Working Paper Series
ISSN 1177-777X

Framework and Proofs for Synthesis of Least Restrictive Controllable Supervisors for Extended Finite-State Machines with Variable Abstraction

Robi Malik and Marcelo Teixeira

Working Paper: 03/2018
October 9, 2018

Department of Computer Science
The University of Waikato
Private Bag 3105
Hamilton, 3240
New Zealand

Abstract

This working paper presents an algorithm that combines modular synthesis for extended finite-state machines (EFSM) with abstraction of variables by symbolic manipulation, in order to compute least restrictive controllable supervisors. Given a modular EFSM system consisting of several components, the proposed algorithm synthesises a separate supervisor for each specification component. To synthesise each supervisor, the algorithm iteratively selects components (plants and variables) from a synchronous composition until a least restrictive controllable solution is obtained. This improves on previous results of the authors where abstraction is only performed by the selection of components and not variables. The working paper explains the theory of EFSM synthesis and abstraction and includes formal proofs of all results. An example of a flexible manufacturing system illustrates how the proposed algorithm works to compute a modular supervisor.

Contents

1 Introduction

Supervisory Control Theory [3, 17] provides a general framework for the synthesis of reactive control functions. Given a model of the system, the *plant*, to be controlled, and a *specification* of the desired behaviour, it is possible to automatically compute, i.e. *synthesise*, a *supervisor* that restricts the plant behaviour while satisfying the specification. Originally, the theory is grounded on the *Finite-state Machines* formalism and several approaches have been developed to make synthesis more efficient [1, 12, 22, 24].

In recent years, Supervisory Control Theory has been generalised for *Extended Finite-state Machines* (EFSM) [4, 14, 23], which include *variables* and improve modelling capabilities for systems with data dependency or software. Variables are more general than approaches with similar purpose [7, 16], including *event distinguishers* [5, 18, 22], and can simplify the modelling task for various discrete event systems. However, they require more sophisticated tools and methods for synthesis. Several synthesis algorithms for EFSMs have been proposed [8, 11, 15, 23], which explore the full system state space, including all possible combinations of variable values. The resulting complexity can be avoided to some extent using *symbolic* representation [11] or *abstraction* [19, 23].

Recently, a modular approach for the synthesis of *least restrictive* and *controllable* supervisors from plants modelled with EFSMs has been proposed [10], which generalises earlier work on modular synthesis without variables [1, 2]. The approach considers only prefix-closed behaviours, and the system model consists of several interacting plant and specification components. In this case, synthesis can be performed separately for each specification EFSM, and the results can be combined to form a modular supervisor. For each specification, the algorithm [10] iteratively selects plant components to be included in synthesis until a least restrictive controllable solution is found. The obtained modular supervisors, in combination, achieve the least restrictive controllable behaviour for the entire system.

This working paper extends the approach of [10] by including the idea of *existential abstraction* [23] of variables. The algorithm of [10] performs abstraction only by selecting EFSM components and always includes all variables of the selected components. Existential abstraction improves on this, because it allows for abstractions to be formed by selecting components and some of their variables. Other variables are quantified out and do not contribute to the state space when synthesis is performed.

In the following, Section 2 introduces the background of extended finite-state machines, and the following sections describe the proposed abstraction method and prove its correctness. Section 3 considers the abstraction of variables from plant components, Section 4 considers the abstraction of variables from specification components, and Section 5 combines these results with other previous work to present an algorithm for synthesis for EFSM systems consisting of multiple plant and specification components. This algorithm is illustrated by an example in Section 6, and afterwards Section 7 adds concluding remarks.

2 Preliminaries

A *finite-state machine (FSM)* consists of a finite set of states linked by transitions, which are labelled by discrete events [3]. This working paper considers *extended finite-state machines (EFSM)*, which add to FSMs *variables* and the ability to read and update these variables on the occurrence of transitions [4, 13].

2.1 Variables and Updates

An *update* is a first-order logic formula [6] constructed from variables, integer constants, Boolean literals, the existential and universal quantifiers ($\exists$ and $\forall$), and the usual arithmetic and logic connectives. Variables can be *bound* to quantifiers or occur *free* in an update. For example, in the update $\exists y\ x > y + 2$, the variable y is bound to the existential quantifier, while x is a free variable. The set of all update formulas is denoted by Π.

In this working paper, formulas are interpreted over finite domains. Every *variable* z is associated with a finite discrete *domain* $\mathrm{dom}(z)$ and an initial value $\hat{z}^\circ \in \mathrm{dom}(z)$. Let $V = \{v_0, \ldots, v_n\}$ be the set of variables with combined domain $\mathrm{dom}(V) = \mathrm{dom}(v_0) \times \cdots \times \mathrm{dom}(v_n)$. An element $\hat{v}$ of $\mathrm{dom}(V)$ is also considered as a *valuation* that assigns to each variable $z \in V$ a value $\hat{v}(z) \in \mathrm{dom}(z)$, and by extension a truth value to each update. The *initial valuation* is $V^\circ \in \mathrm{dom}(V)$ with $V^\circ(z) = \hat{z}^\circ$ for each $z \in V$.

A second set of variables, called *next-state* variables and denoted $V' = \{\, z' \mid z \in V \,\}$ is used to describe the values of the variables after a transition. Variables in V are also referred to as *current-state* variables to differentiate them from the next-state variables in V'. The next-state variable z' has the same domain as its current-state variable z. Given $\hat{v} \in \mathrm{dom}(V)$, the valuation $\hat{v}' \in \mathrm{dom}(V')$ is defined by $\hat{v}'(z') = \hat{v}(z)$ for all $z \in V$. For an update $p \in \Pi$, the term $\mathrm{vars}(p)$ denotes the set of all variables with a free occurrence as current-state or next-state variable in p, and $\mathrm{vars}'(p)$ denotes the set of all variables whose corresponding next-state variables have a free occurrence in p. For example, if $p \equiv \exists z\, x' = y + z + 1$, then $\mathrm{vars}(p) = \{x, y\}$ and $\mathrm{vars}'(p) = \{x\}$. Here and in the following, the relation $\equiv$ denotes syntactic identity of updates to avoid ambiguity when an update contains the equality symbol $=$.

An update $p \in \Pi$ is *satisfiable* if it is true for at least one valuation of its variables, i.e., if there is a valuation $\hat{v} \in \mathrm{dom}(\mathrm{vars}(p))$ such that $\hat{v}(p) = \mathrm{true}$. Otherwise the update p is *unsatisfiable*. An update p is *valid* if it is true for all valuations of its variables, i.e., if $\hat{v}(p) = \mathrm{true}$ for every valuation $\hat{v}$. The *restriction* of a valuation $\hat{v} \in \mathrm{dom}(V)$ to $W \subseteq V$ is $\hat{v}_{|W} \in \mathrm{dom}(W)$ with $\hat{v}_{|W}(z) = \hat{v}(z)$ for all $z \in W$. Two valuations $\hat{v} \in \mathrm{dom}(V)$ and $\hat{w} \in \mathrm{dom}(W)$ can be combined to give $\hat{v} \oplus \hat{w} \in \mathrm{dom}(V \cup W)$ where $(\hat{v} \oplus \hat{w})(z) = \hat{v}(z)$ for $z \in V$ and $(\hat{v} \oplus \hat{w})(z) = \hat{w}(z)$ for $z \in W \setminus V$.

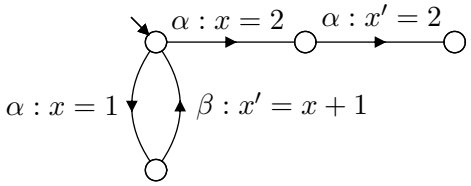

Figure 1: Example of an EFSM.

2.2 Extended Finite-State Machines

Definition 1 An *Extended finite-state machine (EFSM)* is a tuple $F = \langle \Sigma, Q, Q^\circ, \rightarrow \rangle$, where Σ is a finite set of events, Q is a finite set of *locations*, $Q^\circ \subseteq Q$ is the set of *initial locations*, and $\rightarrow \subseteq Q \times \Sigma \times \Pi \times Q$ is the *extended transition relation*.

A transition between locations $x, y \in Q$ with event $\sigma \in \Sigma$ and update $p \in \Pi$ is written $x \overset{\sigma:p}{\rightarrow} y$. It can occur if F is in location x and the update p evaluates to true, and when it occurs, F changes its location to y while updating the variables in $\mathrm{vars}'(p)$ in accordance with p; variables not in $\mathrm{vars}'(p)$ remain unchanged. The transition relation *is extended for events not in the event set* of the EFSM, $\sigma \notin \Sigma$, by defining $x \xrightarrow{\sigma:\mathrm{true}} x$ for all locations $x \in Q$.

Example 1 Consider the EFSM F in Figure 1, which has only one variable x with domain $\mathrm{dom}(x) = \{0, \ldots, 5\}$. The update $x' = x + 1$ of the β-transition changes the variable x by adding 1 to its current value, if it currently is less than 5. Otherwise (if $x = 5$) the transition is disabled. The update $x = 2$ disables its transition unless $x = 2$ in the current state, and the value of x in the next state is unchanged if the transition is taken. Differently, the update $x' = 2$ always enables its transition, and the value of x in the next state is forced to be 2.

Given an EFSM F and event $\sigma \in \Sigma$, the *referenced variable set* is $\mathrm{vars}(F, \sigma) = \bigcup\{\, \mathrm{vars}(p) \mid x \overset{\sigma:p}{\rightarrow} y \,\}$, and $\mathrm{vars}(F) = \bigcup_{\sigma \in \Sigma} \mathrm{vars}(F, \sigma)$. Furthermore, for a set $\mathcal{F}$ of EFSMs, $\mathrm{vars}(\mathcal{F}, \sigma) = \bigcup_{F' \in \mathcal{F}} \mathrm{vars}(F', \sigma)$ and $\mathrm{vars}(\mathcal{F}) = \bigcup_{F' \in \mathcal{F}} \mathrm{vars}(F')$. Analogous notation is defined for vars'.

Definition 2 [10] Let $F = \langle \Sigma, Q, Q^\circ, \rightarrow \rangle$ be an EFSM.

(i) F is *normalised*, if for any two transitions $x_1 \xrightarrow{\sigma:p_1} y_1$ and $x_2 \xrightarrow{\sigma:p_2} y_2$ with the same event $\sigma \in \Sigma$, it holds that $\mathrm{vars}'(p_1) = \mathrm{vars}'(p_2)$.

(ii) F is *pure* if $\mathrm{vars}'(F) = \emptyset$.

(iii) F is *state-deterministic* if $|Q^\circ| \leq 1$, and for all transitions $x \xrightarrow{\sigma:p_1} y_1$ and $x \xrightarrow{\sigma:p_2} y_2$ such that $p_1 \wedge p_2$ is satisfiable, it holds that $y_1 = y_2$.

A set $\mathcal{F} = \{F_1, \ldots, F_n\}$ of EFSMs is normalised, pure, or state-deterministic if every EFSM $F_i \in \mathcal{F}$ has this property.

In a normalised EFSM, the set of variables changed by an event is the same on all transitions. This assumption helps to recognise the implicitly unchanged variables after synchronous composition. Every EFSM can be transformed into a normalised EFSM by a process of renaming similar to normalisation [13]. As a stronger condition, a pure EFSM cannot assign any variables, it only restricts events. State-determinism ensures that the target locations are uniquely determined from the source location, event, and variable assignment. It is needed for supervisors to track the location of the plant by the observation of events and variable values.

Example 2 Consider again the EFSM F in Figure 1. This EFSM is *not* normalised, because it has α-transitions with updates $x = 1$ and $x' = 2$, and $\mathrm{vars}'(x = 1) = \emptyset \neq \{x\} = \mathrm{vars}'(x' = 2)$. That is, some α-transitions explicitly change x while others leave x implicitly unchanged. The EFSM F is also not pure, for example $\mathrm{vars}'(x' = 2) \neq \emptyset$.

On the other hand, F is state-deterministic: although there are two α-transitions with different targets originating from the initial location, the dependence of the guards on the value of x ensures that these transitions cannot be enabled at the same time. The conjunction $x = 1 \wedge x = 2$ is unsatisfiable as x can never have both the values 1 and 2.

In this working paper, plants are modelled by normalised state-deterministic EFSMs, while specifications are pure state-deterministic EFSMs. The synthesised supervisor is also normalised and state-deterministic, but unlike the specification not necessarily pure so that it can restrict variable assignments.

An EFSM $F = \langle \Sigma, Q, Q^\circ, \to \rangle$ can be *unfolded* [13, 23] and interpreted as an FSM with state set $Q \times \mathrm{dom}(\mathrm{vars}(F))$. The states $(x, \hat{v})$ consist of a location $x \in Q$ and a valuation $\hat{v} \in \mathrm{dom}(\mathrm{vars}(F))$. More specifically, the unfolded transition relation is defined as follows.

Definition 3 Let $F = \langle \Sigma, Q, Q^\circ, \to \rangle$ be an EFSM, and let $V \supseteq \mathrm{vars}(E)$. The *unfolded transition relation* $\to \subseteq (Q \times \mathrm{dom}(V)) \times \Sigma \times (Q \times \mathrm{dom}(V))$ is defined such that $(x, \hat{v}) \xrightarrow{\sigma} (y, \hat{w})$ if and only if there exists a transition $x \xrightarrow{\sigma:p} y$ in F such that $(\hat{v} \oplus \hat{w}')(p)$ is true and $\hat{v}(z) = \hat{w}(z)$ for all variables $z \in V \setminus \mathrm{vars}'(p)$.

Thus, an unfolded transition between two states $(x, \hat{v}) \xrightarrow{\sigma} (y, \hat{w})$ exists if F contains a transition $x \xrightarrow{\sigma:p} y$ such that the update p is true, if the current-state variables are interpreted according to $\hat{v}$ and the next-state variables according to $\hat{w}$, and all variables that do not appear as next-state variables in the update p are unchanged between $\hat{v}$ and $\hat{w}$. This transition relation is extended to events not in the EFSM's event set Σ, which are always enabled without changing the EFSM's location or any variables. That is, $(x, \hat{v}) \xrightarrow{\sigma} (x, \hat{v})$, for all $x \in Q$, $\hat{v} \in \mathrm{dom}(V)$, and $\sigma \notin \Sigma$.

The $\to$ notation is extended to traces, state sets, and state machines in the same way as for FSMs. For example, a transition sequence

$$(x^0, \hat{v}^0) \xrightarrow{\sigma_1} (x^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x^n, \hat{v}^n) \tag{2.1}$$

is written $(x^0, \hat{v}^0) \xrightarrow{s} (x^n, \hat{v}^n)$ for $s = \sigma_1 \cdots \sigma_n$. This transition sequence is a *path* in F if it starts in an initial state of F, i.e., if $x^0 \in Q^\circ$ and $\hat{v}^0 = V^\circ$, which is also written as $F \xrightarrow{s} (x^n, \hat{v}^n)$. Based on this, the set of *accessible states* of an EFSM F is

$$Q^{\mathrm{acc}}(F) = \{\, (x, \hat{v}) \in Q \times \mathrm{dom}(\mathrm{vars}(F)) \mid F \xrightarrow{s} (x, \hat{v}) \text{ for some } s \in \Sigma^* \,\} \,. \tag{2.2}$$

With the following definition, an EFSM can be *restricted* to a set of unfolded states, symbolically, by rewriting updates to impose new constraints on the variables.

Definition 4 Let $F = \langle \Sigma, Q, Q^\circ, \to \rangle$ be an EFSM with $V = \mathrm{vars}(F)$, and let $X \subseteq Q \times \mathrm{dom}(V)$. The *symbolic restriction* of F to X is an EFSM $F \restriction X = \langle \Sigma, Q_{|X}, Q^\circ_{|X}, \to_{|X} \rangle$, where

$$Q_{|X} = \{\, x \in Q \mid (x, \hat{v}) \in X \text{ for some } \hat{v} \in \mathrm{dom}(V) \,\} \,; \tag{2.3}$$

$$Q^\circ_{|X} = \{\, x^\circ \in Q^\circ \mid (x^\circ, V^\circ) \in X \,\} \,; \tag{2.4}$$

and $x \xrightarrow{\sigma : p \wedge R_X[p,y]}_{|X} y$, if $x, y \in Q_{|X}$, and $x \xrightarrow{\sigma:p} y$, and $R_X[p,y] \in \Pi$ is an update with $\mathrm{vars}(R_X[p,y]) \subseteq V$ and $\mathrm{vars}'(R_X[p,y]) \subseteq \mathrm{vars}'(p)$ such that, for all valuations $\hat{v}, \hat{w} \in \mathrm{dom}(V)$ it holds that $(y, \hat{v}\restriction_{V \setminus \mathrm{vars}'(p)} \oplus \hat{w}) \in X$ if and only if $(\hat{v} \oplus \hat{w}')(R_X[p,y]) = \mathrm{true}$.

The symbolic restriction formula $R_X[p,y]$ constrains the updates of the EFSM F to ensure that only states in X can be entered *without changing* any variables that are unchanged in F. When constraining a transition with update p, the formula $R_X[p,y]$ only uses next-state variables that also appear as next-state variables in p; other variables are referenced through their current-state name as they remain unchanged. Therefore, the valuation $\hat{v}\restriction_{V \setminus \mathrm{vars}'(p)} \oplus \hat{w}$ after the transition includes the values of unassigned variables $z \notin \mathrm{vars}'(p)$ from $\hat{v}$ before the transition and the values of other variables according to $\hat{w}'$ satisfying the update formula. It is always possible to construct a formula $R_X[p,y]$ as required in the definition, although it is not unique. In the following it is assumed that the formula is obtained deterministically by an appropriate algorithm.

The following lemma confirms that symbolically restricting an EFSM to some set $X \subseteq Q \times \mathrm{dom}(V)$ ensures that the unfolded state space is confined within that set X.

Lemma 1 Let $F = \langle \Sigma, Q, Q^\circ, \to \rangle$ be an EFSM, and let $X \subseteq Q \times \mathrm{dom}(\mathrm{vars}(F))$. Then $Q^{\mathrm{acc}}(F \restriction X) \subseteq X$.

Proof. Let $V = \mathrm{vars}(V)$, and assume $(x, \hat{u}) \in Q^{\mathrm{acc}}(F \restriction X)$ for $\hat{v} \in \mathrm{dom}(V)$. Then there is a path

$$(x^0, \hat{u}^0) \xrightarrow{\sigma_1} (x^1, \hat{u}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x^n, \hat{u}^n) = (x, \hat{u}) \tag{2.5}$$

in $F \restriction X$. Consider two cases.

If $n = 0$, then note that $x^0 \in Q^\circ_{|X}$, which by (2.4) means $(x, \hat{u}) = (x^0, \hat{u}^0) = (x^0, V^\circ) \in X$ as $\hat{u}^0 = V^\circ$ is the initial valuation.

If $n > 0$, then there is a transition $x_{n-1} \xrightarrow{\sigma:p \wedge R}_{|X} x_n$ in $F \restriction X$ with $R \equiv R_X[p, x_n]$ such that $(\hat{u}^{n-1} \oplus (\hat{u}^n)')(p \wedge R) = \text{true}$ and $\hat{u}^{n-1}(z) = \hat{u}^n(z)$ for all $z \in V \setminus \text{vars}'(p \wedge R)$. By Definition 4, the update R is such that $\text{vars}'(R) \subseteq \text{vars}'(p)$ and for all valuations $\hat{v}, \hat{w} \in \text{dom}(V)$ it holds that $(x_n, \hat{v}\restriction_{V \setminus \text{vars}'(p)} \oplus \hat{w}) \in X$ if and only if $(\hat{v} \oplus \hat{w}')(R) = \text{true}$. From $\text{vars}'(R) \subseteq \text{vars}'(p)$ it follows that $\hat{u}^{n-1}(z) = \hat{u}^n(z)$ for all $z \in V \setminus \text{vars}'(p \wedge R) = V \setminus \text{vars}'(p)$. Then note that $\hat{u}^n = \hat{u}^n\restriction_{V \setminus \text{vars}'(p)} \oplus \hat{u}^n = \hat{u}^{n-1}\restriction_{V \setminus \text{vars}'(p)} \oplus \hat{u}^n$. It follows from $(\hat{u}^{n-1} \oplus (\hat{u}^n)')(p \wedge R) = \text{true}$ that $(\hat{u}^{n-1} \oplus (\hat{u}^n)')(R) = \text{true}$ and thus $(x, \hat{u}) = (x^n, \hat{u}^n) = (x^n, \hat{u}^{n-1}\restriction_{V \setminus \text{vars}'(p)} \oplus \hat{u}^n) \in X$. □

2.3 Synchronous Composition

EFSMs are composed using lock-step synchronisation on shared events, like ordinary FSMs, but in addition the updates are combined by conjunction.

Definition 5 The *synchronous composition* of two EFSMs $F_1 = \langle \Sigma_1, Q_1, Q_1^\circ, \to_1 \rangle$ and $F_2 = \langle \Sigma_2, Q_2, Q_2^\circ, \to_2 \rangle$ is

$$F_1 \parallel F_2 = \langle \Sigma_1 \cup \Sigma_2, Q_1 \times Q_2, Q_1^\circ \times Q_2^\circ, \to \rangle \ , \tag{2.6}$$

where $(x_1, x_2) \xrightarrow{\sigma:p_1 \wedge p_2} (y_1, y_2)$ if $x_1 \xrightarrow{\sigma:p_1}_1 y_1$ and $x_2 \xrightarrow{\sigma:p_2}_2 y_2$.

This definition captures EFSMs with different event sets through the extended definition of the transition relation. For example, if $x_1 \xrightarrow{\sigma:p}_1 y_1$ in F_1 and F_2 does not synchronise on this event, $\sigma \notin \Sigma_2$, then the extended transition relation of F_2 includes $x_2 \xrightarrow{\sigma:\text{true}}_2 x_2$ for every location $x_2 \in Q_2$. This results in synchronised transitions $(x_1, x_2) \xrightarrow{\sigma:p \wedge \text{true}} (y_1, x_2)$, or equivalently $(x_1, x_2) \xrightarrow{\sigma:p} (y_1, x_2)$, in $F_1 \parallel F_2$.

As a result of the conjunctive combination of updates, they may cancel each other out. For example, if $x_1 \xrightarrow{\sigma:z'=0}_1 y_1$ in F_1 and $x_2 \xrightarrow{\sigma:z'=1}_2 y_2$ in F_2, then the conjunction $z' = 0 \wedge z' = 1$ is unsatisfiable, or equivalently there is no such transition in $F_1 \parallel F_2$. Synchronous composition can override the assumption of implicitly unchanged variables in an EFSM. If $x_1 \xrightarrow{\sigma:z=0}_1 y_1$ and $x_2 \xrightarrow{\sigma:z'=z+1}_2 y_2$, e.g., then $(x_1, x_2) \xrightarrow{\sigma:z=0 \wedge z'=z+1} (y_1, y_2)$. So the value of z changes from 0 to 1 in $F_1 \parallel F_2$ although implicitly unchanged in F_1.

EFSM synchronous composition is associative and commutative, apart from the renaming of locations and rewriting of updates into equivalent formulas. Synchronous composition is not idempotent as $F \parallel F = F$ does not generally hold for non-deterministic state machines. If F is a state-deterministic EFSM, then $F \parallel F = F$ holds up to isomorphism, after deletion of transitions with unsatisfiable updates and inaccessible locations.

This working paper considers systems modelled as the synchronous composition of several EFSMs. Then the model consists of a *set* of EFSMs, $\mathcal{F} = \{F_1, \ldots, F_n\}$, and the notation $\big\|(\mathcal{F}) = F_1 \parallel \cdots \parallel F_n$ denotes their synchronous composition. As a special case, $\big\|(\emptyset) =$

$\langle \emptyset, \{x^\circ\}, \{x^\circ\}, \emptyset \rangle$ is the neutral element of synchronous composition. This is a one-location EFSM without events, which by definition accepts all events without changing its location or assigning its variables.

The following lemma describes a criterion to determine the presence of a transition in the unfolded transition relation of a synchronous composition. Such a transition exists if each of the composed EFSMs has a transition whose update evaluates is true, and the variables that do not appear as next-state variables in any of these updates are unchanged.

Lemma 2 Let $F_1, \ldots, F_n$ be EFSMs, $V \supseteq \mathrm{vars}(F_1) \cup \cdots \cup \mathrm{vars}(F_n)$, and $\hat{v}, \hat{w} \in \mathrm{dom}(V)$. Then

$$(x_1, \ldots, x_n, \hat{v}) \xrightarrow{\sigma} (y_1, \ldots, y_n, \hat{w}) \qquad \text{in } F_1 \parallel \cdots \parallel F_n \tag{2.7}$$

if and only if each F_i has a transition $x_i \xrightarrow{\sigma:p_i} y_i$ with $(\hat{v} \oplus \hat{w}')(p_i) = \mathrm{true}$, and $\hat{v}\restriction_U = \hat{w}\restriction_U$ where $U = V \setminus (\mathrm{vars}'(p_1) \cup \cdots \cup \mathrm{vars}'(p_n))$.

The proof of this lemma is immediate from Definitions 3 and 5 as the EFSM $F_1 \parallel \cdots \parallel F_n$ must contain a transition $(x_1, \ldots, x_n, \hat{v}) \xrightarrow{\sigma:p_1 \wedge \cdots \wedge p_n} (y_1, \ldots, y_n, \hat{w})$, with the variables not appearing primed in the updates remaining unchanged. Under the assumption of normalisation, the set U of unchanged variables can also be written as $U = V \setminus (\mathrm{vars}'(F_1, \sigma) \cup \cdots \cup \mathrm{vars}'(F_n, \sigma))$. Note that if the event σ is not in the alphabet of some F_i, then by definition $x_i \xrightarrow{\sigma:\mathrm{true}} x_i$.

For ordinary FSMs, it is known that any path in synchronous composition corresponds to a path in each of the composed FSMs. This is not guaranteed form EFSMs, as variable changes performed by one component may affect the path in the other components. Such a path in one component can only be reconstructed if the other components do not change variables. As a first application of Lemma 2, the following Lemma 3 shows this property under the assumption that synchronous composition is performed with a pure EFSM.

Lemma 3 Let A and B be EFSMs, where B is pure, and let

$$(x_A^0, x_B^0, \hat{v}^0) \xrightarrow{\sigma_1} (x_A^1, x_B^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_A^n, x_B^n, \hat{v}^n) \tag{2.8}$$

be a path in $A \parallel B$ where $\hat{v}^i \in \mathrm{dom}(V)$ for some $V \supseteq \mathrm{vars}(A) \cup \mathrm{vars}(B)$. Then

$$(x_A^0, \hat{v}^0) \xrightarrow{\sigma_1} (x_A^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_A^n, \hat{v}^n) \tag{2.9}$$

is a path in A.

Proof. It is clear that x_A^0 and $\hat{v}^0$ are initial locations and variable values from (2.8). Considering the i-th transition on the path (2.8), it follows by Lemma 2 that there are transitions

$$x_A^{i-1} \xrightarrow{\sigma_i:p_A} y_A^i \qquad \text{in } A \qquad \text{with } (\hat{v}^{i-1} \oplus (\hat{v}^i)')(p_A) = \mathrm{true}\ ; \tag{2.10}$$

$$x_B^{i-1} \xrightarrow{\sigma_i:p_B} y_B^i \qquad \text{in } B \qquad \text{with } (\hat{v}^{i-1} \oplus (\hat{v}^i)')(p_B) = \mathrm{true}\ ; \tag{2.11}$$

such that

$$\hat{v}^{i-1}(z) = \hat{v}^i(z) \qquad \text{for all } z \in V \setminus (\mathrm{vars}'(p_A) \cup \mathrm{vars}'(p_B)) = V \setminus \mathrm{vars}'(p_A) \ . \tag{2.12}$$

The last equality holds since $\mathrm{vars}'(p_B) = \emptyset$ as B is pure. By Definition 3, combining (2.10) and (2.12) gives a transition $(x_A^{i-1}, \hat{v}^{i-1}) \xrightarrow{\sigma_i} (x_A^i, \hat{v}^i)$ in A, and the path (2.9) is obtained by repeating this argument for all transitions on the path (2.8). □

Another important property of EFSM synchronous composition is that state-determinism ensures that the sequence of locations visited by an EFSM for a given sequence of events and variable values is uniquely defined, even in synchronous composition with other EFSMs.

Lemma 4 Let A, B, and C be EFSMs, where A is state-deterministic, and let

$$(x_A^0, x_B^0, \hat{v}^0) \xrightarrow{\sigma_1} (x_A^1, x_B^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_A^n, x_B^n, \hat{v}^n) \ ; \tag{2.13}$$

$$(y_A^0, y_C^0, \hat{v}^0) \xrightarrow{\sigma_1} (y_A^1, y_C^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (y_A^n, y_C^n, \hat{v}^n) \tag{2.14}$$

be paths in $A \parallel B$ and $A \parallel C$, respectively. Then $x_A^i = y_A^i$ for $i = 0, \ldots, n$.

Proof. The claim is shown by induction on i. For the base case, $i = 0$, note that x_A^0 and y_A^0 are initial locations of A, and thus $x_A^0 = y_A^0$ by Definition 2 (iii) as A is state-deterministic. Now assume $x_A^i = y_A^i$ for some $i \geq 0$, and consider the next transitions

$$(x_A^i, x_B^i, \hat{v}^i) \xrightarrow{\sigma_{i+1}} (x_A^{i+1}, x_B^{i+1}, \hat{v}^{i+1}) \ ; \tag{2.15}$$

$$(y_A^i, y_C^i, \hat{v}^i) \xrightarrow{\sigma_{i+1}} (y_A^{i+1}, y_C^{i+1}, \hat{v}^{i+1}) \tag{2.16}$$

on the paths (2.13) and (2.14). This means by Lemma 2 that there are transitions in A,

$$x_A^i \xrightarrow{\sigma_{i+1}:p_A} x_A^{i+1} \qquad \text{with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(p_A) = \text{true} \ ; \tag{2.17}$$

$$y_A^i \xrightarrow{\sigma_{i+1}:q_A} y_A^{i+1} \qquad \text{with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(q_A) = \text{true} \ . \tag{2.18}$$

Then $p_A \wedge q_A$ is satisfiable, and noting that $x_A^i = y_A^i$ by inductive assumption, it follows by the state-determinism of A from Definition 2 (iii) and from (2.17) and (2.18) that $x_A^{i+1} = y_A^{i+1}$. □

2.4 Behavioural Inclusion

When comparing EFSMs, variables must be considered in addition to events, so the following notion of behavioural inclusion replaces language inclusion as used for FSMs.

Definition 6 An EFSM F_1 is *behaviourally included* in another EFSM F_2, written $F_1 \subseteq_{\mathrm{v}} F_2$, if for every path

$$(x_0, \hat{v}_0) \xrightarrow{\sigma_1} (x_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_n, \hat{v}_n) \qquad \text{in } F_1 \tag{2.19}$$

with $\hat{v}_i \in \mathrm{dom}(\mathrm{vars}(F_1) \cup \mathrm{vars}(F_2))$, there exists a path

$$(y_0, \hat{v}_0) \xrightarrow{\sigma_1} (y_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (y_n, \hat{v}_n) \qquad \text{in } F_2\ . \tag{2.20}$$

If F_1 is behaviourally included in F_2 then every path in F_1 corresponds to a path in F_2 with the same events and variable assignments. Variables not present in F_1 remain unchanged by F_1 according to Definition 3, and as a consequence must also be unchanged in F_2. This semantics of implicitly unchanged variables makes behavioural inclusion of EFSMs different from language inclusion of FSMs, and several intuitively expected properties do not hold. Therefore, the following lemmas investigate the relation more closely.

First, Lemma 5 shows that behavioural inclusion as defined above is reflexive and transitive. Another desirable property is the *monotonicity* with respect to synchronous composition, i.e., $A \subseteq_{\mathrm{v}} B$ implies $A \parallel C \subseteq_{\mathrm{v}} B \parallel C$. Lemma 6 shows this under the additional assumption that C is pure. Lastly, Lemma 7 shows that symbolic restriction results in an EFSM that is behaviourally included in the original EFSM.

Lemma 5 Let A, B, and C be EFSMs. Then the following properties hold.

(i) $A \subseteq_{\mathrm{v}} A$.

(ii) If $A \subseteq_{\mathrm{v}} B$ and $B \subseteq_{\mathrm{v}} C$ then $A \subseteq_{\mathrm{v}} C$.

Proof.

(i) The reflexivity claim follows immediately from Definition 6.

(ii) Write $V_{AB} = \mathrm{vars}(A) \cup \mathrm{vars}(B)$, $V_{AC} = \mathrm{vars}(A) \cup \mathrm{vars}(C)$, and $V_{BC} = \mathrm{vars}(B) \cup \mathrm{vars}(C)$. Assume that $A \subseteq_{\mathrm{v}} B$ and $B \subseteq_{\mathrm{v}} C$, and consider a path

$$(a_0, \hat{v}_0) \xrightarrow{\sigma_1} (a_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, \hat{v}_n) \tag{2.21}$$

in A with $\hat{v}_i \in \mathrm{dom}(V_{AC})$. Define valuations $\hat{w}_i \in \mathrm{dom}(V_{AB})$ for $i = 0, \ldots, n$ such that $\hat{w}_i = \hat{v}_i{\restriction}_{V_{AB} \cap V_{AC}} \oplus (V_{AB} \setminus V_{AC})^\circ$. Thus, $\hat{w}_i$ is equal to $\hat{v}_i$ for variables that appear in A or C and uses the initial values for variables that appear only in B. Then it can be shown that

$$(a_0, \hat{w}_0) \xrightarrow{\sigma_1} (a_1, \hat{w}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, \hat{w}_n) \tag{2.22}$$

also is a path in A: clearly, a_0 and $\hat{v}_0$ are initial locations and variable values from (2.21), and then $\hat{w}_0$ also only has initial variable values by construction. Also, for each transition

of (2.21) there exists a transition $a_i \xrightarrow{\sigma_{i+1}:p_{i+1}} a_{i+1}$ in A such that $(\hat{v}_i \oplus \hat{v}'_{i+1})(p_{i+1}) = \text{true}$ and $\hat{v}_i(z) = \hat{v}_{i+1}(z)$ for all variables $z \in V_{AC} \setminus \text{vars}'(p_{i+1})$. Note that $\text{vars}(p_{i+1}) \subseteq \text{vars}(A) \subseteq V_{AB} \cap V_{AC}$ and thus

$$\begin{aligned}(\hat{w}_i \oplus \hat{w}'_{i+1})(p_{i+1}) &= (\hat{w}_i{\restriction}_{V_{AB}\cap V_{AC}} \oplus (\hat{w}_{i+1}{\restriction}_{V_{AB}\cap V_{AC}})')(p_{i+1}) \\ &= (\hat{v}_i{\restriction}_{V_{AB}\cap V_{AC}} \oplus (\hat{v}_{i+1}{\restriction}_{V_{AB}\cap V_{AC}})')(p_{i+1}) \\ &= (\hat{v}_i \oplus \hat{v}'_{i+1})(p_{i+1}) \\ &= \text{true} \ .\end{aligned}$$

Also, consider a variable $z \in V_{AB} \setminus \text{vars}'(p_{i+1})$. If $z \in V_{AC}$ then $z \in V_{AB} \cap V_{AC}$ and $z \in V_{AC} \setminus \text{vars}'(p_{i+1})$, i.e, $\hat{w}_i(z) = \hat{v}_i(z) = \hat{v}_{i+1}(z) = \hat{w}_{i+1}(z)$ from above. Otherwise $z \notin V_{AC}$ and thus $z \in V_{AB} \setminus V_{AC}$, i.e., $\hat{w}_i(z) = z^\circ = \hat{w}_{i+1}(z)$ by construction of $\hat{w}_i$. This shows that (2.22) is a path in A. Then, as $A \subseteq_\text{v} B$, there exists a path in B,

$$(b_0, \hat{w}_0) \xrightarrow{\sigma_1} (b_1, \hat{w}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (b_n, \hat{w}_n) \ . \tag{2.23}$$

Define valuations $\hat{u}_i \in \text{dom}(V_{BC})$ for $i = 0, \ldots, n$ such that $\hat{u}_i = \hat{w}_i{\restriction}_{V_{AB}\cap V_{BC}} \oplus (V_{BC} \setminus V_{AB})^\circ$. Then it can be shown that

$$(b_0, \hat{u}_0) \xrightarrow{\sigma_1} (b_1, \hat{u}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (b_n, \hat{u}_n) \tag{2.24}$$

also is a path in B: clearly, b_0 and $\hat{w}_0$ are initial locations and variable values from (2.23), and then $\hat{u}_0$ also only has initial variable values by construction. Also, for each transition of (2.23) there exists a transition $b_i \xrightarrow{\sigma_{i+1}:q_{i+1}} b_{i+1}$ in B such that $(\hat{w}_i \oplus \hat{w}'_{i+1})(q_{i+1}) = \text{true}$ and $\hat{w}_i(z) = \hat{w}_{i+1}(z)$ for all variables $z \in V_{AB} \setminus \text{vars}'(q_{i+1})$. Note that $\text{vars}(q_{i+1}) \subseteq \text{vars}(B) \subseteq V_{AB} \cap V_{BC}$ and thus

$$\begin{aligned}(\hat{u}_i \oplus \hat{u}'_{i+1})(q_{i+1}) &= (\hat{u}_i{\restriction}_{V_{AB}\cap V_{BC}} \oplus (\hat{u}_{i+1}{\restriction}_{V_{AB}\cap V_{BC}})')(q_{i+1}) \\ &= (\hat{w}_i{\restriction}_{V_{AB}\cap V_{BC}} \oplus (\hat{w}_{i+1}{\restriction}_{V_{AB}\cap V_{BC}})')(q_{i+1}) \\ &= (\hat{w}_i \oplus \hat{w}'_{i+1})(q_{i+1}) \\ &= \text{true} \ .\end{aligned}$$

Also, consider a variable $z \in V_{BC} \setminus \text{vars}'(q_{i+1})$. If $z \in V_{AB}$ then $z \in V_{AB} \cap V_{BC}$ and $z \in V_{AB} \setminus \text{vars}'(q_{i+1})$, i.e, $\hat{u}_i(z) = \hat{w}_i(z) = \hat{w}_{i+1}(z) = \hat{u}_{i+1}(z)$ from above. Otherwise $z \notin V_{AB}$ and thus $z \in V_{BC} \setminus V_{AB}$, i.e., $\hat{u}_i(z) = z^\circ = \hat{u}_{i+1}(z)$ by construction of $\hat{u}_i$. This shows that (2.24) is a path in B. Then, as $B \subseteq_\text{v} C$, there exists a path in C,

$$(c_0, \hat{u}_0) \xrightarrow{\sigma_1} (c_1, \hat{u}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (c_n, \hat{u}_n) \ . \tag{2.25}$$

It remains to be shown that then

$$(c_0, \hat{v}_0) \xrightarrow{\sigma_1} (c_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (c_n, \hat{v}_n) \ . \tag{2.26}$$

also is a path in C, where $\hat{v}_i \in \mathrm{dom}(V_{AC})$ are the valuations from (2.21).

Clearly, c_0 is an initial location from (2.25) and $\hat{v}_0$ consists of initial variable values from (2.21). Also, for each transition of (2.25) there exists a transition $c_i \xrightarrow{\sigma_{i+1}:r_{i+1}} c_{i+1}$ in C such that $(\hat{u}_i \oplus \hat{u}'_{i+1})(r_{i+1}) = \mathrm{true}$ and $\hat{u}_i(z) = \hat{u}_{i+1}(z)$ for all variables $z \in V_{BC} \setminus \mathrm{vars}'(r_{i+1})$.

Next, to show that $(\hat{v}_i \oplus \hat{v}'_{i+1})(r_{i+1}) = \mathrm{true}$, consider two cases for $z \in \mathrm{vars}(C) \subseteq V_{AC} \cap V_{BC}$. If $z \in V_{AB}$ then $\hat{v}_i(z) = \hat{v}_i{\restriction}_{V_{AB} \cap V_{BC}}(z) = \hat{w}_i{\restriction}_{V_{AB} \cap V_{BC}}(z) = \hat{w}_i(z) = \hat{w}_i{\restriction}_{V_{AB} \cap V_{AC}}(z) = \hat{u}_i{\restriction}_{V_{AB} \cap V_{AC}}(z) = \hat{u}_i(z)$ by construction of $\hat{w}_i$ and $\hat{u}_i$. If $z \notin V_{AB} \supseteq \mathrm{vars}(A)$ then $\hat{v}_i(z) = z^\circ$ for all i as (2.21) is a path in A, and $z \in V_{BC} \setminus V_{AB}$ so that $\hat{u}_i(z) = \hat{u}_i{\restriction}_{V_{BC} \setminus V_{AB}}(z) = z^\circ$ by construction of $\hat{u}_i$. This shows $\hat{v}_i{\restriction}_{\mathrm{vars}(C)} = \hat{u}_i{\restriction}_{\mathrm{vars}(C)}$. Noting $\mathrm{vars}(r_{i+1}) \subseteq \mathrm{vars}(C)$, it follows that $(\hat{v}_i \oplus \hat{v}'_{i+1})(r_{i+1}) = (\hat{u}_i \oplus \hat{u}'_{i+1})(r_{i+1}) = \mathrm{true}$.

Lastly, to show $\hat{v}_i(z) = \hat{v}_{i+1}(z)$ for $z \in V_{AC} \setminus \mathrm{vars}'(r_{i+1})$, consider $z \in V_{AC} \setminus \mathrm{vars}'(r_{i+1})$. If $z \in \mathrm{vars}(C)$ then $\hat{v}_i(z) = \hat{u}_i(z)$ for $i = 0, \ldots, n$ as just seen, and $z \in \mathrm{vars}(C) \setminus \mathrm{vars}'(r_{i+1}) \subseteq V_{BC} \setminus \mathrm{vars}'(r_{i+1})$, and thus $\hat{v}_i(z) = \hat{u}_i(z) = \hat{u}_{i+1}(z) = \hat{v}_{i+1}(z)$. Otherwise $z \in \mathrm{vars}(A) \setminus \mathrm{vars}(C) \subseteq \mathrm{vars}(A)$, which implies $z \in V_{AB} \cap V_{AC}$ and thus $\hat{v}_i(z) = \hat{v}_i{\restriction}_{V_{AB} \cap V_{AC}}(z) = \hat{w}_i{\restriction}_{V_{AB} \cap V_{AC}}(z) = \hat{w}_i(z)$ by construction of $\hat{w}_i$. If also $z \in \mathrm{vars}(B)$, then $z \in V_{AB} \cap V_{BC}$ which implies $\hat{v}_i(z) = \hat{w}_i(z) = \hat{w}_i{\restriction}_{V_{AB} \cap V_{BC}}(z) = \hat{u}_i{\restriction}_{V_{AB} \cap V_{BC}}(z) = \hat{u}_i(z)$ by construction of $\hat{u}_i$ and thus $\hat{v}_i(z) = \hat{u}_i(z) = \hat{u}_{i+1}(z) = \hat{v}_{i+1}(z)$ as $z \notin \mathrm{vars}(C)$ and (2.25) is a path in C. If on the other hand $z \notin \mathrm{vars}(B)$ then $\hat{w}_i(z) = z^\circ$ for all i as (2.23) is a path in B, and thus $\hat{v}_i(z) = \hat{w}_i(z) = z^\circ = \hat{w}_{i+1}(z) = \hat{v}_{i+1}(z)$.

This completes the proof that (2.26) is a path in C in all cases. □

Lemma 6 Let A, B, C, and E be EFSMs such that E is pure. Then the following properties hold.

(i) $A \parallel E \subseteq_{\mathrm{v}} A$.

(ii) If $A \subseteq_{\mathrm{v}} B$ then $A \parallel E \subseteq_{\mathrm{v}} B \parallel E$.

(iii) If $A \parallel B \subseteq_{\mathrm{v}} A \parallel C$ then $A \parallel B \subseteq_{\mathrm{v}} A \parallel B \parallel C$.

Proof.

(i) Let $V = \mathrm{vars}(A) \cup \mathrm{vars}(E)$, and assume a path

$$(a_0, e_0, \hat{v}_0) \xrightarrow{\sigma_1} (a_1, e_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, e_n, \hat{v}_n) \tag{2.27}$$

in $A \parallel E$ where $\hat{v}_i \in \mathrm{dom}(V)$ for $i = 0, \ldots, n$. By Lemma 3,

$$(a_0, \hat{v}_0) \xrightarrow{\sigma_1} (a_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, \hat{v}_n) \tag{2.28}$$

is a path in A, which is enough to show the claim.

(ii) Let $V = \mathrm{vars}(A) \cup \mathrm{vars}(B) \cup \mathrm{vars}(E)$, and assume a path

$$(a_0, e_0, \hat{v}_0) \xrightarrow{\sigma_1} (a_1, e_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, e_n, \hat{v}_n) \tag{2.29}$$

in $A \parallel E$ where $\hat{v}_i \in \mathrm{dom}(V)$ for $i = 0, \ldots, n$. By Lemma 3,

$$(a_0, \hat{v}_0) \xrightarrow{\sigma_1} (a_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, \hat{v}_n) \tag{2.30}$$

is a path in A. Let $V_{AB} = \mathrm{vars}(A) \cup \mathrm{vars}(B)$. Then

$$(a_0, \hat{v}_0{\restriction}_{V_{AB}}) \xrightarrow{\sigma_1} (a_1, \hat{v}_1{\restriction}_{V_{AB}}) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, \hat{v}_n{\restriction}_{V_{AB}}) \tag{2.31}$$

also is a path in A. As $A \subseteq_{\mathrm{v}} B$, by Definition 6 there exists a path

$$(b_0, \hat{v}_0{\restriction}_{V_{AB}}) \xrightarrow{\sigma_1} (b_1\hat{v}_1{\restriction}_{V_{AB}}) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (b_n, \hat{v}_n{\restriction}_{V_{AB}}) \tag{2.32}$$

in B. Now consider i such that $0 \leq i < n$. From (2.32), it follows that $b_i \xrightarrow{\sigma_{i+1}:p_B} b_{i+1}$ in B with $(\hat{v}_i{\restriction}_{V_{AB}} \oplus (\hat{v}_{i+1}{\restriction}_{V_{AB}})')(p_B) = \mathrm{true}$ and

$$\hat{v}_i{\restriction}_{V_{AB}}(z) = \hat{v}_{i+1}{\restriction}_{V_{AB}}(z) \quad \text{for all } z \in V_{AB} \setminus \mathrm{vars}'(p_B)\ . \tag{2.33}$$

From (2.29) it follows that $e_i \xrightarrow{\sigma_{i+1}:p_E} e_{i+1}$ in E with $(\hat{v}_n \oplus (\hat{v}_{i+1})')(p_E) = \mathrm{true}$. Now consider $z \in V \setminus (\mathrm{vars}'(p_B) \cup \mathrm{vars}'(p_E)) = (V_{AB} \cup \mathrm{vars}(E)) \setminus \mathrm{vars}'(p_B)$ as E is pure. Then either $z \in V_{AB}$ or $z \notin V_{AB}$. If $z \in V_{AB}$, then $z \in V_{AB} \setminus \mathrm{vars}'(p_B)$ and $\hat{v}_i(z) = \hat{v}_i{\restriction}_{V_{AB}}(z) = \hat{v}_{i+1}{\restriction}_{V_{AB}}(z) = \hat{v}_{i+1}(z)$ by (2.33). If $z \notin V_{AB}$ then $z \in \mathrm{vars}(E)$ and $z \notin V_{AB} \supseteq \mathrm{vars}(A) \supseteq \mathrm{vars}'(A) \cup \mathrm{vars}'(E)$ as E is pure, so that $z \in V \setminus (\mathrm{vars}'(A) \cup \mathrm{vars}'(E))$, and $\hat{v}_i(z) = \hat{v}_{i+1}(z)$ follows as variables that do not appear primed in A or E must remain unchanged on the path (2.29) in $A \parallel E$. Thus, $\hat{v}_i(z) = \hat{v}_{i+1}(z)$ for all $z \in V \setminus (\mathrm{vars}'(p_B) \cup \mathrm{vars}'(p_E))$, and noting that b_0, e_0, and $\hat{v}_0$ are initial locations and variable values, it follows that

$$(b_0, e_0, \hat{v}_0) \xrightarrow{\sigma_1} (b_1, e_1\hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (b_n, e_n, \hat{v}_n) \tag{2.34}$$

is a path in $B \parallel E$.

(iii) Let $V = \mathrm{vars}(A) \cup \mathrm{vars}(B) \cup \mathrm{vars}(C)$, and assume a path

$$(a_0, b_0, \hat{v}_0) \xrightarrow{\sigma_1} (a_1, b_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, b_n, \hat{v}_n) \tag{2.35}$$

in $A \parallel B$ where $\hat{v}_i \in \mathrm{dom}(V)$ for $i = 0, \ldots, n$. As $A \parallel B \subseteq_{\mathrm{v}} A \parallel C$, by Definition 6 there is a path

$$(\tilde{a}_0, c_0, \hat{v}_0) \xrightarrow{\sigma_1} (\tilde{a}_1, c_1\hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (\tilde{a}_n, c_n, \hat{v}_n) \tag{2.36}$$

in $A \parallel C$. It will be shown that

$$(a_0, b_0, c_0, \hat{v}_0) \xrightarrow{\sigma_1} (a_1, b_1, c_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (a_n, b_n, c_n, \hat{v}_n) \tag{2.37}$$

is a path in $A \parallel B \parallel C$. Clearly, a_0, b_0, c_0, and $\hat{v}_0$ are initial locations and variable values from (2.35) and (2.36). Considering the i-th transition on the path (2.35), by Lemma 2 there are transitions

$$a_{i-1} \xrightarrow{\sigma_i : p_A} a_i \qquad \text{in } A \qquad \text{with } (\hat{v}_{i-1} \oplus (\hat{v}_i)')(p_A) = \text{true} \; ; \tag{2.38}$$

$$b_{i-1} \xrightarrow{\sigma_i : p_B} b_i \qquad \text{in } B \qquad \text{with } (\hat{v}_{i-1} \oplus (\hat{v}_i)')(p_B) = \text{true} \; ; \tag{2.39}$$

such that

$$\hat{v}_{i-1}(z) = \hat{v}_i(z) \quad \text{for all } z \in V \setminus (\text{vars}'(p_A) \cup \text{vars}'(p_B)) \; . \tag{2.40}$$

Considering the i-th transition on the path (2.36), by Lemma 2 there is a transition

$$c_{i-1} \xrightarrow{\sigma_i : p_C} c_i \qquad \text{in } C \qquad \text{with } (\hat{v}_{i-1} \oplus (\hat{v}_i)')(p_C) = \text{true} \; . \tag{2.41}$$

For a variable $z \in V \setminus (\text{vars}'(p_A) \cup \text{vars}'(p_B) \cup \text{vars}'(p_C)) \subseteq V \setminus (\text{vars}'(p_A) \cup \text{vars}'(p_B))$ it follows from (2.40) that $\hat{v}_{i-1}(z) = \hat{v}_i(z)$. Then it follows by Lemma 2 using (2.38), (2.39), and (2.41) that $(a_{i-1}, b_{i-1}, c_{i-1}, \hat{v}_{i-1}) \xrightarrow{\sigma_i} (a_i, b_i, c_i, \hat{v}_i)$ in $A \parallel B \parallel C$, and the path (2.37) is obtained by repeating this argument for all i. □

Lemma 7 Let $F = \langle \Sigma, Q, Q^\circ, \rightarrow \rangle$ be an EFSM, and let $X \subseteq Q \times \text{dom}(\text{vars}(F))$. Then $F \restriction X \subseteq_\text{v} F$.

Proof. Write $V = \text{vars}(F) = \text{vars}(F \restriction X)$. Assume that

$$(x_0, \hat{v}_0) \xrightarrow{\sigma_1} (x_1, \hat{v}_1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_n, \hat{v}_n) \tag{2.42}$$

is a path in $F \restriction X$, where $\hat{v}_i \in \text{dom}(V)$. Then x_0 is an initial location of $F \restriction X$ and also of F by Definition 4, and $\hat{v}_0$ are initial variable values. Furthermore, for each $i = 0, \ldots, n-1$, there exists a transition $x_i \xrightarrow{\sigma_{i+1} : p_{i+1}} x_{i+1}$ in $F \restriction X$ such that $(\hat{v}_i \oplus \hat{v}'_{i+1})(p_{i+1}) = \text{true}$ and $\hat{v}_i(z) = \hat{v}_{i+1}(z)$ for all variables $z \in V \setminus \text{vars}'(p_{i+1})$. By Definition 4, the update of this transition can be written as $p_{i+1} \equiv q_{i+1} \wedge R_{i+1}$ where $x_i \xrightarrow{\sigma_{i+1} : q_{i+1}} x_{i+1}$ in F and $\text{vars}'(R_{i+1}) \subseteq \text{vars}'(q_{i+1})$. Then clearly $(\hat{v}_i \oplus \hat{v}'_{i+1})(q_{i+1}) = \text{true}$, and $\text{vars}'(p_{i+1}) = \text{vars}'(q_{i+1}) \cup \text{vars}'(R_{i+1}) = \text{vars}'(q_{i+1})$, which implies for $z \in V \setminus \text{vars}'(q_{i+1}) = V \setminus \text{vars}'(p_{i+1})$ that $\hat{v}_i(z) = \hat{v}_{i+1}(z)$ from above. This shows $(x_i, \hat{v}_i) \xrightarrow{\sigma_{i+1}} (x_{i+1}, \hat{v}_{i+1})$ in F, and repeating the argument for all transitions shows that (2.42) is a path in F. □

2.5 Controllability and Synthesis

For the supervisory control of EFSM systems, this working paper assumes that all variables are controlled by the plant [10]. The plant is modelled by a set of normalised EFSMs that represent the possible system behaviour including all possible variable changes. The specification is modelled by one or more pure EFSMs, which only restrict the occurrence of events.

The supervisor can also restrict variable changes associated with controllable events. The following definition of controllability covers specifications and supervisors.

Definition 7 [10] Let $G = \langle \Sigma_G, Q_G, Q_G^\circ, \rightarrow_G \rangle$ and $E = \langle \Sigma_E, Q_E, Q_E^\circ, \rightarrow_E \rangle$ be two EFSMs, and let Σ_u be a set of events. E is Σ_u-*controllable* with respect to G, if for all valuations $\hat{v}, \hat{w} \in \mathrm{dom}(\mathrm{vars}(G) \cup \mathrm{vars}(E))$, all states $(x_G, x_E, \hat{v}) \in Q^{\mathrm{acc}}(G \parallel E)$, and all transitions $(x_G, \hat{v}) \xrightarrow{\mu} (y_G, \hat{w})$ in G such that $\mu \in \Sigma_\mathrm{u}$, there exists a location y_E of E such that $(x_G, x_E, \hat{v}) \xrightarrow{\mu} (y_G, y_E, \hat{w})$ in $G \parallel E$.

Σ_u-controllability means that, from any accessible state in the synchronous composition of the plant G and specification E, if an *uncontrollable* event $\mu \in \Sigma_\mathrm{u}$ is eligible in the plant, then it is also eligible in the specification. In addition, the specification must allow any assignment to next-state variables prescribed by the plant. The condition $(x_G, x_E, \hat{v}) \xrightarrow{\mu} (y_G, y_E, \hat{w})$ is applied to the synchronous composition $G \parallel E$, so it requires the plant and specification to be able to take the transition together. This allows a pure specification to follow the plant's assignments to next-state variables.

In the case that an uncontrollable event is not mentioned in the plant, $\mu \notin \Sigma_G$, based on the extended definition of the transition relation, the transition is always possible in the plant and does not change variables. In order to be controllable, the specification must always enable μ without changing any variables on its occurrence.

Remark 1 Given a plant G and pure specification E, it can be assumed without loss of generality that $\mathrm{vars}(E) \subseteq \mathrm{vars}(G)$. This is because any variable z that appears only in E and not in G, cannot appear as next-state variable in G or E as $\mathrm{vars}'(E) = \emptyset$. This variable z remains unchanged on all transitions of the synchronous composition $G \parallel E$, so all its occurrences can be replaced by a constant representing its initial value $\hat{z}^\circ$, resulting in an EFSM system with equivalent behaviour.

If a specification is not controllable, *synthesis* is used to find a *supervisor*. Unlike the specification, the supervisor may include next-state variables on its updates. Thus, the supervisor can disable (controllable) events completely or under certain circumstances, and it can remove some of the plant's variable assignments from a controllable transition.

Definition 8 Let G and E be two EFSMs, and let Σ_u be a set of events. A *supremal supervisor* for E with respect to G and Σ_u is an EFSM S such that

(i) $G \parallel S \subseteq_\mathrm{v} G \parallel E$;

(ii) S is Σ_u-controllable with respect to G;

(iii) For any EFSM S' that satisfies (i) and (ii), it holds that $G \parallel S' \subseteq_\mathrm{v} G \parallel S$.

Definition 8 characterises the possible synthesis results for a plant G and specification E. A correct supervisor must satisfy the specification through behavioural inclusion after composition with the plant (i), and it must be controllable (ii). It also must be *least restrictive* or *supremal*, i.e., any other supervisor that controllably satisfies the specification has less possible behaviour, again in composition with the plant (iii).

A supervisor satisfying these three conditions can be computed by means of a standard fixpoint iteration on the unfolded state set of $G \parallel E$, using the following operator.

Definition 9 [9] Let $G = \langle \Sigma_G, Q_G, Q_G^\circ, \rightarrow_G \rangle$ and $E = \langle \Sigma_E, Q_E, Q_E^\circ, \rightarrow_E \rangle$ be two EFSMs, let $V = \mathrm{vars}(G) \cup \mathrm{vars}(E)$, and let Σ_u be a set of events. The *extended synthesis step* operator $\Theta_{G,E,\Sigma_\mathrm{u}} \colon 2^{Q_G \times Q_E \times \mathrm{dom}(V)} \to 2^{Q_G \times Q_E \times \mathrm{dom}(V)}$ with respect to G, E, and Σ_u is defined as

$$\begin{aligned}\Theta_{G,E,\Sigma_\mathrm{u}}(X) = \{\, &(x_G, x_E, \hat{v}) \in Q_G \times Q_E \times \mathrm{dom}(V) \mid \text{if } (x_G, \hat{v}) \xrightarrow{\mu} (y_G, \hat{w}) \text{ for some} \\ &\mu \in \Sigma_\mathrm{u} \text{ and } \hat{w} \in \mathrm{dom}(V), \text{ then there exists } y_E \in Q_E \text{ such that} \\ &(x_G, x_E, \hat{v}) \xrightarrow{\mu} (y_G, y_E, \hat{w}) \in X \,\} \,.\end{aligned} \tag{2.43}$$

For a set X of combinations of locations and variable assignments, the operator $\Theta_{G,E,\Sigma_\mathrm{u}}$ removes from X any uncontrollable states, i.e., states where the plant enables some uncontrollable transition not enabled by the specification, and any states from where the system could uncontrollably reach some combination of location and valuation not contained in X. The operator $\Theta_{G,E,\Sigma_\mathrm{u}}$ is monotonic and has a greatest fixpoint $\hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$ [21]. In the finite-state case, this fixpoint is calculated as the limit of the iteration

$$X^0 = Q_Q \times Q_E \times \mathrm{dom}(V) \,; \tag{2.44}$$

$$X^{j+1} = \Theta_{G,E,\Sigma_\mathrm{u}}(X^j) \,. \tag{2.45}$$

The result of EFSM synthesis is then obtained by restricting the system to this fixpoint.

Definition 10 [9] Let G and E be two EFSMs, and let Σ_u be a set of events. The *supremal* Σ_u*-controllable sub-EFSM* of G and E is

$$\sup\mathcal{C}(G, E, \Sigma_\mathrm{u}) = (G \parallel E) \restriction \hat{\Theta}_{G,E,\Sigma_\mathrm{u}} \,, \tag{2.46}$$

where $\hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$ is the greatest fixpoint of the operator $\Theta_{G,E,\Sigma_\mathrm{u}}$ from Definition 9.

It remains to be shown that this fixpoint indeed gives a correct synthesis result according to Definition 8. Care must be taken, as the definition compares the supervisors after composition with the plant, and this requires state-determinism and normalisation to ensure synchrony of the states. The following three lemmas establish preliminary results about state synchrony between plant and supervisor under these assumptions. Based on that, Proposition 11 confirms that the sup$\mathcal{C}$ operation gives a correct synthesis result.

Lemma 8 Let G and E be two EFSMs, such that G is state-deterministic, and let Σ_u be a set of events. For every state $(x_G, (y_G, x_E), \hat{v}) \in Q^\mathrm{acc}(G \parallel \sup\mathcal{C}(G, E, \Sigma_\mathrm{u}))$, it holds that $x_G = y_G$.

Proof. Write $S = \sup\mathcal{C}(G, E, \Sigma_\mathrm{u})$, and let $(x_G, (y_G, x_E), \hat{v}) \in Q^\mathrm{acc}(G \parallel S)$. Then there exists a path

$$(x_G^0, (y_G^0, x_E^0), \hat{v}^0) \xrightarrow{\sigma_1} (x_G^1, (y_G^1, x_E^1), \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, (y_G^n, x_E^n), \hat{v}^n) = (x_G, (y_G, x_E), \hat{v}) \quad (2.47)$$

in $G \parallel S$. It is shown by induction on n that $x_G^n = y_G^n$.

For the base case, $n = 0$, note that x_G^0 and y_G^0 are initial locations of G, and thus $x_G^0 = y_G^0$ by Definition 2 (iii). Now assume $x_G^n = y_G^n$ for some $n \geq 0$, and consider the next transition

$$(x_G^n, (y_G^n, x_E^n), \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, (y_G^{n+1}, x_E^{n+1}), \hat{v}^{n+1}) \quad (2.48)$$

on the path (2.47). This means by Lemma 2 that there are transitions

$$x_G^n \xrightarrow{\sigma_{n+1}:p_G} x_G^{n+1} \qquad \text{in } G \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_G) = \text{true} ; \quad (2.49)$$

$$(y_G^n, x_E^n) \xrightarrow{\sigma_{n+1}:p_S} (y_G^{n+1}, x_E^{n+1}) \qquad \text{in } S \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_S) = \text{true} . \quad (2.50)$$

Recalling that $S = \sup\mathcal{C}(G, E, \Sigma_\mathrm{u})$, by Definitions 4 and 10 and by Lemma 5 the update in (2.65) has the form $p_S \equiv q_G \wedge q_E \wedge R$, and there is a transition

$$y_G^n \xrightarrow{\sigma_{n+1}:q_G} y_G^{n+1} \qquad \text{in } G \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(q_G) = \text{true} . \quad (2.51)$$

Then $p_G \wedge q_G$ is satisfiable, and noting that $x_G^n = y_G^n$ by inductive assumption, it follows by the state-determinism of G from Definition 2 (iii) and from (2.49) and (2.51) that $x_G^{n+1} = y_G^{n+1}$. □

Lemma 9 Let G and E be two EFSMs such that G is normalised, and let Σ_u be a set of events. For every state $(x_G, x_S, \hat{v}) \in Q^\mathrm{acc}(G \parallel \sup\mathcal{C}(G, E, \Sigma_\mathrm{u}))$, it holds that $(x_S, \hat{v}) \in Q^\mathrm{acc}(\sup\mathcal{C}(G, E, \Sigma_\mathrm{u}))$.

Proof. Write $S = \sup\mathcal{C}(G, E, \Sigma_\mathrm{u})$ and $V = \mathrm{vars}(S) = \mathrm{vars}(G) \cup \mathrm{vars}(S)$, and let $(x_G, x_S, \hat{v}) \in Q^\mathrm{acc}(G \parallel S)$. Then $\hat{v} \in \mathrm{dom}(V)$ and there exists a path

$$(x_G^0, x_S^0, \hat{v}^0) \xrightarrow{\sigma_1} (x_G^1, x_S^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, x_S^n, \hat{v}^n) = (x_G, x_S, \hat{v}) \quad (2.52)$$

in $G \parallel S$. It is shown by induction on n that $(x_S^n, \hat{v}^n) \in Q^\mathrm{acc}(S)$.

For the base case, $n = 0$, note that x_S^0 is an initial location of S and $\hat{v}^0$ are initial variable values, and thus $(x_S^0, \hat{v}^0) \in Q^\mathrm{acc}(S)$.

Now assume $(x_S^n, \hat{v}^n) \in Q^{\mathrm{acc}}(S)$ for some $n \geq 0$, and consider the next transition of (2.52),

$$(x_G^n, x_S^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_S^{n+1}, \hat{v}^{n+1}) \ . \tag{2.53}$$

By Lemma 5, this means that there are transitions

$$x_G^n \xrightarrow{\sigma_{n+1}:p_G} x_G^{n+1} \qquad \text{in } G \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_G) = \text{true} \ ; \tag{2.54}$$

$$x_S^n \xrightarrow{\sigma_{n+1}:p_S} x_S^{n+1} \qquad \text{in } S \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_S) = \text{true} \ ; \tag{2.55}$$

and furthermore

$$\hat{v}^n(z) = \hat{v}^{n+1}(z) \quad \text{for all } z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S)) \ . \tag{2.56}$$

Noting that $S = \sup\mathcal{C}(G, E, \Sigma_{\mathrm{u}})$, by Definitions 4 and 5, the update in (2.55) takes the form $p_S \equiv q_G \wedge q_E \wedge R$ for some transition $y_G^n \xrightarrow{\sigma_{n+1}:q_G} y_G^{n+1}$ of G. Note that $\mathrm{vars}'(q_G) = \mathrm{vars}'(G, \sigma_{n+1}) = \mathrm{vars}'(p_G)$ as G is normalised. It follows for $z \in V \setminus \mathrm{vars}'(p_S) = V \setminus (\mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_E) \cup \mathrm{vars}'(R)) = V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_E) \cup \mathrm{vars}'(R)) = V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S))$ that $\hat{v}^n(z) = \hat{v}^{n+1}(z)$ by (2.56). Using (2.55), this shows $(x_S^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_S^{n+1}, \hat{v}^{n+1})$ in S, and the claim $(x_S^{n+1}, \hat{v}^{n+1}) \in Q^{\mathrm{acc}}(S)$ follows by inductive assumption. □

Lemma 10 Let G and E be two EFSMs, and let Σ_{u} be a set of events. Then $\sup\mathcal{C}(G, E, \Sigma_{\mathrm{u}}) \subseteq_{\mathrm{v}} G \parallel \sup\mathcal{C}(G, E, \Sigma_{\mathrm{u}})$.

Proof. Write $S = \sup\mathcal{C}(G, E, \Sigma_{\mathrm{u}})$ and $V = \mathrm{vars}(S) = \mathrm{vars}(G) \cup \mathrm{vars}(E)$. Consider a path

$$(x_G^0, x_E^0, \hat{v}^0) \xrightarrow{\sigma_1} (x_G^1, x_E^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, x_E^n, \hat{v}^n) \tag{2.57}$$

in S, where $\hat{v}^i \in \mathrm{dom}(V)$ for $i = 0, \ldots, n$. It will be shown by induction on n that

$$(x_G^0, (x_G^0, x_E^0), \hat{v}^0) \xrightarrow{\sigma_1} (x_G^1, (x_G^1, x_E^1), \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, (x_G^n, x_E^n), \hat{v}^n) \tag{2.58}$$

is a path in $G \parallel S$.

For the base case, $n = 0$, note that (x_G^0, x_E^0) and thus also x_G^0 are initial locations, and $\hat{v}^0$ are initial variable values from (2.57).

Now assume the path (2.58) in $G \parallel S$ has been constructed up to n, and consider the next transition of (2.57),

$$(x_G^n, x_E^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_E^{n+1}, \hat{v}^{n+1}) \ . \tag{2.59}$$

This means that there is a transition $(x_G^n, x_E^n) \xrightarrow{\sigma_{n+1}:p_S} (x_G^{n+1}, x_E^{n+1})$ in S such that $(\hat{v}^n \oplus (\hat{v}^{n+1})')(p_S) = \text{true}$ and

$$\hat{v}^n(z) = \hat{v}^{n+1}(z) \quad \text{for all variables } z \in V \setminus \mathrm{vars}'(p_S) \ . \tag{2.60}$$

As $S = \sup\mathcal{C}(G, E, \Sigma_u) = (G \parallel E) \upharpoonright \hat{\Theta}_{G,E,\Sigma_u}$, by Definition 4 the update p_S takes the form $p_S \equiv q_G \wedge q_E \wedge R$, and there is a transition $x_G^n \xrightarrow{\sigma_{n+1}:q_G} x_G^{n+1}$ in G with $(\hat{v}^n \oplus (\hat{v}^{n+1})')(q_G) = \text{true}$. For variables $z \in V \setminus \text{vars}'(q_G \wedge p_S) = V \setminus (\text{vars}'(q_G) \cup \text{vars}'(q_G) \cup \text{vars}'(q_E) \cup \text{vars}'(R)) = V \setminus \text{vars}'(p_S)$ it holds by (2.60) that $\hat{v}^n(z) = \hat{v}^{n+1}(z)$. It follows from Lemma 2 that

$$(x_G^n, (x_G^n, x_E^n), \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, (x_G^{n+1}, x_E^{n+1}), \hat{v}^{n+1}) \tag{2.61}$$

in $G \parallel S$, and the existence of the path (2.58) follows by inductive assumption. □

Proposition 11 Let G and E be state-deterministic EFSMs such that G is normalised, and let Σ_u be a set of events. Then $\sup\mathcal{C}(G, E, \Sigma_u)$ is a supremal supervisor for E with respect to G and Σ_u.

Proof. Write $S = \sup\mathcal{C}(G, E, \Sigma_u)$ and $V = \text{vars}(S) = \text{vars}(G) \cup \text{vars}(E)$. It is to be shown that S satisfies conditions (i)–(iii) in Definition 8.

(i) Let

$$(x_G^0, (y_G^0, x_E^0), \hat{v}^0) \xrightarrow{\sigma_1} (x_G^1, (y_G^1, x_E^1), \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, (y_G^n, x_E^n), \hat{v}^n) \tag{2.62}$$

be a path in $G \parallel S = G \parallel \sup\mathcal{C}(G, E, \Sigma_u)$ with $\hat{v}_i \in \text{dom}(\text{vars}(G \parallel S)) = \text{dom}(V)$ for $i = 0, \ldots, n$. Clearly x_G^0, y_G^0, x_E^0, and $\hat{v}^0$ are initial locations and variable values. Consider the i-th transition on the path (2.62),

$$(x_G^i, (y_G^i, x_E^i), \hat{v}^i) \xrightarrow{\sigma_{i+1}} (x_G^{i+1}, (y_G^{i+1}, x_E^{i+1}), \hat{v}^{i+1}) \ . \tag{2.63}$$

By Lemma 2, this means that there are transitions

$$x_G^i \xrightarrow{\sigma_{i+1}:p_G} x_G^{i+1} \qquad \text{in } G \text{ with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(p_G) = \text{true} \ ; \tag{2.64}$$

$$(y_G^i, x_E^i) \xrightarrow{\sigma_{i+1}:p_S} (y_G^{i+1}, x_E^{i+1}) \qquad \text{in } S \text{ with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(p_S) = \text{true} \ ; \tag{2.65}$$

and furthermore

$$\hat{v}^i(z) = \hat{v}^{i+1}(z) \quad \text{for all } z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_S)) \ . \tag{2.66}$$

Recalling that $S = \sup\mathcal{C}(G, E, \Sigma_u)$, by Definitions 4 and 10 and by Lemma 2 the update in (2.65) has the form $p_S \equiv q_G \wedge q_E \wedge R$ with $\text{vars}'(R) \subseteq \text{vars}'(q_G \wedge q_E)$, and there are transitions

$$y_G^i \xrightarrow{\sigma_{i+1}:q_G} y_G^{i+1} \qquad \text{in } G \text{ with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(q_G) = \text{true} \ ; \tag{2.67}$$

$$x_E^i \xrightarrow{\sigma_{i+1}:q_E} x_E^{i+1} \qquad \text{in } E \text{ with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(q_E) = \text{true} \ . \tag{2.68}$$

Consider $z \in V \setminus (\mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_E)) = V \setminus \mathrm{vars}'(q_G \wedge q_E) = V \setminus \mathrm{vars}'(p_S)$. Noting that $z \notin \mathrm{vars}'(q_G) = \mathrm{vars}'(G, \sigma_{i+1}) = \mathrm{vars}'(p_G)$ as G is normalised, it follows that $z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S))$ and thus $\hat{v}^i(z) = \hat{v}^{i+1}(z)$ by (2.66). Thus $(y_G^i, x_E^i, \hat{v}^i) \xrightarrow{\sigma_{i+1}} (y_G^{i+1}, x_E^{i+1}, \hat{v}^{i+1})$ in $G \parallel E$ by Lemma 2. Repeating this argument for all transitions on the path (2.62), it follows that

$$(y_G^0, x_E^0, \hat{v}^0) \xrightarrow{\sigma_1} (y_G^1, x_E^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (y_G^n, x_E^n, \hat{v}^n) \tag{2.69}$$

is a path in $G \parallel E$. This shows $G \parallel S \subseteq_{\mathrm{v}} G \parallel E$ according to Definition 6.

(ii) Let $\hat{v}, \hat{w} \in \mathrm{dom}(V)$, let $(x_G, (\tilde{x}_G, x_E), \hat{v}) \in Q^{\mathrm{acc}}(G \parallel S)$, let $\mu \in \Sigma_{\mathrm{u}}$, and let $(x_G, \hat{v}) \xrightarrow{\mu} (y_G, \hat{w})$ in G. Following Definition 7, it is to be shown that there exists a location y_S of S such that $(x_G, (\tilde{x}_G, x_E), \hat{v}) \xrightarrow{\mu} (y_G, y_S, \hat{w})$ in $G \parallel S$.

Note that $x_G = \tilde{x}_G$ by Lemma 8 and $(\tilde{x}_G, x_E, \hat{v}) \in Q^{\mathrm{acc}}(S)$ by Lemma 9. Using Definition 10, Lemma 1, and the fact that $\hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}}$ is a fixpoint of $\Theta_{G,E,\Sigma_{\mathrm{u}}}$, it follows that $(x_G, x_E, \hat{v}) = (\tilde{x}_G, x_E, \hat{v}) \in Q^{\mathrm{acc}}(S) = Q^{\mathrm{acc}}((G \parallel E) \upharpoonright \hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}}) \subseteq \hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}} = \Theta_{G,E,\Sigma_{\mathrm{u}}}(\hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}})$. Then by Definition 9 there exists a location y_E of E such that $(x_G, x_E, \hat{v}) \xrightarrow{\mu} (y_G, y_E, \hat{w})$ in $G \parallel E$ and $(y_G, y_E, \hat{w}) \in \hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}}$.

This means $(x_G, x_E) \xrightarrow{\mu:p_S} (y_G, y_E)$ in $G \parallel E$ such that $(\hat{v} \oplus \hat{w}')(p_S) = \mathrm{true}$ and

$$\hat{v}(z) = \hat{w}(z) \quad \text{for all } z \in V \setminus \mathrm{vars}'(p_S)\ . \tag{2.70}$$

By Definition 5, the update p_S has the form $p_S \equiv p_G \wedge p_E$ with $x_G \xrightarrow{\mu:p_G} y_G$ in G and $x_E \xrightarrow{\mu:p_E} y_E$ in E. Then also $(\hat{v} \oplus \hat{w}')(p_G) = \mathrm{true}$ as $(\hat{v} \oplus \hat{w}')(p_G \wedge p_E) = (\hat{v} \oplus \hat{w}')(p_S) = \mathrm{true}$. That is,

$$x_G \xrightarrow{\mu:p_G} y_G \quad \text{in } G \qquad \text{with } (\hat{v} \oplus \hat{w}')(p_G) = \mathrm{true}\ . \tag{2.71}$$

Furthermore, it follows according to Definitions 4 and 10 that $(x_G, x_E) \xrightarrow{\mu:p_S \wedge R} (y_G, y_E)$ in $S = \sup\mathcal{C}(G, E) = (G \parallel E) \upharpoonright \hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}}$ where $R \equiv R_{\hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}}}[p_S, (y_G, y_E)]$ is such that $\mathrm{vars}'(R) \subseteq \mathrm{vars}'(p_S)$ and for all valuations $\tilde{v}, \tilde{w} \in \mathrm{dom}(V)$ it holds that $(y_G, y_E, \tilde{v}{\restriction}_{V \setminus \mathrm{vars}'(p_S)} \oplus \tilde{w}) \in \hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}}$ if and only if $(\tilde{v} \oplus \tilde{w}')(R) = \mathrm{true}$. Note that $\hat{v}{\restriction}_{V \setminus \mathrm{vars}'(p_S)} = \hat{w}{\restriction}_{V \setminus \mathrm{vars}'(p_S)}$ by (2.70), so that $(y_G, y_E, \hat{v}{\restriction}_{V \setminus \mathrm{vars}'(p_S)} \oplus \hat{w}) = (y_G, y_E, \hat{w}) \in \hat{\Theta}_{G,E,\Sigma_{\mathrm{u}}}$ and thus $(\hat{v} \oplus \hat{w}')(R) = \mathrm{true}$. Recalling that also $(\hat{v} \oplus \hat{w}')(p_S) = \mathrm{true}$, it follows that

$$(x_G, x_E) \xrightarrow{\mu:p_S \wedge R} (y_G, y_E) \quad \text{in } S \qquad \text{with } (\hat{v} \oplus \hat{w}')(p_S \wedge R) = \mathrm{true}\ . \tag{2.72}$$

Lastly, for a variable $z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S \wedge R)) = V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_E) \cup \mathrm{vars}'(R)) = V \setminus (\mathrm{vars}'(p_S) \cup \mathrm{vars}'(R)) = V \setminus \mathrm{vars}'(p_S)$, it is clear that $\hat{v}(z) = \hat{w}(z)$ from (2.70). Then it follows from (2.71) and (2.72) by Lemma 2 that $(x_G, (\tilde{x}_G, x_E), \hat{v}) = (x_G, (x_G, x_E), \hat{v}) \xrightarrow{\mu} (y_G, (y_G, y_E), \hat{w})$ in $G \parallel S$, which implies the claim with $y_S = (y_G, y_E)$.

(iii) Let S' be an EFSM that satisfies (i) and (ii). It is to be shown that $G \parallel S' \subseteq_{\mathrm{v}} G \parallel S$. As S' satisfies $G \parallel S' \subseteq_{\mathrm{v}} G \parallel E$ (i), by Definition 6 for every path

$$(y_G^0, x_S^0, \hat{w}^0) \xrightarrow{\sigma_1} (y_G^1, x_S^1, \hat{w}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (y_G^n, x_S^n, \hat{w}^n) \tag{2.73}$$

in $G \parallel S'$, with $\hat{w}^i \in \mathrm{dom}(W)$ and $W = \mathrm{vars}(G) \cup \mathrm{vars}(S') \cup \mathrm{vars}(E) = V \cup \mathrm{vars}(S')$, there exists a path

$$(x_G^0, x_E^0, \hat{w}^0) \xrightarrow{\sigma_1} (x_G^1, x_E^1, \hat{w}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, x_E^n, \hat{w}^n) \tag{2.74}$$

in $G \parallel E$. Note that $x_G^i = y_G^i$ for $i = 0, \ldots, n$ by Lemma 4 as G is state-deterministic.

Let $X' \subseteq Q_G \times Q_E \times \mathrm{dom}(W)$ be the set of all end states $(x_G^n, x_E^n, \hat{w}^n)$ of paths (2.74) in $G \parallel E$ obtained from a corresponding path (2.73) in $G \parallel S'$. Further, let $X \subseteq Q_G \times Q_E \times \mathrm{dom}(V)$ be the restriction of this set to variables in V,

$$X = \{\, (x_G, x_E, \hat{w}{\restriction_V}) \mid (x_G, x_E, \hat{w}) \in X' \,\} \,. \tag{2.75}$$

It is next shown that $X \subseteq \Theta_{G,E,\Sigma_{\mathrm{u}}}(X)$, i.e., X is a *post-fixpoint* of $\Theta_{G,E,\Sigma_{\mathrm{u}}}$. Let $(x_G^n, x_E^n, \hat{v}^n) \in X$ and $(x_G^n, \hat{v}^n) \xrightarrow{\mu} (x_G^{n+1}, \hat{v}^{n+1})$ in G for some $\mu \in \Sigma_{\mathrm{u}}$, $x_G^{n+1} \in Q_G$, and $\hat{v}^{n+1} \in \mathrm{dom}(V)$. As $(x_G^n, x_E^n, \hat{v}^n) \in X$, there exists $\hat{w}^n \in \mathrm{dom}(W)$ with $\hat{v}^n = \hat{w}^n{\restriction_V}$ and corresponding paths (2.74) in $G \parallel E$ with end state $(x_G^n, x_E^n, \hat{v}^n)$ and (2.73) in $G \parallel S'$ with end state $(x_G^n, x_S^n, \hat{w}^n) = (y_G^n, x_S^n, \hat{w}^n)$. It follows that $(x_G^n, x_S^n, \hat{w}^n) \in Q^{\mathrm{acc}}(G \parallel S')$. Let $\hat{w}^{n+1} = \hat{v}^{n+1} \oplus \hat{w}^n$.

From $(x_G^n, \hat{v}^n) \xrightarrow{\mu} (x_G^{n+1}, \hat{v}^{n+1})$ it follows that there exists a transition $x_G^n \xrightarrow{\mu : p_G} x_G^{n+1}$ in G with $(\hat{v}^n \oplus (\hat{v}^{n+1})')(p_G) = \mathrm{true}$, thus also $(\hat{w}^n \oplus (\hat{w}^{n+1})')(p_G) = \mathrm{true}$, and $\hat{v}^n(z) = \hat{v}^{n+1}(z)$ for all variables $z \in V \setminus \mathrm{vars}'(p_G)$. For $z \in W \setminus V$ note that $\hat{w}^n(z) = (\hat{v}^{n+1} \oplus \hat{w}^n)(z) = \hat{w}^{n+1}(z)$, and thus $\hat{w}^n(z) = \hat{w}^{n+1}(z)$ for all variables $z \in W \setminus \mathrm{vars}'(p_G)$. It follows that $(x_G^n, \hat{w}^n) \xrightarrow{\mu} (x_G^{n+1}, \hat{w}^{n+1})$ in G, and this implies by the Σ_{u}-controllability of S' with respect to G (ii) that $(x_G^n, x_S^n, \hat{w}^n) \xrightarrow{\mu} (x_G^{n+1}, x_S^{n+1}, \hat{w}^{n+1})$ in $G \parallel S'$ for some location x_S^{n+1} of S'.

As $x_G^n = y_G^n$, this transition extends the path (2.73), so there exists a corresponding path (2.74) in $G \parallel E$,

$$(\tilde{x}_G^0, \tilde{x}_E^0, \hat{w}^0) \xrightarrow{\sigma_1} (\tilde{x}_G^1, \tilde{x}_E^1, \hat{w}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (\tilde{x}_G^n, \tilde{x}_E^n, \hat{w}^n) \xrightarrow{\mu} (\tilde{x}_G^{n+1}, \tilde{x}_E^{n+1}, \hat{w}^{n+1}) \,, \tag{2.76}$$

where $\tilde{x}_G^i = x_G^i$ for $i = 0, \ldots, n+1$ and $\tilde{x}_E^i = x_E^i$ for $i = 0, \ldots, n$ by Lemma 4. Therefore $(x_G^{n+1}, \tilde{x}_E^{n+1}, \hat{v}^{n+1}) = (\tilde{x}_G^{n+1}, \tilde{x}_E^{n+1}, \hat{w}^{n+1}{\restriction_V}) \in X$, and also

$$(x_G^n, x_E^n, \hat{v}^n) = (\tilde{x}_G^n, \tilde{x}_E^n, \hat{w}^n{\restriction_V}) \xrightarrow{\mu} (\tilde{x}_G^{n+1}, \tilde{x}_E^{n+1}, \hat{w}^{n+1}{\restriction_V}) = (x_G^{n+1}, \tilde{x}_E^{n+1}, \hat{v}^{n+1}) \tag{2.77}$$

in $G \parallel E$. It follows that $(x_G^n, x_E^n, \hat{v}^n) \in \Theta_{G,E,\Sigma_\mathrm{u}}(X)$ by Definition 9. As $(x_G^n, x_E^n, \hat{v}^n) \in X$ was chosen arbitrarily, this shows that $X \subseteq \Theta_{G,E,\Sigma_\mathrm{u}}(X)$. It follows from the Knaster-Tarski theorem [21] that $X \subseteq \hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$.

Now consider an arbitrary path (2.73) in $G \parallel S'$. As shown above, there exists a corresponding path (2.74) in $G \parallel E$ such that $(x_G^i, x_E^i, \hat{w}^i{\restriction}_V) \in X \subseteq \hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$ for $i = 0, \ldots, n$. Next, it will be shown by induction on n that (2.74) also is a path in S.

For the base case, $n = 0$, note that x_G^0, x_E^0, and $\hat{w}^0$ are initial locations and variable values, and as $(x_G^0, x_E^0, \hat{w}^0{\restriction}_V) \in \hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$, it follows that (x_G^0, x_E^0) is an initial location of $(G \parallel E) \restriction \hat{\Theta}_{G,E,\Sigma_\mathrm{u}} = S$.

Now assume the path (2.74) exists in S up to n, and consider the next transition of (2.74) in $G \parallel E$,

$$(x_G^n, x_E^n, \hat{w}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_E^{n+1}, \hat{w}^{n+1}) \ . \tag{2.78}$$

This means that there is a transition

$$(x_G^n, x_E^n) \xrightarrow{\sigma_{n+1}:p_S} (x_G^{n+1}, x_E^{n+1}) \tag{2.79}$$

in $G \parallel E$ such that

$$\hat{w}^n(z) = \hat{w}^{n+1}(z) \quad \text{for all variables } z \in W \setminus \mathrm{vars}'(p_S) \ . \tag{2.80}$$

By Definition 4, in $S = (G \parallel E) \restriction \hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$ there exists a transition

$$(x_G^n, x_E^n) \xrightarrow{\sigma_{n+1}:p_S \wedge R} (x_G^{n+1}, x_E^{n+1}) \tag{2.81}$$

where $R \equiv R_{\hat{\Theta}_{G,E,\Sigma_\mathrm{u}}}[p_S, (x_G^{n+1}, x_E^{n+1})]$ is such that $\mathrm{vars}'(R) \subseteq \mathrm{vars}'(p_S)$, and for all valuations $\hat{v}, \hat{w} \in \mathrm{dom}(V)$ it holds that $(x_G^{n+1}, x_E^{n+1}, \hat{v}{\restriction}_{V \setminus \mathrm{vars}'(p_S)} \oplus \hat{w}) \in \hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$ if and only if $(\hat{v} \oplus \hat{w}')(R) = \mathrm{true}$. Note $\hat{w}^{n+1}{\restriction}_V = (\hat{w}^n{\restriction}_{W \setminus \mathrm{vars}'(p_S)} \oplus \hat{w}^{n+1}){\restriction}_V = \hat{w}^n{\restriction}_{V \setminus \mathrm{vars}'(p_S)} \oplus \hat{w}^{n+1}{\restriction}_V$ by (2.80). Then $(x_G^{n+1}, x_E^{n+1}, \hat{w}^n{\restriction}_{V \setminus \mathrm{vars}'(p_S)} \oplus \hat{w}^{n+1}{\restriction}_V) = (x_G^{n+1}, x_E^{n+1}, \hat{w}^{n+1}{\restriction}_V) \in \hat{\Theta}_{G,E,\Sigma_\mathrm{u}}$ and thus $(\hat{w}^n \oplus (\hat{w}^{n+1})')(R) = \mathrm{true}$. Also for variables $z \in W \setminus \mathrm{vars}'(p_S \wedge R) = W \setminus \mathrm{vars}'(p_S)$, it holds that $\hat{w}^n(z) = \hat{w}^{n+1}(z)$ by (2.80). This shows

$$(x_G^n, x_E^n, \hat{w}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_E^{n+1}, \hat{w}^{n+1}) \tag{2.82}$$

in S. Thus, the path (2.74) can be constructed in S for an arbitrary path (2.73) in $G \parallel S'$, which shows $G \parallel S' \subseteq_\mathrm{v} S$.

Thus, $G \parallel S' \subseteq_\mathrm{v} S \subseteq_\mathrm{v} G \parallel S$ by Lemma 10, and the claim $G \parallel S' \subseteq_\mathrm{v} G \parallel S$ follows from Lemma 5 (ii). □

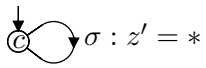

Figure 2: The EFSM chaos(σ, z).

2.6 Chaos Abstraction

This working paper is concerned with methods to modify or rewrite EFSMs or systems of composed EFSMs to simplify them and make synthesis procedures more efficient. Ordinary FSMs have useful modularity properties, according to which synchronous composition of a state machine with another only ever restricts the behaviour [2]. This makes it possible to remove components from a synchronous composition while preserving safety properties such as controllability, i.e., the controllability with respect to a part of the plant implies controllability with respect to the entire plant. EFSMs do not have this property, and therefore this and the following section introduce alternatives for the simplification of EFSM systems.

When EFSMs are combined in synchronous composition, new next-state variables can be added to transitions, possibly changing variables that were implicitly unchanged. To obtain modularity properties similar to those known for FSMs, one solution [10] is to replace the parts of the system not considered in a synthesis attempt by an *abstraction* that includes all possible variable changes. This abstraction is called *chaos EFSM*.

Definition 11 [10] Given an event σ and a variable z, the *chaos EFSM* for σ and z is

$$\text{chaos}(\sigma, z) = \langle \{\sigma\}, \{c\}, \{c\}, \{(c, \sigma, z' = *, c)\} \rangle \ . \tag{2.83}$$

The EFSM chaos(σ, z) is shown in Figure 2. The update $z' = *$ means that the variable z can assume any value from its domain in the next state. Formally, this update is true for all valuations, but it includes the next-state variable z' so that z is no longer implicitly unchanged.

In the synchronous composition $F_1 \parallel F_2$ of two EFSMs, some variables in F_1 may be changed by transitions in F_2. A variable z can be changed after composition of a transition in F_1 that does not mention z' with a transition in F_2 that mentions z'; or by a transition with an event that only appears in F_2. By inspection of the next-state variables on the transitions of F_2, it can be determined that certain variables are not changed in F_2, or are only changed on the occurrence of certain events. The following Lemma 12 shows how to identify the specific chaos EFSMs to capture possible variable changes in another EFSM.

Lemma 12 [10] Let F_1 and F_2 be two EFSMs, and let

$$C = \Big\| (\{\, \text{chaos}(\sigma, z) \mid z \in \text{vars}(F_1) \cap \text{vars}'(F_2, \sigma) \,\}) \ . \tag{2.84}$$

If $(x_1, x_2, \hat{v}) \xrightarrow{\sigma} (y_1, y_2, \hat{w})$ in $F_1 \parallel F_2$ then $(x_1, c, \hat{v}\restriction_{\text{vars}(F_1)}) \xrightarrow{\sigma} (y_1, c, \hat{w}\restriction_{\text{vars}(F_1)})$ in $F_1 \parallel C$, where c is the single location of C.

Proof. Let Σ_1, Σ_2, and Σ_C denote the event sets of F_1, F_2, and C, respectively, and write $V_1 = \mathrm{vars}(F_1)$. Note that $\mathrm{vars}(C) \subseteq V_1$ and thus $V_1 = \mathrm{vars}(F_1 \parallel C)$, and also $\mathrm{vars}'(F_2, \sigma) = \emptyset$ for $\sigma \notin \Sigma_2$ and thus $\Sigma_C \subseteq \Sigma_2$. Assume

$$(x_1, x_2, \hat{v}) \xrightarrow{\sigma} (y_1, y_2, \hat{w}) \tag{2.85}$$

in $F_1 \parallel F_2$ with $\hat{v}, \hat{w} \in \mathrm{dom}(W)$ for some $W \supseteq \mathrm{vars}(F_1) \cup \mathrm{vars}(F_2)$.

If $\sigma \notin \Sigma_1 \cup \Sigma_2$ then $x_1 = y_1$ and $x_2 = y_2$ and $\hat{v} = \hat{w}$, and from $\Sigma_C \subseteq \Sigma_2$ it follows that $\sigma \notin \Sigma_1 \cup \Sigma_C$. Then it is clear that $(x_1, c, \hat{v}) \xrightarrow{\sigma} (x_1, c, \hat{v}) = (y_1, c, \hat{w})$ in $F_1 \parallel C$.

Otherwise $F_1 \parallel F_2$ contains a transition

$$(x_1, x_2) \xrightarrow{\sigma:p} (y_1, y_2) \tag{2.86}$$

such that $(\hat{v} \oplus \hat{w}')(p) = \mathrm{true}$ and $\hat{v}(z) = \hat{w}(z)$ for all variables $z \in W \setminus \mathrm{vars}'(p)$. C contains a single σ-transition $c \xrightarrow{\sigma:p_C} c$, where p_C is the conjunction of $z' = *$ statements over its variables $\mathrm{vars}'(p_C) = V_1 \cap \mathrm{vars}'(F_2, \sigma)$. (If $V_1 \cap \mathrm{vars}'(F_2, \sigma) = \emptyset$ then $\sigma \notin \Sigma_2$ and $c \xrightarrow{\sigma:p_C} c$ still holds for the empty conjunction, $p_C \equiv \mathrm{true}$.) The update p_C is true for all valuations, only its next-state variables are important. From (2.86) it follows that $x_1 \xrightarrow{\sigma:p_1} y_1$ in F_1 and $x_2 \xrightarrow{\sigma:p_2} y_2$ in F_2 such that $p \equiv p_1 \wedge p_2$. Then $F_1 \parallel C$ has a transition

$$(x_1, c) \xrightarrow{\sigma:p_1 \wedge p_C} (y_1, c) \ . \tag{2.87}$$

As $(\hat{v} \oplus \hat{w}')(p) = (\hat{v} \oplus \hat{w}')(p_1 \wedge p_2) = \mathrm{true}$, it follows that $(\hat{v} \oplus \hat{w}')(p_1) = \mathrm{true}$, and therefore also $(\hat{v} \oplus \hat{w}')(p_1 \wedge p_C) = \mathrm{true}$.

Now consider $z \in V_1 \setminus \mathrm{vars}'(p_1 \wedge p_C)$. Then $z \in V_1 \subseteq W$ and $z \notin \mathrm{vars}'(p_1)$ and $z \notin \mathrm{vars}'(p_C) = V_1 \cap \mathrm{vars}'(F_2, \sigma)$, and given $z \in V_1$ also $z \notin \mathrm{vars}'(F_2, \sigma) \supseteq \mathrm{vars}'(p_2)$. This means $z \notin \mathrm{vars}'(p_1 \wedge p_2) = \mathrm{vars}'(p)$ and therefore $\hat{v}(z) = \hat{w}(z)$ from above. This shows the claim $(x_1, c, \hat{v}_{|V_1}) \xrightarrow{\sigma} (y_1, c, \hat{w}_{|V_1})$ in $F_1 \parallel C$. □

In a synchronous composition $F_1 \parallel F_2$, the chaos abstraction of F_2 as defined by (2.84) is the composition of chaos EFSMs for variables in F_1 and events with transitions assigning to these variables in F_2. The condition $z \in \mathrm{vars}(F_1) \cap \mathrm{vars}'(F_2, \sigma)$ in (2.84) can only hold for variables shared between F_1 and F_2 and for events of F_2, so that the construction can be restricted to $\sigma \in \Sigma_2$ and $z \in \mathrm{vars}(F_1) \cap \mathrm{vars}(F_2)$. The composition C of these chaos EFSMs is a one-state EFSM with selfloop transitions $\sigma : z' = *$ for all events $\sigma \in \Sigma_2$ and variables $z \in \mathrm{vars}(F_1) \cap \mathrm{vars}'(F_2, \sigma)$. Lemma 12 allows F_2 to be replaced by this chaos EFSM C, such that all transitions in the composition $F_1 \parallel F_2$ are also possible in the abstraction $F_1 \parallel C$.

2.7 Existential Abstraction

An important feature of the results in this working paper is the ability to simplify EFSMs through *variable abstraction* [23] using the existential quantifier. If $p \in \Pi$ is an update and z

is a variable, then $\exists z\, p$ is an update that is true if and only if p can be made true by choosing some value for the variable z from its domain [6].

In this working paper, quantification is generalised to sets of variables as follows. For $W = \{z_0, z_1, \ldots, z_n\}$, it is defined that

$$\exists W\, p \;\equiv\; \exists z_0 \exists z_0' \exists z_1 \exists z_1' \cdots \exists z_n \exists z_n'\, p\ ; \tag{2.88}$$

$$\exists W'\, p \;\equiv\; \exists z_0' \exists z_1' \cdots \exists z_n'\, p\ . \tag{2.89}$$

That is, an update is quantified over variable set W by quantifying over both the current-state and next-state variables of W. Differently, quantification over W' means to quantify over the next-state variables only. The same notation is introduced for the universal quantifier, so $\forall W\, p$ is true if and only if p is true for all possible values of the current and next-state variables of W.

An EFSM is abstracted by existentially quantifying the updates on all the transitions.

Definition 12 [23] Let $F = \langle \Sigma, Q, Q^\circ, \rightarrow \rangle$ be an EFSM, and let W be a set of variables. The *existential abstraction* of F with respect to W is the EFSM $\exists W\, F = \langle \Sigma, Q, Q^\circ, \rightarrow_\exists \rangle$ where $x \xrightarrow{\sigma:\exists W\, p}_\exists y$ in $\exists W\, F$ if and only if $x \xrightarrow{\sigma:p} y$ in F.

The existential abstraction of a set $\mathcal{F} = \{F_1, \ldots, F_n\}$ of EFSMs is the set $\exists W\, \mathcal{F} = \{\exists W\, F_1, \ldots, \exists W\, F_n\}$ of the abstractions of the individual EFSMs.

Existential abstraction results in an EFSM that is independent of the quantified variables, i.e., $\mathrm{vars}(\exists W\, F) \cap W = \emptyset$. A transition in the abstraction is possible if there exist values for the quantified variables to make the update in the original EFSM F true. It is clear that existential abstraction preserves the EFSM properties of normalisation and purity. State-determinism is not preserved, however, so it has to be required explicitly that an abstraction is state-deterministic in order for it to be used in synthesis.

Lemma 13 Let F be an EFSM, and let W be a set of variables.

(i) If F is normalised, then $\exists W\, F$ is normalised.

(ii) If F is pure, then $\exists W\, F$ is pure.

Proof. Note that $\mathrm{vars}(p)$ denotes the set of *free* variables in an update p, and therefore $\mathrm{vars}'(\exists W\, p) = \mathrm{vars}'(p) \setminus W'$. Then both claims follow directly from Definition 2. □

Another concept related to existential quantification is that of always enabled events. An event in an ordinary FSM is always enabled if it has a transition from every state. With EFSMs, it may additionally be of interest that the event is enabled for all values of variables. The following definition requires an event that is enabled for all current-state values and some next-state values of the variables, while the next-state values for other variables only need to exist.

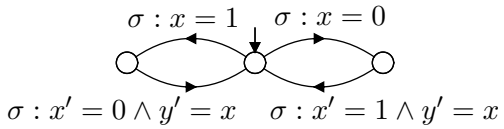

Figure 3: Example of always enabled and unconstrained events.

Definition 13 [10] Let $F = \langle \Sigma, Q, Q^\circ, \rightarrow \rangle$ be an EFSM, and let $W \subseteq \mathrm{vars}(F)$. An event $\sigma \in \Sigma$ is *always enabled* in F with respect to W if for all locations $x \in Q$ with σ-transitions

$$x \xrightarrow{\sigma:p_1} y_1 \quad \cdots \quad x \xrightarrow{\sigma:p_n} y_n \tag{2.90}$$

the formula

$$\exists W' \, (p_1 \vee \cdots \vee p_n) \tag{2.91}$$

is valid. An event set $\Sigma' \subseteq \Sigma$ is always enabled in F with respect to W if every event $\sigma \in \Sigma'$ has this property.

Example 3 Consider the EFSM F in Figure 3 with $\mathrm{dom}(x) = \mathrm{dom}(y) = \{0, 1\}$. Event σ is always enabled in F with respect to $W_1 = \{x, y\}$, because independently of the current value of the variables x and y, it is always possible to choose values for x and y such that some other location can be reached. But σ is *not* always enabled with respect to $W_2 = \{x\}$, because the update $x' = 0 \wedge y' = x$ is not possible when the next value of y is pre-selected to be different from the current value of x. Formally, $\exists x' \, (x' = 0 \wedge y' = x)$ is not valid because, if $x = 1$ and $y' = 0$ then it can never be that $y' = x$.

The following definition introduces the related but slightly different condition of unconstrained events, whose enablement does not depend on certain variables.

Definition 14 Let $F = \langle \Sigma, Q, Q^\circ, \rightarrow \rangle$ be an EFSM, and let $W \subseteq \mathrm{vars}(F)$. An event $\sigma \in \Sigma$ is *unconstrained* in F with respect to W if, for all transitions $x \overset{\sigma:p}{\rightarrow} y$ in F the formula

$$\exists W p \Rightarrow \forall W \exists W' p \tag{2.92}$$

is valid. An event set $\Sigma' \subseteq \Sigma$ is unconstrained in F with respect to W if every event $\sigma \in \Sigma'$ has this property.

The symbol $\Rightarrow$ in (2.92) denotes logical implication [6]. By convention (2.88) the quantification $\exists W$ and $\forall W$ is over both current-state and next-state variables, but in $\forall W \exists W'$ the universal quantification of the next-state variables is immediately overridden by $\exists W'$.

An update is unconstrained by a variable z if, in all cases where the update formula is true for some value of z, then it is also true for all other values of z in the current state,

but possibly with different values of z' in the next state. While an always enabled event is enabled in every location, an unconstrained event, if it is enabled, is enabled independently of given variables.

Example 4 Consider again the EFSM F in Figure 3. The event σ is *not* unconstrained with respect to $W_1 = \{x\}$, because the update $x = 1$ is possible when $x = 1$, so $\exists x \exists x'\, x = 1$ or equivalently $\exists x\ x = 1$ is true, but not for all other values of x, namely $\forall x \exists x'\, x = 1$ or equivalently $\forall x\ x = 1$ is not true. But σ is unconstrained with respect to $W_2 = \{y\}$, because all the transitions are enabled independently of the current value of y. That is, if a transition is enabled for some current-state value of y, then the transition can also be taken for all other values of y, possibly with different values for y' in the next state.

3 Abstracting the Plant

This and the following sections propose modular synthesis algorithms for systems composed of several EFSM components. This section considers the possibilities of abstraction of the plant. It is assumed that the plant is defined as the synchronous composition of a set $\mathcal{G} = \{G_1, \ldots, G_m\}$ of normalised state-deterministic EFSMs, and the specification is given by a single pure state-deterministic EFSM E. More general specifications and their abstraction are considered in the following sections.

3.1 Algorithm

The idea of modular synthesis [1,2] is to simplify the plant $\mathcal{G} = \{G_1, \ldots, G_m\}$ by selecting some of its components, and discarding the others, in such a way that the result is equivalent to that of synthesis with respect to the complete plant. In the EFSM setting, this is now generalised to the selection of plant components *and* variables. Algorithm 1 is further developed from modular FSM synthesis algorithms [1,10] and the modular EFSM synthesis algorithm [10], in that it considers existential abstraction of variables in addition to component selection and chaos abstraction.

The idea is to gradually increase the sets of plants $\mathcal{G}^i$, variables V^i until it is guaranteed that the optimal result has been found. At each iteration, the algorithm considers sets of plants $\mathcal{G}^i \subseteq \mathcal{G}$ and variables $V^i \subseteq \mathrm{vars}(\mathcal{G}) \cup \mathrm{vars}(E)$, and also uncontrollable events $\Sigma_{\mathrm{u}}^i \subseteq \Sigma_{\mathrm{u}}$. Throughout the algorithm, $\bar{\mathcal{G}}^i$ and $\bar{V}^i$ are always the complements of $\mathcal{G}^i$ and V^i.

Initially, synthesis is performed using only the specification E and its variables, as $\mathcal{G}^0 = \emptyset$ (line 3) and $V^0 = \mathrm{vars}(E)$ (line 3). Following Lemma 12, the plants $\bar{\mathcal{G}}^0 = \mathcal{G}$ are replaced by chaos EFSMs $\mathcal{C}^0$ for the included variables (line 7). The set of uncontrollable events Σ_{u}^0 is initially empty (line 2), i.e., synthesis is first performed under the pretence that all events are controllable. In this case, the synthesis result S^0 is equal to the specification E (line 8).

Algorithm 1: Modular abstracting EFSM synthesis for a single specification

Input: normalised state-deterministic plants $\mathcal{G} = \{G_1, \ldots, G_m\}$;
pure state-deterministic specification E; uncontrollable events Σ_u.

Output: supremal supervisor S^k for $\big\|(\mathcal{G})$ with respect to E and Σ_u.

$V \leftarrow \mathrm{vars}(G) \cup \mathrm{vars}(E)$;
$\Sigma_\mathrm{u}^0 \leftarrow \emptyset$;
$\mathcal{G}^0 \leftarrow \emptyset$;
$\bar{\mathcal{G}}^0 \leftarrow \mathcal{G}$;
$V^0 \leftarrow \mathrm{vars}(E)$;
$\bar{V}^0 \leftarrow V \setminus V^0$;
$\mathcal{C}^0 \leftarrow \{\, \mathrm{chaos}(\sigma, v) \mid v \in V^0 \cap \mathrm{vars}'(\bar{\mathcal{G}}^0, \sigma) \,\}$;
$S^0 \leftarrow E$;
$i \leftarrow 0$;
while S^i is not Σ_u-controllable with respect to $\big\|(\exists \bar{V}^i\, \mathcal{G}^i) \parallel \big\|(\mathcal{C}^i)$ **do**
  $\Sigma_\mathrm{u}^{i+1} \leftarrow \Sigma_\mathrm{u}^i \cup \mathrm{uncont}(\big\|(\exists \bar{V}^i\, \mathcal{G}^i) \parallel \big\|(\|\mathcal{C}^i), S^i, \Sigma_\mathrm{u})$;
  Choose $\mathcal{G}^{i+1} \subseteq \mathcal{G}$ and $V^{i+1} \subseteq V$ and $\bar{\mathcal{G}}^{i+1} = \mathcal{G} \setminus \mathcal{G}^{i+1}$ and $\bar{V}^{i+1} = V \setminus V^{i+1}$
    such that $\mathrm{vars}(E) \subseteq V^{i+1} \subseteq \mathrm{vars}(\mathcal{G}^{i+1}) \cup \mathrm{vars}(E)$
    and Σ_u^{i+1} is always enabled in $\bar{\mathcal{G}}^{i+1}$ with respect to $\bar{V}^{i+1}$
    and Σ_u^{i+1} is unconstrained in $\mathcal{G}^{i+1}$ with respect to $\bar{V}^{i+1}$
    and $\mathrm{vars}'(\mathcal{G}^{i+1}, \mu) \cap \mathrm{vars}'(\bar{\mathcal{G}}^{i+1}, \mu) \cap \bar{V}^{i+1} = \emptyset$ for each $\mu \in \Sigma_\mathrm{u}^{i+1}$
    and $\exists \bar{V}^{i+1}\, \mathcal{G}^{i+1}$ is state-deterministic;
  $\mathcal{C}^{i+1} \leftarrow \{\, \mathrm{chaos}(\sigma, z) \mid z \in V^{i+1} \cap \mathrm{vars}'(\bar{\mathcal{G}}^{i+1}, \sigma) \,\}$;
  $S^{i+1} \leftarrow \mathrm{sup}\mathcal{C}\big(\big\|(\exists \bar{V}^{i+1}\, \mathcal{G}^{i+1}) \parallel \big\|(\mathcal{C}^{i+1}), E, \Sigma_\mathrm{u}^{i+1}\big)$;
  $i \leftarrow i + 1$;
end
return S^i

Therefore, on entering the loop for the first time, the loop entry condition in line 10 checks whether the specification $S^0 = E$ is controllable with respect to only the chaos EFSM $\mathcal{C}^0$ (recall that $\mathcal{G}^0 = \emptyset$), based on the full set Σ_u of uncontrollable events. This may succeed if, for example, the specification has only controllable events, in which case $S^0 = E$ is returned as the least restrictive solution. Otherwise the loop is entered and synthesis is performed with respect to increased subsets of plants, variables, and uncontrollable events, which are computed as follows.

First, line 11 calculates a new set Σ_u^{i+1} of uncontrollable events. As the current supervisor S^i is not controllable by the loop entry condition, there must be some uncontrollable event that is possible in the plant but not in the specification. These events are called the *causes of uncontrollability*, as per the following definition.

Definition 15 Let G and E be two EFSMs, and let Σ_u be a set of events. The set of *causes of* Σ_u*-uncontrollability* of E with respect to G is the set of events

$$\begin{aligned}\mathrm{uncont}(G, E, \Sigma_\mathrm{u}) = \{\, \mu \in \Sigma_\mathrm{u} \mid \text{there exist } (x_G, x_E, \hat{v}) \in Q^{\mathrm{acc}}(G \parallel E) \text{ and } (x_G, \hat{v}) \xrightarrow{\mu} \\ (x_G, \hat{w}) \text{ in } G, \text{ and there is no location } y_E \text{ in } E \text{ such that} \\ (x_G, x_E, \hat{v}) \xrightarrow{\mu} (y_G, y_E, \hat{w}) \,\} \,.\end{aligned} \tag{3.1}$$

It is clear that the set of causes of uncontrollability is empty, $\mathrm{uncont}(G, E, \Sigma_\mathrm{u}) = \emptyset$, if and only if G is Σ_u-controllable with respect to E. As the loop entry condition has found the supervisor S^i to be not Σ_u^i-controllable with respect to the plant abstraction $\big\|(\exists \bar{V}^i\, \mathcal{G}^i) \,\big\| \big\|(\mathcal{C}^i)$, there exist some causes of uncontrollability, which are included in the next set Σ_u^{i+1} of uncontrollable events on line 11. This ensures that they are treated as uncontrollable for the next synthesis attempt. For other uncontrollable events, the algorithm will continue to pretend that they are controllable.

Next, line 12 chooses new plants $\mathcal{G}^{i+1}$ and variables V^{i+1} to form an improved approximation. First, all variables used in the specification are retained,

$$\mathrm{vars}(E) \subseteq V^{i+1} \,. \tag{3.2}$$

In this section, variables that appear in the specification are not abstracted. Later, in Section 4, it is shown how the amount of variables in the specification can be reduced by specification abstraction before Algorithm 1 is invoked.

The main part of the logic in Algorithm 1 is the selection of plant components and variables from them. To ensure a least restrictive synthesis result, all plant components that can disable some uncontrollable event are included [1,10]. Therefore it is required that the uncontrollable events must be always enabled by the plants $\bar{\mathcal{G}}^{i+1}$ and variables $\bar{V}^{i+1}$ not included in synthesis according to Definition 13,

$$\Sigma_\mathrm{u}^{i+1} \text{ is always enabled in } \bar{\mathcal{G}}^{i+1} \text{ with respect to } \bar{V}^{i+1}. \tag{3.3}$$

This condition ensures that any plant components that could cause disablement of some uncontrollable event from $\Sigma_{\mathrm{u}}^{i+1}$ are included in the abstraction. Furthermore, all variables that can constrain these uncontrollable events in the selected plant components $\mathcal{G}^{i+1}$ must also be included. This can be ensured by including all variables that appear in the selected plants on transitions with the selected uncontrollable events, or as weaker condition it is enough that the uncontrollable events are unconstrained by the other variables according to Definition 14,

$$\Sigma_{\mathrm{u}}^{i+1} \text{ is unconstrained in } \mathcal{G}^{i+1} \text{ with respect to } \bar{V}^{i+1}. \tag{3.4}$$

In order to consider the conditions (3.3) and (3.4) separately for $\mathcal{G}^{i+1}$ and $\bar{\mathcal{G}}^{i+1}$, it is furthermore necessary that there are no conflicting assignments to the same abstracted variable. Therefore it is required that $\mathcal{G}^{i+1}$ and $\bar{\mathcal{G}}^{i+1}$ do not share these variables in their primed form,

$$\mathrm{vars}'(\mathcal{G}^{i+1}, \mu) \cap \mathrm{vars}'(\bar{\mathcal{G}}^{i+1}, \mu) \cap \bar{V}^{i+1} = \emptyset \quad \text{for each } \mu \in \Sigma_{\mathrm{u}}^{i+1} \ . \tag{3.5}$$

This condition can be checked separately for each event μ considered as uncontrollable in the current iteration.

To summarise, in addition to including all variables from the specification (3.2), the selected uncontrollable events must be always enabled (3.3) in the plants $\bar{\mathcal{G}}^{i+1}$ that are *not* included in the approximation and unconstrained (3.4) in the plants $\mathcal{G}^{i+1}$ that are included, and the two parts $\mathcal{G}^{i+1}$ and $\bar{\mathcal{G}}^{i+1}$ of the plant must not use any of the abstracted variables in their primed form with the same uncontrollable event (3.5).

The variables not included in V^{i+1}, i.e., those in $\bar{V}^{i+1}$, are removed by existential abstraction. In order for synthesis to be well-defined, the abstraction must remain state-deterministic. This is ensured by the last condition,

$$\exists \bar{V}^{i+1}\, \mathcal{G}^{i+1} \text{ is state-deterministic} \ . \tag{3.6}$$

After the plants and variables for the next step have been chosen, line 13 introduces chaos EFSMs $\mathcal{C}^{i+1}$ to replace the plants $\bar{\mathcal{G}}^{i+1}$ that are not included, such that any possible changes to the included variables V^{i+1} are reflected in the abstraction [10].

Then line 14 performs synthesis for the plant abstraction $\left\|(\exists \bar{V}^{i+1}\, \mathcal{G}^{i+1}) \right\| \left\|(\mathcal{C}^{i+1})\right.$ and the chosen set $\Sigma_{\mathrm{u}}^{i+1}$ of uncontrollable events. If the resulting supervisor is controllable with respect to the full set Σ_{u} of uncontrollable events (line 10), then it is returned as the result. Otherwise more uncontrollable events need to be included, resulting in a new plant abstraction. The loop continues until a Σ_{u}-controllable solution is found.

Line 12 of Algorithm 1 may be difficult to implement as it is not specified how the plant components $\mathcal{G}^{i+1}$ and variables V^{i+1} can be chosen such that conditions (3.2)–(3.6) are satisfied at the same time. A simple approach is to start with the variables of the specification (3.2) and the plants that disable an uncontrollable event from Σ_{u}^{i} in some location (3.3), and then gradually add more plants and variables until all conditions are satisfied. The search may be

simpler for well-designed EFSM models in practice, such as the flexible manufacturing system presented in Section 6 where not all conditions need to be considered.

The following subsections contain a correctness proof of Algorithm 1. Clearly, termination is guaranteed because the set Σ_{u}^i of uncontrollable events increases with every iteration, and it is bounded by the finite set Σ_{u} of all uncontrollable events. It remains to be shown that the algorithm returns a correct result, i.e., that the supervisor S^i returned from line 17 is indeed a supremal supervisor for E with respect to $\mathcal{G}$ and Σ_{u}.

It is clear from line 14 that

$$S^i = \mathrm{sup}\mathcal{C}\big(\, \|(\exists \bar{V}^i\, \mathcal{G}^i) \parallel \|(\mathcal{C}^i), E, \Sigma_{\mathrm{u}}^i\big) \tag{3.7}$$

is a supremal supervisor for E with respect to $\|(\exists \bar{V}^i\, \mathcal{G}^i) \parallel \|(\mathcal{C}^i)$ and Σ_{u}^i. To prove the correctness claim, it will first be shown that, for all i,

$$S^i \text{ is a supremal supervisor for } E \text{ with respect to } \mathcal{G} \text{ and } \Sigma_{\mathrm{u}}^i\ . \tag{3.8}$$

This is shown by Proposition 20 in Section 3.5 below. Then the only difference between S^i and the desired result is the uncontrollable event set: S^i is synthesised with respect to $\Sigma_{\mathrm{u}}^i \subseteq \Sigma_{\mathrm{u}}$. As S^i uses fewer uncontrollable events, it can secondly be shown that it over-approximates the synthesis result with respect to the full uncontrollable event set Σ_{u},

$$\mathrm{sup}\mathcal{C}(\|(\mathcal{G}), E, \Sigma_{\mathrm{u}}) \subseteq_{\mathrm{v}} S^i\ . \tag{3.9}$$

On termination of the loop, S^i is not only Σ_{u}^i-controllable but also Σ_{u}-controllable. At this point, it follows from (3.8) and (3.9) that S^i is a supremal supervisor for E with respect to $\mathcal{G}$ and Σ_{u}.

3.2 Proof of Behavioural Inclusion

As a first step towards the proof of (3.8), the following Proposition 14 shows that a supervisor computed from any configuration of variable abstraction satisfies the specification. It is well-known for ordinary FSMs that any supervisor synthesised for some specification E results in a behaviour more restrictive than E. In the case of EFSMs there is a complication, because the supervisor could theoretically increase behaviour by the addition of variables in the synthesised updates. This possibility can be ruled out through the assumption of normalisation.

The following proposition shows that the supervisor obtained after every iteration of Algorithm 1 satisfies the given specification. In this and the following propositions, G is the part of the plant included in the current approximation, while H is the part not included and replaced by chaos EFSMs C. The variables included in the current approximation are denoted W, and $\bar{W}$ is their complement. That is, $G = \|(\mathcal{G}^{i+1})$, $H = \|(\bar{\mathcal{G}}^{i+1})$, $C = \|(\mathcal{C}^{i+1})$, and $W = V^{i+1}$ in the notation of Algorithm 1.

Proposition 14 Let G, H, and E be EFSMs, where G and H are normalised, let $V = \mathrm{vars}(G) \cup \mathrm{vars}(H) \cup \mathrm{vars}(E)$ and $\Sigma_{\mathrm{u}} \subseteq \Sigma$. Let $W \subseteq V$ and $\bar{W} = V \setminus W$, $C = \|(\{\, \mathrm{chaos}(\sigma, z) \mid z \in W \cap \mathrm{vars}'(H, \sigma) \,\})$, and $S = \mathrm{sup}\mathcal{C}((\exists \bar{W} G) \parallel C, E, \Sigma_{\mathrm{u}})$. Then

$$G \parallel H \parallel S \subseteq_{\mathrm{v}} G \parallel H \parallel E \ . \tag{3.10}$$

Proof. Note that $\mathrm{vars}(S) = \mathrm{vars}(\mathrm{sup}\mathcal{C}((\exists \bar{W} G) \parallel C, E, \Sigma_{\mathrm{u}})) \subseteq V$ and therefore $\mathrm{vars}(G \parallel H \parallel S) = \mathrm{vars}(G \parallel H \parallel E) = V$. Then by Definition 6 it is enough to show for any path

$$(x_G^0, x_H^0, (\tilde{x}_G^0, c, x_E^0), \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_H^n, (\tilde{x}_G^n, c, x_E^n), \hat{v}^n) \tag{3.11}$$

in $G \parallel H \parallel S$, with $\hat{v}^i \in \mathrm{dom}(V)$ for $i = 0, \ldots, n$, that

$$(x_G^0, x_H^0, x_E^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_H^n, x_E^n, \hat{v}^n) \tag{3.12}$$

is a path in $G \parallel H \parallel E$. Clearly x_G^0, x_H^0, x_E^0, and $\hat{v}^0$ are initial locations and variable values from (3.11). Now consider a transition on the path (3.11)

$$(x_G^j, x_H^j, (\tilde{x}_G^j, c, x_E^j), \hat{v}^j) \xrightarrow{\sigma_{1+1}} (x_G^{j+1}, x_H^{j+1}, (\tilde{x}_G^{j+1}, c, x_E^{j+1}), \hat{v}^{j+1}) \ . \tag{3.13}$$

in $G \parallel H \parallel S$. By Lemma 2, there exist transitions

$$x_G^j \xrightarrow{\sigma_j : p_G} x_G^{j+1} \qquad \text{with } (\hat{v}^j \oplus (\hat{v}^{j+1})')(p_G) = \mathrm{true} \qquad \text{in } G \ ; \tag{3.14}$$

$$x_H^j \xrightarrow{\sigma_j : p_H} x_H^{j+1} \qquad \text{with } (\hat{v}^j \oplus (\hat{v}^{j+1})')(p_H) = \mathrm{true} \qquad \text{in } H \ ; \tag{3.15}$$

$$(\tilde{x}_G^j, c, x_E^j) \xrightarrow{\sigma_j : p_S} (\tilde{x}_G^{j+1}, c, x_E^{j+1}) \qquad \text{with } (\hat{v}^j \oplus (\hat{v}^{j+1})')(p_S) = \mathrm{true} \qquad \text{in } S \ ; \tag{3.16}$$

such that

$$\hat{v}^j(z) = \hat{v}^{j+1}(z) \quad \text{for all } z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H) \cup \mathrm{vars}'(p_S)) \ . \tag{3.17}$$

The update of the transition (3.16) in S takes the form $p_S \equiv q_G \wedge q_C \wedge q_E \wedge R$ with $\mathrm{vars}'(R) \subseteq \mathrm{vars}'(q_G \wedge q_C \wedge q_E)$, for some transitions $x_G^j \xrightarrow{\sigma_j : q_G} x_G^{j+1}$ in $\exists \bar{W} G$, $c \xrightarrow{\sigma_j : q_C} c$ in C, and

$$x_E^j \xrightarrow{\sigma_j : q_E} x_E^{j+1} \qquad \text{with } (\hat{v}^j \oplus (\hat{v}^{j+1})')(q_E) = \mathrm{true} \qquad \text{in } E \ . \tag{3.18}$$

Now consider a variable $z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H) \cup \mathrm{vars}'(q_E))$. Then since G and H are normalised, $z \notin \mathrm{vars}'(p_G) = \mathrm{vars}'(G, \sigma_j) \supseteq \mathrm{vars}'(\exists \bar{W} G, \sigma_j) = \mathrm{vars}'(q_G)$ and $z \notin \mathrm{vars}'(p_H) = \mathrm{vars}'(H, \sigma_j) \supseteq \mathrm{vars}'(C, \sigma_j) = \mathrm{vars}'(q_C)$, and then also $z \notin \mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_C) \cup \mathrm{vars}'(q_E) \supseteq \mathrm{vars}'(R)$. Then also $z \notin \mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_C) \cup \mathrm{vars}'(q_E) \cup \mathrm{vars}'(R) = \mathrm{vars}'(p_S)$. It follows that $\hat{v}^j(z) = \hat{v}^{j+1}(z)$ by (3.17). Combining this with (3.14), (3.15), and (3.18), it follows using Lemma 2 that

$$(x_G^j, x_H^j, x_E^j, \hat{v}^j) \xrightarrow{\sigma_j} (x_G^{j+1}, x_H^{j+1}, x_E^{j+1}, \hat{v}^{j+1}) \tag{3.19}$$

in $G \| H \| E$. The path (3.12) is obtained by repeating this argument for all the steps in (3.11). □

3.3 Proof of Controllability

The second step towards the proof of (3.8) is to show that the supervisor obtained after every iteration of Algorithm 1 is controllable with respect to the original plant. As each supervisor is synthesised for a plant abstraction, it must be controllable with respect to that abstraction. For ordinary FSMs it is known [2] that a supervisor controllable with respect to a part of a composed plant is also controllable with respect to the complete plant. For EFSMs, this result has been generalised [10] for the inclusion of chaos abstractions, and in Proposition 16 in this section, it is further generalised to cover the existential abstraction performed by Algorithm 1.

The proposition depends on the following lemma about the reachability of states. Every state reachable in the original plant corresponds to a reachable state in the plant abstraction used by Algorithm 1. This means that the abstraction allows more behaviour than the original plant.

Lemma 15 Let G, H, and S be EFSMs with $V = \mathrm{vars}(G) \cup \mathrm{vars}(H) \cup \mathrm{vars}(S)$. Let $W \subseteq V$ and $\bar{W} = V \setminus W$ and $C = \big\|(\{\, \mathrm{chaos}(\sigma, z) \mid z \in W \cap \mathrm{vars}'(H, \sigma) \,\})$ such that $\mathrm{vars}(S) \subseteq W$. If $(x_G, x_H, x_S, \hat{v}) \in Q^{\mathrm{acc}}(G \parallel H \parallel S)$, then $(x_G, c, x_S, \hat{v}\restriction_W) \in Q^{\mathrm{acc}}((\exists \bar{W} G) \parallel C \parallel S)$ where c is the single location of C.

Proof. Note that $\mathrm{vars}((\exists \bar{W} G) \parallel C) \subseteq W$, which given $\mathrm{vars}(S) \subseteq W$ implies $\mathrm{vars}((\exists \bar{W} G) \parallel C \parallel S) \subseteq W$. Let $(x_G, x_H, x_S, \hat{v}) \in Q^{\mathrm{acc}}(G \parallel H \parallel S)$ for some $\hat{v} \in \mathrm{dom}(V)$. Then there exists a path

$$(x_G^0, x_H^0, x_S^0, \hat{v}^0) \xrightarrow{\sigma_1} (x_G^1, x_H^1, x_S^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, x_H^n, x_S^n, \hat{v}^n) = (x_G, x_H, x_S, \hat{v}) \tag{3.20}$$

in $G \parallel H \parallel S$. It will be shown that

$$(x_G^0, c, x_S^0, \hat{v}^0\restriction_W) \xrightarrow{\sigma_1} (x_G^1, c, x_S^1, \hat{v}^1\restriction_W) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_G^n, c, x_S^n, \hat{v}^n\restriction_W) = (x_G, c, x_S, \hat{v}\restriction_W) \tag{3.21}$$

is a path in $(\exists \bar{W} G) \parallel C \parallel S$.

Clearly, x_G^0, c, x_S^0, and $\hat{v}^0\restriction_W$ are initial locations and variable values. Now consider a step

$$(x_G^j, x_H^j, x_S^j, \hat{v}^j) \xrightarrow{\sigma_j} (x_G^{j+1}, x_H^{j+1}, x_S^{j+1}, \hat{v}^{j+1}) \tag{3.22}$$

on the path (3.20). By Lemma 2, there are transitions $x_G^j \xrightarrow{\sigma_j : p_G} x_G^{j+1}$ with $(\hat{v}^j \oplus (\hat{v}^{j+1})')(p_G) = \text{true}$ in G, $x_H^j \xrightarrow{\sigma_j : p_H} x_H^{j+1}$ with $(\hat{v}^j \oplus (\hat{v}^{j+1})')(p_H) = \text{true}$ in H, and $x_S^j \xrightarrow{\sigma_j : p_S} x_S^{j+1}$ with $(\hat{v}^j \oplus (\hat{v}^{j+1})')(p_S) = \text{true}$ in S, such that

$$\hat{v}^j(z) = \hat{v}^{j+1}(z) \quad \text{for all } z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H) \cup \mathrm{vars}'(p_S)) \,. \tag{3.23}$$

From $x_G^j \xrightarrow{\sigma_j : p_G} x_G^{j+1}$ in G it follows that

$$x_G^j \xrightarrow{\sigma_j : \exists \bar{W} p_G} x_G^{j+1} \quad \text{with } (\hat{v}^j \oplus (\hat{v}^{j+1})')(\exists \bar{W} p_G) = \text{true} \quad \text{in } \exists \bar{W} G \,. \tag{3.24}$$

By construction of the chaos EFSMs, it is clear that $c \xrightarrow{\sigma_j : p_C} c$ with $(\hat{v}^j \oplus (\hat{v}^{j+1})')(p_C) = \text{true}$ in C.

Now consider $z \in W \setminus (\text{vars}'(\exists \bar{W} p_G) \cup \text{vars}'(p_C) \cup \text{vars}'(p_S))$. Then $z \in W \subseteq V$ and $z \notin \text{vars}'(\exists \bar{W} p_G)$ and $z \notin \text{vars}'(p_C)$ and $z \notin \text{vars}'(p_S)$. From $z \notin \text{vars}'(\exists \bar{W} p_G) = \text{vars}'(p_G) \setminus \bar{W} = \text{vars}'(p_G) \cap W$ and $z \in W$ it is clear that $z \notin \text{vars}'(p_G)$, and $z \notin \text{vars}'(p_C) = \text{vars}'(C, \sigma_j)$ means that C cannot include $\text{chaos}(\sigma_j, z)$, which given $z \in W$ implies $z \notin \text{vars}'(H, \sigma_j)$ by construction of C. Thus, $z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H) \cup \text{vars}'(p_S))$, which implies $\hat{v}^j(z) = \hat{v}^{j+1}(z)$ by (3.23). By Lemma 2, it follows that

$$(x_G^j, c, x_S^j, \hat{v}^j \restriction_W) \xrightarrow{\sigma_j} (x_G^{j+1}, c, x_S^{j+1}, \hat{v}^{j+1} \restriction_W) \tag{3.25}$$

in $(\exists \bar{W} G) \parallel C \parallel S$. The path (3.21) is obtained by repeating this argument for all the steps in (3.20). □

Using the above lemma, it is now possible to prove this section's result about controllability with respect to a plant abstraction. If a supervisor is controllable with respect to an abstraction in Algorithm 1, it is also controllable with respect to the original plant. This means that the supervisor obtained after every iteration of Algorithm 1 is controllable with respect to the original plant.

Proposition 16 Let G, H, and S be EFSMs with $V = \text{vars}(G) \cup \text{vars}(H) \cup \text{vars}(S)$ and $\Sigma_u \subseteq \Sigma$. Let $W \subseteq V$ and $\bar{W} = V \setminus W$ and $C = \big\|(\{\, \text{chaos}(\sigma, z) \mid z \in W \cap \text{vars}'(H, \sigma) \,\})$ such that $\text{vars}(S) \subseteq W$. If S is Σ_u-controllable with respect to $(\exists \bar{W} G) \| C$, then S is Σ_u-controllable with respect to $G \parallel H$.

Proof. Note that $\text{vars}((\exists \bar{W} G) \parallel C) \subseteq W$. Let $(x_G, x_H, x_S, \hat{v}) \in Q^{\text{acc}}(G \parallel H \parallel S)$, $\mu \in \Sigma_u$, and

$$(x_G, x_H, \hat{v}) \xrightarrow{\mu} (y_G, y_H, \hat{w}) \quad \text{in } G \parallel H \ , \tag{3.26}$$

where $\hat{v}, \hat{w} \in \text{dom}(V)$. Following Definition 7, it is to be shown that there exists a location y_S of S such that

$$(x_G, x_H, x_S, \hat{v}) \xrightarrow{\mu} (y_G, y_H, y_S, \hat{w}) \quad \text{in } G \parallel H \parallel S \ . \tag{3.27}$$

First, it will be shown that

$$(x_G, c, \hat{v} \restriction_W) \xrightarrow{\mu} (y_G, c, \hat{w} \restriction_W) \quad \text{in } (\exists \bar{W} G) \parallel C \ , \tag{3.28}$$

where c is the single location of C. The assumption (3.26) means by Lemma 2 that there are transitions $x_G \xrightarrow{\mu : p_G} y_G$ with $(\hat{v} \oplus \hat{w}')(p_G) = \text{true}$ in G and $x_H \xrightarrow{\mu : p_H} y_H$ with $(\hat{v} \oplus \hat{w}')(p_H) = \text{true}$ in H, such that

$$\hat{v}(z) = \hat{w}(z) \quad \text{for all } z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H)) \ . \tag{3.29}$$

From $x_G \xrightarrow{\mu:p_G} y_G$ in G it follows that $x_G \xrightarrow{\mu:\exists \bar{W} p_G} y_G$ in $\exists \bar{W} G$ with $(\hat{v} \oplus \hat{w}')(\exists \bar{W} p_G) = \text{true}$. By construction of the chaos EFSMs, it is clear that $c \xrightarrow{\mu:p_C} c$ with $(\hat{v} \oplus \hat{w}')(p_C) = \text{true}$ in C. Now consider $z \in W \setminus (\text{vars}'(\exists \bar{W} p_G) \cup \text{vars}'(p_C))$. Then $z \in W \subseteq V$ and $z \notin \text{vars}'(\exists \bar{W} p_G)$ and $z \notin \text{vars}'(p_C)$. From $z \notin \text{vars}'(\exists \bar{W} p_G) = \text{vars}'(p_G) \setminus \bar{W} = \text{vars}'(p_G) \cap W$ and $z \in W$ it is clear that $z \notin \text{vars}'(p_G)$, and from $z \notin \text{vars}'(p_C) = \text{vars}'(C, \mu)$, it follows that C cannot include $\text{chaos}(\mu, z)$, which given $z \in W$ means $z \notin \text{vars}'(H, \mu) \supseteq \text{vars}'(p_H)$ by construction of C. Thus, $z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H))$, which implies $\hat{v}(z) = \hat{w}(z)$ by (3.29), and the claim (3.28) follows by Lemma 2.

Second, it follows from Lemma 15 that $(x_G, c, x_S, \hat{v}\restriction_W) \in Q^{\text{acc}}((\exists \bar{W} G) \parallel C \parallel S)$.

Then, given (3.28) and $\text{vars}(S) \subseteq W$ and because S is Σ_{u}-controllable with respect to $(\exists \bar{W} G) \parallel C$, by Definition 7 there exists a location y_S of S such that

$$(x_G, c, x_S, \hat{v}\restriction_W) \xrightarrow{\mu} (y_G, c, y_S, \hat{w}\restriction_W) \quad \text{in } (\exists \bar{W} G) \parallel C \parallel S \ . \tag{3.30}$$

By Lemma 2, this implies $x_S \xrightarrow{\mu:p_S} y_S$ with $(\hat{v}\restriction_W \oplus (\hat{w}\restriction_W)')(p_S) = \text{true}$ in S. Furthermore, for any variable $z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H) \cup \text{vars}'(p_S)) \subseteq V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H))$, it holds that $\hat{v}(z) = \hat{w}(z)$ by (3.29). Then the claim (3.27) follows by combining this with the above observations about transitions in G, H, and S, using Lemma 2. □

3.4 Proof of Least Restrictiveness

The third and last step towards of proof of (3.8) is to show that every iteration of Algorithm 1 produces a least restrictive supervisor with respect to the original plant. This is the most difficult part of the proof, which depends on the precise conditions (3.2)–(3.5) how the variables are selected and the plant abstraction is formed.

When synthesising for ordinary FSMs, least restrictiveness can be ensured by including all plant components that may disable an uncontrollable event [1, 10]. This is generalised for EFSMs by requiring that the uncontrollable events are always enabled by the plant EFSMs not included in the abstraction (3.3), and that the uncontrollable events are unconstrained by the variables not included in the abstraction (3.4).

The least restrictiveness result is shown in Proposition 18 below, and depends on the following Lemma 17 about the existence of states in the synthesis result for an abstraction. The lemma shows under the assumptions (3.2)–(3.5) that every state encountered during synthesis of the original system, corresponds to a state encountered during synthesis for the plant abstraction.

Lemma 17 Let G, H, and E be normalised EFSMs with $V = \text{vars}(G) \cup \text{vars}(H) \cup \text{vars}(E)$ and $\Sigma_{\text{u}} \subseteq \Sigma$. Let $W \subseteq V$ and $\bar{W} = V \setminus W$ such that $\text{vars}(E) \subseteq W \subseteq \text{vars}(G) \cup \text{vars}(E)$, and Σ_{u} is unconstrained in G with respect to $\bar{W}$, and Σ_{u} is always enabled in H with respect to $\bar{W}$, and $\text{vars}'(G, \mu) \cap \text{vars}'(H, \mu) \cap \bar{W} = \emptyset$ for each $\mu \in \Sigma_{\text{u}}$, and let $C = \big\|(\{\, \text{chaos}(\sigma, z) \mid$

$z \in W \cap \mathrm{vars}'(H,\sigma)\,\}$). If $(x_G, x_H, x_E, \hat{v}) \in \hat{\Theta}_{G\|H,E,\Sigma_\mathrm{u}}$ then $(x_G, c, x_E, \hat{v}\restriction_W) \in \hat{\Theta}_{(\exists \bar{V} G)\|C,E,\Sigma_\mathrm{u}}$ where c is the single location of C.

Proof. Write $Q = Q_G \times \{c\} \times Q_E \times \mathrm{dom}(W)$, and assume $(x_G, x_H, x_E, \hat{v}) \in \hat{\Theta}_{G\|H,E,\Sigma_\mathrm{u}}$ for some $\hat{v} \in \mathrm{dom}(V)$. It is shown by induction on j that

$$(x_G, c, x_E, \hat{v}\restriction_W) \in \Theta^j_{(\exists \bar{W} G)\|C,E,\Sigma_\mathrm{u}}(Q) \tag{3.31}$$

for all $j \geq 0$.

For the base case, $j = 0$, note that $(x_G, c, x_E, \hat{v}\restriction_W) \in Q = \Theta^0_{(\exists \bar{W} G)\|C,E,\Sigma_\mathrm{u}}(Q)$.

To show the claim (3.31) for $j+1$, assume that there is some $j \geq 0$ such that (3.31) holds for all $(x_G, x_H, x_E, \hat{v}) \in \hat{\Theta}_{G\|H,E,\Sigma_\mathrm{u}}$. Note that

$$\Theta^{j+1}_{(\exists \bar{W} G)\|C,E,\Sigma_\mathrm{u}}(Q) = \Theta_{(\exists \bar{W} G)\|C,E,\Sigma_\mathrm{u}}(\Theta^j_{(\exists \bar{W} G)\|C,E,\Sigma_\mathrm{u}}(Q)) \ . \tag{3.32}$$

Following Definition 9, to show that $(x_G, c, x_E, \hat{v}\restriction_W)$ is contained in this set, assume

$$(x_G, c, \hat{v}\restriction_W) \xrightarrow{\mu} (y_G, c, \tilde{w}) \quad \text{in } (\exists \bar{W} G) \parallel C \tag{3.33}$$

for some $\mu \in \Sigma_\mathrm{u}$ and $\tilde{w} \in \mathrm{dom}(W)$. This means by Lemma 2 that $x_G \xrightarrow{\mu:\tilde{p}_G} y_G$ in $\exists \bar{W} G$ with $(\hat{v}\restriction_W \oplus \tilde{w}')(\tilde{p}_G) = \mathrm{true}$, and $c \xrightarrow{\mu:p_C} c$ in C with $(\hat{v}\restriction_W \oplus \tilde{w}')(p_C) = \mathrm{true}$, and

$$\hat{v}\restriction_W(z) = \tilde{w}(z) \quad \text{for all variables } z \in W \setminus (\mathrm{vars}'(\tilde{p}_G) \cup \mathrm{vars}'(p_C)) \ . \tag{3.34}$$

Here, $x_G \xrightarrow{\mu:\tilde{p}_G} y_G$ in $\exists \bar{W} G$ means that $x_G \xrightarrow{\mu:p_G} y_G$ in G where $\tilde{p}_G \equiv \exists \bar{W} p_G$. Thus, $(\hat{v}\restriction_W \oplus \tilde{w}')(\exists \bar{W} p_G) = (\hat{v}\restriction_W \oplus \tilde{w}')(\tilde{p}_G) = \mathrm{true}$. Since $\mu \in \Sigma_\mathrm{u}$ is unconstrained in G with respect to $\bar{W}$, the formula $\exists \bar{W} p_G \Rightarrow \forall \bar{W} \exists \bar{W}' p_G$ is valid. Then $(\hat{v}\restriction_W \oplus \tilde{w}')(\forall \bar{W} \exists \bar{W}' p_G) = \mathrm{true}$ and thus $(\hat{v} \oplus \tilde{w}')(\exists \bar{W}' p_G) = \mathrm{true}$, and also $(\hat{v} \oplus \tilde{w}')(\exists \bar{W}'_G p_G) = \mathrm{true}$ where $\bar{W}_G = \mathrm{vars}'(p_G) \cap \bar{W}$. Then there exists $\tilde{u}_G \in \mathrm{dom}(\bar{W}_G)$ such that $(\hat{v} \oplus (\tilde{u}_G \oplus \tilde{w})')(p_G) = \mathrm{true}$.

As $\mu \in \Sigma_\mathrm{u}$ is always enabled in H with respect to $\bar{W}$, by Definition 13 there are transitions

$$x_H \xrightarrow{\mu:p^1_H} y^1_H \quad \cdots \quad x_H \xrightarrow{\mu:p^n_H} y^n_H \qquad \text{in } H \tag{3.35}$$

such that the formula $\exists \bar{W}'(p^1_H \vee \cdots \vee p^n_H)$ is valid. Then, $(\hat{v} \oplus (\tilde{u}_G \oplus \tilde{w})')(\exists \bar{W}' p^1_H \vee \cdots \vee \exists \bar{W}' p^n_H)) = (\hat{v} \oplus (\tilde{u}_G \oplus \tilde{w})')(\exists \bar{W}'(p^1_H \vee \cdots \vee p^n_H)) = \mathrm{true}$, which means $(\hat{v} \oplus (\tilde{u}_G \oplus \tilde{w})')(\exists \bar{W}' p^i_H) = \mathrm{true}$ for some i. Let $p_H \equiv p^i_H$ and $y_H = y^i_H$ and $\bar{W}_H = \mathrm{vars}'(p_H) \cap \bar{W}$. Then $(\hat{v} \oplus (\tilde{u}_G \oplus \tilde{w})')(\exists \bar{W}'_H p_H) = \mathrm{true}$, and there exists a valuation $\tilde{u}_H \in \mathrm{dom}(\bar{W}_H)$ such that $(\hat{v} \oplus (\tilde{u}_H \oplus \tilde{u}_G \oplus \tilde{w})')(p_H) = \mathrm{true}$.

Note that $\bar{W}_G \cap \bar{W}_H = \mathrm{vars}'(p_G) \cap \bar{W} \cap \mathrm{vars}'(p_H) \cap \bar{W} \subseteq \mathrm{vars}'(G,\mu) \cap \mathrm{vars}'(H,\mu) \cap \bar{W} = \emptyset$ by assumption. As also $\tilde{w} \in \mathrm{dom}(W)$, with $W \cap \bar{W} = \emptyset$, the valuations $\tilde{w}$, $\tilde{u}_G$, and $\tilde{u}_H$ do not share any variables, so that $\tilde{u}_H \oplus \tilde{u}_G \oplus \tilde{w} \geq \tilde{u}_G \oplus \tilde{w} \geq \tilde{w}$.

Let

$$\hat{w} = \tilde{u}_H \oplus \tilde{u}_G \oplus \tilde{w} \oplus \hat{v}\restriction_U \quad \text{where} \quad U = V \setminus (W \cup \bar{W}_G \cup \bar{W}_H). \tag{3.36}$$

Then $x_G \xrightarrow{\mu:p_G} y_G$ with $(\hat{v} \oplus \hat{w}')(p_G) = (\hat{v} \oplus (\tilde{u}_G \oplus \tilde{w})')(p_G) = \text{true}$ and $x_H \xrightarrow{\mu:p_H} y_H$ with $(\hat{v} \oplus \hat{w}')(p_H) = (\hat{v} \oplus (\tilde{u}_H \oplus \tilde{u}_G \oplus \tilde{w})')(p_H) = \text{true}$. Now consider $z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H))$. Then $z \in V$, and $z \notin \text{vars}'(p_G) \supseteq \text{vars}'(\exists \bar{W} p_G)$ and $z \notin \text{vars}'(p_H) = \text{vars}'(H, \mu) \supseteq \text{vars}'(C, \mu) = \text{vars}'(p_C)$ because H is normalised. If $z \in W$, then $z \in W \setminus (\text{vars}'(\exists \bar{W} p_G) \cup \text{vars}'(p_C))$ and thus $\hat{v}(z) = \hat{v}\restriction_W(z) = \tilde{w}(z) = \hat{w}(z)$ by (3.34). If $z \notin W$, then $z \in V \setminus (W \cup \text{vars}'(p_G) \cup \text{vars}'(p_H)) \subseteq V \setminus (W \cup (\text{vars}'(p_G) \cap \bar{W}) \cup (\text{vars}'(p_H) \cap \bar{W})) = V \setminus (W \cup \bar{W}_G \cup \bar{W}_H) = U$ and thus $\hat{v}(z) = \hat{v}\restriction_U(z) = \hat{w}(z)$ by (3.36). This shows

$$\hat{v}(z) = \hat{w}(z) \text{ for all } z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H)). \tag{3.37}$$

It follows by Lemma 2 that

$$(x_G, x_H, \hat{v}) \xrightarrow{\mu} (y_G, y_H, \hat{w}) \text{ in } G \parallel H. \tag{3.38}$$

As $(x_G, x_H, x_E, \hat{v}) \in \hat{\Theta}_{G\parallel H, E, \Sigma_u} = \Theta_{G\parallel H, E, \Sigma_u}(\hat{\Theta}_{G\parallel H, E, \Sigma_u})$ it follows from Definition 9 that there exists a location y_E of E such that $(x_G, x_H, x_E, \hat{v}) \xrightarrow{\mu} (y_G, y_H, y_E, \hat{w})$ in $G \parallel H \parallel E$ and $(y_G, y_H, y_E, \hat{w}) \in \hat{\Theta}_{G\parallel H, E, \Sigma_u}$. This also means $x_E \xrightarrow{\mu:p_E} y_E$ in E with $(\hat{v} \oplus \hat{w}')(p_E) = \text{true}$.

Then $x_G \xrightarrow{\mu:\tilde{p}_G} y_G$ in $\exists \bar{W} G$ with $(\hat{v}\restriction_W \oplus \tilde{w}')(\tilde{p}_G) = \text{true}$, and $c \xrightarrow{\mu:p_C} c$ in C with $(\hat{v}\restriction_W \oplus \tilde{w}')(p_C) = \text{true}$ because $\text{vars}(p_C) \subseteq \text{vars}(C) \subseteq W$, and $x_E \xrightarrow{\mu:p_E} y_E$ in E with $(\hat{v}\restriction_W \oplus \tilde{w}')(\tilde{p}_E) = (\hat{v} \oplus \hat{w}')(p_E) = \text{true}$ because $\text{vars}(p_E) \subseteq \text{vars}(E) \subseteq W$ and $\hat{w} \geq \tilde{w}$.

Consider $z \in W \setminus (\text{vars}'(\tilde{p_G}) \cup \text{vars}'(p_C) \cup \text{vars}'(p_E))$. Then $z \in W \subseteq V$, and $z \notin \text{vars}'(\tilde{p_G}) = \text{vars}'(\exists \bar{W} p_G) = \text{vars}'(p_G) \setminus \bar{W} = \text{vars}'(p_G) \cap W$ so that $z \notin \text{vars}'(p_G)$ as $z \in W$, and $z \notin \text{vars}'(p_C) = \text{vars}'(C, \mu) = W \cap \text{vars}'(H, \mu)$ so that $z \notin \text{vars}'(H, \mu) = \text{vars}'(p_H)$ as $z \in W$ and H is normalised. So $z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_H))$ and it follows by (3.37) that $\hat{v}\restriction_W(z) = \hat{v}(z) = \hat{w}(z) = \tilde{w}(z)$.

It follows from Lemma 2 that $(x_G, c, x_E, \hat{v}\restriction_W) \xrightarrow{\mu} (y_G, c, y_E, \tilde{w})$ in $G \parallel H \parallel E$ where $(y_G, c, y_E, \tilde{w}) = (y_G, c, y_E, \hat{w}\restriction_W) \in \Theta^j_{\exists \bar{W} G \parallel C, E, \Sigma_u}(Q)$ by inductive assumption as $(y_G, y_H, y_E, \hat{w}) \in \hat{\Theta}_{G\parallel H, E, \Sigma_u}$. The claim $(x_G, c, x_E, \hat{v}\restriction_W) \in \Theta^{j+1}_{\exists \bar{W} G \parallel C, E, \Sigma_u}(Q)$ follows from Definition 9. □

According to Lemma 17, the synthesis fixpoint for the original system is somehow contained in the synthesis fixpoint computed using an abstraction. This observation forms the base for the proof of least restrictiveness in the following Proposition 18, which depends on the same assumptions (3.2)–(3.5) as Lemma 17. The proof lifts the result about the inclusion of the synthesis fixpoints to show that any controllable supervisor for the original plant and specification results in behaviour that is included in that of a system controlled by the supervisor computed using an abstraction.

Proposition 18 Let G, H, and E be normalised EFSMs, E pure, with $V = \mathrm{vars}(G) \cup \mathrm{vars}(H) \cup \mathrm{vars}(E)$, and $\Sigma_\mathrm{u} \subseteq \Sigma$. Let $W \subseteq V$ and $\bar{W} = V \setminus W$ such that $\mathrm{vars}(E) \subseteq W \subseteq \mathrm{vars}(G) \cup \mathrm{vars}(E)$, and Σ_u is unconstrained in G with respect to $\bar{W}$, and Σ_u is always enabled in H with respect to $\bar{W}$, and $\mathrm{vars}'(G,\mu) \cap \mathrm{vars}'(H,\mu) \cap \bar{W} = \emptyset$ for each $\mu \in \Sigma_\mathrm{u}$, and let $C = \big\|(\{\, \mathrm{chaos}(\sigma, z) \mid z \in W \cap \mathrm{vars}'(H,\sigma) \,\})$. Let S' be an EFSM such that $G \parallel H \parallel S' \subseteq_\mathrm{v} G \parallel H \parallel E$ and S' is Σ_u-controllable with respect to $G \parallel H$. Then

$$G \parallel H \parallel S' \subseteq_\mathrm{v} G \parallel H \parallel \mathrm{sup}\mathcal{C}((\exists \bar{W} G) \parallel C, E, \Sigma_\mathrm{u}) \; . \tag{3.39}$$

Proof. Write $S = \mathrm{sup}\mathcal{C}(G \parallel H, E, \Sigma_\mathrm{u})$ and $S_\exists = \mathrm{sup}\mathcal{C}((\exists \bar{W} G) \parallel C, E, \Sigma_\mathrm{u})$. Consider a path

$$(\ddot{x}_G^0, \ddot{x}_H^0, \ddot{x}_S^0, \hat{v}^0) \xrightarrow{\sigma_0} \cdots \xrightarrow{\sigma_\eta} (\ddot{x}_G^n, \ddot{x}_H^n, \ddot{x}_S^n, \hat{v}^n). \tag{3.40}$$

in $G \parallel H \parallel S'$ with $\hat{v}^j \in \mathrm{dom}(\ddot{V})$ and $\ddot{V} = V \cup \mathrm{vars}(S')$. Since $G \parallel H \parallel S' \subseteq_\mathrm{v} G \parallel H \parallel S$ by Proposition 11, there exists a path

$$(x_G^0, x_H^0, (\tilde{x}_G^0, \tilde{x}_H^0, x_E^0), \hat{v}^0) \xrightarrow{\sigma_0} \cdots \xrightarrow{\sigma_\eta} (x_G^n, x_H^n, (\tilde{x}_G^n, \tilde{x}_H^n, x_E^n), \hat{v}^n) \tag{3.41}$$

in $G \parallel H \parallel S$. It is shown by induction on n that, for all $n \geq 0$,

$$(\tilde{x}_G^n, \tilde{x}_H^n, x_E^n, \hat{v}^n{\restriction_V}) \in Q^\mathrm{acc}(S) \tag{3.42}$$

and

$$(x_G^0, c, (\tilde{x}_G^0, c, x_E^0), \hat{v}^0) \xrightarrow{\sigma_0} \cdots \xrightarrow{\sigma_\eta} (x_G^n, c, (\tilde{x}_G^n, c, x_E^n), \hat{v}^n). \tag{3.43}$$

is a path in $G \parallel H \parallel S_\exists$ where c is the single location of C.

For the base case, $n = 0$, it is clear that $(\tilde{x}_G^0, \tilde{x}_H^0, x_E^0, \hat{v}^0{\restriction_V})$ is initial in S by (3.41), which already shows (3.42). It follows by Lemma 1 that $(\tilde{x}_G^0, \tilde{x}_H^0, x_E^0, \hat{v}^0{\restriction_V}) \in Q^\mathrm{acc}(S) \subseteq \hat{\Theta}_{G \parallel H, E, \Sigma_\mathrm{u}}$, which implies by Lemma 17 that $(\tilde{x}_G^0, c, x_E^0, \hat{v}^0{\restriction_W}) \in \hat{\Theta}_{(\exists \bar{V} G) \parallel C, E, \Sigma_\mathrm{u}}$. Note that x_G^0, x_H^0, $\tilde{x}_G^0$, x_E^0, and $\hat{v}^0$ are initial locations and variable values from (3.41), so $(\tilde{x}_G^0, c, x_E^0)$ is initial in $S_\exists$ by Definition 4, and then

$$(x_G^0, x_H^0, (\tilde{x}_G^0, c, x_E^0), \hat{v}^0) \tag{3.44}$$

is a path in $G \parallel H \parallel S_\exists$.

Assume that (3.42) and (3.43) have been shown for some n, and consider the $(n+1)$-th transition of the path (3.41) in $G \parallel H \parallel S$,

$$(x_G^n, x_H^n, (\tilde{x}_G^n, \tilde{x}_H^n, x_E^n), \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_H^{n+1}, (\tilde{x}_G^{n+1}, \tilde{x}_H^{n+1}, x_E^{n+1}), \hat{v}^{n+1}) \tag{3.45}$$

This means by Lemma 2 that there are transitions

$$x_G^n \xrightarrow{\sigma_{n+1} : p_G} x_G^{n+1} \qquad \text{in } G \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_G) = \mathrm{true} \; ; \tag{3.46}$$

$$x_H^n \xrightarrow{\sigma_{n+1} : p_H} x_H^{n+1} \qquad \text{in } H \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_H) = \mathrm{true} \; ; \tag{3.47}$$

$$(\tilde{x}_G^n, \tilde{x}_H^n, x_E^n) \xrightarrow{\sigma_{n+1} : p_S} (\tilde{x}_G^{n+1}, \tilde{x}_H^{n+1}, x_E^{n+1}) \qquad \text{in } S \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_S) = \mathrm{true} \; ; \tag{3.48}$$

and furthermore

$$\hat{v}^n(z) = \hat{v}^{n+1}(z) \quad \text{for all } z \in \ddot{V} \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H) \cup \mathrm{vars}'(p_S)) \ . \tag{3.49}$$

The update of the transition (3.48) takes the form $p_S \equiv q_G \wedge q_H \wedge p_E \wedge R$ with $\mathrm{vars}'(R) \subseteq \mathrm{vars}'(q_G \wedge q_H \wedge p_E)$, for some transitions $\tilde{x}_G^n \xrightarrow{\sigma_{n+1}:q_G} \tilde{x}_G^{n+1}$ in G and $\tilde{x}_H^n \xrightarrow{\sigma_{n+1}:q_H} \tilde{x}_H^{n+1}$ in H and $x_E^n \xrightarrow{\sigma_{n+1}:p_E} x_E^{n+1}$ in E. Since G and H are normalised and E is pure, it follows that

$$\begin{aligned}\mathrm{vars}'(p_S) &= \mathrm{vars}'(q_G \wedge q_H \wedge p_E) \cup \mathrm{vars}'(p_R) = \mathrm{vars}'(q_G \wedge q_H \wedge p_E) \\ &= \mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_H) \cup \mathrm{vars}'(p_E) = \mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_H) \\ &= \mathrm{vars}'(G, \sigma_{n+1}) \cup \mathrm{vars}'(H, \sigma_{n+1}) = \mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H) \ . \end{aligned} \tag{3.50}$$

Then (3.49) means $\hat{v}^n(z) = \hat{v}^{n+1}(z)$ for all variables $z \in \ddot{V} \setminus \mathrm{vars}'(p_S)$. Then it can be said because of (3.48) that $(\tilde{x}_G^n, \tilde{x}_H^n, x_E^n, \hat{v}^n\restriction_V) \xrightarrow{\sigma_{n+1}} (\tilde{x}_G^{n+1}, \tilde{x}_H^{n+1}, x_E^{n+1}, \hat{v}^{n+1}\restriction_V)$ in S. As $(\tilde{x}_G^n, \tilde{x}_H^n, x_E^n, \hat{v}^n\restriction_V) \in Q^{\mathrm{acc}}(S)$ by inductive assumption (3.42), it follows that $(\tilde{x}_G^{n+1}, \tilde{x}_H^{n+1}, x_E^{n+1}, \hat{v}^{n+1}\restriction_V) \in Q^{\mathrm{acc}}(S)$, showing the inductive claim (3.42) for $n+1$. Also $(\tilde{x}_G^{n+1}, \tilde{x}_H^{n+1}, x_E^{n+1}, \hat{v}^{n+1}\restriction_V) \in Q^{\mathrm{acc}}(S) \subseteq \hat{\Theta}_{G \| H, E, \Sigma_\mathrm{u}}$ by Lemma 1, which implies by Lemma 17 that

$$(\tilde{x}_G^{n+1}, c, x_E^{n+1}, \hat{v}^{n+1}\restriction_W) \in \hat{\Theta}_{(\exists \bar{V} G) \| C, E, \Sigma_\mathrm{u}} \ . \tag{3.51}$$

Further, it follows from $(\hat{v}^n \oplus (\hat{v}^{n+1})')(q_G \wedge q_H \wedge p_E \wedge R) = (\hat{v}^n \oplus (\hat{v}^{n+1})')(p_S) = \mathrm{true}$ that $(\hat{v}^n \oplus (\hat{v}^{n+1})')(\exists \bar{W} q_G) = \mathrm{true}$ and $(\hat{v}^n \oplus (\hat{v}^{n+1})')(p_E) = \mathrm{true}$. As $\tilde{x}_G^n \xrightarrow{\sigma_{n+1}:q_G} \tilde{x}_G^{n+1}$ in G, there is a transition $\tilde{x}_G^n \xrightarrow{\sigma_{n+1}:\exists \bar{W} q_G} \tilde{x}_G^{n+1}$ in $\exists \bar{W} G$. Clearly $c \xrightarrow{\sigma_{n+1}:p_C} c$ in C with $(\hat{v}^n \oplus (\hat{v}^{n+1})')(p_C) = \mathrm{true}$ by construction of C. Let $p \equiv (\exists \bar{W} q_G) \wedge p_C \wedge p_E$, so that $(\hat{v}^n \oplus (\hat{v}^{n+1})')(p) = \mathrm{true}$. By Definition 4, the following transition exists in $S_\exists = \sup \mathcal{C}((\exists \bar{W} G) \parallel C, E, \Sigma_\mathrm{u})$,

$$(\tilde{x}_G^n, c, x_E^n) \xrightarrow{\sigma_{n+1}:p \wedge R_\exists} (\tilde{x}_G^{n+1}, c, x_E^{n+1}) \tag{3.52}$$

where $R_\exists \equiv R_{\hat{\Theta}_{(\exists \bar{V} G) \| C, E, \Sigma_\mathrm{u}}}[p, (\tilde{x}_G^{n+1}, c, x_E^{n+1})]$ is such that $\mathrm{vars}'(R_\exists) \subseteq \mathrm{vars}'(p)$ and for all valuations $\hat{v}, \hat{w} \in \mathrm{dom}(W)$ it holds that $(\tilde{x}_G^{n+1}, c, x_E^{n+1}, \hat{v}\restriction_{W \setminus \mathrm{vars}'(p)} \oplus \hat{w}) \in \hat{\Theta}_{(\exists \bar{V} G) \| C, E, \Sigma_\mathrm{u}}$ if and only if $(\hat{v} \oplus \hat{w}')(R_\exists) = \mathrm{true}$.

Now consider a variable

$$z \in W \setminus \mathrm{vars}'(p) = W \setminus (\mathrm{vars}'(\exists \bar{W} q_G) \cup \mathrm{vars}'(p_C) \cup \mathrm{vars}'(p_E)) \ . \tag{3.53}$$

Then $z \in W \subseteq \ddot{V}$ and $z \notin \mathrm{vars}'(\exists \bar{W} q_G) = \mathrm{vars}'(q_G) \cap W$, i.e., $z \notin \mathrm{vars}'(q_G) = \mathrm{vars}'(G, \sigma_{n+1}) = \mathrm{vars}'(p_G)$, and $z \notin \mathrm{vars}'(p_C) = \mathrm{vars}'(C, \sigma_{n+1}) \supseteq \mathrm{vars}'(H, \sigma_{n+1}) = \mathrm{vars}'(p_H)$. Thus, $z \in \ddot{V} \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H)) = \ddot{V} \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H) \cup \mathrm{vars}'(p_S))$, and it follows that $\hat{v}^n(z) = \hat{v}^{n+1}(z)$ by (3.49). This means $\hat{v}^{n+1}\restriction_W = \hat{v}^n\restriction_{W \setminus \mathrm{vars}'(p)} \oplus \hat{v}^{n+1}\restriction_W$, so that $(\tilde{x}_G^{n+1}, c,$

$x_E^{n+1}, \hat{v}^n\restriction_{W\setminus \mathrm{vars}'(p)} \oplus \hat{v}^{n+1}\restriction_W) = (\tilde{x}_G^{n+1}, c, x_E^{n+1}, \hat{v}^{n+1}\restriction_W) \in \hat{\Theta}_{(\exists \bar{V} G)\parallel C,E,\Sigma_\mathrm{u}}$ by (3.51). Then it follows from the construction of $R_\exists$ that $(\hat{v}^n \oplus (\hat{v}^{n+1})')(R_\exists) = \mathrm{true}$.

Combining this with (3.46), (3.47), and (3.52), there is a transition

$$(x_G^n, x_H^n, (\tilde{x}_G^n, c, x_E^n)) \xrightarrow{\sigma_{n+1}:p_G \wedge p_H \wedge p \wedge R_\exists} (x_G^{n+1}, x_H^{n+1}, (\tilde{x}_G^{n+1}, c, x_E^{n+1})) \quad \text{in } G \parallel H \parallel S_\exists \tag{3.54}$$

such that $(\hat{v}^n \oplus (\hat{v}^{n+1})')(p_G \wedge p_H \wedge p \wedge R_\exists) = \mathrm{true}$. Lastly, consider a variable $z \in \ddot{V} \setminus (\mathrm{vars}'(p_G \wedge p_H \wedge p \wedge R_\exists)) \subseteq \ddot{V} \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H)) = \ddot{V} \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_H) \cup \mathrm{vars}'(p_S))$ using (3.50). It follows that $\hat{v}^n(z) = \hat{v}^{n+1}(z)$ by (3.49) and

$$(x_G^n, x_H^n, (\tilde{x}_1^n, c, x_E^n), \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_H^{n+1}(\tilde{x}_1^{n+1}, c, x_E^{n+1}), \hat{v}^{n+1}) \quad \text{in } G \parallel H \parallel S_\exists\ . \tag{3.55}$$

This adds the $(n+1)$-th step to the path (3.43) and completes the induction. The claim (3.39) follows by Definition 6 from (3.43). □

3.5 Correctness Proof of Algorithm 1

This section combines the preceding results from Propositions 14, 16, and 18 to show that Algorithm 1 correctly computes a supremal supervisor for an input specification E and a set of plants $\mathcal{G}$. It can be shown that the algorithm's main loop maintains the following invariant: before and after each iteration,

$$S^i \text{ is a supremal supervisor for } E \text{ with respect to } \Big\|(\mathcal{G}) \text{ and } \Sigma_\mathrm{u}^i \tag{3.56}$$

and

$$\Big\|(\mathcal{G}) \parallel \sup\mathcal{C}(\Big\|(\mathcal{G}), E, \Sigma_\mathrm{u}) \subseteq_\mathrm{v} \Big\|(\mathcal{G}) \parallel S^i\ . \tag{3.57}$$

That is, each iteration results in a correct solution S^i based on the full plant $\mathcal{G}$ and the reduced uncontrollable event set Σ_u^i (3.56), which also is an over-approximation of the supremal supervisor based on the full event set Σ_u (3.57). As on exit of the loop, the computed supervisor S^i is also Σ_u-controllable with respect to all uncontrollable events, it can then be shown to be the desired result.

Next, Proposition 19 shows that the loop invariant holds initially for $S^0 = E$ and $\Sigma_\mathrm{u}^0 = \emptyset$. Afterwards, Proposition 20 shows that (3.56) holds after each iteration, and Proposition 21 shows that (3.57) holds after each iteration, both of which are consequences of Propositions 14, 16, and 18.

Proposition 19 Let G and E be state-deterministic EFSMs such that G is normalised, and let Σ_u be a set of events.

(i) E is a supremal supervisor for E with respect to G and $\emptyset$.

(ii) $G \parallel \sup\mathcal{C}(G, E, \Sigma_\mathrm{u}) \subseteq_\mathrm{v} G \parallel E$.

Proof.

(i) It is to be shown that E satisfies conditions (i)–(iii) of Definition 8. Obviously, $G \parallel E \subseteq_{\mathrm{v}} G \parallel E$, showing (i). Also, it follows from Definition 7 that E is trivially $\emptyset$-controllable, showing (ii). Lastly, let S' be an EFSM such that $G \parallel S' \subseteq_{\mathrm{v}} G \parallel E$. Then S' already satisfies (iii).

(ii) As $\sup\mathcal{C}(G, E, \Sigma_{\mathrm{u}})$ is a supremal supervisor for E with respect to G by Proposition 11, it is clear by Definition 8 (i) that $G \parallel \sup\mathcal{C}(G, E, \Sigma_{\mathrm{u}}) \subseteq_{\mathrm{v}} G \parallel E$. □

Proposition 20 After each iteration of the loop in Algorithm 1, it holds that S^i is a supremal supervisor for E with respect to $\big\|(\mathcal{G})$ and Σ_{u}^i.

Proof. It is clear from lines 14 and 13 that

$$S^i = \sup\mathcal{C}\big(\, \big\|(\exists \bar{V}^i \mathcal{G}^i) \parallel \big\|(\mathcal{C}^i), E, \Sigma_{\mathrm{u}}^i\big) \;; \tag{3.58}$$

$$\big\|(\mathcal{C}^i) = \big\|(\{\, \mathrm{chaos}(\sigma, z) \mid z \in V^i \cap \mathrm{vars}'(\bar{\mathcal{G}}^i, \sigma) \,\}) \;. \tag{3.59}$$

It is to be shown that S^i satisfies the conditions for a supremal supervisor according to Definition 8.

(i) The preconditions of Proposition 14 are satisfied for $G = \big\|(\mathcal{G}^i)$, $H = \big\|(\bar{\mathcal{G}}^i)$, $\Sigma_{\mathrm{u}} = \Sigma_{\mathrm{u}}^i$, $W = V^i$, $\bar{W} = \bar{V}^i$, $C = \big\|(\mathcal{C}^i)$, and $S = S^i$. Then Proposition 14 gives $\big\|(\mathcal{G}) \parallel S^i = G \parallel H \parallel S \subseteq_{\mathrm{v}} G \parallel H \parallel E = \big\|(\mathcal{G}) \parallel E$.

(ii) The preconditions of Proposition 16 are satisfied for $G = \big\|(\mathcal{G}^i)$, $H = \big\|(\bar{\mathcal{G}}^i)$, $S = S^i$, $\Sigma_{\mathrm{u}} = \Sigma_{\mathrm{u}}^i$, $W = V^i$, $\bar{W} = \bar{V}^i$, and $C = \big\|(\mathcal{C}^i)$. Note that $\mathrm{vars}(S) = \mathrm{vars}(S^i) \subseteq \mathrm{vars}(\exists \bar{V}^i \mathcal{G}^i) \cup \mathrm{vars}(\mathcal{C}^i) \cup \mathrm{vars}(E) = (\mathrm{vars}(\mathcal{G}^i) \cap V^i) \cup \mathrm{vars}(\mathcal{C}^i) \cup \mathrm{vars}(E) \subseteq V^i = W$ by (3.58) and (3.59) and because $\mathrm{vars}(E) \subseteq V^i$ from line 14 of Algorithm 1. Also $S = S^i$ is Σ_{u}^i-controllable with respect to $\big\|(\exists \bar{V}^i \mathcal{G}^i) \parallel \big\|(\mathcal{C}^i) = (\exists \bar{W} G) \parallel C$ by (3.58) and Definition 8 (ii), so by Proposition 16 it follows that $S^i = S$ is Σ_{u}^i-controllable with respect to $G \parallel H = \big\|(\mathcal{G})$.

(iii) Let S' be an EFSM that satisfies (i) and (ii). i.e., $G \parallel H \parallel S' \subseteq_{\mathrm{v}} G \parallel H \parallel E$ and S' is Σ_{u}-controllable with respect to $G \parallel H$. Line 14 of Algorithm 1 ensures that the preconditions of Proposition 18 are satisfied for $G = \big\|(\mathcal{G}^i)$, $H = \big\|(\bar{\mathcal{G}}^i)$, $\Sigma_{\mathrm{u}} = \Sigma_{\mathrm{u}}^i$, $W = V^i$, $\bar{W} = \bar{V}^i$, and $C = \big\|(\mathcal{C}^i)$. Then Proposition 18 gives $\big\|(\mathcal{G}) \parallel S' = G \parallel H \parallel S' \subseteq_{\mathrm{v}} G \parallel H \parallel \sup\mathcal{C}(\exists \bar{W} G \parallel C, E, \Sigma_{\mathrm{u}}^i) = \big\|(\mathcal{G}) \parallel \sup\mathcal{C}\big(\, \big\|(\exists \bar{V}^i \mathcal{G}^i) \parallel \big\|(\mathcal{C}^i), E, \Sigma_{\mathrm{u}}^i\big) = \big\|(\mathcal{G}) \parallel S^i$. □

Proposition 21 After each iteration of the loop in Algorithm 1, it holds that

$$\big\|(\mathcal{G}) \parallel \sup\mathcal{C}(\big\|(\mathcal{G}), E, \Sigma_{\mathrm{u}}) \subseteq_{\mathrm{v}} \big\|(\mathcal{G}) \parallel S^i \;. \tag{3.60}$$

Proof. Write $S = \sup\mathcal{C}(\|(\mathcal{G}), E, \Sigma_u)$. S is a supremal supervisor for E with respect to $\|(\mathcal{G})$ and Σ_u by Proposition 11, i.e., by Definition 8 (i) $\|(\mathcal{G}) \parallel S \subseteq_v \|(\mathcal{G}) \parallel E$ and (ii) S is Σ_u-controllable with respect to $\|(\mathcal{G})$. Noting that $\Sigma_u^i \subseteq \Sigma_u$, it follows from Definition 7 that S is also Σ_u^i-controllable with respect to $\|(\mathcal{G})$. As S^i is a supremal supervisor for E with respect to $\|(\mathcal{G})$ and Σ_u^i by Proposition 20, the claim (3.60) follows from Definition 8 (iii). □

Propositions 19–21 confirm that the loop invariant consisting of (3.56) and (3.57) holds before and after each iteration of the main loop of Algorithm 1. When the loop terminates, the loop-entry condition on line 10 no longer holds, i.e., the computed supervisor S^i is Σ_u-controllable with respect to the current plant abstraction, using the full set of uncontrollable events. This is now enough to show that this result is also a supremal solution with respect to the entire plant and entire set of uncontrollable events.

Theorem 22 Upon termination of Algorithm 1, the result S^i is a supremal supervisor for E with respect to $\|(\mathcal{G})$ and Σ_u.

Proof. It is to be shown that S^i satisfies the conditions for a supremal supervisor according to Definition 8.

(i) As each S^i is a supremal supervisor for E with respect to $\|(\mathcal{G})$ and Σ_u^i by Propositions 19 and 20, it is clear from Definition 8 (i) that $\|(\mathcal{G}) \parallel S^i \subseteq_v \|(\mathcal{G}) \parallel E$.

(ii) From line 10 of Algorithm 1, it is clear that S^i is Σ_u-controllable with respect to $\|(\exists \bar{V} \mathcal{G}^i) \parallel \|(\mathcal{C}^i)$ upon termination of the loop. Note that $\mathrm{vars}(S^i) \subseteq \mathrm{vars}(\exists \bar{V}^i \mathcal{G}^i) \cup \mathrm{vars}(\mathcal{C}^i) \cup \mathrm{vars}(E) = (\mathrm{vars}(\mathcal{G}^i) \cap V^i) \cup \mathrm{vars}(\mathcal{C}^i) \cup \mathrm{vars}(E) \subseteq V^i$ as seen in the proof of Proposition 20 (ii). Then S^i is Σ_u-controllable with respect to $\|(\mathcal{G})$ by Proposition 16.

(iii) Let S' be an EFSM that satisfies (i) and (ii). i.e., $G \parallel H \parallel S' \subseteq_v G \parallel H \parallel E$ and S' is Σ_u-controllable with respect to $G \parallel H$. As $\sup\mathcal{C}(\|(\mathcal{G}), E, \Sigma_u)$ is a supremal supremal for E with respect to $\|(\mathcal{G})$ and Σ_u by Proposition 11, it follows from Definition 8 (iii) and Proposition 21 that $\|(\mathcal{G}) \parallel S' \subseteq_v \|(\mathcal{G}) \parallel \sup\mathcal{C}(\|(\mathcal{G}), E, \Sigma_u) \subseteq_v \|(\mathcal{G}) \parallel S^i$. It follows by Lemma 5 (ii) that $\|(\mathcal{G}) \parallel S' \subseteq_v \|(\mathcal{G}) \parallel S^i$. □

It is also clear that Algorithm 1 terminates, because the set Σ_u^i of uncontrollable events increases with each iteration. This is because, if the loop is entered again, then S^i is not Σ_u-controllable by line 10 but Σ_u^i-controllable by line 14, which means that there must exist $\mu \in \mathrm{uncont}(\|(\exists \bar{V}^i \mathcal{G}^i) \| \|(\|\mathcal{C}^i), S^i, \Sigma_u) \setminus \Sigma_u^i$ on line 11. Yet, Σ_u^i cannot increase forever, because it is bounded by the finite set Σ_u of all uncontrollable events. This is enough to complete the total correctness proof, i.e., Algorithm 1 terminates and returns the correct supremal supervisor for all inputs.

4 Abstracting the Specification

This section considers the case of a single specification EFSM E, assumed to be state-deterministic and pure, and a single normalised and state-deterministic plant EFSM G. If the plant is more structured than that, it can be abstracted using the methods in Section 3 above. The concern here is whether any variables can be existentially abstracted from the specification E.

As noted in Remark 1 on page 16, variables that appear only in the pure specification and not in the plant, can be removed by replacing them with a constant representing their initial value. This trivial case is not considered further. The question then is whether any variables shared between the plant and specification can be existentially abstracted from the specification.

By closely inspecting the synthesis process, it can be observed that only the updates of the uncontrollable events are relevant for the removal of states. This suggests the existential quantification of variables that are only used controllably in the specification. That is, if a set of variables $\bar{V} \subseteq \mathrm{vars}(E)$ does not contain any variables used uncontrollably in E,

$$\bar{V} \cap \mathrm{vars}(E, \Sigma_{\mathrm{u}}) = \emptyset \ , \tag{4.1}$$

then it is enough to synthesise for the abstracted specification $\exists \bar{V} E$ instead of E. Such synthesis only makes sense for a state-deterministic abstraction, so a second assumption is made that

$$\exists \bar{V} E \text{ is state-deterministic} \ . \tag{4.2}$$

Then synthesis will ensure that all constraints associated with uncontrollable events in E are satisfied controllably. However, the constraints associated with controllable events are not properly included in the abstraction $\exists \bar{V} E$ and may not be carried forward in the synthesis result. Fortunately, controllable constraints can easily be enforced in a supervisor without the need for synthesis—it is enough to use the updates on the controllable transitions in E on the corresponding transitions in the synthesis result. Under the assumption of state-determinism, this can be achieved by composing the synthesis result for the abstracted specification with the original specification.

Therefore, to compute a supervisor for a specification E, it is possible to first find an abstraction $\exists \bar{V} E$ subject to (4.1) and (4.2), and then compute a supervisor $S_\exists = \sup\mathcal{C}(G, \exists \bar{V} E, \Sigma_{\mathrm{u}})$ for the abstraction, e.g., using Algorithm 1. Then a supervisor for the original specification is obtained by composing the result $S_\exists$ obtained with the abstraction with the original specification E, i.e., $S_\exists \parallel E$. As the main result for specification abstraction, Theorem 28 at the end of this section shows under the assumptions (4.1) and (4.2) that

$$S_\exists \parallel E \text{ is a supremal supervisor for } E \text{ with respect to } G \text{ and } \Sigma_{\mathrm{u}} \tag{4.3}$$

for every supremal supervisor $S_\exists$ for $\exists \bar{V} E$ with respect to G and Σ_{u}.

To use this result for synthesis, one has to find a set $\bar{V}$ of variables satisfying (4.1) and (4.2) and compute the abstraction $\exists \bar{V} E$. While the variables are easily found by removing the variables used uncontrollably from the set of all variables in the specification, the requirement (4.2) of state-determinism is more difficult to ensure algorithmically. A simple solution is to start with all variables used only controllably in the specification, $\bar{V} = \mathrm{vars}(E) \setminus \mathrm{vars}(E, \Sigma_\mathrm{u})$, and gradually remove variables that cause failure of $\exists \bar{V} E$ being state-deterministic, until (4.2) is satisfied. Once an appropriate set $\bar{V}$ of variables for abstraction is found, Algorithm 1 can be used to compute a synthesis result for $\sup\mathcal{C}(G, \exists \bar{V} E, \Sigma_\mathrm{u})$, which then can be combined with the specification E to obtain a correct supervisor for the original synthesis problem.

The proof of (4.3) requires to establish the three conditions for a supremal supervisor from Definition 8, namely behavioural inclusion in the specification, controllability, and least restrictiveness. These conditions are established in the following Sections 4.1–4.3, and then combined in Section 4.4 to give the final result.

4.1 Proof of Behavioural Inclusion

The first step of the proof of (4.3) requires to show that the combined supervisor $S_\exists \parallel E$ is behaviourally included in the original specification E. This easily follows from the properties of EFSM synchronous composition because the specification E is pure and already part of the combined supervisor $S_\exists \parallel E$.

Proposition 23 Let G and E be two EFSMs such that E is pure, let $\bar{V} \subseteq \mathrm{vars}(E)$, and let Σ_u be a set of events. If $S_\exists$ is a supremal supervisor for $\exists \bar{V} E$ with respect to G and Σ_u, then $G \parallel S_\exists \parallel E \subseteq_\mathrm{v} G \parallel E$.

Proof. As $S_\exists$ is a supremal supervisor for $\exists \bar{V} E$ with respect to G, it holds by Definition 8 (i) that $G \parallel S_\exists \subseteq_\mathrm{v} G \parallel \exists \bar{V} E$. As E and thus also $\exists \bar{V} E$ is pure, it holds by Lemma 6 (i) that $G \parallel \exists \bar{V} E \subseteq_\mathrm{v} G$. Then it follows by Lemma 5 (ii) that $G \parallel S_\exists \subseteq_\mathrm{v} G$. Finally, since E is pure, it follows by Lemma 6 (ii) that $G \parallel S_\exists \parallel E \subseteq_\mathrm{v} G \parallel E$. □

4.2 Proof of Controllability

The second step towards the proof of (4.3) is to show that the supervisor synthesised from the abstraction combined with the specification, $S_\exists \parallel E$, is controllable. It is clear that $S_\exists$ is controllable as it is a supremal supervisor, but the crucial issue is that it remains controllable when composed with the original specification E.

Here, assumption (4.1) is important, because it ensures that the updates associated with uncontrollable events are the same in the specification E and its abstraction $\exists \bar{V} E$. As $S_\exists$ is synthesised for the abstracted specification, its behaviour must be included in that of $\exists \bar{V} E$. Then assumption (4.1) ensures that its uncontrollable transitions are also possible in E.

Proving this requires to link the locations of the specification E and its abstraction $\exists \bar{V} E$ in paths with the same variable assignments. The following Lemma 24 uses the assumption (4.2) of the state-determinism of the abstraction to show that E and $\exists \bar{V} E$ are always in the same locations.

Lemma 24 Let A, B, and E be EFSMs, let $\bar{V} \subseteq \mathrm{vars}(E)$ such that $\exists \bar{V} E$ is state-deterministic, and assume there are paths

$$(x_A^0, x_E^0, \hat{v}^0) \xrightarrow{\sigma_1} (x_A^1, x_E^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (x_A^n, x_E^n, \hat{v}^n) \qquad \text{in } A \parallel E \ ; \tag{4.4}$$

$$(y_B^0, y_E^0, \hat{v}^0) \xrightarrow{\sigma_1} (y_B^1, y_E^1, \hat{v}^1) \xrightarrow{\sigma_2} \cdots \xrightarrow{\sigma_n} (y_B^n, y_E^n, \hat{v}^n) \qquad \text{in } B \parallel \exists \bar{V} E \ ; \tag{4.5}$$

where $\hat{v}^i \in \mathrm{dom}(V)$ for some $V \supseteq \mathrm{vars}(A) \cup \mathrm{vars}(B) \cup \mathrm{vars}(E)$. Then $x_E^i = y_E^i$ for $i = 0, \ldots, n$.

Proof. The claim is shown by induction on i.

For the base case, $i = 0$, note that x_E^0 is an initial location of E and thus also of $\exists \bar{V} E$ by Definition 12. As y_E^0 also is an initial location of $\exists \bar{V} E$, it follows that $x_E^0 = y_E^0$ by Definition 2 (iii) as $\exists \bar{V} E$ is state-deterministic.

Now assume $x_E^i = y_E^i$ for some $i \geq 0$. Considering the $(i+1)$-th transition on the path (4.4), by Lemma 2 there is a transition

$$x_E^i \xrightarrow{\sigma_{i+1} : p} x_E^{i+1} \qquad \text{in } E \text{ with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(p) = \text{true} \ . \tag{4.6}$$

Then by Definition 12 there is a transition

$$x_E^i \xrightarrow{\sigma_{i+1} : \exists \bar{V} p} x_E^{i+1} \qquad \text{in } \exists \bar{V} E \text{ with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(\exists \bar{V} p) = \text{true} \ . \tag{4.7}$$

Considering the $(i+1)$-th transition on the path (4.5), by Lemma 2 there is a transition

$$y_E^i \xrightarrow{\sigma_{i+1} : q} y_E^{i+1} \qquad \text{in } \exists \bar{V} E \text{ with } (\hat{v}^i \oplus (\hat{v}^{i+1})')(q) = \text{true} \ . \tag{4.8}$$

Then $\exists \bar{V} p \wedge q$ is satisfiable, and noting that $x_E^i = y_E^i$ by inductive assumption, it follows by the state-determinism of $\exists \bar{V} E$ from Definition 2 (iii) and from (4.7) and (4.8) that $x_E^{i+1} = y_E^{i+1}$. □

Given the consistency of the reachable locations between the abstraction $\exists \bar{V} E$ and the original specification E from Lemma 24, the following Proposition 25 establishes controllability of the combined supervisor $S_\exists \parallel E$, under the assumptions (4.1) and (4.2).

Proposition 25 Let G and E be two EFSMs, where E is pure, let Σ_u be a set of events and $\bar{V} \subseteq \mathrm{vars}(E)$ be a set of variables such that $\bar{V} \cap \mathrm{vars}(E, \Sigma_\mathrm{u}) = \emptyset$ and $\exists \bar{V} E$ is state-deterministic, and let $S_\exists$ be a supremal supervisor for $\exists \bar{V} E$ with respect to G and Σ_u. Then $S_\exists \parallel E$ is Σ_u-controllable with respect to G.

Proof. Write $V = \mathrm{vars}(G) \cup \mathrm{vars}(S_\exists) \cup \mathrm{vars}(E)$. Following Definition 7, let

$$(x_G, x_S, x_E, \hat{v}) \in Q^{\mathrm{acc}}(G \parallel S_\exists \parallel E) \ , \tag{4.9}$$

where $\hat{v} \in \mathrm{dom}(V)$, and let $\mu \in \Sigma_\mathrm{u}$ and $\hat{w} \in \mathrm{dom}(V)$ such that

$$(x_G, \hat{v}) \xrightarrow{\mu} (y_G, \hat{w}) \quad \text{in } G \ . \tag{4.10}$$

It is enough to show that there exist locations y_S of $S_\exists$ and y_E of E such that the following transition exists in $G \parallel S_\exists \parallel E$:

$$(x_G, x_S, x_E, \hat{v}) \xrightarrow{\mu} (y_G, y_S, y_E, \hat{w}) \ . \tag{4.11}$$

It follows from (4.9) that there is a path

$$(x_G^0, x_S^0, x_E^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_S^n, x_E^n, \hat{v}^n) = (x_G, x_S, x_E, \hat{v}) \quad \text{in } G \parallel S_\exists \parallel E \ . \tag{4.12}$$

Since E is pure, by Lemma 3

$$(x_G^0, x_S^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_S^n, \hat{v}^n) = (x_G, x_S, \hat{v}) \tag{4.13}$$

is a path in $G \parallel S_\exists$. Then $(x_G, x_S, \hat{v}{\restriction_W}) \in Q^{\mathrm{acc}}(G \parallel S_\exists)$ where $W = \mathrm{vars}(G) \cup \mathrm{vars}(S_\exists)$, and $(x_G, \hat{v}{\restriction_W}) \xrightarrow{\mu} (y_G, \hat{w}{\restriction_W})$ in G by (4.10). Then, since $S_\exists$ is Σ_u-controllable with respect to G, there exists a location y_S of $S_\exists$ such that

$$(x_G, x_S, \hat{v}{\restriction_W}) \xrightarrow{\mu} (y_G, y_S, \hat{w}{\restriction_W}) \qquad \text{in } G \parallel S_\exists \ . \tag{4.14}$$

This means by Lemma 2 that there are transitions

$$x_G \xrightarrow{\mu : p_G} y_G \qquad \text{in } G \qquad \text{with } (\hat{v} \oplus \hat{w}')(p_G) = (\hat{v}{\restriction_W} \oplus (\hat{w}{\restriction_W})')(p_G) = \mathrm{true} \ ; \tag{4.15}$$

$$x_S \xrightarrow{\mu : p_S} y_S \qquad \text{in } S_\exists \qquad \text{with } (\hat{v} \oplus \hat{w}')(p_S) = (\hat{v}{\restriction_W} \oplus (\hat{w}{\restriction_W})')(p_S) = \mathrm{true} \ ; \tag{4.16}$$

and furthermore $\hat{v}(z) = \hat{w}(z)$ for all $z \in W \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S))$. Note for $z \in V \setminus W = V \setminus (\mathrm{vars}'(G) \cup \mathrm{vars}'(S_\exists)) = V \setminus (\mathrm{vars}'(G) \cup \mathrm{vars}'(S_\exists) \cup \mathrm{vars}'(E))$ as E is pure, that $\hat{v}(z) = \hat{w}(z)$ by (4.11). Thus,

$$\hat{v}(z) = \hat{w}(z) \quad \text{for all } z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S)) \ , \tag{4.17}$$

and it also holds that

$$(x_G, x_S, \hat{v}) \xrightarrow{\mu} (y_G, y_S, \hat{w}) \qquad \text{in } G \parallel S_\exists \ . \tag{4.18}$$

Combining (4.13) and (4.18), it follows that

$$(x_G^0, x_S^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_S^n, \hat{v}^n) = (x_G, x_S, \hat{v}) \xrightarrow{\mu} (y_G, y_S, \hat{w}) \quad \text{in } G \parallel S_\exists \ . \tag{4.19}$$

As $S_\exists$ is a supremal supervisor for $\exists \bar{V} E$ with respect to G, it holds by Definition 8 (i) that $G \parallel S_\exists \subseteq_\mathrm{v} G \parallel \exists \bar{V} E$, so by Definition 6 there is a path

$$(y_G^0, y_E^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (y_G^n, y_E^n, \hat{v}^n) = (y_G^n, y_E^n, \hat{v}) \xrightarrow{\mu} (y_G^{n+1}, y_E^{n+1}, \hat{w}) \quad \text{in } G \parallel \exists \bar{V} E \ . \tag{4.20}$$

As $\exists \bar{V} E$ is state-deterministic, it follows by Lemma 24 from (4.12) and (4.20) that $y_E^n = x_E^n = x_E$. Let $y_E = y_E^{n+1}$. From the last step of (4.20) it follows by Lemma 2 that

$$x_E = y_E^n \xrightarrow{\mu : q_E} y_E^{n+1} = y_E \quad \text{in } \exists \bar{V} E \quad \text{with } (\hat{v} \oplus \hat{w}')(q_E) = \text{true} \ . \tag{4.21}$$

By Definition 12, the update q_E has the form $q_E \equiv \exists \bar{V} p_E$ for some transition $x_E \xrightarrow{\mu : p_E} y_E$ in E. But $\mathrm{vars}(p_E) \cap \bar{V} \subseteq \mathrm{vars}(E, \mu) \cap \bar{V} \subseteq \mathrm{vars}(E, \Sigma_\mathrm{u}) \cap \bar{V} = \emptyset$ by assumption, so $q_E \equiv \exists \bar{V} p_E$ and p_E are logically equivalent and thus

$$x_E \xrightarrow{\mu : p_E} y_E \quad \text{in } E \quad \text{with } (\hat{v} \oplus \hat{w}')(p_E) = \text{true} \ . \tag{4.22}$$

Lastly, note that $\mathrm{vars}'(p_E) = \emptyset$ as E is pure, so that $\hat{v}(z) = \hat{w}(z)$ for all variables $z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S) \cup \mathrm{vars}'(p_E)) = V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_S))$ by (4.17). Then the existence of the transition (4.11) follows by Lemma 2 using (4.15), (4.16), and (4.22). □

4.3 Proof of Least Restrictiveness

The third and last step towards of proof of (4.3) is to show that synthesis with the abstracted specification results in a least restrictive result with respect to the original specification. This is easier to show than the above controllability result, because the abstracted specification is weaker than the original specification, and therefore gives a less restrictive result. The assumptions (4.1) and (4.2) are not needed here.

As a first step, the following Lemma 26 shows that the original specification E is behaviourally contained in its abstraction $\exists \bar{V} E$, in composition with every plant G.

Lemma 26 Let G and E be EFSMs such that E is pure, and let $\bar{V} \subseteq \mathrm{vars}(E)$. Then $G \parallel E \subseteq_\mathrm{v} G \parallel \exists \bar{V} E$.

Proof. Write $V = \mathrm{vars}(G) \cup \mathrm{vars}(E)$, and consider a path

$$(x_G^0, x_E^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_E^n, \hat{v}^n) \quad \text{in } G \parallel E \tag{4.23}$$

where $\hat{v}^0, \ldots, \hat{v}^n \in \mathrm{dom}(V)$. It will be shown that (4.23) also is a path in $G \parallel \exists \bar{V} E$. Clearly, x_G^0 and $\hat{v}^0$ are initial locations and variable values from (4.23), and x_E^0 is an initial location of E and by Definition 12 also of $\exists \bar{V} E$. Considering the i-th transition of the path (4.23), by Lemma 2 there are transitions

$$x_G^{i-1} \xrightarrow{\sigma_i : p_G} x_G^i \qquad \text{in } G \qquad \text{with } (\hat{v}^{i-1} \oplus (\hat{v}^i)')(p_G) = \text{true} \ ; \tag{4.24}$$

$$x_E^{i-1} \xrightarrow{\sigma_i : p_E} x_E^i \qquad \text{in } E \qquad \text{with } (\hat{v}^{i-1} \oplus (\hat{v}^i)')(p_E) = \text{true} \ ; \tag{4.25}$$

such that

$$\hat{v}^{i-1}(z) = \hat{v}^i(z) \quad \text{for all } z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_E)) \ . \tag{4.26}$$

From (4.25), it follows by Definition 12 that there is a transition

$$x_E^{i-1} \xrightarrow{\sigma_i : \exists \bar{V} p_E} x_E^i \qquad \text{in } \exists \bar{V} E \qquad \text{with } (\hat{v}^{i-1} \oplus (\hat{v}^i)')(\exists \bar{V} p_E) = \text{true} \ . \tag{4.27}$$

Note that $\text{vars}'(p_E) = \emptyset$ as E is pure, and then also $\text{vars}'(\exists \bar{V} p_E) = \emptyset$. Then for a variable $z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(\exists \bar{V} p_E)) = V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_E))$, it follows that $\hat{v}^{i-1}(z) = \hat{v}^i(z)$ by (4.26). It follows by Lemma 2 that $(x_G^{i-1}, x_E^{i-1}, \hat{v}^{i-1}) \xrightarrow{\sigma_i} (x_G^i, x_E^i, \hat{v}^i)$ in $G \parallel \exists \bar{V} E$, and by repeating this argument for all i it is shown that the path (4.23) exists in $G \parallel \exists \bar{V} E$. □

The containment result from Lemma 26 is enough to prove least the least restrictiveness result for a specification abstraction. The following Proposition 27 shows that, if $S_\exists$ is a supremal supervisor for the abstraction $\exists \bar{V} E$ and S' is a controllable supervisor that satisfies the full specification E, then S' is more restrictive than the combined supervisor $S_\exists \parallel E$ computed using the abstraction.

Proposition 27 Let G and E be EFSMs such that E is pure, let Σ_u be a set of events, and let $\bar{V} \subseteq \text{vars}(E)$. Further, let S' be an EFSM such that $G \parallel S' \subseteq_\text{v} G \parallel E$ and S' is Σ_u-controllable with respect to G and Σ_u, and let $S_\exists$ be a supremal supervisor for $\exists \bar{V} E$ with respect to G and Σ_u. Then $G \parallel S' \subseteq_\text{v} G \parallel S_\exists \parallel E$.

Proof. As $G \parallel S' \subseteq_\text{v} G \parallel E$ by assumption and $G \parallel E \subseteq_\text{v} G \parallel \exists \bar{V} E$ by Lemma 26, it follows by Lemma 5 (ii) that $G \parallel S' \subseteq_\text{v} G \parallel \exists \bar{V} E$. As S' is also Σ_u-controllable with respect to G and Σ_u by assumption, and $S_\exists$ be a supremal supervisor for $\exists \bar{V} E$ with respect to G and Σ_u, it follows by Definition 8 (iii) that $G \parallel S' \subseteq_\text{v} G \parallel S_\exists$. This implies

$$G \parallel S' \parallel E \subseteq_\text{v} G \parallel S_\exists \parallel E \tag{4.28}$$

by Lemma 6 (ii) since E is pure. Furthermore, it follows from the assumption $G \parallel S' \subseteq_\text{v} G \parallel E$ by Lemma 6 (iii) that $G \parallel S' \subseteq_\text{v} G \parallel S' \parallel E$. Then also $G \parallel S' \subseteq_\text{v} G \parallel S_\exists \parallel E$ by Lemma 5 (ii) □

4.4 Proof of Main Result for Specification Abstraction

The following Theorem 28 combines the preceding results from Sections 4.1, 4.2, and 4.3. If the abstracted variables are chosen according to the assumptions (4.1) and (4.2), it is possible to synthesise a supervisor for an existentially abstracted specification. The result, when composed with the original specification, can serve as least restrictive supervisor.

Theorem 28 Let G and E be EFSMs such that E is pure, let Σ_u be a set of events, and let $\bar{V} \subseteq \text{vars}(G) \cap \text{vars}(E)$ such that $\bar{V} \cap \text{vars}(E, \Sigma_\text{u}) = \emptyset$ and $\exists \bar{V} E$ is state-deterministic. If $S_\exists$ is a supremal supervisor for $\exists \bar{V} E$ with respect to G and Σ_u, then $S_\exists \parallel E$ is a supremal supervisor for E with respect to G and Σ_u.

Algorithm 2: Abstraction-based modular synthesis with multiple specifications

Input: normalised state-deterministic plants $\mathcal{G} = \{G_1, \ldots, G_m\}$; pure specifications $\mathcal{E} = \{E_1, \ldots, E_k\}$; uncontrollable events Σ_u.

Output: collection $\mathcal{S}$ of supervisors such that $\big\|(\mathcal{S})$ is a supremal supervisor for $\big\|(\mathcal{E})$ with respect to $\big\|(\mathcal{G})$ and Σ_u.

1 $\mathcal{S} \leftarrow \emptyset$;
2 **foreach** $E_j \in \mathcal{E}$ **do**
3 | Choose $\bar{V}_j \subseteq \mathrm{vars}(E_j) \setminus \mathrm{vars}(E_j, \Sigma_\mathrm{u})$ such that $\exists \bar{V}_j\, E_j$ is state-deterministic;
4 | Calculate S_j using Algorithm 1 with $E = \exists \bar{V}_j\, E_j$;
5 | $\mathcal{S} \leftarrow \mathcal{S} \cup \{S_j, E_j\}$;
6 **end**
7 **return** $\mathcal{S}$

Proof. It is to be shown that $S_\exists \parallel E$ satisfies the conditions (i)–(iii) from Definition 8 for a supremal supervisor. Condition (i) follows from Proposition 23, condition (ii) follows from Proposition 25, and condition (iii) follows from Proposition 27. □

5 Synthesis with Multiple Specifications

The results presented in the previous Sections 3 and 4 show how abstractions can be used to compute a supervisor for a single specification. In general, the specification is given in modular form, as a synchronous composition $E_1 \parallel \cdots \parallel E_k$ of several EFSMs. In this case, it is known [2] for ordinary FSMs that synthesis can be performed separately for each specification, and the resulting supervisors can be combined to form a least restrictive controllable supervisor for the combined specification. Under the assumption of pure specifications, these results can be generalised directly for EFSMs [9, 10].

Algorithm 2 uses this idea to synthesise a modular supervisor for an EFSM system composed of several plants $G_1 \parallel \cdots \parallel G_m$ and specifications $E_1 \parallel \cdots \parallel E_k$, while incorporating the results from Sections 3 and 4. The loop on line 2 processes each specification E_j, by first abstracting it according to Section 4 and then computing a supervisor using plant abstractions according to Section 3. On line 3, the variables $\bar{V}_j$ for abstraction of the specification E_j are chosen to satisfy assumptions (4.1) and (4.2), and then line 4 invokes Algorithm 1 to compute a supremal supervisor S_j for the specification abstraction $\exists \bar{V}_j\, E_j$. In this case, Theorem 28 states that the composition $S_j \parallel E_j$ of the supervisor computed using the specification abstraction and the original specification is a supremal supervisor, and therefore both EFSMs S_j and E_j are added to the modular supervisor $\mathcal{S}$ on line 5.

The remainder of this section is devoted to the correctness proof of Algorithm 2. The main argument is based on the observation that synthesis for a modular specification can

be performed separately for each specification. It is first shown for a modular specification $E_1 \parallel E_2$ consisting of two EFSMs that the composition $S_1 \parallel S_2$ of some least restrictive supervisors for E_1 and E_2 is a least restrictive supervisor for the combined specification. This requires to establish the three conditions for a supremal supervisor from Definition 8, namely behavioural inclusion in the combined specification, controllability, and least restrictiveness, which are shown separately in Sections 5.1–5.3. These proofs appear in similar form in [9], and are adapted here to the revised definition of behavioural inclusion.

Afterwards, Section 5.4 combines and generalises the results for any number of specifications. In conjunction with the results from Sections 3 and 4, it proves the correctness of Algorithm 2.

5.1 Proof of Behavioural Inclusion

In this and the following subsections, it is assumed that there is a single normalised plant EFSM G and two pure specification EFSMs E_1 and E_2, and supremal supervisors S_1 and S_2 have been synthesised separately for these two specifications. It is to be shown that the composition $S_1 \parallel S_2$ of the supervisors is a supremal supervisor for the composition $E_1 \parallel E_2$ of the specifications.

As a first step, Proposition 30 in this subsection shows that the composition of the supervisors is behaviourally included in the combined specification. This is clear for ordinary FSMs because, if the behaviour of each supervisor is included in one specification, then their combined behaviour must be included in both. For EFSMs, Lemma 29 first ensures that the composition of the supervisors is possible without any conflict in variable updates. That is, any path in the composition of the two supervisors also is a path in each of the supervisors. This is true because both supervisors are synthesised with respect to the same normalised plant, so they both must follow the plant's variable updates.

Lemma 29 Let G be a normalised EFSM, let Σ_u be a set of events, and let S_1 and S_2 be supremal supervisors for some pure EFSMs E_1 and E_2, respectively, with respect to G and Σ_u. If for some $V \supseteq \mathrm{vars}(G) \cup \mathrm{vars}(S_1) \cup \mathrm{vars}(S_2)$ and $\hat{v}^0, \ldots, \hat{v}^n \in \mathrm{dom}(V)$,

$$(x_G^0, x_{S1}^0, x_{S2}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_{S1}^n, x_{S2}^n, \hat{v}^n) \tag{5.1}$$

is a path in $G \parallel S_1 \parallel S_2$, then

$$(x_G^0, x_{S1}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_{S1}^n, \hat{v}^n) \tag{5.2}$$

$$(x_G^0, x_{S2}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_{S2}^n, \hat{v}^n) \tag{5.3}$$

are paths in $G \parallel S_1$ and $G \parallel S_2$, respectively.

Proof. The claim is shown by induction on n.

Clearly, x_G^0, x_{S1}^0, x_{S2}^0, and $\hat{v}^0$ are all initial locations and variable values, which shows the claim for $n = 0$.

Now assume the paths (5.2) and (5.3) have been constructed up to length n, and consider the next transition

$$(x_G^n, x_{S1}^n, x_{S2}^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_{S1}^{n+1}, x_{S2}^{n+1}, \hat{v}^{n+1}) \tag{5.4}$$

on the path (5.1). This means by Lemma 2 that there are transitions

$$x_G^n \xrightarrow{\sigma_{n+1}:p_G} x_G^{n+1} \qquad \text{in } G \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1}))(p_G) = \text{true} ; \tag{5.5}$$

$$x_{S1}^n \xrightarrow{\sigma_{n+1}:p_{S1}} x_{S1}^{n+1} \qquad \text{in } S_1 \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1}))(p_{S1}) = \text{true} ; \tag{5.6}$$

$$x_{S2}^n \xrightarrow{\sigma_{n+1}:p_{S2}} x_{S2}^{n+1} \qquad \text{in } S_2 \text{ with } (\hat{v}^n \oplus (\hat{v}^{n+1}))(p_{S2}) = \text{true} ; \tag{5.7}$$

such that

$$\hat{v}^n(z) = \hat{v}^{n+1}(z) \qquad \text{for all } z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_{S1}) \cup \text{vars}'(p_{S2})) . \tag{5.8}$$

Construct $\hat{w}_1^{n+1} = \hat{v}^{n+1}{\restriction}_{\text{vars}'(p_G)\cup\text{vars}'(p_{S1})} \oplus \hat{v}^n$. That is, $\hat{w}_1^{n+1}$ is the same as $\hat{v}^{n+1}$ for variables that appear primed in p_G or p_{S1} and keeps the values from $\hat{v}^n$ for other variables. Then $(\hat{v}^n \oplus (\hat{w}_1^{n+1})')(p_G) = \text{true}$ and $(\hat{v}^n \oplus (\hat{w}_1^{n+1})')(p_{S1}) = \text{true}$ and $\hat{v}^n(z) = \hat{w}_1^{n+1}(z)$ for $z \in V \setminus (\text{vars}'(p_G) \cup \text{vars}'(p_{S1}))$. By inductive assumption (5.2) and Lemma 2, there is a path

$$(x_G^0, x_{S1}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_{S1}^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_{S1}^{n+1}, \hat{w}_1^{n+1}) \tag{5.9}$$

in $G \| S_1$. Since S_1 is a supremal supervisor for E_1 with respect to G, it holds by Definition 8 (i) that $G \parallel S_1 \subseteq_{\text{v}} G \parallel E_1$, so there exists a path

$$(y_G^0, y_{E1}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (y_G^n, y_{E1}^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (y_G^{n+1}, y_{E1}^{n+1}, \hat{w}_1^{n+1}) \tag{5.10}$$

in $G \parallel E_1$. Now for the last transition on this path, by Lemma 2 there are transitions $y_G^n \xrightarrow{\sigma_{n+1}:q_G} y_G^{n+1}$ in G and $y_{E1}^n \xrightarrow{\sigma_{n+1}:q_{E1}} y_{E1}^{n+1}$ in E_1 such that $\hat{v}^n(z) = \hat{w}_1^{n+1}(z)$ for all

$$z \in V \setminus (\text{vars}'(q_G) \cup \text{vars}'(q_{E1})) = V \setminus \text{vars}'(q_G) = V \setminus \text{vars}'(G, \sigma_{n+1}) = V \setminus \text{vars}'(p_G) \tag{5.11}$$

because E_1 is pure and G is normalised. Now consider an arbitrary variable $z \in \text{vars}'(p_{S1}) \setminus \text{vars}'(p_G) \subseteq V \setminus \text{vars}'(p_G)$. Then $\hat{v}^n(z) = \hat{w}_1^{n+1}(z) = \hat{v}^{n+1}{\restriction}_{\text{vars}'(p_G)\cup\text{vars}'(p_{S1})}(z) = \hat{v}^{n+1}(z)$ by construction of $\hat{w}_1^{n+1}$.

By analogous argumentation, it is shown that $\hat{v}^n(z) = \hat{v}^{n+1}(z)$ for an arbitrary variable $z \in \text{vars}'(p_{S2}) \setminus \text{vars}'(p_G)$. Combining these observations with (5.8) gives $\hat{v}^n(z) = \hat{v}^{n+1}(z)$ for all variables $z \in V \setminus \text{vars}'(p_G)$. Then given (5.5)–(5.7), by Lemma 2 there are transitions

$$(x_G^n, x_{S1}^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_{S1}^{n+1}, \hat{v}^{n+1}) \qquad \text{in } G \parallel S_1 ; \tag{5.12}$$

$$(x_G^n, x_{S2}^n, \hat{v}^n) \xrightarrow{\sigma_{n+1}} (x_G^{n+1}, x_{S2}^{n+1}, \hat{v}^{n+1}) \qquad \text{in } G \parallel S_2 ; \tag{5.13}$$

which together with the inductive assumption extend the paths (5.2) and (5.3) up to $n+1$. □

Given the result from Lemma 29 about the composition of two supervisors, it can now be shown that together they ensure behavioural inclusion in the combined specification. This follows from the fact that each supervisor ensures inclusion in its specification, under the assumption of purity.

Proposition 30 Let G be a normalised EFSM, let E_1 and E_2 be pure EFSMs, and let Σ_u be a set of events. Let S_1 and S_2 be supremal supervisors for E_1 and E_2, respectively, with respect to G and Σ_u. Then $G \parallel S_1 \parallel S_2 \subseteq_\mathrm{v} G \parallel E_1 \parallel E_2$.

Proof. Write $V = \mathrm{vars}(G) \cup \mathrm{vars}(E_1) \cup \mathrm{vars}(E_2) \cup \mathrm{vars}(S_1) \cup \mathrm{vars}(S_2)$. Let

$$(x_G^0, x_{S1}^0, x_{S2}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_{S1}^n, x_{S2}^n, \hat{v}^n) \tag{5.14}$$

be a path in $G \parallel S_1 \parallel S_2$ where $\hat{v}^i \in \mathrm{dom}(V)$ for $i = 0, \ldots, n$. By Lemma 29 there are paths

$$(x_G^0, x_{S1}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_{S1}^n, \hat{v}^n) \qquad \text{in } G \parallel S_1 \ ; \tag{5.15}$$

$$(x_G^0, x_{S2}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_{S2}^n, \hat{v}^n) \qquad \text{in } G \parallel S_2 \ . \tag{5.16}$$

Since S_1 and S_2 are supremal supervisors for E_1 and E_2 with respect to G, it holds by Definition 8 (i) that $G \parallel S_1 \subseteq_\mathrm{v} G \parallel E_1$ and $G \parallel S_2 \subseteq_\mathrm{v} G \parallel E_2$, so there are paths

$$(y_{G1}^0, y_{E1}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (y_{G1}^n, y_{E1}^n, \hat{v}^n) \qquad \text{in } G \parallel E_1 \ ; \tag{5.17}$$

$$(y_{G2}^0, y_{E2}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (y_{G2}^n, y_{E2}^n, \hat{v}^n) \qquad \text{in } G \parallel E_2 \ . \tag{5.18}$$

It will be shown that

$$(y_{G1}^0, y_{E1}^0, y_{E2}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (y_{G1}^n, y_{E1}^n, y_{E2}^n, \hat{v}^n) \tag{5.19}$$

is a path in $G \parallel E_1 \parallel E_2$. Clearly, y_{G1}^0, y_{E1}^0, y_{E2}^0, and $\hat{v}^0$ are initial locations and variable values. Now fix $i = 0, \ldots, n-1$. By Lemma 2 and (5.17) there are transitions $y_{G1}^i \xrightarrow{\sigma_{i+1}:p_{G1}} y_{G1}^{i+1}$ in G with $(\hat{v}^i \oplus (\hat{v}^{i+1})')(p_{G1}) = \text{true}$ and $y_{E1}^i \xrightarrow{\sigma_{i+1}:p_{E1}} y_{E1}^{i+1}$ in E_1 with $(\hat{v}^i \oplus (\hat{v}^{i+1})')(p_{E1}) = \text{true}$ such that $\hat{v}^i(z) = \hat{v}^{i+1}(z)$ for all variables $z \in V \setminus (\mathrm{vars}'(p_{G1}) \cup \mathrm{vars}'(p_{E1}))$. Likewise by (5.18) there is a transition $y_{E2}^i \xrightarrow{\sigma_{i+1}:p_{E2}} y_{E2}^{i+1}$ in E_2 with $(\hat{v}^i \oplus (\hat{v}^{i+1})')(p_{E2}) = \text{true}$. Note that $\mathrm{vars}'(p_{E2}) = \emptyset$ as E_2 is pure, so for an arbitrary variable $z \in V \setminus (\mathrm{vars}'(p_{G1}) \cup \mathrm{vars}'(p_{E1}) \cup \mathrm{vars}'(p_{E2})) = V \setminus (\mathrm{vars}'(p_{G1}) \cup \mathrm{vars}'(p_{E1}))$ it also holds that $\hat{v}^i(z) = \hat{v}^{i+1}(z)$. Then by Lemma 2,

$$(y_{G1}^i, y_{E1}^i, y_{E2}^i, \hat{v}^0) \xrightarrow{\sigma_{1+1}} (y_{G1}^{i+1}, y_{E1}^{i+1}, y_{E2}^{i+1}, \hat{v}^{i+1}) \tag{5.20}$$

in $G \parallel E_1 \parallel E_2$, and the path (5.19) is constructed by repeating this argument for all i. □

5.2 Proof of Controllability

As a second step towards the proof for two separate supervisors, the following Proposition 31 shows that the combination of two supremal supervisors for the same plant remains controllable. This is known [2] for ordinary FSMs, and is easily generalised for EFSMs using the result from Lemma 29 about the paths in the composition of supervisors.

Proposition 31 Let G be a normalised EFSM, let E_1 and E_2 be pure EFSMs, and let Σ_u be a set of events. Let S_1 and S_2 be supremal supervisors for E_1 and E_2, respectively, with respect to G and Σ_u. Then $S_1 \parallel S_2$ is Σ_u-controllable with respect to G.

Proof. Write $V_1 = \mathrm{vars}(G) \cup \mathrm{vars}(S_1)$, $V_2 = \mathrm{vars}(G) \cup \mathrm{vars}(S_2)$, and $V = V_1 \cup V_2 = \mathrm{vars}(G) \cup \mathrm{vars}(S_1) \cup \mathrm{vars}(S_2)$. Assume $\hat{v}, \hat{w} \in \mathrm{dom}(V)$ and $(x_G, x_1, x_2, \hat{v}) \in Q^{\mathrm{acc}}(G \parallel S_1 \parallel S_2)$ and $\mu \in \Sigma_u$ and $(x_G, \hat{v}) \xrightarrow{\mu} (y_G, \hat{w})$ in G. Following Definition 7, it is to be shown that there exists a location (y_1, y_2) of $S_1 \parallel S_2$ such that

$$(x_G, x_1, x_2, \hat{v}) \xrightarrow{\mu} (y_G, y_1, y_2, \hat{w}) \qquad \text{in } G \parallel S_1 \parallel S_2 \ . \tag{5.21}$$

By Lemma 29 it follows from $(x_G, x_1, x_2, \hat{v}) \in Q^{\mathrm{acc}}(G \parallel S_1 \parallel S_2)$ that $(x_G, x_1, \hat{v}{\restriction_{V_1}}) \in Q^{\mathrm{acc}}(G \parallel S_1)$. Since S_1 is a supremal supervisor for E_1 with respect to G and Σ_u, it holds by Definition 8 (ii) that S_1 is Σ_u-controllable with respect to G, so by Definition 7 there exists a location y_1 of S_1 such that $(x_G, x_1, \hat{v}{\restriction_{V_1}}) \xrightarrow{\mu} (y_G, y_1, \hat{w}{\restriction_{V_1}})$ in $G \parallel S_1$. This means by Lemma 2 that there are transitions $x_G \xrightarrow{\mu:p_G} y_G$ in G with $(\hat{v}{\restriction_{V_1}} \oplus (\hat{w}{\restriction_{V_1}})')(p_G) = \mathrm{true}$ and $x_1 \xrightarrow{\mu:p_1} y_1$ in S_1 with $(\hat{v}{\restriction_{V_1}} \oplus (\hat{w}{\restriction_{V_1}})')(p_1) = \mathrm{true}$ such that

$$\hat{v}{\restriction_{V_1}}(z) = \hat{w}{\restriction_{V_1}}(z) \qquad \text{for all } z \in V_1 \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_1)) \ . \tag{5.22}$$

Likewise $(x_G, x_2, \hat{v}{\restriction_{V_1}}) \in Q^{\mathrm{acc}}(G \parallel S_2)$, and there are transitions $x_G \xrightarrow{\mu:q_G} y_G$ in G and $x_2 \xrightarrow{\mu:q_2} y_2$ in S_2 with $(\hat{v}{\restriction_{V_2}} \oplus (\hat{w}{\restriction_{V_2}})')(q_2) = \mathrm{true}$ such that

$$\hat{v}{\restriction_{V_2}}(z) = \hat{w}{\restriction_{V_2}}(z) \qquad \text{for all } z \in V_2 \setminus (\mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_2)) \ . \tag{5.23}$$

Consider a variable $z \in V \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_1) \cup \mathrm{vars}'(q_2))$. Then $z \in V_1$ or $z \in V_2$. If $z \in V_1$ then $z \in V_1 \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_1) \cup \mathrm{vars}'(q_2)) \subseteq V_1 \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_1))$ and $\hat{v}(z) = \hat{v}{\restriction_{V_1}}(z) = \hat{w}{\restriction_{V_1}}(z) = \hat{w}(z)$ by (5.22). If $z \in V_2$ then $z \in V_2 \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(p_1) \cup \mathrm{vars}'(q_2)) \subseteq V_2 \setminus (\mathrm{vars}'(p_G) \cup \mathrm{vars}'(q_2)) = V_2 \setminus (\mathrm{vars}'(G, \mu) \cup \mathrm{vars}'(q_2)) = V_2 \setminus (\mathrm{vars}'(q_G) \cup \mathrm{vars}'(q_2))$ as G is normalised, and $\hat{v}(z) = \hat{v}{\restriction_{V_2}}(z) = \hat{w}{\restriction_{V_2}}(z) = \hat{w}(z)$ by (5.23). The claim (5.21) follows by Lemma 2. □

5.3 Proof of Least Restrictiveness

The last step of the proof for two separate supervisors is to show the least restrictiveness of the combined supervisor. For ordinary EFSMs, if S_1 is least restrictive for E_1 and S_2 is least restrictive for E_2, then any alternative supervisor for $E_1 \parallel E_2$ must be behaviourally included in each of E_1 and E_2, and still be controllable with respect to the same plant, so it is more restrictive than both the least restrictive supervisors S_1 and S_2, and then also their combination. The proof of the following Proposition 32 performs these arguments for EFSMs.

Proposition 32 Let G be a normalised EFSM, let E_1 and E_2 be pure EFSMs, and let Σ_u be a set of events. Further let S_1 and S_2 be supremal supervisors for E_1 and E_2, respectively, with respect to G and Σ_u, and let S' be an EFSM such that $G \parallel S' \subseteq_v G \parallel E_1 \parallel E_2$ and S' is Σ_u-controllable with respect to G. Then $G \parallel S' \subseteq_v G \parallel S_1 \parallel S_2$.

Proof. Note that $G \| S' \subseteq_v G \| E_1$ by Lemmas 6 (i) and 5 (ii) as $G \| S' \subseteq_v G \| E_1 \| E_2 \subseteq_v G \| E_1$. As S' is also Σ_u-controllable with respect to G and S_1 is a supremal supervisor for E_1 with respect to G and Σ_u, it holds by Definition 8 (iii) that $G \parallel S' \subseteq_v G \parallel S_1$. Likewise it can be shown that $G \parallel S' \subseteq_v G \parallel S_2$.

Let $V_1 = \mathrm{vars}(G) \cup \mathrm{vars}(S') \cup \mathrm{vars}(S_1)$ and $V_2 = \mathrm{vars}(G) \cup \mathrm{vars}(S') \cup \mathrm{vars}(S_2)$ and $V = V_1 \cup V_2 = \mathrm{vars}(G) \cup \mathrm{vars}(S') \cup \mathrm{vars}(S_1) \cup \mathrm{vars}(S_2)$, and consider a path

$$(x_G^0, x_S^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_G^n, x_S^n, \hat{v}^n) \qquad \text{in } G \parallel S' \tag{5.24}$$

where $\hat{v}^i \in \mathrm{dom}(V)$. As $G \parallel S' \subseteq_v G \parallel S_1$ and $G \parallel S' \subseteq_v G \parallel S_2$, by Definition 6 there exist paths

$$(x_{G1}^0, x_{S1}^0, \hat{v}^0{\restriction}_{V_1}) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_{G1}^n, x_{S1}^n, \hat{v}^n{\restriction}_{V_1}) \qquad \text{in } G \parallel S_1 \ ; \tag{5.25}$$

$$(x_{G2}^0, x_{S2}^0, \hat{v}^0{\restriction}_{V_2}) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_{G2}^n, x_{S2}^n, \hat{v}^n{\restriction}_{V_2}) \qquad \text{in } G \parallel S_2 \ . \tag{5.26}$$

It will be shown that

$$(x_{G1}^0, x_{S1}^0, x_{S2}^0, \hat{v}^0) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_n} (x_{G1}^n, x_{S1}^n, x_{S2}^n, \hat{v}^n) \qquad \text{in } G \parallel S_1 \parallel S_2 \ . \tag{5.27}$$

Clearly, x_{G1}^0, x_{S1}^0, x_{S2}^0, and $\hat{v}^0$ are initial locations and variable values from (5.24)–(5.26). Now fix $i = 0, \ldots, n-1$. By (5.25) there are transitions $x_{G1}^i \xrightarrow{\sigma_{i+1}:p_{G1}} x_{G1}^{i+1}$ in G with $(\hat{v}^i{\restriction}_{V_1} \oplus (\hat{v}^{i+1}{\restriction}_{V_1})')(p_{G1}) = \text{true}$ and $x_{S1}^i \xrightarrow{\sigma_{i+1}:p_{S1}} x_{S1}^{i+1}$ in S_1 with $(\hat{v}^i{\restriction}_{V_1} \oplus (\hat{v}^{i+1}{\restriction}_{V_1})')(p_{S1}) = \text{true}$ such that

$$\hat{v}^i{\restriction}_{V_1}(z) = \hat{v}^{i+1}{\restriction}_{V_1}(z) \qquad \text{for all } z \in V_1 \setminus (\mathrm{vars}'(p_{G1}) \cup \mathrm{vars}'(p_{S1})) \ . \tag{5.28}$$

By (5.26) there are transitions $x_{G2}^i \xrightarrow{\sigma_{i+1}:p_{G2}} x_{G2}^{i+1}$ in G and $x_{S2}^i \xrightarrow{\sigma_{i+1}:p_{S2}} x_{S2}^{i+1}$ in S_2 with $(\hat{v}^i{\restriction}_{V_2} \oplus (\hat{v}^{i+1}{\restriction}_{V_2})')(p_{S2}) = \text{true}$ such that

$$\hat{v}^i{\restriction}_{V_2}(z) = \hat{v}^{i+1}{\restriction}_{V_2}(z) \qquad \text{for all } z \in V_2 \setminus (\mathrm{vars}'(p_{G2}) \cup \mathrm{vars}'(p_{S2})) \ . \tag{5.29}$$

Then clearly $(\hat{v}^i \oplus (\hat{v}^{i+1})')(p_{G1}) = \text{true}$ and $(\hat{v}^i \oplus (\hat{v}^{i+1})')(p_{S1}) = \text{true}$ and $(\hat{v}^i \oplus (\hat{v}^{i+1})')(p_{S2}) = \text{true}$. Consider $z \in V \setminus (\text{vars}'(p_{G1}) \cup \text{vars}'(p_{S1}) \cup \text{vars}'(p_{S2}))$. Then $z \in V_1$ or $z \in V_2$. If $z \in V_1$ then $z \in V_1 \setminus (\text{vars}'(p_{G1}) \cup \text{vars}'(p_{S1}) \cup \text{vars}'(p_{S2})) \subseteq V_1 \setminus (\text{vars}'(p_{G1}) \cup \text{vars}'(p_{S1}))$ and $\hat{v}^i(z) = \hat{v}^{i+1}(z)$ by (5.28). If $z \in V_2$ then $z \in V_2 \setminus (\text{vars}'(p_{G1}) \cup \text{vars}'(p_{S1}) \cup \text{vars}'(p_{S2})) \subseteq V_2 \setminus (\text{vars}'(p_{G1}) \cup \text{vars}'(p_{S2})) = V_2 \setminus (\text{vars}'(G, \sigma_{i+1}) \cup \text{vars}'(p_{S2})) = V_2 \setminus (\text{vars}'(p_{G2}) \cup \text{vars}'(p_{S2}))$ as G is normalised, and $\hat{v}^i(z) = \hat{v}^{i+1}(z)$ by (5.29). It follows by Lemma 2 that $(x^i_{G1}, x^i_{S1}, x^i_{S2}, \hat{v}^i) \xrightarrow{\sigma_{i+1}} (x^{i+1}_{G1}, x^{i+1}_{S1}, x^{i+1}_{S2}, \hat{v}^{i+1})$ in $G \| S_1 \| S_2$, and the path (5.27) is obtained by repeating this argument for all i. □

5.4 Correctness Proof of Algorithm 2

This section combines and extends the results from the preceding Sections 5.1–5.3 to show the correctness of Algorithm 2. First, Proposition 33 shows that the composition of two supervisors synthesised for two different specifications is a supremal supervisor for the composition of these specifications, and then Proposition 34 generalises this result for an arbitrary number of specifications using induction.

Proposition 33 Let G be a normalised EFSM, let E_1 and E_2 be pure EFSMs, and let Σ_u be a set of events. Let S_1 and S_2 be supremal supervisors for E_1 and E_2, respectively, with respect to G and Σ_u. Then $S_1 \parallel S_2$ is a supremal supervisor for $E_1 \parallel E_2$ with respect to G and Σ_u.

Proof. It is to be shown that $S_1 \parallel S_2$ satisfies the conditions (i)–(iii) from Definition 8 for a supremal supervisor. Condition (i) follows from Proposition 30, condition (ii) follows from Proposition 31, and condition (iii) follows from Proposition 32. □

Proposition 34 Let G be a normalised EFSM, let $E_1, \ldots, E_k$ be pure EFSMs for some $k \geq 0$, let Σ_u be a set of events, and let S_j be a supremal supervisor for E_j with respect to G and Σ_u for $j = 1, \ldots, k$. Then $S_1 \parallel \cdots \parallel S_k$ is a supremal supervisor for $E_1 \parallel \cdots \parallel E_k$ with respect to G and Σ_u.

Proof. Write $E = E_1 \parallel \cdots \parallel E_k$ and $S = S_1 \parallel \cdots \parallel S_k$. The claim is shown by induction on k.

For the base case, $k = 0$, let $U = \big\|(\emptyset)$ denote the neutral element of synchronous composition which satisfies $F \parallel U = F$ for every EFSM F, and note that $E = E_1 \parallel \cdots \parallel E_k = U$ and $S = S_1 \parallel \cdots \parallel S_k = U$ for $k = 0$. Then it is to be shown that $S = U$ satisfies the conditions (i)–(iii) from Definition 8 for a supremal supervisor. For condition (i), note that $G \parallel S = G \parallel U \subseteq_\text{v} G \parallel U = G \parallel E$ by Lemma 5 (i). For condition (ii), note that U is trivially controllable with respect to any plant and uncontrollable event set by Definition 7. For condition (iii), assume an alternative supervisor S' such that $G \parallel S' \subseteq_\text{v} G \parallel E$. Then clearly $G \parallel S' \subseteq_\text{v} G \parallel E = G \parallel U = G \parallel S$.

Now assume the claim has been shown for some $k \geq 0$, and consider $E = E_1 \| \cdots \| E_{k+1}$ and $S = S_1 \| \cdots \| S_{k+1}$. By inductive assumption $S_1 \| \cdots \| S_k$ is a supremal supervisor for $E_1 \| \cdots \| E_k$ with respect to G and Σ_u, and by assumption S_{k+1} is a supremal supervisor for E_{k+1} with respect to G and Σ_u. Then it follows by Proposition 33 that $S = (S_1 \parallel \cdots \parallel S_k) \parallel S_{k+1}$ is a supremal supervisor for $E = (E_1 \parallel \cdots \parallel E_k) \parallel E_{k+1}$ with respect to G and Σ_u. □

Finally, the result about separate synthesis is used to prove the correctness of Algorithm 2. Each iteration in the loop gives a supremal supervisor by Theorems 22 and 28, and then their combination also is a supremal supervisor by Proposition 34.

Theorem 35 Upon termination of Algorithm 2, the composition $\big\|(\mathcal{S})$ of the results is a supremal supervisor for the composed specification $\big\|(\mathcal{E})$ with respect to the composed plant $\big\|(\mathcal{G})$ and uncontrollable event set Σ_u.

Proof. Write $\big\|(\mathcal{G}) = G_1 \| \cdots \| G_m$ and $\big\|(\mathcal{E}) = E_1 \| \cdots \| E_k$. The loop on line 2 of Algorithm 2 calculates S_j for each $j = 1, \ldots, k$ and on line 5 adds both S_j and E_j to the result $\mathcal{S}$, making

$$\big\|(\mathcal{S}) = (S_1 \parallel E_1) \parallel \cdots \parallel (S_k \parallel E_k) \ . \tag{5.30}$$

Line 4 calculates S_j using Algorithm 1 with $E = \exists \bar{V}_j \, E_j$, so it follows from Theorem 22 that S_j is a supremal supervisor for $\exists \bar{V}_j \, E_j$ with respect to $\big\|(\mathcal{G})$ and Σ_u. On line 3, the variables are chosen such that $\bar{V}_j \subseteq \mathrm{vars}(E_j) \setminus \mathrm{vars}(E_j, \Sigma_\mathrm{u})$, which implies $\bar{V}_j \cap \mathrm{vars}(E_j, \Sigma_\mathrm{u}) = \emptyset$, and $\exists \bar{V}_j \, E_j$ is state-deterministic, so it follows from Theorem 28 that each $S_j \parallel E_j$ is a supremal supervisor for E_j with respect to $\big\|(\mathcal{G})$ and Σ_u. Then it follows by Proposition 34 from (5.30) that $\big\|(\mathcal{S})$ is a supremal supervisor for $\big\|(\mathcal{E}) = E_1 \parallel \cdots \parallel E_k$ with respect to $\big\|(\mathcal{G})$ and Σ_u. □

It is also clear that Algorithm 2 terminates because the loop performs exactly one iteration for each specification $E_1, \ldots, E_k$. Therefore, it is concluded that the algorithm is totally correct and can be used to compute a least restrictive supervisor for any combination of normalised state-deterministic plant and pure state-deterministic specification EFSMs.

6 Flexible Manufacturing System Example

This section applies the proposed synthesis procedure from Algorithm 2 to compute a modular least restrictive controllable supervisor for an EFSM model of a flexible manufacturing system. Figure 4 shows the layout of this example.

Workpieces enter the system through one of two feeders (F_1 or F_2) and are placed on the first conveyor (C_1), which delivers them to the first production line (L_1). In L_1, the workpieces may be processed by the first machine (M_1) and then put on the second conveyor (C_2), or they may be put on C_2 immediately without processing. After passing conveyor C_2 the workpieces

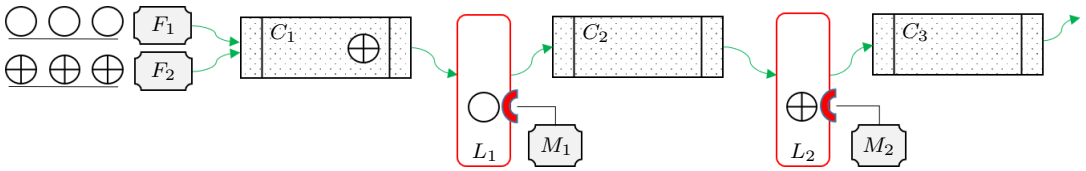

Figure 4: Flexible Manufacturing System Layout.

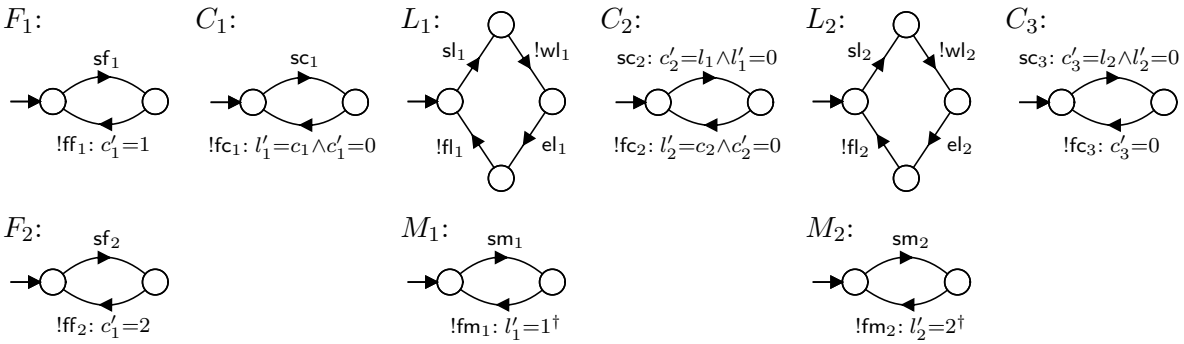

Figure 5: Flexible Manufacturing System Plants.

may be processed by a similar production line (L_2) and machine (M_2), and after that they are put on the last conveyor (C_3) before exiting the system.

The two feeders F_1 and F_2 provide two different types of workpieces: F_1 delivers type 1 workpieces, and F_2 delivers type 2 workpieces. The decision which feeder is used is outside of the scope of the model. The objective is to control the system in such a way that type 1 workpieces are only processed by machine M_1 and do not enter production line L_2, and likewise type 2 workpieces are only processed by M_2.

The modelling of the different workpiece types is facilitated by the use of EFSM variables, as demonstrated in the plant model in Figure 5. The variables c_1, c_2, c_3, l_1, and l_2 represent the contents of the conveyors and production lines. Their domain is $\{0, 1, 2, 1^\dagger, 2^\dagger\}$, where a value of 0 means that the corresponding conveyor or production line is empty, a value of 1 or 2 indicates the presence of a raw workpiece of type 1 or 2, and a value of $1^\dagger$ or $2^\dagger$ indicates the presence of a workpiece of type 1 or 2 that has been processed by its corresponding machine M_1 or M_2.

In the model, uncontrollable events are prefixed with an exclamation mark (!) to distinguish them from the controllable events. The plant EFSM F_1 shows that feeder F_1 can be started controllably with event sf$_1$, then finishes uncontrollably with event !ff$_1$ and upon finishing puts a type 1 workpiece on conveyor C_1 as indicated by the update $c'_1 = 1$. Plant F_2 describes the analogous behaviour of feeder F_2. Similarly, conveyor C_1 is started with sc$_1$, and upon finishing with !fc$_1$ its workpiece is put into the first production line, $l'_1 = c_1$, and

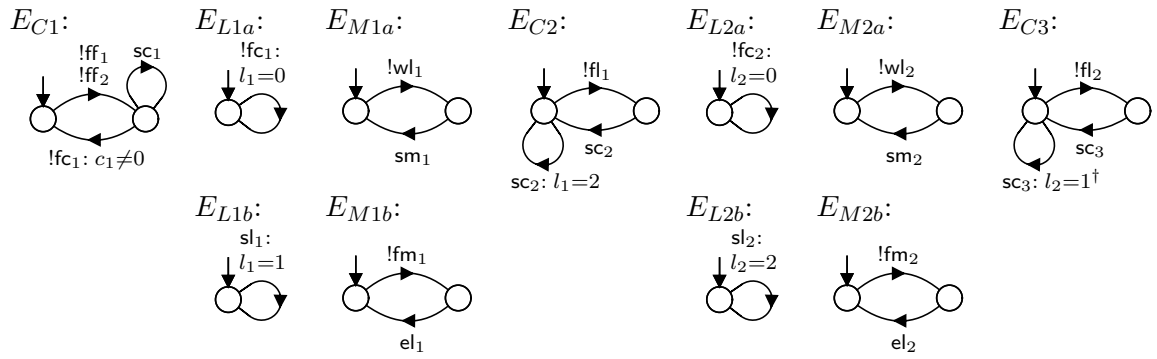

Figure 6: Flexible Manufacturing System Specifications.

removed from the conveyor, $c_1' = 0$. Conveyors C_2 and C_3 are similar, but in addition remove a workpiece from the production line in front of them when starting. Production line L_1 is requested to pick up a workpiece with controllable event sl_1, and the completion of the pick-up is indicated by uncontrollable event $\mathsf{!wl}_1$ after which the workpiece is available for machine M_1. Then the production line can be requested to eject the workpiece (el_1) and upon completion ($\mathsf{!fl}_1$) the workpiece again becomes available for conveyor C_2. Machine M_1 can be requested to start processing (sm_1), and when it finishes ($\mathsf{!fm}_1$) it changes the workpiece in the production line, $l_1' = 1^\dagger$, to indicate a processed workpiece of type 1. Production line L_2 and machine M_2 work in the same way.

Figure 6 shows specification EFSMs that capture several control requirements for the flexible manufacturing system. Specification E_{C1} describes the requirement that conveyor C_1 can only start (sc_1) after having been loaded with a workpiece, i.e., after one of the feeders has completed ($\mathsf{!ff}_1$ or $\mathsf{!ff}_2$), and through the guard $c_1 \neq 0$ also requires that there must be a workpiece on the conveyor when it finishes. Specification E_{L1a} rules out overflow of production line L_1, because conveyor C_1 is only allowed to deliver a workpiece ($\mathsf{!fc}_1$) when the line is empty, $l_1 = 0$. Specification E_{L1b} requires that only unprocessed type 1 workpiece may enter production line L_1. Specifications E_{M1a} and E_{M1b} require that machine M_1 only starts (sm_1) when there is a workpiece for it to process ($\mathsf{!wl}_1$), and the workpiece is only ejected (el_1) after being processed by the machine ($\mathsf{!fm}_1$). Specification E_{C2} constrains the starting (sc_2) of conveyor C_2: conveyor C_2 may start when production line L_1 contains a type 2 workpiece, $l_1 = 2$, as these workpieces should bypass L_1, or after production line L_1 has returned a processed workpiece ($\mathsf{!fl}_1$). Specifications E_{L2a}, E_{L2b}, E_{M2a}, E_{M2b}, and E_{C3} constrain the behaviour of production line L_2 and conveyor C_3 in a similar way.

It is clear that the EFSM model satisfies the structural requirements outlined for Algorithms 1 and 2. All the plants are normalised and all the specifications are pure. Also, all the EFSMs are state-deterministic, and so are all possible abstractions as no location has more

than one outgoing transition for any given event. Moreover, it can be seen in Figure 5 that for any given event, no next-state variable appears in more than one EFSM on transitions with that event, so that condition (3.5) will be trivially satisfied for any abstraction considered. It is quite typical for well-designed EFSM models to have such properties, particularly in the manufacturing context.

To synthesise a least restrictive supervisor, Algorithm 2 processes each of the specifications in Figure 6. The order in which the specifications are processed is not important, so the following explanation starts with the easiest cases. The supervisors computed by Algorithm 1 are shown in Figure 7.

- Specification E_{L1b} has only one controllable event, sl_1, so its variable l_1 is only used controllably and can be abstracted. Algorithm 2 forms the abstraction $\exists l_1 E_{L1b}$, and as $\exists l_1\ l_1 = 1$ is true, this simplifies to a one-state FSM with a controllable selfloop, which is trivially controllable and returned unchanged by Algorithm 1.

  Then the abstraction $\exists l_1 E_{L1b}$ becomes the first supervisor collected by Algorithm 2. It appears in Figure 7 as S_{L1b}. It performs no control as a supervisor. Therefore Algorithm 2 also includes the original specification E_{L1b} as a supervisor, which ensures through the update $l_1 = 1$ that production line L_1 only starts (sl_1) when a workpiece of type 1 is available.

- Specification E_{M1a} has no variables, so Algorithm 2 passes it to Algorithm 1 unchanged. As E_{M1a} disables the uncontrollable event $!\mathsf{wl}_1$, it is found not controllable at the beginning of Algorithm 1, so the algorithm assigns $\Sigma_{\mathrm{u}}^1 = \{!\mathsf{wl}_1\}$ and searches for plants that disable this cause of uncontrollability. This yields L_1, which also has no variables. Therefore Algorithm 1 chooses $\mathcal{G}^1 = \{L_1\}$, $V^1 = \emptyset$, and $\mathcal{C}^1 = \emptyset$. Synthesis results in the supervisor $S^1 = \sup\mathcal{C}(L_1, E_{M1a}, \{!\mathsf{wl}_1\})$, which is also $!\mathsf{fl}_1$-controllable.

  This supervisor is shown as S_{M1a} in Figure 7. It is returned from Algorithm 1 and collected by Algorithm 2. The supervisor S_{M1a} ensures that machine M_1 is only started (sm_1) when a workpiece is available, i.e., after $!\mathsf{wl}_1$ has occurred.

- Specification E_{M1b} has no variables, so Algorithm 2 passes it to Algorithm 1 unchanged. As E_{M1b} disables the uncontrollable event $!\mathsf{fm}_1$, it is found not controllable at the beginning of Algorithm 1, so the algorithm assigns $\Sigma_{\mathrm{u}}^1 = \{!\mathsf{fm}_1\}$ and searches for plants that disable this cause of uncontrollability. This yields M_1, which includes the variable l_1. Yet on closer inspection $!\mathsf{fm}_1$ is unconstrained in M_1 with respect to l_1 (note that $\forall l_1 \exists l_1'\ l_1' = 1$ is true, which is enough to establish the validity of (2.92) in Definition 14). Therefore Algorithm 1 chooses the plant abstraction $\mathcal{G}^1 = \{\exists l_1 M_1\}$, no variables, $V^1 = \emptyset$, and no chaos EFSMs, $\mathcal{C}^1 = \emptyset$. Then synthesis results in the supervisor $S^1 = \sup\mathcal{C}(\exists l_1 M_1, E_{M1b}, \{!\mathsf{fm}_1\})$, which is controllable as no other uncontrollable events are involved.

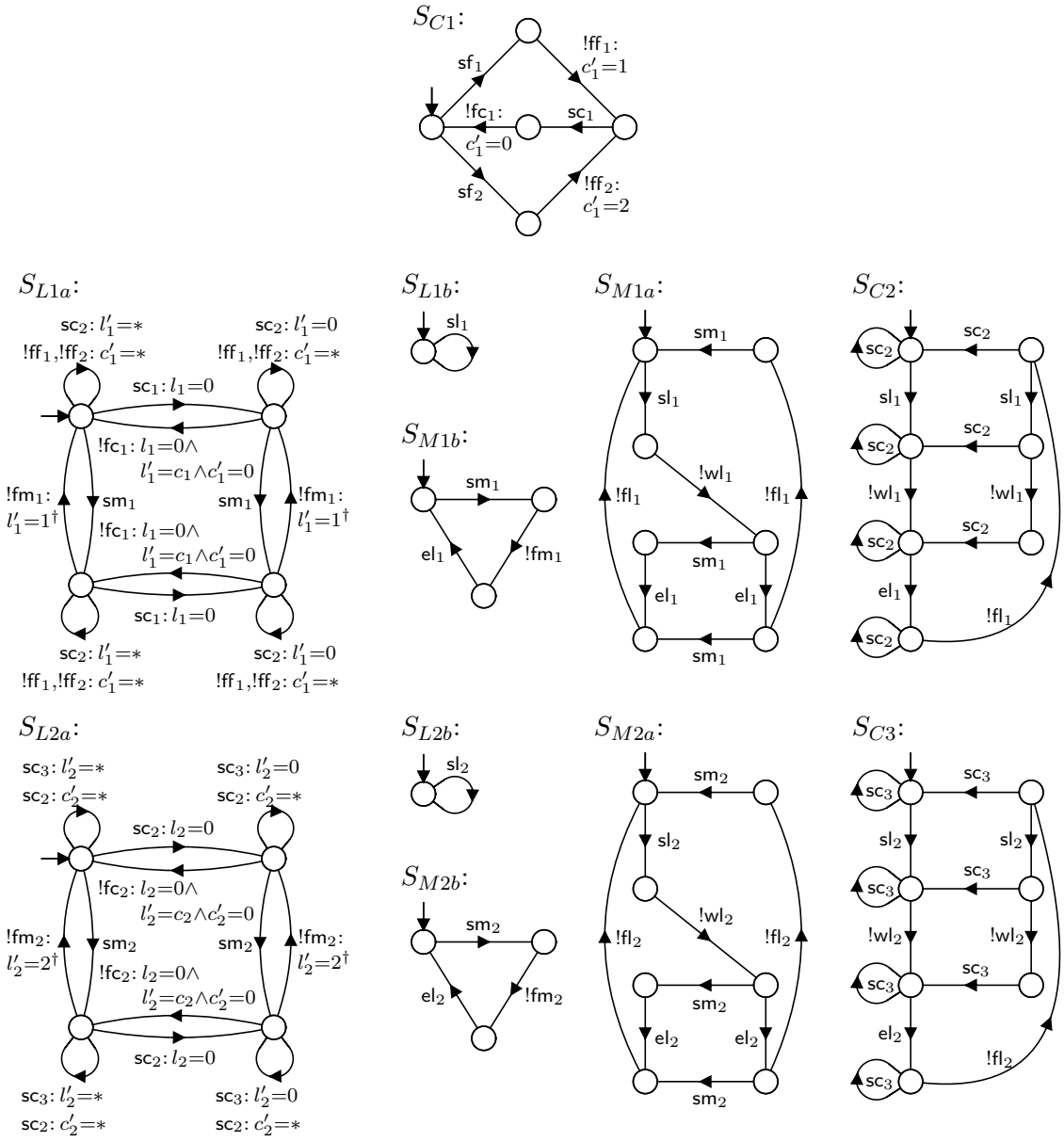

Figure 7: Synthesised supervisors for flexible manufacturing system.

This supervisor, shown as S_{M1b} in Figure 7, is returned from Algorithm 1 and collected by Algorithm 2. It ensures that production line L_1 only starts the ejection (el_1) of a workpiece after it has been processed ($!\mathsf{fm}_1$) by machine M_1.

- Specification E_{C1} includes the variable c_1, which is used uncontrollably with event $!\mathsf{fc}_1$ and cannot be abstracted by Algorithm 2, so E_{C1} is passed to Algorithm 1 unchanged. This specification is not controllable by itself as it disables uncontrollable events $!\mathsf{ff}_1$, $!\mathsf{ff}_2$, and $!\mathsf{fc}_1$. In its first iteration, Algorithm 1 selects $\Sigma_u^1 = \{!\mathsf{ff}_1, !\mathsf{ff}_2, !\mathsf{fc}_1\}$, which leads it to identify the plants F_1, F_2, and C_1 and the variable c_1 that appears in E_{C1}. Plant C_1 also includes the variable l_1, but Σ_u^1 is unconstrained in C_1 with respect to l_1, because for the $!\mathsf{fc}_1$-transition in C_1 the condition (2.92) in Definition 14 becomes

$$\exists l_1 \exists l_1' (l_1' = c_1 \wedge c_1' = 0) \Rightarrow \forall l_1 \exists l_1' (l_1' = c_1 \wedge c_1' = 0) \ , \tag{6.1}$$

  which can be simplified by removing the unused variable l_1 from quantification to give

$$\exists l_1' (l_1' = c_1 \wedge c_1' = 0) \Rightarrow \exists l_1' (l_1' = c_1 \wedge c_1' = 0) \ , \tag{6.2}$$

  and the latter is clearly valid. This shows that the transition is possible independently of the current value of l_1. Therefore Algorithm 1 replaces C_1 by the abstraction $\exists l_1 C_1$, which amounts to changing the update of the $!\mathsf{fc}_1$-transition to $c_1' = 0$. No chaos EFSMs are needed as c_1 does not appear in any other plant, so that $\mathcal{G}^1 = \{F_1, F_2, \exists l_1 C_1\}$, $V^1 = \{c_1\}$, and $\mathcal{C}^1 = \emptyset$. Synthesis results in $S^1 = \sup\mathcal{C}(F_1 \parallel F_2 \parallel \exists l_1 C_1, E_{C1}, \{!\mathsf{ff}_1, !\mathsf{ff}_2, !\mathsf{fc}_1\})$, which is controllable as no other uncontrollable events are involved.

  This supervisor, shown as S_{C1} in Figure 7, is returned and collected by Algorithm 2. It ensures that conveyor C_1 is only started after delivery of a workpiece from a feeder, and the feeders only start when the conveyor is empty.

- Specification E_{C2} includes the variable l_1, but it is only used with the controllable event sc_2. Then Algorithm 2 passes $\exists l_1 E_{C2}$ to Algorithm 1, which amounts to deletion of the update from E_{C2}. In Algorithm 1, the specification is not controllable by itself as it disables the uncontrollable event $!\mathsf{fl}_1$. So the algorithm sets $\Sigma_u^1 = \{!\mathsf{fl}_1\}$ and finds that $!\mathsf{fl}_1$ is only disabled by plant L_1, which has no variables. Then $\mathcal{G}^1 = \{L_1\}$, $V^1 = \emptyset$, $\mathcal{C}^1 = \emptyset$, and synthesis gives $S^1 = \sup\mathcal{C}(L_1, \exists l_1 E_{C2}, \{!\mathsf{fl}_1\})$, which is also found to be $!\mathsf{wl}_1$-controllable

  This supervisor, shown as S_{C2} in Figure 7, is returned and collected by Algorithm 2. Together with the original specification E_{C2}, which also is collected by Algorithm 2, it ensures that conveyor C_2 only removes type 2 workpieces that should not be processed by production line L_1 or type 1 workpieces processed by L_1.

- Specification E_{L1a} includes variable l_1, which is used with the uncontrollable event $!\mathsf{fc}_1$ and therefore cannot be abstracted. Thus Algorithm 2 passes E_{L1a} unchanged to Algorithm 1. At the beginning of Algorithm 1, chaos EFSMs are constructed for all

events that can change the variable l_1 in some plant, resulting in $\mathcal{C}^0 = \{\text{chaos}(!\mathsf{fc}_1, l_1), \text{chaos}(!\mathsf{fm}_1, l_1), \text{chaos}(\mathsf{sc}_2, l_1)\}$. Then E_{L1} is not controllable with respect to $\|(\mathcal{C}^0)$ because l_1 can uncontrollably change from its initial value 0 by $\text{chaos}(!\mathsf{fc}_1, l_1)$, and afterwards $!\mathsf{fc}_1$ is possible with the specification's guard $l_1 = 0$ being false.

The only uncontrollable event in the specification and cause of uncontrollability is $!\mathsf{fc}_1$, so $\Sigma_{\text{u}}^1 = \{!\mathsf{fc}_1\}$. This uncontrollable event is disabled by plant C_1, which therefore is included in the plant abstraction. C_1 also includes the variable c_1, which is *not* unconstrained, so that $\mathcal{G}^1 = \{C_1\}$ and $V^1 = \{c_1, l_1\}$. Apart from the selected plant C_1, the variable l_1 is still assigned on event $!\mathsf{fm}_1$ in M_1 and on event sc_2 in C_2, and the variable c_1 is assigned on event $!\mathsf{ff}_1$ in F_1 and on event $!\mathsf{ff}_2$ in F_2. Therefore the chaos EFSMs $\mathcal{C}^1 = \{\text{chaos}(!\mathsf{ff}_1, c_1), \text{chaos}(!\mathsf{ff}_2, c_2), \text{chaos}(!\mathsf{fm}_1, l_1), \text{chaos}(\mathsf{sc}_2, l_1)\}$ are included. Synthesis gives a supervisor

$$S^1 = \sup\mathcal{C}(C_1 \parallel \text{chaos}(!\mathsf{ff}_1, c_1) \parallel \text{chaos}(!\mathsf{ff}_2, c_1) \parallel \text{chaos}(!\mathsf{fm}_1, l_1) \parallel \text{chaos}(\mathsf{sc}_2, l_1), \quad E_{L1a}, \{!\mathsf{fc}_1\}) \ , \tag{6.3}$$

but it is not $!\mathsf{fm}_1$-controllable. This is because S^1 is synthesised under the pretence that $!\mathsf{fm}_1$ is controllable, and then S^1 can constrain $!\mathsf{fm}_1$ to prevent the system from entering states with $l_1 \neq 0$ where $!\mathsf{fc}_1$ is enabled. For example, S^1 allows sc_1 when initially $l_1 = 0$ and then disables $!\mathsf{fm}_1$ to prevent it from changing l_1 to a non-zero value before $!\mathsf{fc}_1$ occurs. But this is not acceptable since $!\mathsf{fm}_1$ really is uncontrollable.

Therefore Algorithm 1 enters another iteration with $\Sigma_{\text{u}}^2 = \{!\mathsf{fc}_1, !\mathsf{fm}_1\}$. Now plant M_1 must also be included as it disables $!\mathsf{fm}_1$. There are no additional variables in M_1 so that $\mathcal{G}^2 = \{C_1, M_1\}$ and $V^2 = \{c_1, l_1\}$. Outside of the selected plants C_1 and M_1, the variable l_1 is only assigned on sc_2 in C_2, so that $\mathcal{C}^2 = \{\text{chaos}(!\mathsf{ff}_1, c_1), \text{chaos}(!\mathsf{ff}_2, c_2), \text{chaos}(\mathsf{sc}_2, l_1)\}$. Then another supervisor is synthesised,

$$S^2 = \sup\mathcal{C}(C_1 \parallel M_1 \parallel \text{chaos}(!\mathsf{ff}_1, c_1) \parallel \text{chaos}(!\mathsf{ff}_2, c_1) \parallel \text{chaos}(\mathsf{sc}_2, l_1), \quad E_{L1a}, \{!\mathsf{fc}_1, !\mathsf{fm}_1\}) \ , \tag{6.4}$$

and this supervisor is also found to be controllable with respect to the remaining uncontrollable events $!\mathsf{ff}_1$ and $!\mathsf{ff}_2$.

The supervisor, shown as S_{L1a} in Figure 7, is returned and collected by Algorithm 2. It avoids overflow of production line L_1, because it only allows the conveyor C_1 to start delivery of a new workpiece (sc_1) when the production line is empty, $l_1 = 0$.

- The remaining specifications E_{L2b}, E_{M2a}, E_{M2b}, E_{C3}, and E_{L2a} are processed in a similar way as those above, resulting in further supervisors S_{L2b}, S_{M2a}, S_{M2b}, S_{C3}, and S_{L2a}.

On completion, Algorithm 2 returns the supervisors shown in Figure 7 plus the original specifications in Figure 6, which together control the flexible manufacturing system in the least restrictive controllable way. The largest supervisor EFSMs have seven locations, and the largest state spaces encountered are 100 unfolded states during synthesis of S_{L1a} and S_{L2a}. In comparison, a full monolithic synthesis for all the plants and specifications together, explores a state space of 14580 unfolded states and results in a single supervisor EFSM with 464 locations and 1551 unfolded states.

7 Conclusions

This working paper presents an algorithm that combines modular and abstraction-based synthesis for extended finite-state machines (EFSM). The approach allows to calculate supervisors that control a system in the least restrictive controllable way. Through a combination of component selection and symbolic manipulation by means of existential quantification, the method avoids the exploration of the full state space as normally required in synthesis. The resulting supervisors are modular and can be presented as the composition of several small EFSMs, which also facilitates human readability.

These results improve the authors' previous work [10,23] in that they allow abstraction by both component and variable selection. In future research, the authors would like to consider the modular synthesis of nonblocking supervisors for EFSMs using abstractions. It is also of interest whether supervisor reduction techniques [20] can be used to remove redundant transitions and simplify guards in the synthesised supervisor EFSMs.

References

[1] Åkesson, K., Flordal, H., Fabian, M.: Exploiting modularity for synthesis and verification of supervisors. In: Proceedings of the 15th IFAC World Congress on Automatic Control (2002)

[2] Brandin, B.A., Malik, R., Malik, P.: Incremental verification and synthesis of discrete-event systems guided by counter-examples. IEEE Transactions on Control Systems Technology **12**(3), 387–401 (2004). DOI 10.1109/TCST.2004.824795

[3] Cassandras, C.G., Lafortune, S.: Introduction to Discrete Event Systems, 2 edn. Springer Science & Business Media, New York, NY, USA (2008)

[4] Chen, Y., Lin, F.: Modeling of discrete event systems using finite state machines with parameters. In: Proceedings of 2000 IEEE International Conference on Control Applications (CCA), pp. 941–946 (2000). DOI 10.1109/CCA.2000.897591

[5] Cury, J.E.R., de Queiroz, M.H., Bouzon, G., Teixeira, M.: Supervisory control of discrete event systems with distinguishers. Automatica **56**, 93–104 (2015). DOI 10.1016/j.automatica.2015.03.025

[6] Huth, M., Ryan, M.: Logic in Computer Science. Cambridge University Press, Cambridge, UK (2004)

[7] Komenda, J., van Schuppen, J.H.: Control of discrete-event systems with modular or distributed structure. Theoretical Computer Science **388**, 199–226 (2007). DOI 10.1016/j.tcs.2007.07.049

[8] Le Gall, T., Jeannet, B., Marchand, H.: Supervisory control of infinite symbolic systems using abstract interpretation. In: Proceedings of the 46th IEEE Conference on Decision and Control, CDC '05, pp. 30–35 (2005). DOI 10.1109/CDC.2005.1582126

[9] Malik, R., Teixeira, M.: An algorithm for the synthesis of least restrictive controllable supervisors for extended finite-state machines. Working Paper 01/2016, Department of Computer Science, University of Waikato, Hamilton, New Zealand (2016). URL `http://hdl.handle.net/10289/9841`

[10] Malik, R., Teixeira, M.: Modular supervisor synthesis for extended finite-state machines subject to controllability. In: Proceedings of the 13th International Workshop on Discrete Event Systems, WODES'16, pp. 117–122. IEEE (2016). DOI 10.1109/WODES.2016.7497831

[11] Miremadi, S., Åkesson, K., Lennartson, B.: Symbolic computation of reduced guards in supervisory control. IEEE Transactions on Automation Science and Engineering **8**(4), 754–764 (2011). DOI 10.1109/TASE.2011.2146249

[12] Mohajerani, S., Malik, R., Fabian, M.: A framework for compositional synthesis of modular nonblocking supervisors. IEEE Transactions on Automatic Control **59**(1), 150–162 (2014). DOI 10.1109/TAC.2013.2283109

[13] Mohajerani, S., Malik, R., Fabian, M.: A framework for compositional nonblocking verification of extended finite-state machines. Discrete Event Dynamic Systems: Theory and Applications **26**(1), 33–84 (2016). DOI 10.1007/s10626-015-0217-y

[14] Mohajerani, S., Malik, R., Fabian, M.: Compositional synthesis of supervisors in the form of state machines and state maps. Automatica **76**, 277–281 (2017). DOI http://dx.doi.org/10.1016/j.automatica.2016.10.012

[15] Ouedraogo, L., Kumar, R., Malik, R., Åkesson, K.: Nonblocking and safe control of discrete-event systems modeled as extended finite automata. IEEE Transactions on

Automation Science and Engineering **8**(3), 560–569 (2011). DOI 10.1109/TASE.2011.2124457

[16] de Queiroz, M.H., Cury, J.E.R.: Modular supervisory control of large scale discrete event systems. In: R. Boel, G. Stremersch (eds.) Discrete Event Systems: Analysis and Control, SECS 569, pp. 103–118. Kluwer Academic Publishers (2000)

[17] Ramadge, P.J.G., Wonham, W.M.: The control of discrete event systems. Proceedings of the IEEE **77**(1), 81–98 (1989). DOI 10.1109/5.21072

[18] Rosa, M., Teixeira, M., Malik, R.: Exploiting approximations in supervisory control with distinguishers. IFAC PapersOnLine **51**(7), 13–18 (2018). DOI 10.1016/j.ifacol.2018.06.272

[19] Shoaei, M.R., Feng, L., Lennartson, B.: Abstractions for nonblocking supervisory control of extended finite automata. In: Proceedings of the 8th International Conference on Automation Science and Engineering, CASE 2012, pp. 364–370 (2012). DOI 10.1109/CoASE.2012.6386446

[20] Su, R., Wonham, W.M.: Supervisor reduction for discrete-event systems. Discrete Event Dynamic Systems: Theory and Applications **14**(1), 31–53 (2004). DOI 10.1023/B:DISC.0000005009.40749.b6

[21] Tarski, A.: A lattice-theoretical fixpoint theorem and its applications. Pacific Journal of Mathematics **5**(2), 285–309 (1955)

[22] Teixeira, M., Cury, J.E.R., de Queiroz, M.H.: Exploiting distinguishers in local modular control of discrete-event systems. IEEE Transactions on Automation Science and Engineering **15**(3), 1431–1437 (2018). DOI 10.1109/TASE.2018.2793963

[23] Teixeira, M., Malik, R., Cury, J.E.R., de Queiroz, M.H.: Supervisory control of DES with extended finite-state machines and variable abstraction. IEEE Transactions on Automatic Control **60**(1), 118–129 (2015). DOI 10.1109/TAC.2014.2337411

[24] Wong, K.C., Wonham, W.M.: Modular control and coordination of discrete-event systems. Discrete Event Dynamic Systems: Theory and Applications **8**(3), 247–297 (1998). DOI 10.1023/A:1008210519960