the existence of $r \in \Sigma^*$ such that $\delta_h(q, tr) \in Q_m$. This implies $str \in L_f \cap E \subseteq L_f \cap L$, and, hence, properties (b) and (c) in Proposition 4. Thus, we have established $s \in \operatorname{cfx} E$. □

For the converse implication, we demonstrate that any state reachable via a string from the controllability prefix must be a winning state.

Lemma 12. Given a plant $L \subseteq \Sigma^*$, and a relatively closed specification E, $\emptyset \neq E = (\operatorname{pfx} E) \cap L$, both regular, consider a realisation $G = (Q, \Sigma, \delta, q_o, Q_m)$, with $L(G) = L$, $L_m(G) = E$ and induced equivalence $\equiv_G$ at least as fine as $\equiv_L$ and $\equiv_E$. Let $W \subseteq Q$ denote the winning states as defined by Eq. (8). Then

$$\forall\, s \in \Sigma^* . s \in \operatorname{cfx} E \rightarrow \delta(q_o, s) \in W . \tag{24}$$

Proof. For a preliminary result, part (A) of the proof, consider a reachable state q such that

$$\forall\, s \in \Sigma^* . \delta(q_o, s) = q \rightarrow s \in \operatorname{cfx} E . \tag{25}$$

Denote the equivalence class of $\equiv_G$ associated with q by

$$S_q := \{ s \in \Sigma^* \mid \delta(q_o, s) = q \} . \tag{26}$$

Pick any such $s \in S_q$ and denote F_s the set of all supervisors with properties (a) – (c) as specified in Proposition 4 in order to consider the control patterns

$$\gamma_s := \cup\{ f(s) \mid s \in F_s \} \quad \text{and} \quad \gamma_q := \cup\{ \gamma_s \mid s \in S_q \} . \tag{27}$$

Then, by Proposition 4 and Eq. (5), $s\sigma \in \operatorname{cfx} E$ for all $s \in S_q$ and all $\sigma \in \gamma_s$ with $\delta(q, \sigma)!$. By Proposition 6 and the prerequisite that $\equiv_G$ is at least as fine as $\equiv_L$ and $\equiv_E$, we infer that $s'\sigma \in \operatorname{cfx} E$ for all $s, s' \in S_q$ and all $\sigma \in \gamma_s$. This implies $s\sigma \in \operatorname{cfx} E$ for all $s \in S_q$ and all $\sigma \in \gamma_q$. Also, for any $s \in S_q$ and $\sigma \in \Sigma$ such that $\delta(q_o\, s\sigma) \in \delta(q, \gamma_q)$ we have $s\sigma \equiv_G s\sigma'$ for some $\sigma' \in \gamma_q$, Again by Proposition 6 and and the prerequisite regarding $\equiv_G$, we obtain $s\sigma \in \operatorname{cfx} E$ for all $s \in S_q$ and all $\sigma \in \Sigma$ with $\delta(q_o\, s\sigma) \in \delta(q, \gamma_q)$. Referring again to Proposition 6, we finally obtain for all $p \in \delta(q, \gamma_q)$ that

$$\forall\, t \in \Sigma^* . \delta(q_o, t) = p \rightarrow t \in \operatorname{cfx} E . \tag{28}$$

We summarise our preliminary result (A) so far as follows: given a reachable state q such that implication (25) holds, we can apply the control pattern γ_q such that effectively the same implication (28) holds for all successor states $p \in \delta(q, \gamma_q)$.

For the second part (B) of the proof, consider any $s \in \operatorname{cfx} E$ and observe, again by Proposition 6, that the state $q := \delta(q_o, s)$ qualifies as candidate for our preliminary result (A). We can then iteratively define a state feedback $h : Q \rightarrow \Gamma$ with $h(p) = \gamma_p$ such that $\delta(q_o, r) = p$ implies $r \in \operatorname{cfx} E$ for any state p which is closed-loop reachable from q, i.e., for any $p \in V := \delta_h(q, \Sigma^*)$. Now pick an arbitrary $p \in V$ and $r \in \Sigma^*$ such that $\delta(q_o, r) = p$. Since $r \in \operatorname{cfx} E$ there exists $t \in \Sigma^*$ such that $rt \in E$ and, hence, $\delta(p, t) \in Q_m$. We choose the shortest such t. If it is the case that $\delta_h(p, t)$ is defined, we have $\delta_h(p, t) = \delta(p, t)$ and, hence, $\delta_h(p, \Sigma^*) \cap Q_m \neq \emptyset$, i.e., p is co-reachable in the closed-loop configuration. If, on the other hand, $\delta_h(p, t)$ is not defined, we consider the longest prefix $u \in \operatorname{pfx} t$ such that $\delta_h(p, u)!$. By our choice of u we have $p' := \delta_h(p, u) \notin Q_m$ and, hence, $ru \notin E$. By $p' \in V$ we have $ru \in \operatorname{cfx} E$ and we pick a supervisor f for $ru \in \operatorname{cfx} E$ via Proposition 4. Referring to property (c), there must exist $\sigma \in f(ru)$ such that $ru\sigma \in \operatorname{pfx} L$. By $h(p') = \gamma_{p'}$ we obtain $\sigma \in f(ru) \subseteq h(p')$ and conclude $\delta_h(p', r\sigma)!$. This contradicts the case prerequisite and, hence, we always have that $\delta_h(p, t) = \delta(p, t) \in Q_m$ and conclude $\delta_h(p, \Sigma^*) \cap Q_m \neq \emptyset$ for all states $p \in V = \delta_h(q, \Sigma^*)$. Thus, h constitutes a state feedback which qualifies for the existential quantification in the definition of the winning states, Eq. (8), and thereby demonstrates $\delta(q_o, s) = q \in W$. Since the choice of $s \in \operatorname{cfx} E$ was arbitrary, this concludes the proof. □

The following theorem is an immediate consequence of the previous two lemmata and summarises our main result.

Theorem 13. Given a plant $L \subseteq \Sigma^*$ and a relatively closed specification E, $\emptyset \neq E = (\operatorname{pfx} E) \cap L$, both regular, consider a realisation $G = (Q, \Sigma, \delta, q_o, Q_m)$, with $L(G) = L$, $L_m(G) = E$ and induced equivalence $\equiv_G$ at least as fine as $\equiv_L$ and $\equiv_E$. Let $W \subseteq Q$ denote the winning states as defined in Eq. (8). Then

$$\forall\, s \in \Sigma^* . s \in \operatorname{cfx} E \leftrightarrow \delta(q_o, s) \in W . \tag{29}$$

□

This concludes our main argument: the fixpoint Z, Eq. (15), by Theorem 10 matches the set of winning states, Eq. (8), which in turn by Theorem 13 characterises the controllability prefix; i.e., given an automaton $G = (Q, \Sigma, \delta, q_o, Q_m)$ that qualifies for Theorem 13, compute the fixpoint Z and let $G_Z := (Q, \Sigma, \delta, q_o, Z)$ to obtain $\operatorname{cfx} E = L_m(G_Z)$. As a corollary, we can use the same computational procedure to obtain a realisation of the supremal controllable and relatively closed sublanguage.

Corollary 14. Consider $L, E \subseteq \Sigma^*$, $\emptyset \neq E = (\operatorname{pfx} E) \cap L$, a realisation $G = (Q, \Sigma, \delta, q_o, Q_m)$, and the set of winning states W, all under the same hypothesis as in Theorem 13. Apply the state feedback $h : Q \rightarrow \Gamma$ defined by Eqs. (18a/b) in the context of Lemma 8, to obtain the closed-loop system $G_h = (Q, \Sigma, \delta_h, q_o, Q_m)$. If $q_o \in W \subseteq Q$, then $L_m(G_h) = \sup \mathcal{CF}(L, E) \neq \emptyset$. Else, $q_o \notin W \subseteq Q$ and $\sup \mathcal{CF}(L, E) = \emptyset$.

Proof. For the case $q_o \notin W \subseteq Q$ we apply Theorem 13 to obtain $\epsilon \notin \operatorname{cfx} E$ and, hence, by Definition 3, $K^\uparrow := \sup \mathcal{CF}(L, E) = \emptyset$. Thus, we may assume $q_o \in W \subseteq Q$ from now on.

Pick an arbitrary $s \in L_m(G_h)$. Observe that $\delta_h(q_o, \operatorname{pfx} s) \subseteq \delta_h(q_o, \Sigma^*) \subseteq \delta_h(W, \Sigma^*) \subseteq W$, where the last inclusion follows with Lemma 8. We apply Theorem 13 and obtain $\operatorname{pfx} s \subseteq \operatorname{cfx} E$. Since $\operatorname{pfx} s$ is closed, this implies $\operatorname{pfx} s \subseteq \sup \mathcal{P}(\operatorname{cfx} E)$. Together with $s \in L_m(G_h) \subseteq E \subseteq L$ we conclude $s \in K^\uparrow$ via Lemma 5. Hence, $L_m(G_h) \subseteq K^\uparrow$.

For the converse inclusion $K^\uparrow \subseteq L_m(G_h)$ pick an arbitrary $s \in K^\uparrow$. Since $s \in K^\uparrow \subseteq E = L_m(G)$, it is sufficient to establish that $s \in L(G_h)$. This is trivially the case if $s = \epsilon$. From now on consider the case $s \neq \epsilon$. Referring to Lemma 5, we have that $s \in \sup \mathcal{P}(\operatorname{cfx} E)$. This implies $\operatorname{pfx} s \subseteq \sup \mathcal{P}(\operatorname{cfx} E) \subseteq \operatorname{cfx} E$. Applying Theorem 13, we obtain $\delta(q_o, \operatorname{pfx} s) \subseteq W$. Decompose s by $r\sigma \in \operatorname{pfx} s$ with $r \in \Sigma^*$ and $\sigma \in \Sigma$, and assume that

$r \in \mathrm{L}(G_h)$. With $q := \delta(q_o, r) \in W$ we have that $q' := \delta(q, \sigma) = \delta(q_o, r\sigma) \in W$. Referring to the definition of h, case (18a), this implies $\sigma \in h(q)$ and, hence, $r\sigma \in \mathrm{L}(G_h)$. Since $\epsilon \in \mathrm{L}(G_h)$, we obtain $r \in \mathrm{L}(G_h)$ for all $r \in \mathrm{pfx}\, s$ inductively. In particular, we have that $s \in \mathrm{L}(G_h)$. □

Example. Consider the alphabet $\Sigma = \{\alpha, \beta\}$, $\Sigma_c = \{\alpha\}$, $\Sigma_{uc} = \{\beta\}$, the plant $L \subseteq \Sigma^*$, and the specification E, $\emptyset \neq E = (\mathrm{pfx}\, E) \cap L$, with a trim and a full realisation given in Figure 3, respectively. The product automaton G is shown in Figure 4 and satisfies the prerequisites of Theorem 13. The set of winning states $W = Z$ obtained via our fixpoint (15) fails to include the initial state. We conclude by Corollary 14 that $\emptyset = \sup \mathcal{CF}(L, E)$, i.e., the synthesis problem has no solution. Nevertheless, we can inspect G_h obtained by applying the state feedback h given by Eqs. (18a/b); i.e., by removing transitions as highlighted (dashed) in Figure 4. This feedback will be functional under the additional hypothesis that either the sequence $s_1 = \beta$ will not be generated or else that $s_2 = \beta\beta\beta$ will not be generated. If the plant model has been obtained by abstraction, the latter should be refined regarding those two sequences. In contrast, any refinement that addresses the behaviour after $s_3 = \alpha$ is not worth the effort since this sequence leads to a winning state anyway. Note that for the given problem parameters the standard procedure for the computation of $\sup \mathcal{CF}(L, E)$ will simply return the empty automaton. □

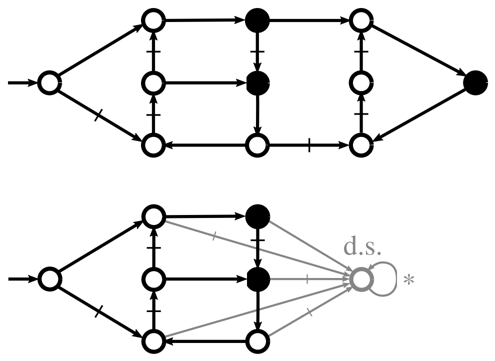

Fig. 3. Realisation of plant L (top) and specification E (bottom)

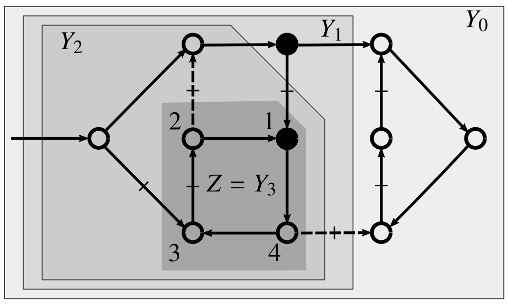

Fig. 4. Automaton G with winning states $W = Z$

CONCLUSION

We have discussed the controllability prefix for the basic supervisory control problem with an upper-bound language-inclusion specification. Extending the algebraic properties identified in earlier work (Moor and Schmidt, 2015, 2017), we have derived a novel characterisation, Lemma 5, of the supremal controllable and relatively closed sublanguage and thereby parallel the situation of ω-languages as discussed by Thistle and Wonham (1994b). Turning attention to automata realisations and non-blocking state feedback, we take a game theoretic perspective and characterise the set of winning states by a two-nested fixpoint, Theorem 10. Finally, we identify a one-to-one correspondence of the winning states for a strategically constructed automaton and the controllability prefix, Theorem 13. This enables the effective computation of the controllability prefix with quadratic complexity and constitutes our main result. As a corollary, the procedure can also be used as an alternative for the computation of the supremal controllable and relatively closed sublanguage commonly studied in supervisory control.

REFERENCES

Cassandras, C.G. and Lafortune, S. (2008). *Introduction to Discrete Event Systems*. Springer, second edition.

Ehlers, R., Lafortune, S., Tripakis, S., and Vardi, M.Y. (2017). Supervisory control and reactive synthesis: a comparative introduction. *Discrete Event Dynamic Systems*, 27(2), 209–260.

Finkbeiner, B. (2016). Synthesis of reactive systems. In *Dependable Software Systems Engineering*, volume 45 of *NATO Science for Peace and Security Series, D: Information and Communication Security*, 72–98.

Moor, T. and Schmidt, K.W. (2015). Fault-tolerant control of discrete-event systems with lower-bound specifications. *Workshop on Dependable Control of Discrete Systems (DCDS)*.

Moor, T. and Schmidt, K.W. (2017). The controllability prefix for supervisory control under partial observation. *20th IFAC World Congress, invited track DCDS*, 14206–14211.

Ramadge, P.J. and Wonham, W.M. (1987). Supervisory control of a class of discrete event processes. *SIAM J. Control and Optimization*, 25, 206–230.

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

Schmuck, A.K., Moor, T., and Majumdar, R. (2020). On the relation between reactive synthesis and supervisory control of non-terminating processes. *Discrete Event Dynamic Systems*, 30, 81–124.

Thistle, J.G. and Wonham, W.M. (1994a). Control of infinite behavior of finite automata. *SIAM J. Control and Optimization*, 32, 1075–1097.

Thistle, J.G. and Wonham, W.M. (1994b). Supervision of infinite behavior of discrete event systems. *SIAM J. Control and Optimization*, 32, 1098–1113.

Wonham, W.M. and Cai, K. (2019). *Supervisory control of discrete-event systems*. Communications and Control Engineering. Springer.

Wonham, W.M. and Ramadge, P.J. (1987). On the supremal controllable sublanguage of a given language. *SIAM Journal on Control and Optimization*, 25(3), 637–659.

Yang, J.M., Moor, T., and Raisch, J. (2020). Refinements of behavioural abstractions for the supervisory control of hybrid systems. *Accepted for publication in Discrete Event Dynamic Systems*.

Yari, F. and Hashtrudi-Zad, S. (2016). Computational procedures for robust nonblocking supervisory control. In *Canadian Conf. on Electrical and Computer Engineering*, 1–5.

Ziller, R.M. and Cury, J.E.R. (1994). On the supremal Lm-closed and the supremal Lm-closed and L-controllable sublanguages of a given language. *LNCIS*, 199, 80–85.