12th IFAC/IEEE Workshop on Discrete Event Systems
Cachan, France. May 14-16, 2014

Reducing an Operational Supervisory Control Problem by Decomposition for Deterministic Pushdown Automata

S. Schneider * **A.-K. Schmuck** ** **U. Nestmann** * **J. Raisch** **,***

* *Modelle und Theorie Verteilter Systeme, Technische Universität Berlin*
** *Regelungssysteme, Technische Universität Berlin*
*** *Max-Planck-Institut für Dynamik komplexer technischer Systeme*

Abstract: The purpose of Supervisory Control Theory (SCT) is to synthesize a controller for a plant and a specification such that the desired closed-loop behavior is enforced. Effective solvers have been constructed in the past for the setting of plants and specifications modeled by Deterministic Finite Automata (DFA). We extend the domain of the specification to Deterministic Pushdown Automata (DPDA) and verify an effective solver (up to two basic building blocks which ensure controllability and blockfreeness, effectively solved for this setting in two companion papers). We verify the enforcement of desired operational criteria, which are, in contrast to the setting of DFA, partly oblivious to the (un)marked language of the closed loop. Our general approach trivially covers the setting of DFA and can be reused and adapted to develop effective solvers for other settings as the realizability of solutions to the supervisory control problem (SCP) is considered on an abstract level.

Keywords: Supervisory Control Theory, Pushdown Systems, Operational Properties, Fully Abstract Denotational Semantics, Suprema Characterizations, Fixed Point Computations

The SCP was introduced and solved for plants and specifications modeled by DFA by Ramadge and Wonham (1984); Wonham and Ramadge (1987). Subsequently, the SCP was considered for other settings including certain Petri nets by, e.g., Giua and DiCesare (1994, 1995). Later, Griffin (2008, 2007) addressed the SCP for DFA plants and DPDA specifications with prefix-closed marked languages. However, one key criterion of the SCP, namely minimal restrictiveness, is violated by his approach as explained in Schmuck, Schneider, Raisch, and Nestmann (2014).

Wonham and Ramadge (1987) synthesize the desired controller language C, formally defined by the SCP, for regular plant and specification languages (P and S, respectively), by operating on their DFA representations, generating a DFA representation of C. We perceive the involved languages P, S, and C as trace abstractions of the finite operational DFA-models. While this standard trace abstraction is sufficient for DFA and observable Petri nets we show that it is insufficient for DPDA with their unobservable λ-steps. This insight enforces the perspective of synthesizing automata realizations instead of languages.

In Section 1, the models, relevant to this paper, including DPDA, are introduced. Afterwards we define trace abstractions for DPDA and analyze their basic properties in Section 2. In Section 3 we provide operational criteria of the desired (least restrictive) controller, and formally define a reduction of synthesizing a controller satisfying these constraints to the SCP by introducing an operational SCP (OSCP) based on trace abstractions introduced before.

The operational computation of the concrete solver for the DFA setting of Ramadge and Wonham (1984), which is extensively used in the control of discrete event systems, does not correspond well to their trace-abstract characterization of the desired closed-loop behavior. To bridge this gap, (i) in Section 4, we formalize the correspondence between operational and trace-abstract solutions by suitably characterizing satisfactory controllers and verify their equivalence to the standard SCP solutions; and (ii) in Section 5, we introduce adequate fixed-point algorithms which (a) determine only satisfactory controllers and (b) are constructed over operations ensuring controllability and blockfreeness for the underlying finite state models. These fixed-point algorithms constitute solvers for the SCP for DFA plants and DPDA specifications when using the effective procedures for ensuring controllability and blockfreeness for DPDA, discussed in the two companion papers Schmuck, Schneider, Raisch, and Nestmann (2014); Schneider and Nestmann (2014). In summary, we consider the implications of choosing DPDA as specifications on SCT (a) by formalizing operational criteria for the desired controllers, (b) by reducing the synthesis problem to the standard SCP by means of an OSCP using adequate trace abstractions which guarantee the desired criteria for controller realizations, (c) by expressing the SCP by supremal elements of a complete lattice over our trace abstractions, and (d) by decomposing the suprema-based characterization into fixed-point algorithms consisting of basic building blocks implemented in the companion papers cited above.

While outlining the process of verifying the adequacy of the trace abstractions of Section 2, we have verified the automata foundations of Section 1 and the central results of Section 4 and Section 5 in the interactive theorem prover Isabelle/HOL (Paulson et al., 2011).

1. MODELS OF BEHAVIOR

We assume a fixed set of possible events Σ shared by all models contained in this paper. This set is, as usual, partitioned into a set of controllable events Σ_{c} and a set of uncontrollable events Σ_{uc}. In examples we assume $\Sigma_{\mathsf{c}} = \{a, b, c, d, e\}$ and $\Sigma_{\mathsf{uc}} = \{u, v\}$.

We use the following notation throughout the paper.

Notation 1. Let A be a set. Then (i) A^* denotes the set of finite words over A, (ii) $A^{\omega *}$ denotes the set of finite and infinite words over A, (iii) single symbols are denoted by greek letters (except for λ, the empty word), (iv) words are denoted by s, w, (v) $\cdot$ is the (usually omitted) concatenation operation on words (and languages), (vi) $\sqsubseteq$ is the prefix relation, (vii) $\overline{A}$ is the prefix-closure of A, (viii) $\sqsupseteq$ is the suffix relation, and (ix) $\lambda x.f(x)$ is the nameless function equal to f (e.g., $f(x){=}x^2$ implies $f{=}\lambda x.x^2$). □

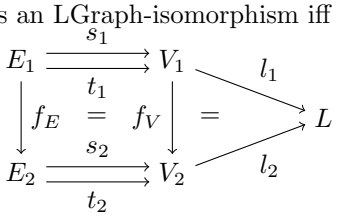
1.1 Labeled Graphs

We use labeled graphs as representations of discrete (operational) behavior, where edges correspond to steps.

Definition 1. (Labelled Graphs). $G = (V, E, L, s, t, l) \in$ LGraph iff (i) V is a set of vertices, (ii) E is a set of edges, (iii) L is a set of labels, (iv) $s : E \to V$ maps each edge to its source-vertex, (v) $t : E \to V$ maps each edge to its target-vertex, and (vi) $l : V \to 2^L$ maps each vertex to the set of its labels. □

Two operational behaviors (given as LGraphs) are equivalent iff they are renamings of each other.

Definition 2. (LGraph-Isomorphisms).
Let $G_1 = (V_1, E_1, L, s_1, t_1, l_1)$ and $G_2 = (V_2, E_2, L, s_2, t_2, l_2)$ be two LGraphs with identical sets of labels. Then $f = (f_V : V_1 \to V_2, f_E : E_1 \to E_2) : G_1 \to G_2$ containing mappings for vertices and edges is an LGraph-isomorphism iff (i) f_V and f_E are bijections, (ii) sources are preserved: $s_2 \circ f_E = f_V \circ s_1$, (iii) targets are preserved: $t_2 \circ f_E = f_V \circ t_1$, and (iv) labels are preserved: $l_1 = l_2 \circ f_V$.



Furthermore, $G_1 \cong G_2$ iff there is an LGraph-isomorphism $f : G_1 \to G_2$. □

1.2 Discrete Event Systems

We introduce Discrete Event Systems (DES) as a denotational model which is entirely decoupled from the syntax and semantics of concrete (e.g., automata) realizations. The DES is given by the sets of (i) all (possible) observations—the unmarked language L_{um} and (ii) all (possible) desired observations—the marked language L_m.

Definition 3. (Discrete Event System).
$D = \langle\!| L_{um}, L_m |\!\rangle \in$ DES iff $L_m \subseteq L_{um} = \overline{L_{um}} \subseteq \Sigma^*$. $\mathrm{L_{um}}(D)$ and $\mathrm{L_m}(D)$ denote the two components of D. □

We repeat the well known notions of language-blockfreeness and language-controllability of DES which are central to the standard SCT (cf. Ramadge and Wonham (1984)).

Definition 4. (Blockfree and Controllable DES).
Let D_1, $D_2 \in$ DES. Then (i) D_1 is language-blockfree [1] iff $\mathrm{L_{um}}(D_1) \subseteq \overline{\mathrm{L_m}(D_1)}$ and (ii) D_1 is language-controllable [2] w.r.t D_2 (denoted by $\mathrm{LCont}(\mathrm{L_{um}}(D_1), \mathrm{L_{um}}(D_2), \Sigma_{\mathsf{uc}})$)iff $(\mathrm{L_{um}}(D_1){\cdot}\Sigma_{\mathsf{uc}}) \cap \mathrm{L_{um}}(D_2) \subseteq \mathrm{L_{um}}(D_1)$. □

[1] Every observation is a prefix of a desired observation.

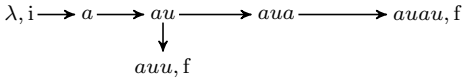
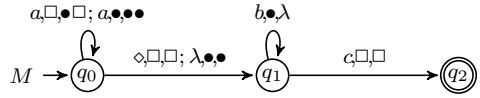


Figure 1. $P = (\overline{\{auau, auu\}}, \{auau, auu\}) \in$ DES is represented as an LGraph: the names of the vertices in the visualization are the labels of the vertices.



$(\bot, (q_0,\lambda,\square)) \vdash_M ((q_0,a,\square,\bullet\square,q_0), (q_0,a,\bullet\square))$
$\vdash_M ((q_0,a,\bullet,\bullet\bullet,q_0), (q_0,aa,\bullet\bullet\square)) \vdash_M ((q_0,\lambda,\bullet,\bullet,q_1), (q_1,aa,\bullet\bullet\square))$
$\vdash_M ((q_1,b,\bullet,\lambda,q_1), (q_1,aab,\bullet\square)) \vdash_M ((q_1,b,\bullet,\lambda,q_1), (q_1,aabb,\square))$
$\vdash_M ((q_1,c,\square,\square,q_2), (q_2,aabbc,\square))$

Figure 2. $M \in$ FPDA with $\mathrm{L_m}(M) = \{a^{n+1}b^{n+1}c \mid n \in \mathbf{N}\}$ and an exemplary initial derivation in which edges are printed in gray.

Definition 5. (Natural Operational Behaviour of DES).
Let $D \in$ DES. Then $[\![D]\!]^{\mathrm{DES}}_{\mathrm{LGraph}} = (V, E, L, s, t, l)$ is the (natural) LGraph-representation of D iff (i) $V = \mathrm{L_{um}}(D)$, (ii) $E = \{(w, w{\cdot}\sigma) \mid w{\cdot}\sigma \in \mathrm{L_{um}}(D)\}$, (iii) $L = \{\mathrm{i}, \mathrm{f}\} \cup \mathrm{L_{um}}(D)$, (iv) $s(w, w') = w$, (v) $t(w, w') = w'$, and (vi) $l(w) = \{w\} \cup \{\mathrm{i} \mid w = \lambda\} \cup \{\mathrm{f} \mid w \in \mathrm{L_m}(D)\}$. □

An example of this encoding is given in Figure 1 where a DES is visualized as an LGraph.

Obviously, there is an isomorphism between the LGraph-representations of two DES iff the DES are identical.

Lemma 2. (Sound Encoding). Let $D_1, D_2 \in$ DES.
Then $[\![D_1]\!]^{\mathrm{DES}}_{\mathrm{LGraph}} \cong [\![D_2]\!]^{\mathrm{DES}}_{\mathrm{LGraph}}$ iff $D_1 = D_2$. ■

We will use the following complete lattice in Section 4 to characterize important DES. Suprema and infima of sets of languages are denoted using $\cup$ and $\cap$ in this paper.

Lemma 1. (Complete Lattice of DES).
DES forms a complete lattice using the following operations where $\{A, B\} \cup M \subseteq$ DES. (i) the least element: $\bot = \langle\!| \emptyset, \emptyset |\!\rangle$, (ii) the greatest element: $\top = \langle\!| \Sigma^*, \Sigma^* |\!\rangle$, (iii) the inclusion: $A \leq B$ iff $\mathrm{L_{um}}(A) \subseteq \mathrm{L_{um}}(B)$ and $\mathrm{L_m}(A) \subseteq \mathrm{L_m}(B)$. (iv) the strict inclusion: $A < B$ iff $A \leq B$ and $A \neq B$. (v) the synchronous (infimal) composition (denoted by $A{\times}B$ in the rest of the paper): $\inf(A, B) = \langle\!| \mathrm{L_{um}}(A) \cap \mathrm{L_{um}}(B), \mathrm{L_m}(A) \cap \mathrm{L_m}(B) |\!\rangle$, (vi) the alternative (supremal) composition: $\sup(A, B) = \langle\!| \mathrm{L_{um}}(A) \cup \mathrm{L_{um}}(B), \mathrm{L_m}(A) \cup \mathrm{L_m}(B) |\!\rangle$, (vii) the maximal DES included in all DES from M [3]: $\mathrm{Inf}(M) = \langle\!| \cap \mathrm{L_{um}}(M), \cap \mathrm{L_m}(M) |\!\rangle$, and (viii) the least DES which includes all DES from M: $\mathrm{Sup}(M) = \langle\!| \cup \mathrm{L_{um}}(M), \cup \mathrm{L_m}(M) |\!\rangle$. ■

1.3 Finalizing Pushdown Automata (FPDA)

FPDA, introduced here, are DFA enriched with a single stack-variable which can be used to remember aspects for later reuse of a generated word. An example of FPDA

[2] Whenever D_2 has the observation w, D_1 does not prevent w, and $w{\cdot}u$ is an observation of D_2 (for an uncontrollable event u), then $w{\cdot}u$ is not prevented by D_1.

[3] Remark: $\mathrm{L_m}$ and $\mathrm{L_{um}}$ are here computing images, i.e., the sets of the (un)marked languages of DES from M.

generating a language unacceptable by any DFA is given in Figure 2: The FPDA M operates by remembering the number of generated a as an equally long sequence of $\bullet$ in its stack—then, for any generated b one $\bullet$ is popped from the stack. Furthermore, we additionally assume that the automaton can decide to stop generating symbols by "generating" the end-of-output marker $\diamond$. This intuitive explanation is formalized in the following definition of FPDA and their operational semantics.

Definition 6. (Finalizing Pushdown Automata (FPDA)). $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F, \diamond) \in$ FPDA iff (i) the states Q, the output alphabet Σ, the stack alphabet Γ, and the set of edges δ are finite, (ii) $\delta : Q \times (\Sigma \cup \{\lambda, \diamond\}) \times \Gamma \times \Gamma^* \times Q$, (iii) the end-of-output marker $\diamond$ is not contained in Σ, (iv) the end-of-stack marker $\square$ is contained in Γ, (v) the end-of-stack marker is never removed from the stack ($(q, \sigma, \square, s', q')$ implies $s' \sqsupseteq \square$), (vi) the marking states F and the initial state q_0 are contained in Q. □

We provide the slightly nonstandard branching semantics of an FPDA M which utilizes a history variable in the configurations to greatly simplify the definitions of the trace abstractions presented in Section 2. Furthermore, this branching semantics corresponds to the intuition that the finite state realizations are generators rather than acceptors of languages.

Definition 7. (Semantics of FPDA). (i) the set of configurations $\mathcal{C}(M) = Q \times \Sigma^* \cdot \{\lambda, \diamond\} \times \Gamma^+$ where $(q, w, s) \in \mathcal{C}(M)$ consists of a state q, a history variable w (storing the symbols generated), and a stack variable s, (ii) the initial configuration $\mathcal{C}_{\text{init}}(M)$ is $(q_0, \lambda, \square)$, (iii) the set of marking configurations $\mathcal{C}_{\text{m}}(M)$ is defined by $\{(q, w, s) \in \mathcal{C}(M) \mid q \in F\}$, (iv) the annotated configurations $\mathcal{C}^\delta(M) = ((\delta \cup \{\bot\}) \times \mathcal{C}(M))$ additionally contain "pre-edges" from δ (v) the single-step relation (operating on the annotated configurations) $\vdash_M : \mathcal{C}^\delta(M) \times \mathcal{C}^\delta(M)$ is defined by $(e, (p, w, s \cdot s'')) \vdash_M ((p, w', s, s', p'), (p', w \cdot w', s' \cdot s''))$ (i.e., the state is changed to q', w' is added to the history variable, and the prefix s of the stack-variable $s \cdot s''$ is replaced by s') where $w \sqsupseteq \diamond$ implies $w' = \lambda$ (i.e., once the end-of-output marker $\diamond$ has been generated, the history variable cannot be extended), (vi) the set of derivations $\mathcal{D}(M)$ contains all elements from $\mathcal{C}^\delta(M)^{\omega *}$ starting in a configuration of the form $(\bot, c)$ where all adjacent $(e_1, c_1), (e_2, c_2) \in \mathcal{C}^\delta(M)$ satisfy $(e_1, c_1) \vdash_M (e_2, c_2)$, (vii) the set of initial derivations $\mathcal{D}_{\text{I}}(M)$ contains all elements of $\mathcal{D}(M)$ starting with the initial configuration (i.e., $(\bot, (q_0, \lambda, \square))$), (viii) the reachable configurations $\mathcal{C}_{\text{reach}}(M)$ are defined by $\{c \in \mathcal{C}(M) \mid \exists d \in \mathcal{D}_{\text{I}}(M), n \in \mathbf{N} \ . \ d(n) = (e, c)\}$, (ix) let out : $(\mathcal{C}(M) \cup \mathcal{C}^\delta(M)) \to \Sigma^*$ be defined by $\text{out}(e, (q, w, s)) = \text{out}(q, w, s) = (\text{if } w \sqsupseteq \diamond \text{ then butlast}(w) \text{ else } w)$ (i.e., we drop the possibly contained end-of-output marker $\diamond$ from the history variable to obtain the output of a configuration), (x) the marked language $\text{L}_{\text{m}}(M)$ is defined by $\text{out}(\mathcal{C}_{\text{m}}(M) \cap \mathcal{C}_{\text{reach}}(M))$, and (xi) the unmarked language $\text{L}_{\text{um}}(M)$ is defined by $\text{out}(\mathcal{C}_{\text{reach}}(M))$.
The concatenation of $d_1, d_2 \in \mathcal{D}(M)$ at index $n \in \mathbf{N}$ is given by $(d_1 \cdot_n d_2) = \lambda i.$ if $i \leq n$ then $d_1(i)$ else $d_2(i - n)$. □

An example of an FPDA-derivation is given in Figure 2. The well known sub-classes of FPDA having one ore more of the properties below are defined in Table 1.

	FPDA	PDA	DPDA	NFA	DFA
deterministic			✓		✓
λ-step-free				✓	✓
$\diamond$-step-free		✓	✓	✓	✓
stack-free				✓	✓

Table 1. Subclasses of FPDA.

Definition 8. (Sub-classes of FPDA). An FPDA is deterministic iff for every reachable configuration all two distinct steps append distinct elements of $\Sigma \cup \{\diamond\}$ to the history variable. An FPDA is λ-step-free iff no edge in δ is of the form (p, λ, s, s', p'). An FPDA is $\diamond$-step-free iff no edge in δ is of the form $(p, \diamond, s, s', p')$. An FPDA is stack-free iff every edge in δ is of the form $(p, a, \square, \square, p')$. □

Remark 1. There is no complete lattice over DFA (DPDA, FPDA) since regular languages (deterministic context free languages, context free languages) are not closed under infinite union [4] and intersection [5]. Therefore, similarly to Ramadge and Wonham (1984), we state the SCP over the complete lattice of trace abstractions. In particular, we are using the complete lattice over DES from Lemma 1. □

2. ADEQUACY OF DES W.R.T. FPDA

Since we want to express the SCP for FPDA in terms of an SCP over DES it is essential that DES adequately describe the operational behavior of FPDA. We provide three variations of encodings of (i) FPDA into DES and (ii) FPDA into LGraphs and investigate in each of the three variations whether the FPDA to DES encoding preserves the operational behavior, i.e., we compare the operational behavior of the FPDA and the resulting DES by translating both into LGraphs using the encodings DES to LGraph and FPDA to LGraph. In fact, we provide adequate encodings for DFA and DPDA.

2.1 Adequate Encoding for DFA

The following encoding defines for an FPDA (and its operational semantics) the *observable* operational behavior, expressed as an LGraph. In this variation we expect every step to be fully observable which is only true for DFA and λ-step free DPDA.

Definition 9. (Natural Operational Behaviour of FPDA). Let $M \in$ FPDA. Then $[\![M]\!]^{\text{FPDA}}_{\text{LGraph}} = (E, V, L, s, t, l)$ is the (natural) LGraph-representation of M iff (i) V is the smallest set containing $\mathcal{C}_{\text{init}}(M)$ which is closed under $\vdash_M$, (ii) $E = \vdash_M$, (iii) $L = \{\text{i}, \text{f}\} \cup \text{L}_{\text{um}}(M)$, (iv) $s(c_1, e, c_2) = c_1$, (v) $t(c_1, e, c_2) = c_2$, and (vi) $l(c) = \{\text{out}(c)\} \cup \{\text{i} \mid c = \mathcal{C}_{\text{init}}(M)\} \cup \{\text{f} \mid c \in \mathcal{C}_{\text{m}}(M)\}$. □

Since each step is observable, two FPDA are equivalent w.r.t. $[\![\cdot]\!]^{\text{FPDA}}_{\text{LGraph}}$ iff they are renamings of each other.

Proposition 1. (Sound Encoding). Let $M^1, M^2 \in$ FPDA be accessible. Then $[\![M^1]\!]^{\text{FPDA}}_{\text{LGraph}} \cong [\![M^2]\!]^{\text{FPDA}}_{\text{LGraph}}$ iff M^1 is a renaming (states, stack-elements, stack-end-marker, and end-of-output-marker) of M^2. ■

Observe that the standard/natural encoding $[\![\cdot]\!]^{\text{FPDA}}_{\text{DES}}$ is the (implicitly used) trace abstraction from FPDA into DES

[4] A union of infinitely many singletons is not context free: $\cup\{\{a^n b^n\} \mid n \in \mathbf{N}, n \text{ prime}\} = \{a^n b^n \mid n \in \mathbf{N}, n \text{ prime}\}$

[5] An intersection of infinitely many singletons is not context free: $\cap\{\Sigma^* \setminus \{a^n b^n\} \mid n \in \mathbf{N}, n \text{ prime}\} = \Sigma^* \setminus \{a^n b^n \mid n \in \mathbf{N}, n \text{ prime}\}$

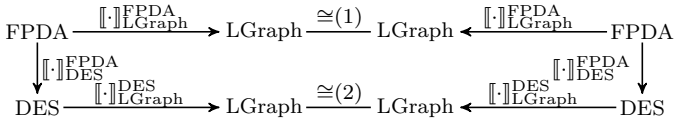


Figure 3. For Theorem 1: (1) holds iff (2) holds.

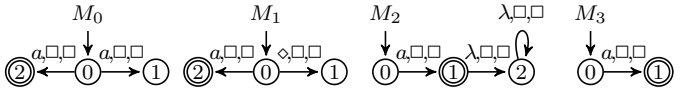


Figure 4. M_0, M_1, M_2, and M_3 have the same $[\![\cdot]\!]^{\text{FPDA}}_{\text{DES}}$ image $\langle\!\langle \{\lambda, a\}, \{a\} \rangle\!\rangle$ which is language-blockfree. Only M_3 is operationally-blockfree.

used by Wonham and Ramadge (1987) as a denotational description of the operational behavior of DFA.

Definition 10. (Natural Encoding $[\![\cdot]\!]^{\text{FPDA}}_{\text{DES}}$).
Let $M \in$ FPDA.
Then $[\![M]\!]^{\text{FPDA}}_{\text{DES}} = \langle\!\langle\, \mathrm{L_{um}}(M), \mathrm{L_m}(M) \rangle\!\rangle \in$ DES. □

We can conclude that for two FPDA which are DFA or λ-step free DPDA the observable operational behavior coincides iff their DES representations have equivalent observable operational behaviors, i.e., they have identical (un)marked languages.

Theorem 1. ($[\![\cdot]\!]^{\text{FPDA}}_{\text{DES}}$ is Fully Abstract w.r.t. $[\![\cdot]\!]^{\text{FPDA}}_{\text{LGraph}}$).
Let $M^1, M^2 \in$ FPDA, deterministic, $\diamond$-step-free, and λ-step-free. Then, $[\![M^1]\!]^{\text{FPDA}}_{\text{LGraph}} \cong [\![M^2]\!]^{\text{FPDA}}_{\text{LGraph}}$ iff $[\![[\![M^1]\!]^{\text{FPDA}}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}} \cong [\![[\![M^2]\!]^{\text{FPDA}}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}}$. ■

Confer to Figure 3 for a visualization of Theorem 1.
However, since none of the assumptions of Theorem 1 can be dropped, as stated in the following corollary, the encodings of FPDA into LGraph and DES are unsatisfactory (e.g., consider DPDA with λ-steps).

Corollary 1. Let $M^1 \in$ FPDA which is not deterministic, not $\diamond$-step-free, or not λ-step-free and let $M^2 \in$ DFA. Then, $[\![M^1]\!]^{\text{FPDA}}_{\text{LGraph}} \cong [\![M^2]\!]^{\text{FPDA}}_{\text{LGraph}}$ *not if but only if* $[\![[\![M^1]\!]^{\text{FPDA}}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}} \cong [\![[\![M^2]\!]^{\text{FPDA}}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}}$. ■

For example, consider the FPDA in Figure 4. When choosing M_0, M_1, or M_2 for M^1 and M_3 for M^2 in Corollary 1 then $[\![[\![M^1]\!]^{\text{FPDA}}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}} \cong [\![[\![M^2]\!]^{\text{FPDA}}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}}$ is satisfied but not $[\![M^1]\!]^{\text{FPDA}}_{\text{LGraph}} \cong [\![M^2]\!]^{\text{FPDA}}_{\text{LGraph}}$.

2.2 Adequate Encoding for DPDA

Corollary 1 states that DES are no sound denotational model for FPDA (including DPDA) w.r.t. the full observability defined via $[\![\cdot]\!]^{\text{FPDA}}_{\text{LGraph}}$. The problem stems from the steps which are invisible in the DES (λ-steps, $\diamond$-steps, and nondeterministic choices (note that this kind of step is also not explicitly contained in the operational semantics)) but visible to the behavioral equivalence $[\![M^1]\!]^{\text{FPDA}}_{\text{LGraph}} \cong [\![M^2]\!]^{\text{FPDA}}_{\text{LGraph}}$. These steps could be made visible by modification of the FPDA, however, this is not reasonable for non-determinism and the end-of-output marker (see Remark 2 in Section 3). In this section we consider FPDA which are deterministic and $\diamond$-step-free (i.e., DPDA): as the λ-steps represent the internal steps of the controller, properties on its occurrence in executions of the closed loop may be of great importance.

We distinguish between two kinds of λ-step-sequences: finite sequences and infinite sequences.

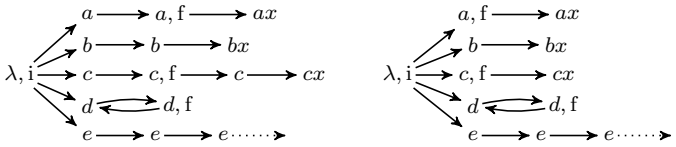


Figure 5. The operation $[\![\cdot]\!]^{\text{FPDA-}\lambda}_{\text{LGraph}}$ transforms the LGraph returned by $[\![\cdot]\!]^{\text{FPDA}}_{\text{LGraph}}$ on the left into the LGraph on the right.

►*Finite λ-step-sequences* do not necessarily need to be observable for DPDA because DPDA do not have a worst-case-execution-time in general [6]. Thus, we assume in this paper that there are no properties to be enforced on these finite sequences. Observe that in DPDA at most finitely many λ-steps, which are executed deterministically, occur between two visible steps. We can therefore replace maximal finite sequences by a single vertex (with label f iff some vertex in the finite sequence had the label f) by $[\![\cdot]\!]^{\text{FPDA-}\lambda}_{\text{LGraph}}$ as exemplified in Figure 5 (page 4). We omit the formal construction (which is contained in Schneider and Schmuck (2013)) due to space restrictions.

►*Infinite λ-step-sequences* represent livelocks which can be contracted in the operational semantics into single steps which makes such steps visible with the symbol $\circlearrowleft \in \Sigma_{\mathsf{uc}}$.

Definition 11. (Encoding $[\![\cdot]\!]^{\text{FPDA-}\omega}_{\text{DES}}$). Let $M \in$ DPDA. Then $[\![M]\!]^{\text{FPDA-}\omega}_{\text{DES}} = \langle\!\langle\, \text{noteLL}(\mathrm{L_{um}}(M)), \text{noteLL}(\mathrm{L_m}(M)) \rangle\!\rangle \in$ DES where $\text{noteLL}(A) = \{w \circlearrowleft | \; w \in A \wedge \exists d \in \mathcal{D}_{\mathrm{I}}(M), N \in \mathbf{N} \,.\, \forall k \geq N \,.\, \text{out}(d(k)) = w\}$. □

We can conclude that when finite sequences of λ-steps are not to be observed (by using $[\![\cdot]\!]^{\text{FPDA-}\lambda}_{\text{LGraph}}$) and livelocks are made visible in the DES (by using $[\![\cdot]\!]^{\text{FPDA-}\omega}_{\text{DES}}$), then for two DPDA the observable operational behavior coincides iff their DES representations have equivalent observable operational behaviors.

Theorem 2. ($[\![\cdot]\!]^{\text{FPDA-}\omega}_{\text{DES}}$ is Fully Abstract w.r.t. $[\![\cdot]\!]^{\text{FPDA-}\lambda}_{\text{LGraph}}$).
Let $M^1, M^2 \in$ DPDA. Then, $[\![M^1]\!]^{\text{FPDA-}\lambda}_{\text{LGraph}} \cong [\![M^2]\!]^{\text{FPDA-}\lambda}_{\text{LGraph}}$ iff $[\![[\![M^1]\!]^{\text{FPDA-}\omega}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}} \cong [\![[\![M^2]\!]^{\text{FPDA-}\omega}_{\text{DES}}]\!]^{\text{DES}}_{\text{LGraph}}$. ■

Finally, since the encodings preserve the observable behavior (except for the $\circlearrowleft$-appending encoding $[\![\cdot]\!]^{\text{FPDA-}\omega}_{\text{DES}}$), they preserve in particular the (un)marked languages.

Corollary 2. (Preservation of (un)marked languages).
If $F \in \{[\![\cdot]\!]^{\text{DES}}_{\text{LGraph}}, [\![\cdot]\!]^{\text{FPDA}}_{\text{LGraph}}, [\![\cdot]\!]^{\text{FPDA-}\lambda}_{\text{LGraph}}, [\![\cdot]\!]^{\text{FPDA}}_{\text{DES}}\}$
then $\mathrm{L_m}(X) = \mathrm{L_m}(F(X))$ and $\mathrm{L_{um}}(X) = \mathrm{L_{um}}(F(X))$.
Furthermore, for $[\![\cdot]\!]^{\text{FPDA-}\omega}_{\text{DES}}$: $\mathrm{L_m}(X) = \mathrm{L_m}([\![X]\!]^{\text{FPDA-}\omega}_{\text{DES}}) \cap \Sigma^*$ and $\mathrm{L_{um}}(X) = \mathrm{L_{um}}([\![X]\!]^{\text{FPDA-}\omega}_{\text{DES}}) \cap \Sigma^*$. ■

2.3 Adequate Encodings for FPDA

To determine adequate encodings of all FPDA, the uncovering of livelocks can be extended to the uncovering of all formerly invisible steps (then also including nondeterminism, $\diamond$-steps, and possibly even finite sequences of λ-steps which were hidden in the previous subsection). For such an encoding, a similar soundness theorem can be formulated. However, we will explain in Remark 2 in Section 3 that this encoding fails to be satisfactory for the task of properly reducing the synthesis problem by means of an OSCP as some operational criteria are no longer enforced on realizations of controllers.

[6] Consider $\{a^n b^m c^n \mid n, m \in \mathbf{N}\} \cup \{a^n b^m d^m \mid n, m \in \mathbf{N}\}$: all a and b have to be recorded by the stack: when reaching a c all records of b have to be removed in an unbounded number of steps.

3. (OPERATIONAL) SUPERVISORY CONTROL PROBLEM

According to the previous section, livelocks (i.e., infinite λ-sequences) are observable when using the trace abstraction $[\![\cdot]\!]_{\mathrm{DES}}^{\mathrm{FPDA}\text{-}\omega}$. This allows us to specify an operational SCP (OSCP) preventing livelocks in FPDA realizations of constructed controllers. Before introducing the OSCP we formalize the SCP as introduced by Wonham and Ramadge (1987) in our notation using DES as a fundamental model rather than the marked language alone.

Definition 12. (SCP). Let $P, S \in$ DES be a plant and a specification. Then $\mathrm{SCP}(P,S)$ contains the least restrictive controllers $C \in$ DES w.r.t. (i) $\mathrm{L_m}(C{\times}P) \subseteq \mathrm{L_m}(S)$, (ii) $C{\times}P$ is language-Σ_{uc}-controllable w.r.t. P, and (iii) $C{\times}P$ is language-blockfree. Least restrictive means that $C'{\times}P{\leq}C{\times}P$ for any $C' \in$ DES satisfying (i)–(iii). □

Based on the SCP, we introduce the following OSCP.

Definition 13. (Operational SCP (OSCP)). Given a plant $P \in$ DFA and a specification $S \in$ DPDA. Then $\mathrm{OSCP}(P,S)$ is the set of all $C \in$ DPDA satisfying $[\![C]\!]_{\mathrm{DES}}^{\mathrm{FPDA}\text{-}\omega} \in \mathrm{SCP}([\![P]\!]_{\mathrm{DES}}^{\mathrm{FPDA}}, [\![S]\!]_{\mathrm{DES}}^{\mathrm{FPDA}})$. □

Since the closed-loop construction relies on the synchronization of a controller $C \in$ DPDA (broader classes for C from FPDA are only considered in Remark 2) and a plant $P \in$ DFA ($P \in$ DFA throughout this paper), we give such a construction which returns a DPDA. This construction is quite similar to the synchronous composition of DPDA with DFA from, for example, (Hopcroft and Ullman, 1979, page 135).

Definition 14. (FPDA-DFA-Synchronous Composition). Let $M^1 = (Q^1, \Sigma, \Gamma, \delta^1, q_0^1, \Box, F^1, \diamond) \in$ FPDA. Let $M^2 = (Q^2, \Sigma, \Gamma', \delta^2, q_0^2, \Box', F^2, \diamond') \in$ DFA. Then $M^1 \times M^2 = M = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F, \diamond)$ is given by (i) $Q{=}Q^1{\times}Q^2$ (ii) $((q_1,q_2), w, s, s', (q_1', q_2')) \in \delta$ iff $(q_1, w, s, s', q_1') \in \delta^1$ and either $w \in \Sigma, (q_2, w, \Box', \Box', q_2'){\in}\delta^2$ or $w \in \{\lambda, \diamond\}, q_2 = q_2'$, (iii) $q_0 = (q_0^1, q_0^2)$, and (iv) $F = F^1 \times F^2$. □

The main criteria for reasonability of the OSCP definition are the properties of closed loops for controllers which are solutions to the OSCP. After giving such relevant criteria in the following definition, we present our first main result.

Definition 15. (Properties of FPDA). Let $M^1 \in$ FPDA. Let $M^2 \in$ FPDA. Then (i) M^1 has a livelock iff for some infinite $d \in \mathcal{D}_{\mathrm{I}}(M^1)$ there is an $N \in \mathbf{N}$ such that the unmarked language of d is constant after N (i.e., for all $k \geq N$: $\mathrm{out}(d(N)) = \mathrm{out}(d(k))$), (ii) M^1 is operational-blockfree iff for any finite $d_i \in \mathcal{D}_{\mathrm{I}}(M^1)$ of length $n \in \mathbf{N}$ ending in $d_i(n) = (e,c)$ there is a continuation $d_c \in \mathcal{D}(M^1)$ such that $d_i{\cdot_n}d_c$ is a marking derivation and d_i and d_c match at the gluing point n (i.e., $d_c(0) = (\bot, c)$), (iii) M^1 is operational-satisfying M^2 iff for any $d_1 \in \mathcal{D}_{\mathrm{I}}(M^1)$ of length n_1 ending in a marking state, there is some $d_2 \in \mathcal{D}_{\mathrm{I}}(M^2)$ of length n_2 ending in a marking state such that $\mathrm{out}(d_1(n_1)) = \mathrm{out}(d_2(n_2))$, and (iv) M^1 is operational-Σ_{uc}-controllable w.r.t. M^2 iff whenever $d_1 \in \mathcal{D}_{\mathrm{I}}(M^1)$ is of length $n_1 \in \mathbf{N}$ ending in $d_1(n_1) = (e_1, c_1)$, $d_2 \in \mathcal{D}_{\mathrm{I}}(M^2)$ is of length $n_2 \in \mathbf{N}$, $\mathrm{out}(d_1(n_1)) = \mathrm{out}(d_2(n_2))$, $d_2(n_2) \vdash_{M^2} (e_2, c_2)$, $\mathrm{out}(e_2, c_2) = \mathrm{out}(d_2(n_2))u$, and $u \in \Sigma_{\mathsf{uc}}$, then there is a continuation $d_c \in \mathcal{D}(M^1)$ of length $n_3 \in \mathbf{N}$ such that $\mathrm{out}((d_1{\cdot_{n_1}}d_c)(n_1 + n_3)) = \mathrm{out}(d_2(n_2))u$ and d_1 and d_c match at the gluing point n_1 (i.e., $d_c(0) = (\bot, c_1)$). □

Incidentally (mainly due to the determinism), all of the above properties are satisfied for closed loops generated from solutions to the OSCP from Definition 13.

Theorem 3. (Further Properties of OSCP controllers). Given a plant $P \in$ DFA and a specification $S \in$ DPDA. Let $C \in \mathrm{OSCP}(P,S)$. Then (i) $C \times P$ is operational-satisfying S, (ii) $C \times P$ is operational-Σ_{uc}-controllable w.r.t. P, (iii) $C \times P$ has no livelocks, and (iv) $C \times P$ is operational-blockfree. ■

Remark 2. (Nonextendability of Theorem 3). Theorem 3 cannot be extended to $S \in$ NFA or $S \in$ FPDA which are not $\diamond$-step-free since language-blockfreeness is insufficient for operational-blockfreeness in general (see M_0 and M_1 in Figure 4 page 4). □

We have integrated enough information into the DES representations of DPDA and DFA to be able to reuse the unmodified SCP. The more direct approach, of modifying the SCP to include conditions enforcing, e.g., livelockfreeness, operational blockfreeness, or even more advanced properties where cost-optimal controllers are to be synthesized, is nontrivial because the validity of the SCP has to be verified as well. Even with extra conditions, the set of sound controllers must be closed under arbitrary union, which is not true if for example livelockfreeness is to be enforced (for example, if $\tau \in \Sigma_{\mathsf{c}}$ represents a λ-step, the supremal language of bounded executions of τ allows livelock executions of τ: $\cup\{\{\tau^n\} \mid n \in \mathbf{N}\} = \{\tau\}^*$).

Using Theorem 2 we can conclude that all closed loops for solutions to the OSCP have equivalent observable behavior; therefore, the concrete choice of a controller realization is irrelevant.

Theorem 4. (OSCP Solutions are Equivalent). Given a plant $P \in$ DFA, a specification $S \in$ DPDA, and $C_1, C_2 \in \mathrm{OSCP}(P,S)$. Then $[\![C_1 \times P]\!]_{\mathrm{LGraph}}^{\mathrm{FPDA}\text{-}\lambda} \cong [\![C_2 \times P]\!]_{\mathrm{LGraph}}^{\mathrm{FPDA}\text{-}\lambda}$. ■

4. SCP CHARACTERIZATIONS VIA SUPREMA

In this section we formalize the SCP over DES by defining sound and maximal (i.e., least restrictive) controllers and compare this formalization to the supremal characterization of Ramadge and Wonham (1984). The DES based representations are used in Section 5 to verify solvers for the SCP which are also adequate for DPDA specifications. These solvers deterministically generate the sound and maximal controllers which are additionally smallest (i.e., contain the least set of words).

Definition 16. (Sound, Maximal, and Smallest Solutions). Given a plant $P \in$ DES and a specification $S \in$ DES. Let $C \in$ DES. Then (by using Equations (1)–(3) in Table 2 on page 6) (i) C is a sound controller iff the closed loop $P{\times}C$ is safe in the sense of (i)–(iii) from Definition 12, formally $C \in \mathcal{S}_{\mathrm{sat}}(P,S)$, (ii) C is a maximal controller iff the closed loop $P{\times}C$ is sound and less restrictive than any other safe closed loop, formally $C \in \mathcal{S}_{\max}(P,S)$, and (iii) C is a smallest maximal controller iff it is a maximal controller and any strictly smaller maximal controller produces a more restrictive closed loop, formally $C \in \mathcal{S}_{\max}^{\min}(P,S)$. □

We explain the basic differences between these types of controllers by an example.

Example 1. (Comparison of Solutions). Let P be a language-blockfree plant. Let $S = \langle\!|\Sigma^*, \Sigma^*|\!\rangle$ be a specification.

$$C \in \mathcal{S}_{\text{sat}}(P,S) \text{ iff } ❶\, \mathrm{L_m}(P{\times}C) \subseteq \mathrm{L_m}(S) \quad ❷\, \mathrm{LCont}(\mathrm{L_{um}}(P{\times}C), \mathrm{L_{um}}(P), \Sigma_{\mathsf{uc}}) \quad ❸\, \mathrm{L_{um}}(P{\times}C) \subseteq \overline{\mathrm{L_m}(P{\times}C)} \tag{1}$$

$$C \in \mathcal{S}_{\max}(P,S) \text{ iff } ❶ C \in \mathcal{S}_{\text{sat}}(P,S) \quad ❷ \forall C' \in \mathcal{S}_{\text{sat}}(P,S)\,.\, P{\times}C' \leq P{\times}C \tag{2}$$

$$C \in \mathcal{S}^{\min}_{\max}(P,S) \text{ iff } ❶ C \in \mathcal{S}_{\max}(P,S) \quad ❷ \forall C' \in \mathcal{S}_{\max}(P,S)\,.\, C' < C \to P{\times}C' < P{\times}C \tag{3}$$

$$\Phi_{\text{mm}}(A_m, S_m) \triangleq ❶ A_m \subseteq S_m \quad ❷\, \mathrm{LCont}(\overline{A_m}, \mathrm{L_{um}}(P), \Sigma_{\mathsf{uc}}) \tag{4}$$

$$\Phi_{\text{um}}(A_{um}, S_m) \triangleq ❶ A_{um} \subseteq \overline{S_m} \quad ❷\, \mathrm{LCont}(A_{um}, \mathrm{L_{um}}(P), \Sigma_{\mathsf{uc}}) \quad ❸ A_{um} = \overline{A_{um}} \quad ❹ A_{um} \subseteq \overline{A_{um} \cap S_m} \tag{5}$$

$$C \in \mathcal{L}_{\text{mm}}(P,S) \text{ iff } ❶\, \mathrm{L_m}(P{\times}C) = \cup\{A \mid \Phi_{\text{mm}}(A, \mathrm{L_m}(S))\} \quad ❷\, \mathrm{L_{um}}(P{\times}C) = \overline{\mathrm{L_m}(P{\times}C)} \tag{6}$$

$$C \in \mathcal{L}_{\text{um}}(P,S) \text{ iff } ❶\, \mathrm{L_{um}}(P{\times}C) = \cup\{A \mid \Phi_{\text{um}}(A, \mathrm{L_m}(S))\} \quad ❷\, \mathrm{L_m}(P{\times}C) = \mathrm{L_m}(S) \cap \mathrm{L_{um}}(P{\times}C) \tag{7}$$

$$\mathcal{L}^{\text{si}}_{\text{mm}}(P,S) \triangleq \mathcal{L}_{\text{mm}}(P, P{\times}S) \tag{8}$$

$$\mathcal{L}^{\text{si}}_{\text{um}}(P,S) \triangleq \mathcal{L}_{\text{um}}(P, P{\times}S) \tag{9}$$

$$C \in \mathcal{L}^{\text{si}}_{\text{mm}}(P,S) \text{ iff } ❶\, \mathrm{L_m}(C) = \cup\{A \mid \Phi_{\text{mm}}(A, \mathrm{L_m}(P{\times}S))\} \quad ❷\, \mathrm{L_{um}}(C) = \overline{\mathrm{L_m}(P{\times}C)} \tag{10}$$

$$C \in \mathcal{L}^{\text{si}}_{\text{um}}(P,S) \text{ iff } ❶\, \mathrm{L_{um}}(C) = \cup\{A \mid \Phi_{\text{um}}(A, \mathrm{L_m}(P{\times}S))\} \quad ❷\, \mathrm{L_m}(C) = \mathrm{L_m}(S) \cap \mathrm{L_{um}}(P{\times}C) \tag{11}$$

$$C \in \mathcal{D}_{\text{mm}}(P,S) \text{ iff } P{\times}C = \mathrm{Sup}(\{\langle\!|\overline{A}, A|\!\rangle \mid \Phi_{\text{mm}}(A, \mathrm{L_m}(S))\}) \tag{12}$$

$$C \in \mathcal{D}_{\text{um}}(P,S) \text{ iff } P{\times}C = \mathrm{Sup}(\{\langle\!|A, \mathrm{L_m}(S) \cap A|\!\rangle \mid \Phi_{\text{um}}(A, \mathrm{L_m}(S))\}) \tag{13}$$

$$\mathcal{D}^{\text{si}}_{\text{mm}}(P,S) \triangleq \mathcal{D}_{\text{mm}}(P, P{\times}S) \tag{14}$$

$$\mathcal{D}^{\text{si}}_{\text{um}}(P,S) \triangleq \mathcal{D}_{\text{um}}(P, P{\times}S) \tag{15}$$

Table 2. Sound, maximal, and smallest maximal controllers: Equations (1)–(3); language based supremal closed loops: Equations (6)–(9); DES based supremal closed loops: Equations (12)–(15). Sets of controllers obtained by suprema over languages (DES) are denoted here by $\mathcal{L}(\mathcal{D})$ with markers.

The controller $C_1 = \langle\!|\emptyset, \emptyset|\!\rangle$ is sound but not maximal. The controller $C_2 = S$ is maximal. The controller $C_3 = P$ is smallest maximal. While C_1 is not desirable (and just defined for presentation purposes), there is no difference between C_2 and C_3 w.r.t. the overall goal of the SCT to determine a controller enforcing the desired observable operational behavior on the closed loop. For implementation on a physical device, C_2 is for $P \neq S$ more compact and requires therefore less space. Nevertheless, the solvers, we are aware of, produce the controller C_3. The problem of determining the size-optimal automata realization of some sound and maximal controller is left for the future. □

Similarly to Theorem 4 we can state that the plant P is consistently restricted by all controllers $C \in \mathcal{S}_{\max}(P,S)$.

Theorem 5. (Consistent Restriction). Given a plant $P \in$ DES and a specification $S \in$ DES. Let $C, C' \in \mathcal{S}_{\max}(P,S)$. Then $P{\times}C = P{\times}C'$. ■

4.1 Language-Based Characterization by Suprema

Ramadge and Wonham (1984) have, based on the supremum over languages in Equation (6), introduced the desired marked language of the desired closed loop and by assuming language-blockfreeness also the unmarked language of the desired closed loop. In Equation (6) we have given the marked-maximal solution $\mathcal{L}_{\text{mm}}$ in our notation which is based on the supremum over the marked languages of controller candidates. While Equation (6) has the advantage to be compact and obviously sound (being directly related to, for example, Definition 16) we propose the alternative characterization in Equation (7) which is based on a supremum over the unmarked language of the desired closed loop. Primary differences are the translation from marked to unmarked languages using the prefix-closure operator and the different enforcement of blockfreeness. Both characterizations of the desired closed loop are equivalent and produce the desired closed-loop behavior. The alternative characterization is advantageous because, e.g., the effective solver in the DFA-setting (presented by Wonham and Ramadge (1987, Lemma 5.1)) removes unmarked words with controllability and blocking problems. The supremal characterization $\mathcal{L}_{\text{um}}(P,S)$ thereby describes the operation of the fixed-point algorithms more precisely.

Theorem 6. $\mathcal{L}_{\text{mm}}(P,S) = \mathcal{L}_{\text{um}}(P,S) \subseteq \mathcal{S}_{\max}(P,S)$ ■

Effective solvers usually produce deterministically a unique result which is the smallest maximal sound controller. This is achieved by simplifying the input by restricting the specification to the behavior of the plant: this is done in Equations (8) and (9) where the simple input marked maximal and simple input unmarked maximal solutions $\mathcal{L}^{\text{si}}_{\text{mm}}$ and $\mathcal{L}^{\text{si}}_{\text{um}}$ are defined. Incidentally, the controllers described with this restriction are all contained in $\mathcal{S}^{\min}_{\max}$.

Theorem 7. $\mathcal{L}^{\text{si}}_{\text{mm}}(P,S) = \mathcal{L}^{\text{si}}_{\text{um}}(P,S) \subseteq \mathcal{S}^{\min}_{\max}(P,S)$ ■

Furthermore, the controllers in $\mathcal{L}^{\text{si}}_{\text{mm}}$ and $\mathcal{L}^{\text{si}}_{\text{um}}$ produce identical closed loops as the controllers in $\mathcal{L}_{\text{mm}}$ and $\mathcal{L}_{\text{um}}$.

A commonality of the four characterizations in Equations (6)–(9) is that they are based on characterizations of the closed loop and not on the controller. In Equations (10) and (11) we explain that the actual result as obtained by $\mathcal{L}^{\text{si}}_{\text{mm}}$ and $\mathcal{L}^{\text{si}}_{\text{um}}$ is the controller and the closed loop which basically follows from the adherence of the closed loop to the specification due to its intersection with the plant.

4.2 DES-Based Characterization by Suprema

Furthermore, the characterizations in Equations (6)–(9) do not properly reflect the execution of iterative solvers which modify both, the marked and the unmarked language (using operations from Figure 6). In our formal approach we handle the modifications to both languages explicitly, in contrast to Ramadge and Wonham (1984) who focus on the modifications to the marked language exclusively, by including the statements on the unmarked (in the case of $\mathcal{L}_{\text{mm}}$) and marked language (in the case of $\mathcal{L}_{\text{um}}$) into the supremum statement using the lattice of DES.

Proposition 2. $\mathcal{D}_{\text{mm}}(P,S) = \mathcal{D}_{\text{um}}(P,S) = \mathcal{L}_{\text{mm}}(P,S)$ ■

The results on the set-based characterization can easily be transferred to the DES-based characterization. In the next section we decompose a supremum into a greatest fixed point of a composed operation.

5. SCP CHARACTERIZATIONS VIA GREATEST FIXED-POINTS

Usually, the desired controller is calculated by iterative application of a function $F : \mathrm{DES} \to \mathrm{DES}$. Let $\mathcal{F}$ denote the set of all these iterators ($\mathcal{F} = \mathrm{DES}^{\mathrm{DES}}$). Good iterators have the property that they do not skip beyond greatest fixed points which is achieved by including the last property (v) in the next definition. If $F \in \mathcal{G}(F_{\mathrm{inp}}, F_{\mathrm{fp}}, F_{\mathrm{out}})$ in the following definition, then (i) F_{inp} specifies the set of DES to which F should only be applied to, (ii) F_{fp} specifies the set of DES (from F_{inp}) for which F returns its input, and (iii) F_{out} specifies the set of DES which are returned by F (when executed on a DES from F_{inp}).

Definition 17. (Good Iterator). $F \in \mathcal{F}$ is a good iterator on $F_{\mathrm{inp}}, F_{\mathrm{fp}}, F_{\mathrm{out}} \subseteq \mathrm{DES}$ (written $F \in \mathcal{G}(F_{\mathrm{inp}}, F_{\mathrm{fp}}, F_{\mathrm{out}})$) iff whenever $X, Y \in F_{\mathrm{inp}}$ then (i) $F(X) \leq X$, (ii) $X \in F_{\mathrm{fp}}$ iff $F(X) = X$, (iii) $F(X) \in F_{\mathrm{out}}$, (iv) if $X \leq Y$ then $F(X) \leq F(Y)$, and (v) if $F(X) < X$, $Y < X$, $F(Y) = Y$, then $Y \leq F(X)$. □

Good iterators are closed under unconditional $\circ$ and conditional $\rhd$ composition, used to obtain fixed point algorithms computing the desired solutions in Section 5.1.

Lemma 3. (Composition of Good Iterators using $\circ$).
Let $F \in \mathcal{G}(F_{\mathrm{inp}}, F_{\mathrm{fp}}, F_{\mathrm{out}})$. Let $G \in \mathcal{G}(G_{\mathrm{inp}}, G_{\mathrm{fp}}, G_{\mathrm{out}})$.
If $F_{\mathrm{out}} \subseteq G_{\mathrm{inp}}$, then $G \circ F \in \mathcal{G}(F_{\mathrm{inp}}, F_{\mathrm{fp}} \cap G_{\mathrm{fp}}, G_{\mathrm{out}})$. ■

Definition 18. (Conditional Composition). Let $F, G \in \mathcal{F}$ then $G \rhd F = \lambda x.$ if $F(x) = x$ then x else $G(F(x))$. □

Lemma 4. (Composition of Good Iterators using $\rhd$).
Let $F \in \mathcal{G}(F_{\mathrm{inp}}, F_{\mathrm{fp}}, F_{\mathrm{out}})$. Let $G \in \mathcal{G}(G_{\mathrm{inp}}, G_{\mathrm{fp}}, G_{\mathrm{out}})$. If $F_{\mathrm{out}} \subseteq G_{\mathrm{inp}}$ and $F_{\mathrm{inp}} \subseteq G_{\mathrm{fp}}$, then $G \rhd F \in \mathcal{G}(F_{\mathrm{inp}}, \lambda X.$ if $F(X) = X$ then $X \in F_{\mathrm{fp}} \cap F_{\mathrm{inp}}$ else $X \in F_{\mathrm{fp}} \cap G_{\mathrm{fp}}, (F_{\mathrm{inp}} \cap F_{\mathrm{fp}} \cap F_{\mathrm{out}}) \cup G_{\mathrm{out}})$. ■

Good iterators do not skip beyond greatest fixed points.

Corollary 3. (Good Iterators do not Skip a GFP).
Let $F \in \mathcal{G}(\mathrm{DES}, F_{\mathrm{fp}}, \mathrm{DES})$ and $Y \in \mathrm{DES}$.
Then $\mathrm{gfp}(\lambda X.F(X \times Y)) = \mathrm{gfp}(\lambda X.F(X \times F(Y)))$. ■

The actual synthesis computation is then given by the following algorithm $\mathcal{U}$.

Definition 19. (Universal Computation).
Let $F \in \mathcal{F}$ and $D \in \mathrm{DES}$.
Then $\mathcal{U}(F, D) = (\text{if } D = F(D) \text{ then } D \text{ else } \mathcal{U}(F, F(D)))$. □

Then, greatest fixed points are calculated (assuming termination) as follows.

Theorem 8. ($\mathcal{U}$ computes the GFP).
Let $F \in \mathcal{G}(\mathrm{DES}, F_{\mathrm{fp}}, \mathrm{DES})$. If $\mathcal{U}(F, \top)$ terminates, then $\mathcal{U}(F, \top) = \mathrm{Sup}(F_{\mathrm{fp}}) = \mathrm{gfp}(F)$. ■

We conclude that it is sufficient for concrete applications (e.g., DFA and DPDA) to verify the goodness of the used iterator to prove that the generated controller solves the SCP. Furthermore, the composition lemmas above give a proof-strategy by composition of a good iterator from multiple good iterators: this is exemplified in the next subsection where we introduce concrete iterators. As demonstrated in Section 5.1, it is occasionally advantageous to execute the universal algorithm on some value different from the $\top$ element.

Theorem 9. (Initialized $\mathcal{U}$ computes the GFP).
Let $F \in \mathcal{G}(\mathrm{DES}, F_{\mathrm{fp}}, F_{\mathrm{out}})$. If $\mathcal{U}(F, S)$ terminates, then $\mathcal{U}(F, S) = \mathrm{Sup}(\{X \mid X \in F_{\mathrm{fp}}, X \leq S\}) = \mathrm{gfp}(\lambda X.F(X \times S))$. ■

$$
\begin{array}{l}
\mathrm{op}_{\mathrm{cw}}(w, U) = \forall u \in U \,.\, w{\cdot}u \in \mathrm{L_{um}}(P) \to w{\cdot}u \in \mathrm{L_{um}}(C) \\
\mathrm{op}_{\mathrm{c}}(A, U) = \forall w' \in A \,.\, \mathrm{op}_{\mathrm{cw}}(w', U, \mathrm{L_{um}}(P), \mathrm{L_{um}}(C)) \\
\quad \text{where } A = \cup\{A \mid A = \overline{A} \wedge A \subseteq \mathrm{L_{um}}(D)\} \\
\mathrm{F_{bf}}(D) = \langle\!|\overline{\mathrm{L_m}(D)}, \mathrm{L_m}(D)|\!\rangle \\
\mathrm{F_{c1}}(D) = \langle\!|A, \mathrm{L_m}(D) \cap A|\!\rangle \\
\quad \text{where } A = \{w \in \mathrm{L_{um}}(D) \mid \mathrm{op_c}(\overline{\{w\}}, \Sigma_{\mathsf{uc}}^*)\} \\
\mathrm{F_{c2}}(D) = \langle\!|A, \mathrm{L_m}(D) \cap A|\!\rangle \\
\quad \text{where } A = \{w \in \mathrm{L_{um}}(D) \mid \mathrm{op_c}(\overline{\{w\}}, \Sigma_{\mathsf{uc}})\} \\
\mathrm{F_{c3}}(D) = \langle\!|A, B|\!\rangle \\
\quad \text{where } B = \{w \in \mathrm{L_m}(D) \mid \mathrm{op_c}(\overline{\{w\}}, \Sigma_{\mathsf{uc}})\} \\
\quad \text{and } A = \{w \in \mathrm{L_{um}}(D) \mid \begin{pmatrix} \wedge \mathrm{op_c}(\overline{\{w\}} \setminus \{w\}, \Sigma_{\mathsf{uc}}) \\ \wedge (\neg\, \mathrm{op_{cw}}(w, \Sigma_{\mathsf{uc}}) \to w \notin \overline{B}) \end{pmatrix}\} \\
\mathrm{F_{spec}}(D) = D \times S
\end{array}
$$

Figure 6. The good iterators $\mathrm{F_{bf}}$, $\mathrm{F_{c1}}$, $\mathrm{F_{c2}}$, $\mathrm{F_{c3}}$, and $\mathrm{F_{spec}}$ where the plant P and the specification S are omitted parameters.

5.1 Examples of Good Iterators

The iterators to be discussed are given in Table 6.

►*Blockfreeness:* The iterator $\mathrm{F_{bf}}$ generates the least restrictive DES that is blockfree and contained in the input. An implementation of this iterator is presented by Wonham and Ramadge (1987, Lemma 5.1) for DFA and in the companion paper by Schneider and Nestmann (2014) for DPDA. Let $\Phi_{\mathrm{bf}}(D) = \mathrm{L_{um}}(D) \subseteq \mathrm{L_m}(D)$.

Lemma 5. $\mathrm{F_{bf}} \in \mathcal{G}(\mathrm{DES}, \mathrm{DES} \cap \Phi_{\mathrm{bf}}, \mathrm{DES} \cap \Phi_{\mathrm{bf}})$. ■

►$\star$-*Controllability:* The iterator $\mathrm{F_{c1}}$ generates the least restrictive DES that is controllable and contained in the input. Let $\Phi_{\mathrm{cont}}(D) = \mathrm{LCont}(\mathrm{L_{um}}(D), \mathrm{L_{um}}(P), \Sigma_{\mathsf{uc}})$.

Lemma 6. Given a plant $P \in \mathrm{DES}$ over the uncontrollable symbols Σ_{uc}. $\mathrm{F_{c1}} \in \mathcal{G}(\mathrm{DES}, \mathrm{DES} \cap \Phi_{\mathrm{cont}}, \mathrm{DES} \cap \Phi_{\mathrm{cont}})$. ■

►1-*Controllability:* The difference between $\star$-controllability and the 1-controllability is that $\mathrm{F_{c1}}$ removes controllability problems for every finite sequence of uncontrollable symbols whereas $\mathrm{F_{c2}}$ only removes single-step controllability problems. Thereby, $\mathrm{F_{c2}}$ may produce "new" controllability problems which are not removed in the single application of $\mathrm{F_{c2}}$. Therefore, iterative application of $\mathrm{F_{c2}}$ always produces $\mathrm{F_{c1}}$: $\mathrm{gfp}(\lambda X.\mathrm{F_{c2}}(X \times Y)) = \mathrm{F_{c1}}(Y)$. An implementation of this iterator is presented by Wonham and Ramadge (1987, Lemma 5.1) for DFA.

Lemma 7. Given a plant $P \in \mathrm{DES}$ over the uncontrollable symbols Σ_{uc}. $\mathrm{F_{c2}} \in \mathcal{G}(\mathrm{DES}, \mathrm{DES} \cap \Phi_{\mathrm{cont}}, \mathrm{DES})$. ■

►*M-Controllability:* $\mathrm{F_{c3}}$ requires a blockfree input to ensure the welldefinedness of the resulting DES. For $\mathrm{F_{c3}}$, an unmarked word (which is not also a marked word) with a controllability problem (but where all strict prefixes are controllable) is not removed, if it cannot be extended to the created marked language. Therefore, $\mathrm{F_{c3}}$ is equivalent to $\mathrm{F_{c2}}$ up to blockfreeness: $\mathrm{F_{bf}} \circ \mathrm{F_{c3}} = \mathrm{F_{bf}} \circ \mathrm{F_{c2}}$. An implementation of this iterator is presented in the companion paper by Schmuck, Schneider, Raisch, and Nestmann (2014) for DPDA.

Example 2. (Differences between $\mathrm{F_{c1}}$, $\mathrm{F_{c2}}$ and $\mathrm{F_{c3}}$).
Let $D = \langle\!|\overline{\{aua\}}, \{aua\}|\!\rangle$ and P as given in Figure 1. Then $\mathrm{F_{c1}}(D) = \langle\!|\{\lambda\}, \emptyset|\!\rangle$, $\mathrm{F_{c2}}(D) = \langle\!|\overline{\{a\}}, \emptyset|\!\rangle$, and $\mathrm{F_{c3}}(D) = \langle\!|\overline{\{au\}}, \emptyset|\!\rangle$. For $\mathrm{F_{c3}}$: the word au is not removed because all its strict prefixes are controllable and even though it is not controllable it can also not be extended into the created marked language $\emptyset$. □

Lemma 8. Given a plant $P \in \mathrm{DES}$ over the uncontrollable symbols Σ_{uc}. $\mathrm{F}_{\mathrm{c3}} \in \mathcal{G}(\mathrm{DES} \cap \Phi_{\mathrm{bf}}, \mathrm{DES} \cap \Phi_{\mathrm{cont}}, \mathrm{DES})$. ∎

►*Satisfaction of the Specification:* Finally, the iterator $\mathrm{F}_{\mathrm{spec}}$ generates the least restrictive DES that is contained in the input and the specification. Let $\Phi_{\mathrm{spec}}(D) = D \leq S$.

Lemma 9. Given a specification $S \in \mathrm{DES}$. Then $\mathrm{F}_{\mathrm{spec}} \in \mathcal{G}(\mathrm{DES}, \mathrm{DES} \cap \Phi_{\mathrm{spec}}, \mathrm{DES} \cap \Phi_{\mathrm{spec}})$ ∎

Using the above properties on the individual good iterators, we can compose three good iterators.

Theorem 10. (Computation of $\mathcal{D}^{\mathrm{si}}_{\mathrm{mm}}$).
Given a plant $P \in \mathrm{DES}$ and a specification $S \in \mathrm{DES}$. Let $F_c \in \{\mathrm{F}_{\mathrm{c1}}, \mathrm{F}_{\mathrm{c2}}, \mathrm{F}_{\mathrm{c3}}\}$. Let $F = F_c \circ \mathrm{F}_{\mathrm{bf}} \circ \mathrm{F}_{\mathrm{spec}}$. Then $F \in \mathcal{G}(\mathrm{DES}, \{\langle\!|\overline{A}, A|\!\rangle \mid \Phi_{\mathrm{mm}}(A, \mathrm{L_m}(S))\}, \mathrm{DES})$. Finally, if $\mathcal{U}(F, \top)$ terminates, then $\mathcal{U}(F, \top) = \mathcal{D}^{\mathrm{si}}_{\mathrm{mm}}(P, S)$. ∎

That is, we have verified an approach, up to termination and fixed-point-detection, which synthesizes a least restrictive controller. However, (i) while the operation $\mathrm{F}_{\mathrm{spec}}$ is neutral from the second iteration onwards, this cannot be formally captured in the context of the complete lattice and therefore an implementation which follows the universal computation strictly has to re-execute $\mathrm{F}_{\mathrm{spec}}$ in every iteration and (ii) as F_{c3} requires a blockfree input, we are forced to operate with the input assumption of blockfree DES. While there is a decision procedure (Sénizergues, 1997) to determine whether our implementation of F_{bf} modifies its input, we are using the $\rhd$-composition in Theorem 11 to avoid the execution of such a computationally expensive equivalence test. We obtain the following result, which is also adequate for the DFA setting.

Theorem 11. (DPDA-Computation of $\mathcal{D}^{\mathrm{si}}_{\mathrm{mm}}$).
Given a plant $P \in \mathrm{DES}$ and a specification $S \in \mathrm{DES}$. Let $I = \mathrm{F}_{\mathrm{bf}}(\mathrm{F}_{\mathrm{spec}}(P))$ where the restricted specification $P \times S$ is used. Let $F = \mathrm{F}_{\mathrm{bf}} \rhd \mathrm{F}_{\mathrm{c3}}$. Then $F \in \mathcal{G}(\mathrm{DES} \cap \Phi_{\mathrm{bf}}, \mathrm{DES} \cap \Phi_{\mathrm{bf}} \cap \Phi_{\mathrm{cont}}, \mathrm{DES} \cap \Phi_{\mathrm{bf}})$. Finally, if $\mathcal{U}(F, I)$ terminates, then $\mathcal{U}(F, I) = \mathcal{D}^{\mathrm{si}}_{\mathrm{mm}}(P, S)$. ∎

The last theorem states that the universal algorithm, started with the initial argument I and iteratively executing the implementations of F_{bf} and F_{c3} presented in the two companion papers, determines (up to termination) the smallest maximal controller defined by the suprema-based characterization $\mathcal{D}^{\mathrm{si}}_{\mathrm{mm}}(P, S)$.

6. CONCLUSION

We have introduced a methodology for the extension of the SCP to domains broader than DFA. The technical problems resulting in such extensions are exemplified by considering DPDA-specifications which allow for λ-steps which are unobservable (when part of finite sequences) or observable (when part of infinite sequences). By means of nondeterminism and finalization (using an end-of-output marker $\diamond$) we have shown that the operational criteria (specifically operational blockfreeness) are more suitable than the trace-abstract criteria as they are strictly more demanding and adequate w.r.t. the task of characterization of the desired observable operational behavior (see Remark 2). While we have investigated how the operational criteria are satisfied by using certain encodings of the involved finite automata models, future research may also allow for the direct extension of the SCP by additional trace-abstract criteria. This work may also be extended by (i) adding further operational criteria as for example worst-case-execution times and (ii) by establishing initial results on size-optimal realizations of constructed controllers. Finally, we intend to prove the results on the encodings in Sections 2 and 3 in the interactive theorem prover Isabelle/HOL (Paulson et al., 2011) as it is already done for automata models from Section 1 and the results of Sections 4 and 5.

REFERENCES

Giua, A. and DiCesare, F. (1994). Blocking and controllability of petri nets in supervisory control. *IEEE Transactions on Automatic Control*, 39(4), 818–823. doi: 10.1109/9.286260.

Giua, A. and DiCesare, F. (1995). Decidability and closure properties of weak petri net languages in supervisory control. *IEEE Transactions on Automatic Control*, 40(5), 906–910. doi:10.1109/9.384227.

Griffin, C. (2008). A note on the properties of the supremal controllable sublanguage in pushdown systems. *IEEE Transactions on Automatic Control*, 53(3), 826 –829.

Griffin, C. (2007). *Decidability and optimality in pushdown control systems: A new approach to discrete event control*. Ph.D. thesis, The Pensylvania State University.

Hopcroft, J.E. and Ullman, J.D. (1979). *Introduction to Automata Theory, languages and computation*. Addison-Wesley Publishing company.

Paulson, L., Nipkow, T., and Wenzel, M. (2011). Isabelle/HOL. URL `http://isabelle.in.tum.de`.

Ramadge, P. and Wonham, W. (1984). Supervisory control of a class of discrete event processes. In A. Bensoussan and J. Lions (eds.), *Analysis and Optimization of Systems*, volume 63 of *Lecture Notes in Control and Information Sciences*, 475–498. Springer Berlin Heidelberg.

Schmuck, A.-K., Schneider, S., Raisch, J., and Nestmann, U. (2014). Extending supervisory controller synthesis to deterministic pushdown automata—enforcing controllability least restrictively. *Proceedings of the 12th IFAC - IEEE International Workshop on Discrete Event Systems*.

Schneider, S. and Nestmann, U. (2014). Enforcing operational properties including blockfreeness for deterministic pushdown automata. `http://arxiv.org/abs/1403.5081`.

Schneider, S. and Schmuck, A.-K. (2013). Supervisory controller synthesis for deterministic pushdown automata specifications. Technical report, Technical University of Berlin, URL `http://www.tu-berlin.de/?25631`.

Sénizergues, G. (1997). The equivalence problem for deterministic pushdown automata is decidable. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela (eds.), *ICALP*, volume 1256 of *Lecture Notes in Computer Science*, 671–681. Springer.

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