Characterizing Weakly Terminating Partners of Open Systems

Christian Stahl[1], Daniela Weinberg[2], and Karsten Wolf[2]

[1] Department of Mathematics and Computer Science, Technische Universiteit
Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
C.Stahl@tue.nl
[2] Universität Rostock, Institut für Informatik, 18051 Rostock, Germany
{Daniela.Weinberg, Karsten.Wolf}@uni-rostock.de

Abstract. We study open systems modeled as finite state machines with an interface for asynchronous communication with other open systems. An open system P is a *partner* of an open system S, if the two systems can be composed to a closed system $S \oplus P$ that is weakly terminating (does not have deadlocks nor livelocks). Known controller synthesis techniques allow us to construct a most permissive partner $MP(S)$ to a given open system S. Our contribution is to enhance $MP(S)$ with Boolean annotations, yielding a finite characterization of *all* partners of S. We demonstrate the usefulness of this characterization by involving it in a decision procedure for *accordance*. Accordance is a canonical preorder on open systems: R accords with S if every partner of S is a partner of R. Our decision procedure reduces accordance to finding a simulation relation between $MP(S)$ and $MP(R)$ and checking validity of implications between related Boolean formulas.

1 Introduction

Open systems occur in many areas of computer science, called modules, services, agents, or components. An open system is typically seen as a unit that evolves autonomously; that is, it can be changed (or replaced) in the context of an overall system that is not necessarily fully transparent to the component designer. Formal methods can support the process of changing or replacing open systems by establishing accordance relations between open systems that give guarantees on the effect to the whole system.

In this paper, we model an open system as a finite state machine with labels that represent asynchronous message exchange with the remainder of the overall system. As the correctness criterion we want to assert, we choose *weak termination*—that is, the absence of deadlocks and livelocks in the overall system. Weak termination is a property of highest practical relevance in the fields of workflows (called soundness [1]), services (e.g., [13]), and discrete event systems (called non-conflicting or nonblocking [43,17]. Consequently, we aim at an accordance relation between open systems that preserves weak termination.

From this setting, we can canonically derive the concept of a *partner* P of an open system S. This is another open system (playing the role of the remaining system or the environment) such that the composition $S \oplus P$ (to be defined) is a weakly terminating closed system. Then, again canonically, R accords with S if every partner of S is a partner of R, too. Accordance is obviously the coarsest preorder on open systems such that replacing S with an accordant R preserves weak termination in every environment P. It has been introduced in [31,13] for synchronous communication. In [31,13,36], the authors prove that should testing [44] is the coarsest traditional precongruence that implies accordance, but accordance does not imply should testing. All mentioned concepts are introduced in Sect. 2.

Our key to deciding an inclusion on (typically infinite) sets of partners is a *finite characterization of all partners* of a given open system S, called the operating guideline of S. The operating guideline consists of two ingredients: (1) a specific most permissive partner $MP(S)$ of S that can be obtained using standard controller synthesis techniques, and (2) a Boolean annotation to every strongly connected subgraph of $MP(S)$. While $MP(S)$ establishes an upper limit to the behavior of a partner of S, the annotations establish lower limits—that is, behavior that must be present at minimum to avoid deadlocks and livelocks in composition with S. In Sect. 3, we develop the concept of operating guidelines. We start with the construction of a most permissive partner. Although this is a standard controller synthesis approach—however, for possibly nondeterministic open systems and asynchronous communication—the explicit presentation permits us to introduce the *specific* controller we are using. Then we introduce the annotations, define a matching relation between open systems and operating guidelines, and finally establish the characterization: $S \oplus P$ is weakly terminating if and only if P matches with the operating guideline of S.

For deciding accordance between open systems S and R, we relate the most permissive partners of S and R by a simulation relation. This way, we obtain a relation between strongly connected subgraphs of the two most permissive partners. We show that deciding accordance amounts to checking whether implications between related formulas are valid. The result is spelled out in Sect. 4. Existing algorithms provide with should testing only a sufficient criterion [13] or assume synchronous communication [50].

Our contribution extends previous work where we used deadlock freedom as correctness criterion and it was sufficient to have Boolean annotations to *states* rather than strongly connected subgraphs. For this (much simpler) setting, we identified several useful applications for operating guidelines—for example, finding for an incorrect partner of S a most similar one for a suitable similarity measure [28] and the implementation of union, intersection, and complement on sets of partners [23]—which may serve as additional motivation for the partner characterization given here. Our previous work and links to related work by other authors are discussed in Sect. 5.

2 Preliminaries

For a set X, let X^* be the set of finite sequences (words) over X. For any ternary relation $\delta \subseteq A \times B \times C$, $a \xrightarrow{b}_\delta c$ denotes $(a,b,c) \in \delta$. For a ternary relation $\delta \subseteq A \times B \times A$, relation $\delta^* \subseteq A \times B^* \times A$ is its reflexive and transitive closure defined by $a \xrightarrow{b_1 \dots b_n}_{\delta^*} c$ iff there are $a_0, \dots, a_n$ such that $a = a_0$, $c = a_n$, and, for all $1 \leq i \leq n$, $a_{i-1} \xrightarrow{b_i}_\delta a_i$. If any of the elements a, b, or c is omitted, we mean the existence of such an element; $a \rightarrow_{\delta^*} c$ denotes that c is reachable from a in δ.

2.1 State Machines for Modeling Open Systems

An open system consists of a control structure describing its *internal behavior*. Transitions are labeled. Some of the transition labels are considered as input or output labels to describe the interaction of the state machine with its environment. We consider state machines with final states. Final states shall help us to distinguish between desired and undesired termination.

Definition 1 (state machine). A *state machine* $S = (Q, \hat{Q}, I, O, T, \delta, F)$ consists of

- a (countably infinite) set Q of *states*,
- a nonempty set of *initial states* $\hat{Q} \subseteq Q$,
- a finite set I of *input labels*, a finite set O of *output labels*, and a finite set T of *internal labels* such that I, O, and T are pairwise disjoint,
- a *transition relation* $\delta \subseteq Q \times (I \cup O \cup T) \times Q$, and
- a (countably infinite) set $F \subseteq Q$ of *final states.*

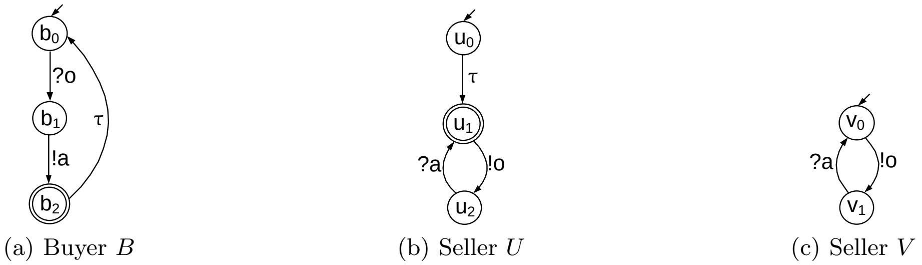
For a state $q \in Q$, define by $en(q) = \{x \mid q \xrightarrow{x}_\delta\}$ the set of outgoing transitions of q. S is *finite* if the set $\{q \mid \exists \hat{q} \in \hat{Q} : \hat{q} \rightarrow_{\delta^*} q\}$ of *reachable* states is finite.

We shall assume throughout this paper that a state machine has ingredients named as in Definition 1. If multiple state machines are involved, we shall use indices to distinguish their ingredients.

Next, we will coin some state machines as *deterministic*. Unlike standard automata theory, we will permit multiple initial states, internal state changes, and equally labeled transitions in $en(q)$. We will restrict all these permissions, however, to cases where resulting states can be distinguished by their nature (i.e., their *final* status).

Definition 2 (deterministic state machine). A state machine S is *deterministic* if, for all $q, q', q'' \in Q$, the following three conditions hold:

- $card(\hat{Q} \cap F) \leq 1$ and $card(\hat{Q} \setminus F) \leq 1$,
- if, for $x \in I \cup O$, $q \xrightarrow{x}_\delta q'$ and $q \xrightarrow{x}_\delta q''$ then $q' = q''$ or $card(\{q', q''\} \cap F) = 1$,
- if, for $\sigma \in T^*$, $q \xrightarrow{\sigma}_{\delta^*} q'$ then $q = q'$ or $card(\{q, q'\} \cap F) = 1$.



(a) Buyer B (b) Seller U (c) Seller V

Fig. 1. State machines modeling a buyer and two sellers. The symbol ? and ! precedes an output and an input label, respectively.

Note that by Definition 1 and the first item of Definition 2, we have $card(\hat{Q} \cap F) = 1$ or $card(\hat{Q} \setminus F) = 1$.

Example 1. As a running example, we use the open system of a buyer, which is modeled as the state machine B in Fig. 1. The buyer receives an offer sent by a seller ($?o$) and then accepts the order by sending an acceptance message ($!a$) to the seller. Thereby, the buyer returns to a state where it can receive further offers. This is modeled by an internal event.

The state machines U and V in Fig. 1 model two possible sellers of the buyer. Seller U sends an offer and requires a notification of acceptance. Seller V continuously sends new offers to the buyer.

Open systems are linked via message channels. The names of the channels correspond to the input and output labels of the system. For each message channel m, the state of the channel is represented by the number of pending messages. Consequently, we pool the state of all message channels connected to an open system S into a multiset (bag) $\mathcal{B} : I \cup O \to \mathbb{N}$ over $I \cup O$. Let $[\,]$ denote the empty multiset (assigning 0 to all arguments), and let $Bags(M)$ denote the set of all multisets over M.

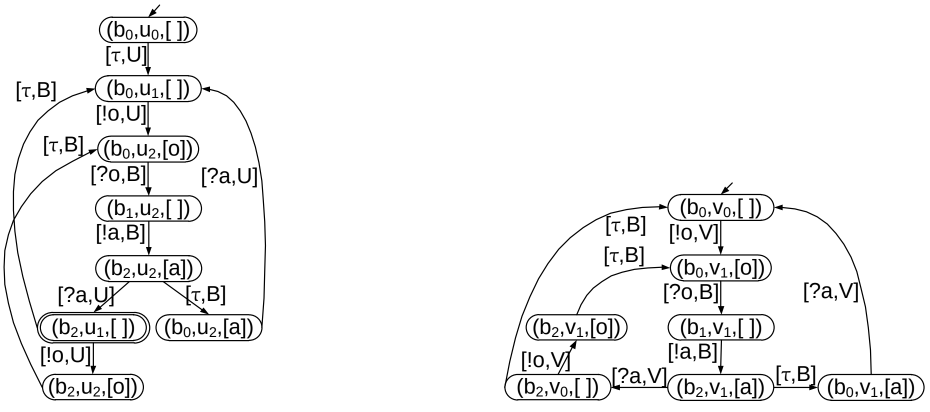
Using the multiset representation of message channels, we can lift the internal behavior $\delta \subseteq Q \times (I \cup O \cup T) \times Q$ to the *external behavior* Δ of S.

Definition 3 (external behavior). For a state machine S, define its *external behavior* $\Delta \subseteq (Q \times Bags(I \cup O)) \times (I \cup O \cup T) \times (Q \times Bags(I \cup O))$ by the following laws. For $q \xrightarrow{x}_{\delta} q'$ and $\mathcal{B} \in Bags(I \cup O)$:

- if $x \in I$ then $(q, \mathcal{B} + [x]) \xrightarrow{x}_{\Delta} (q', \mathcal{B})$,
- if $x \in O$ then $(q, \mathcal{B}) \xrightarrow{x}_{\Delta} (q', \mathcal{B} + [x])$,
- if $x \in T$ then $(q, \mathcal{B}) \xrightarrow{x}_{\Delta} (q', \mathcal{B})$.

Open systems interact by operating on the same message channels. In this paper, we only consider the interaction between two open systems.

Two state machines S and P are *composable* if the input channels of one state machine are the output channels of the other. The result is a closed system; that



(a) $B \oplus U$

(b) $B \oplus V$

Fig. 2. The compositions of the buyer and the sellers.

is, a system that has only internal transitions. For the sake of preserving the origin of transitions, we tag the internal labels of the composed system with either "S" or "P".

Definition 4 (composition). State machines $S = (Q_S, \hat{Q}_S, I_S, O_S, T_S, \delta_S, F_S)$ and $P = (Q_P, \hat{Q}_P, I_P, O_P, T_P, \delta_P, F_P)$ are *composable* if $I_S = O_P$ and $I_P = O_S$. The *composition* of two composable state machines S and P is the state machine $S \oplus P = (Q, \hat{Q}, I, O, T, \delta, F)$ defined as

- $Q = Q_S \times Q_P \times Bags(I_S \cup I_P)$,
- $\hat{Q} = \hat{Q}_S \times \hat{Q}_P \times \{[\,]\}$,
- $I = O = \emptyset$,
- $T = (I_S \cup O_S \cup T_S) \times \{S\} \cup (I_P \cup O_P \cup T_P) \times \{R\}$,
- $(q_S, q_P, \mathcal{B}) \xrightarrow{[x,S]}_{\delta} (q'_S, q_P, \mathcal{B}')$ if $(q_S, \mathcal{B}) \xrightarrow{x}_{\Delta_S} (q'_S, \mathcal{B}')$,
- $(q_S, q_P, \mathcal{B}) \xrightarrow{[x,P]}_{\delta} (q_S, q'_P, \mathcal{B}')$ if $(q_P, \mathcal{B}) \xrightarrow{x}_{\Delta_P} (q'_P, \mathcal{B}')$, and
- $F = F_S \times F_P$.

Having tagged the labels in $S \oplus P$, we can define projections of sequences over T onto S and P, respectively. For a sequence $\sigma \in T^*$, $\sigma|_S = x_1 \ldots x_n$ is its *projection* onto S if $\sigma = \tau_0[x_1, S]\tau_1 \ldots [x_n, S]\tau_n$ such that $\tau_0 \ldots \tau_n \in ((I_P \cup O_P \cup T_P) \times \{P\})^*$. The projection $\sigma|_P$ is defined accordingly. We obtain the simple lemma:

Lemma 5. *If* $(q_S, q_P, \mathcal{B}) \xrightarrow{\sigma}_{\delta^*_{S \oplus P}} (q'_S, q'_P, \mathcal{B}')$ *then* $q_S \xrightarrow{\sigma|_S}_{\delta^*_S} q'_S$ *and* $q_P \xrightarrow{\sigma|_P}_{\delta^*_P} q'_P$*.*

Example 2. In Fig. 1, the state machine B is composable to both state machines U and V. Figure 2 shows the compositions of B and U and V. Throughout this paper, we do not depict states that are not reachable from the initial state.

To compare the behavior of two given state machines S and R, we use a (strong) simulation relation [34]. "Strong" means that an internal transition of one system must be matched by an internal transition of the other system, but not necessarily the same one. We further demand that S and R have the same set of input and output channels, respectively. As usual, two states q_S and q_R are only related by a simulation relation if both are either final states or not.

Definition 6 (simulation relation). Let $S = (Q_S, I, O, T_S, \hat{Q}_S, \delta_S, F_S)$ and $R = (Q_R, I, O, T_R, \hat{Q}_R, \delta_R, F_R)$ be state machines. A binary relation $\varrho \subseteq Q_S \times Q_R$ is a *simulation relation of S by R* if

- for all $\hat{q}_S \in \hat{Q}_S$, there exists an $\hat{q}_R \in \hat{Q}_R$ such that $(\hat{q}_S, \hat{q}_R) \in \varrho$;
- for all $(q_S, q_R) \in \varrho$, $a \in I \cup O \cup T_S$, $q'_S \in Q_S$ such that $(q_S, a, q'_S) \in \delta_S$, there exists a state $q'_R \in Q_R$ and an $a' \in I \cup O \cup T_R$ such that $(q_R, a', q'_R) \in \delta_R$, $(q'_S, q'_R) \in \varrho$, and ($a = a'$ or ($a \in T_S$ and $a' \in T_R$)); and
- for all $(q_S, q_R) \in \varrho$, $q_S \in F_S$ iff $q_R \in F_R$.

If such a ϱ exists, we say that R *simulates* S. If R is deterministic then there exists a unique *minimal simulation relation* ϱ of S by R, i.e., $\varrho \subseteq \varrho'$, for all simulation relations ϱ' of S by R.

The minimal simulation relation can be computed inductively over δ_S (starting with $\hat{Q}_S$) because, by determinism of R, the respective matching q_R is uniquely determined by the requirements for simulation relations.

2.2 Weak Termination and Partner

We are interested in "correct" collaborations of open systems. In this paper, "correct" means that the composition neither contains deadlocks nor livelocks.

Definition 7 (strongly connected sets (SCS) and components (SCC)). Two states $q, q' \in Q$ of a state machine are *mutually reachable* if $q \rightarrow_{\delta^*} q'$ and $q' \rightarrow_{\delta^*} q$. Mutual reachability is an equivalence relation on states of a state machine, and its equivalence classes are *strongly connected components* (SCCs). Every set of states where all elements are pairwise mutually reachable is a *strongly connected set* (SCS). An SCS (or SCC) C is *reachable* if $\hat{q} \rightarrow_{\delta^*} q$ for any $\hat{q} \in \hat{Q}$ and any $q \in C$. An SCC C is *terminal* (TSCC) if no state of another SCC is reachable from any state of C.

SCCs are the largest SCSs. If an SCS is reachable, all its elements are reachable from an initial state.

A reachable TSCC with a singleton nonfinal state without a self-loop is a *deadlock*; every other reachable TSCC, which does not contain a final state, is a *livelock*. Note that our definition of a deadlock differs from the standard definition in literature, as we discriminate between final states and deadlocks.

Weak termination requires that from every reachable state, a final state is reachable.

Definition 8 (weak termination). A finite state machine is *weakly terminating* if, for all $\hat{q} \in \hat{Q}$ and all $q \in Q$ with $\hat{q} \rightarrow_{\delta^*} q$, there is a $q_f \in F$ such that $q \rightarrow_{\delta^*} q_f$.

For finite state machines, there are several alternative characterizations for weak termination.

Proposition 9. *For any finite state machine S, the following five statements are equivalent:*

1. *S is weakly terminating.*
2. *S contains neither deadlocks nor livelocks.*
3. *Every set of nonfinal states can be left; that is, for all nonempty reachable sets $C \subseteq Q$, $C \cap F = \emptyset$ implies there are states $q \in C$ and $q' \notin C$ with $q \rightarrow_{\delta} q'$.*
4. *Every reachable TSCC contains a final state.*
5. *S satisfies the CTL formula AG EF final where final is an atomic proposition that is true of all states in F and false of all other states.*

If the collaboration of two open systems is weakly terminating, each system is referred to as a *partner* of the other one. The collaboration of two finite systems is not necessarily finite. In particular, indefinitely growing numbers of messages may yield infinitely many states. For the sake of staying in the realm of finite state systems, we introduce the notion of a *b-partner*.

Definition 10 (b-partner). Let $b \in \mathbb{N}$. A finite state machine P is a *b-partner* of a finite state machine S if $S \oplus P$ is weakly terminating and for all reachable states $(q_S, q_P, \mathcal{B}) \in Q_{S \oplus P}$ and all $x \in I_S \cup I_P$, $\mathcal{B}(x) \leq b$.

The introduction of the parameter b establishes an artificial limit on the number of pending messages in a single channel. Technically, it assures (for any value of b) that the collaboration has a finite number of reachable states. This is a prerequisite for the algorithms studied in the sequel, because without this message bound the existence of a partner for a state machine is undecidable [33]. Pragmatically, the bound could either represent a reasonable buffer size in the middleware—for example, the result of a static analysis of the communication behavior of a system—or simply be chosen sufficiently large.

Example 3. The composition $B \oplus U$ in Fig. 2(a) is weakly terminating, because a final state $(b_2, u_1, [\,])$ is reachable from every reachable state. In contrast, the composition $B \oplus V$ in Fig. 2(b) is not weakly terminating because it does not contain a final state. Thus, U is a 1-partner of B but V is not.

In the remainder of this subsection, we distinguish particular b-partners.

Definition 11 (most permissive). A b-partner P' of S is *most permissive* if P' is a deterministic state machine and simulates all b-partners P of S.

A state machine may have several different most permissive b-partners. As, by definition, all of them simulate each other and, in addition, the minimal simulation relations are unique, these simulation relations are reversible, and we can state the following proposition:

Proposition 12. *Any two most permissive b-partners of S are bisimilar.*

Despite bisimilarity, most permissive b-partners may exhibit differences. One of them is concerned with *precision*. We introduce this concept for measuring, how much information ("knowledge") about S is represented in a state q_P of a b-partner P. In other words, we are interested in the states S might be in while P is in q_P. We generally prefer small sets $k_S(q_P)$, because then the transitions of the partner can be specific to only few situations in the collaboration.

Definition 13 (knowledge, precision). Let S and P be composable. For $q_P \in Q_P$, the set $k_S(q_P) = \{(q_S, \mathcal{B}) \mid (q_S, q_P, \mathcal{B}) \text{ reachable in } S \oplus P\}$ defines the *knowledge of* q_P. Let P_1 and P_2 be both deterministic and composable to S. If P_1 is simulated by P_2 via the minimal simulation relation ϱ, we say that P_1 is *more precise* than P_2 w.r.t. S if, for all $(q_1, q_2) \in \varrho$, $k_S(q_1) \subseteq k_S(q_2)$.

A most precise b-partner is not necessarily the one that can be obtained by bisimulation minimization. In fact, merging states corresponds to building the union of involved k_S-sets.

In the next section, we will show how to construct a finite minimal b-partner for S among those which are most permissive and most precise. As this b-partner is unique up to isomorphism, we shall refer to it as $MP_b(S)$.

3 Partner Characterization

An operating guideline for a state machine S characterizes the possibly infinite set of b-partners of S in a finite manner. It is based on the minimal most permissive and most precise b-partner $MP_b(S)$ announced in the previous section. Being most permissive, any other b-partner of S can be understood as having a more restricted behavior than $MP_b(S)$. However, the restricted behavior must still be rich enough to preserve weak termination. To this end, we annotate every strongly connected subgraph of $MP_b(S)$ with a Boolean formula. This formula spells out the requirement that sufficiently many transitions remain in a b-partner P to allow to leave all sets of nonfinal states in $S \oplus P$ as required in Proposition 9(3).

3.1 Constructing a Most Permissive and Most Precise Partner

First, we aim to construct the minimal most permissive and most precise b-partner $MP_b(S)$ to a given finite state machine S. The construction is based on computing minimal candidates for values of $k(q)$ to assure most preciseness and then to identify a state with its k_S-image ($q := k_S(q)$) to assure minimality.

Most permissiveness is proven subsequently. We believe that our construction is standard in automata theory and controller synthesis—although it is often restricted deterministic automata or synchronous communication—but we spell out the construction, as we will exploit details of the construction later on.

Let S be a finite state machine, and let q be a state of a composable finite state machine P. First, we observe that the presence of some states in $k_S(q)$ implies that some other states are unavoidably present in $k_S(q)$. This is reflected in the following definition.

Definition 14 (closure, b-set). Let $b \in \mathbb{N}$ and S be a finite state machine. Define the *closure* of $Q \subseteq Q_S \times Bags(I \cup O)$ as the set $cl(Q) = \{(q', \mathcal{B}') \mid \exists (q, \mathcal{B}) \in Q : (q, \mathcal{B}) \rightarrow_{\Delta_S^*} (q', \mathcal{B}')\}$. If for all $(q, \mathcal{B}) \in Q$ and all $x \in I \cup O, \mathcal{B}(x) \leq b$ then Q is a *b-set*.

From Definition 4, we can immediately deduce:

Proposition 15. *If P is composable to S and $q_P \in Q_P$ then $Q \subseteq k(q_P)$ implies $cl(Q) \subseteq k(q_P)$. If $q_P \in \hat{Q}_P$ then $cl(\hat{Q}_S \times \{[\,]\}) \subseteq k(q_P)$.*

In subsequent constructions, we aim to replace inclusion by equality thus achieving maximum precision.

Next, we model the effect of transitions in P on sets $k_S(q)$. This operation will be used to define the transitions of $MP_b(S)$. While input and output labels of P are determined by S, we are free to chose a set of internal labels T_P. We do so by intentionally leaving the domain of the second argument in the subsequent definition undefined.

Definition 16 (set-step). Let S be a finite state machine. For $Q \subseteq Q_S \times Bags(I \cup O)$, define the *set-step* function as follows:

- if $x \in I_S$, $\text{set-step}(Q, x) = \{(q, \mathcal{B} + [x]) \mid (q, \mathcal{B}) \in Q\}$,
- if $x \in O_S$, $\text{set-step}(Q, x) = \{(q, \mathcal{B}) \mid (q, \mathcal{B} + [x]) \in Q\}$,
- if $x \notin I_S \cup O_S$, $\text{set-step}(Q, x) = Q$.

Again, Definition 4 yields:

Proposition 17. *If P is composable to S and $q_P \xrightarrow{x}_{\delta_P} q'_P$, then set-step$(k(q_P)) \subseteq k(q'_P)$.*

Having a closer look at the concepts introduced so far, we can even state the following result:

Proposition 18. *If P is composable to S then, for all unreachable $q_P \in Q_P$, $k(q_P) = \emptyset$ while for all reachable $q_P \in Q_P$,*

$$k(q_P) = cl(Q_0) \cup \bigcup_{q'_P, x : q'_P \xrightarrow{x}_{\delta_P} q_P} cl(\textit{set-step}(k(q'_P), x))$$

where $Q_0 = \hat{Q}_S \times \{[\,]\}$ if $q_P \in \hat{Q}_P$ and $Q_0 = \emptyset$, else.

Next, we construct an overapproximation of $MP_b(S)$ of S. We define states based on candidate sets for their k_S-values. Each state occurs in a nonfinal and a final version. This is achieved by tagging each set with either 1 (nonfinal) or 2 (final). Transitions are defined such that they do not cause any loss in precision. Furthermore, we insert states only if their k-values are b-closures, for some given b. This way, our construction is finite. Nevertheless, the construction covers all b-partners, as for every state q of a b-partner P of S, $k(q)$ must be a b-set.

Definition 19 (overapproximation). Let $b \in \mathbb{N}$. For a finite state machine $S = (Q_S, \hat{Q}_S, I, O, T, \delta_S, F_S)$, define the state machine $TS_b^0(S) = (\mathcal{Q}, I, O, \{\tau\}, \hat{\mathcal{Q}}, \delta, \mathcal{F})$ inductively as follows:

- Base: Let $C = cl(\hat{Q}_S \times \{[\,]\})$. If C is a b-set, then $\hat{\mathcal{Q}} = C \times \{1, 2\}$; otherwise, $\hat{\mathcal{Q}} = \emptyset$. Let initially $\mathcal{Q}$ be $\hat{\mathcal{Q}}$ and $\delta = \emptyset$.
- Step: If $(C, i) \in Q$, $x \in I \cup O \cup \{\tau\}$, let $C' = cl(\text{set-step}((C, i), x))$. If C' is a b-set then let $C' \times \{1, 2\} \subseteq \mathcal{Q}$ and $\{((C, i), x, (C', i)), ((C, i), x, (C', 3-i))\} \subseteq \delta$.
- Define $F = \{(C, i) \in \mathcal{Q} \mid i = 2\}$.

This construction may result in a state machine with empty set of states. This is the case if already the construction of the initial states violates the bound b for the pending messages. In this case, S does not have b-partners as b can be violated by transitions of S which cannot be avoided by any b-partner.

Using Propositions 15, 17 and 18, it is easy to verify:

Proposition 20. *For all $(C, i) \in Q_{TS_b^0(S)}$, $k_S((C, i)) = C$.*

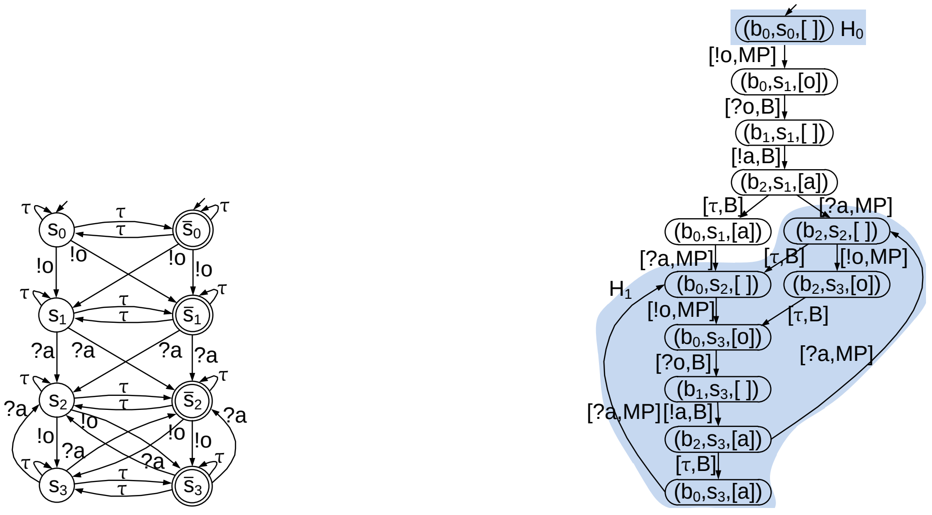
The next lemma shows that $TS_b^0(S)$ overapproximates all b-partners of S.

Lemma 21 (overapproximation simulates all b-partners). *Let $b \in \mathbb{N}$ and S be a finite state machine. Then, $TS_b^0(S)$ simulates every b-partner P of S.*

Proof. Let P be a b-partner of S. Then $TS_b^0(S)$ exists. Because $TS_b^0(S)$ is deterministic, there is a single candidate for a minimal simulation relation that relates one of the initial states of P to either the final or the nonfinal initial state of $TS_b^0(S)$ and, for each state, any x-successor of a state $q_P \in Q_P$ with $(q_P, Q) \in \varrho$ to the unique x-successor of Q that has the fitting final status. We show that this candidate is indeed a simulation relation. Furthermore, we show that $(q_P, (C, i)) \in \varrho$ implies that $C \subseteq k(q_P)$.

Let $\hat{q}_P \in \hat{Q}_P$. Because $TS_b^0(S)$ exists, it has an initial state (C, i) where i corresponds to the final status of $\hat{q}_P$. So, we have $(\hat{q}_P, (C, i)) \in \varrho$ fulfilling the requirements for initial states in a simulation relation. By Proposition 15, $C \subseteq k(q_P)$.

Next, let $(q_P, q) \in \varrho$ and $q_P \xrightarrow{x}_P q'_P$. If no x-successor (C, i) to q exists in $TS_b^0(S)$, C is not a b-set as this is the only condition for omitting x-successors. In this case, however, Proposition 17 and the inductive assumption would immediately reveal that P itself cannot be a b-partner, contradicting the general assumption. If, however, an x-successor exists, there are actually two of them where one is final and the other is nonfinal. Consequently, the simulation property continues to hold. In addition, it is easy to re-establish the condition $C \subseteq k(q'_P)$. □



(a) $MP_1(B)$

(b) $B \oplus MP_1(B)$

Fig. 3. The most permissive 1-partner $MP_1(B)$ of B and the composition of its nonfinal states with B. To improve readability, the $[\tau, MP]$-selfloops at every state of $B \oplus MP_1(B)$ are not shown.

We restrict $TS_b^0(S)$ to weak termination by removing all states from $TS_b^0(S)$ that result in a deadlock or livelock in the composed system. The latter task must be repeated, as the removal of states may introduce new deadlocks or livelocks.

Definition 22 (construction). Let $b \in \mathbb{N}$ and S be a finite state machine. For $i \geq 0$, obtain $TS_b^{i+1}(S)$ from $TS_b^i(S)$ by removing all states Q where there exists $(q_S, \mathcal{B}) \in Q$ such that $(q_S, Q, \mathcal{B})$ is a member of a TSCC in $S \oplus TS_b^i(S)$ without final state. Also remove all adjacent transitions and all states that become unreachable from initial states. Let $MP_b(S)$ be the fixed point reached by this construction.

Example 4. Figure 3(a) shows the state machine $MP_1(B)$. For every nonfinal state s, there exists a corresponding final state $\overline{s}$ and s and $\overline{s}$ are mutually reachable. In addition, every state has a τ-labeled selfloop to ensure strong simulation of any 1-partner of B. To illustrate the construction if Definitions 19 and 22, let $C = \{(b_0, [\,])\}$ and $C' = \{(b_0, [o]), (b_0, [a]), (b_1, [\,]), (b_2, [a])\}$. Then, $s_0 = (C, 1)$, $s_1 = (C', 1)$, $s_2 = (C \cup \{(b_2, [\,])\}, 1)$, and $s_3 = (C' \cup \{(b_2, [o])\}, 1)$. Each final state has the same knowledge as its corresponding nonfinal state, but is labeled 2.

We show that if the state machine $MP_b(S)$ has at least one state, then it is a most permissive b-partner of S; otherwise, no b-partner exists.

Lemma 23 (most permissive partner). *Let $b \in \mathbb{N}$. A finite state machine S has a b-partner iff the set $\mathcal{Q}$ of states of $MP_b(S)$ is nonempty.*

Proof. $\Rightarrow$: Suppose P is a b-partner of S. Let i be the largest i such that there is a simulation of P by $TS_b^i(S)$. If there is no simulation of P by $TS_b^{i+1}(S)$, then $S \oplus P$ must violate weak termination (if $i = 0$) as this is the only reason for removing additional states from $TS_b^i(S)$.

$\Leftarrow$: The construction of $MP_b(S)$ terminates because we start with a finite structure and only remove states and transitions. If the resulting structure is not empty, it must be a b-partner because all reasons not to be a b-partner have been erased: Removal of ingoing transitions does not change the condition $k((C, i)) = C$ for remaining reachable states. Thus, the message bound b cannot be violated as all k-values are b-sets. Weak termination cannot be violated because, upon termination, no TSCC without final state are there to be removed. □

Now, we conclude the main result of this section:

Theorem 24. *Let $b \in \mathbb{N}$ and S be a finite state machine. The state machine $MP_b(S)$ is the most permissive and most precise b-partner of S.*

Proof. Most permissive follows from Lemma 21 and Lemma 23. The overapproximation $TS_b^0(S)$ of $TS_b(S)$ is most precise by Proposition 18. As the construction of $TS_b(S)$ does not change this property, we conclude that $TS_b(S)$ is most precise. □

Example 5. With Theorem 24, we conclude that $MP_1(B)$ in Fig. 3(a) is the most permissive and most precise 1-partner of B.

In the remainder of this paper, we restrict ourselves to finite state machines and assume $b \in \mathbb{N}$.

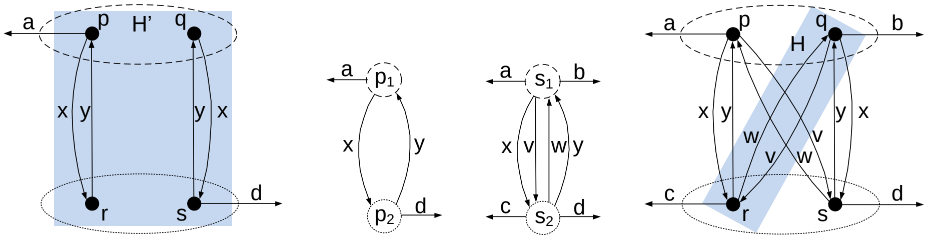
3.2 Representing all Partners in a Finite Manner

With the most permissive and most precise b-partner of a state machine, we constructed the first ingredient of its operating guideline. In the following, we show how the second ingredient, the annotation, can be computed.

We first recapitulate some terms from graph theory.

Definition 25 (subgraph, SCSG). A *subgraph* G of a state machine $S = (Q, \hat{Q}, I, O, T, \delta, F)$ is a triple (Q_G, δ_G, F_G) with $Q_G \subseteq Q$, $\delta_G \subseteq \delta$, and $F_G = F \cap Q_G$. If the set Q_G is an SCS of G, then G is a *strongly connected subgraph* (SCSG). An SCSG $G = (Q_G, \delta_G, F_G)$ is *induced by* Q_G if for all $q, q' \in Q_G$ and for all $x \in I \cup O \cup T$, $(q, x, q') \in \delta$ iff $(q, x, q') \in \delta_G$.

We also need to relate subgraphs H in the composition $S \oplus P$ with an SCSG G of P. We require that the projection of H onto the states and transitions of P is G. In addition, every transition originating from S and having a source in H must be present in H as well as every transition originating from P being present in G and having a source in H, must be present in H.



(a) $S \oplus P$ (b) P (c) $MP(S)$ (d) $S \oplus MP(S)$

Fig. 4. Example motivating the annotation of SCSGs in operating guidelines.

Definition 26 (valid subgraph). Let $S \oplus P$ be the composition of the state machines S and P, and let G be an SCSG of P. A *subgraph H of $S \oplus P$ is valid* w.r.t. G if the following two conditions hold:

1. The projection of H onto the states and transitions of P yields G, i.e.,
   - $Q_G = \{q_P \mid (q_S, q_P, \mathcal{B}) \in Q_H\}$,
   - $\delta_G = \{(q_P, x, q'_P) \mid (q_S, q_P, \mathcal{B}) \xrightarrow{[x,P]}_{\delta_H} (q_S, q'_P, \mathcal{B}')\}$,
   - $F_G = Q_G \cap F_P$.
2. The transition relation δ_H is complete, i.e.,
   - $\forall e = (q_S, q_P, \mathcal{B}) \xrightarrow{[x,S]}_{\delta_{S \oplus P}} (q'_S, q_P, \mathcal{B}') : (q_S, q_P, \mathcal{B}) \in Q_H$ implies $e \in \delta_H$, and
   - $\forall (q_P, x, q'_P) \in \delta_G \, \forall (q_S, q_P, \mathcal{B}) \in Q_H : e = (q_S, q_P, \mathcal{B}) \xrightarrow{[x,P]}_{\delta_{S \oplus P}} (q_S, q'_P, \mathcal{B}')$ implies $e \in \delta_H$.

An operating guideline specifies which actions $MP(S)$ has to perform to leave the SCSs of $S \oplus MP(S)$. This information is encoded as Boolean formulae. To enable P to use this information, we relate the states of P and $MP(S)$ using the minimal simulation relation of P by $MP(S)$ and determine, based on the outgoing edges of P, whether the formulae of the operating guideline is evaluated to true and, thus, $S \oplus P$ is weakly terminating. It turns out that we have to annotate all SCSGs of $MP(S)$ rather than its SCSs or SCCs. The example in Fig. 4 illustrates this.

Example 6. Figure 4 shows (a part of) $MP(S)$ and the corresponding parts of P, $S \oplus P$ and $S \oplus MP(S)$. Assume we annotate every SCS G of $MP(S)$ specifying how to leave (move on in) every subgraph H of $S \oplus MP(S)$ that is valid w.r.t. G. Let $G = \{s_1, s_2\}$. Then there exists the subgraph H in Fig. 4(d) in $S \oplus MP(S)$ that is valid w.r.t. G. To leave H (or at least to move on), at least one of the edges b, c, v, w has to be present, resulting in the annotation $b \vee c \vee v \vee w$. Consider P and its SCS $G' = \{p_1, p_2\}$. The edges leaving G' (i.e., a and d) do not define a satisfying assignment to the formula $b \vee c \vee v \vee w$, and we would (wrongly) assume that $S \oplus P$ is not weakly terminating; however, the subgraph H' in Fig. 4(a) can be left. Thus, annotating SCSs may exclude partners.

This can be circumvented by annotating SCSGs instead. In this case, the SCSG induced by G' is used to evaluate the annotation of those SCSGs G^* of $MP(S)$ that contain the edges x and y but neither v nor w. The respective formula of G^* is only defined by subgraphs of $S \oplus MP(S)$ that are valid w.r.t. G^*. Consequently, the the subgraph H does not contribute to the definition of the annotation of G^*.

Having this in mind, we can define an operating guideline of S. As subgraphs in the composition of $S \oplus P$ can only cause a problem if P cannot perform certain transitions, only subgraphs H that neither contain a final state nor are left by a transition originating from S define the annotation of an SCSG of $MP(S)$. The formula is then the conjunction over all those subgraphs H.

Definition 27 (operating guideline). Let S be a state machine for which $MP_b(S)$ exists. For an SCSG G of $MP_b(S)$, the set $\mathcal{H}_G$ denotes all subgraphs H of $S \oplus MP_b(S)$ that are valid w.r.t. G such that

- H does not contain a final state; and
- No transition originating from S leaves H.

A pair $(MP_b(S), \Phi)$ is an *b-operating guideline* of S if, for all SCSGs G, the Boolean annotation of G is defined as

$$\Phi(G) = \bigwedge_{H : H \in \mathcal{H}_G} \quad \bigvee_{e \in \{(q,x,q') \in \delta_{MP_b(S)} | \exists (q_S,q,B) \in Q_H : (q_S,q,B) \xrightarrow{x} \delta_{S \oplus MP_b(S)}\}} e \quad .$$

Each clause of the formula $\Phi(G)$ is defined by an SCSG H. It is a disjunction over all transitions originating in $MP_b(S)$ and having a state of Q_H as its source. In other words, we collect transitions that leave H but also transitions being internal to H.

Example 7. The 1-operating guideline of B consists of $MP_1(B)$ (see Fig. 3(a)) and a formula Φ. We give some example annotations, thereby using the part of $B \oplus MP_1(B)$, which is in Fig. 3(b). For the SCSG $G_0 = (\{s_0\}, \{\}, \{\})$, there is only one subgraph H_0 in $B \oplus MP_1(B)$ (highlighted in Fig. 3(b)) yielding $\Phi(G_0) = (s_0, \tau, s_0) \vee (s_0, \tau, \overline{s}_0) \vee (s_0, !o, s_1) \vee (s_0, !o, \overline{s}_1)$. Accordingly, $\Phi(G_1) = (\overline{s}_0, \tau, \overline{s}_0) \vee (\overline{s}_0, \tau, s_0) \vee (\overline{s}_0, !o, \overline{s}_1) \vee (\overline{s}_0, !o, s_1)$ for the SCSG $G_1 = (\{\overline{s}_0\}, \{\}, \{\})$ (not shown in Fig. 3(b)). For $G_3 = (\{s_2, s_3\}, \{(s_2, !o, s_3), (s_3, ?a, s_2)\}, \{\})$, the largest subgraph, H_1, is highlighted in Fig. 3(b). All subgraphs require that at least one of the eight edges with a source in G_3 are present; thus, $\Phi(G_3)$ is a disjunction over these edges.

We relate the states of P to the states of $MP_b(S)$ using the minimal simulation relation, which is a necessary requirement for P being a b-partner of S. To avoid the consideration of unfolded cycles, we consider the state machine which is defined by this simulation relation.

Definition 28 (matching graph). Let ϱ be the minimal simulation relation of a state machine $R = (Q_R, \hat{Q}_R, I, O, T_R, \delta_R, F_R)$ and a deterministic state machine $S = (Q_S, \hat{Q}_S, I, O, T_S, \delta_S, F_S)$. Define the state machine $Z = (Q_Z, \hat{Q}_Z, I, O, T_Z, \delta_Z, F_Z)$ by

- $Q_Z = \varrho$,
- $\hat{Q}_Z = Q_Z \cap (\hat{Q}_R \times \hat{Q}_S)$,
- $T_Z = T_R$,
- $\delta_Z = \{((q_R, q_S), (x, x'), (q'_R, q'_S)) \mid (q_R, q_S), (q'_R, q'_S) \in \varrho \wedge (q_R, x, q'_R) \in \delta_R \wedge (q_S, x', q'_S) \in \delta_S \wedge (x = x' \vee (x \in T_R \wedge x' \in T_S))\}$, and
- $F_Z = Q_Z \cap (F_R \times F_S)$.

The *matching graph* $MG(R, S) = (Q, \hat{Q}, I, O, T, \delta, F)$ of R and S is the reachable portion of Z with

- $Q = \{q \mid \exists \hat{q} \in \hat{Q} : \hat{q} \rightarrow_{\delta_Z^*} q\}$,
- $\hat{Q} = \hat{Q}_Z$,
- $T = T_Z$,
- $\delta = \delta_Z \cap (Q \times (I \cup O \cup T_Z) \times Q)$, and
- $F = Q \cap F_Z$.

Note that we could also define δ_Z such that it only considers the transition label x rather than a pair (x, x'), because x' is uniquely defined.

The projection of an SCS of a matching graph onto the states of its components is an SCS in the respective component.

Proposition 29. *If G is an SCS of $MG(R, S)$, then the projection G_R of G onto the states of R is an SCS of R and the projection G_S of G onto the states of S is an SCS of S.*

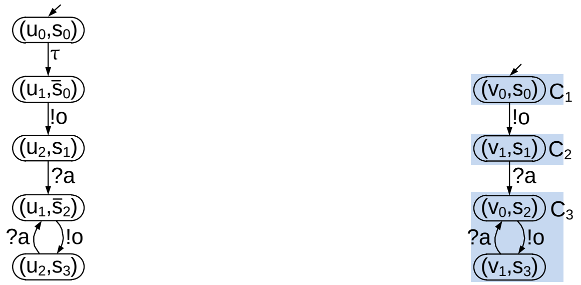
An SCC of $MG(P, MP_b(S))$ induces an SCSG of P and of $MP_b(S)$. Existence of these two SCSGs is justified by Proposition 29.

Definition 30 (induced SCSG). Let the SCSG $G = (Q_G, \delta_G, F_G)$ be induced by an SCC of a matching graph $MG(R, S) = (Q, \hat{Q}, I, O, T, \delta, F)$. The SCSG G *induces the SCSG* $G_R = (Q', \delta', F')$ in R with

- $Q' = \{q_R \mid (q_R, q_S) \in Q_G\}$,
- $\delta' = \{(q_R, x, q'_R) \mid \exists q_S, q'_S \in Q_S : ((q_R, q_S), (x, x'), (q'_R, q'_S)) \in \delta_G\}$, and
- $F' = \{q_R \mid (q_R, q_S) \in F_G\}$.

Likewise, G *induces the SCSG* $G_S = (Q'', \delta'', F'')$ in S with

- $Q'' = \{q_S \mid (q_R, q_S) \in Q_G\}$,
- $\delta'' = \{(q_S, x', q'_S) \mid \exists q_R, q'_R \in Q_R : ((q_R, q_S), (x, x'), (q'_R, q'_S)) \in \delta_G\}$, and
- $F'' = \{q_S \mid (q_R, q_S) \in F_G\}$.



(a) $MG(U, MP_1(B))$ (b) $MG(V, MP_1(B))$

Fig. 5. Two matching graphs

The procedure for checking containment of a state machine P in a b-operating guideline $(MP_b(S), \varPhi)$ of S consists of two steps. First, we construct the matching graph of P and $MP_b(S)$. Second, we check, for all SCCs G in the matching graph, whether the induced SCSG G_P of (the SCSG induced by) G evaluates the annotation of the induced SCGS $G_{MP_b(S)}$ to true.

Definition 31 (matching). A state machine P *matches* with a b-operating guideline $(MP_b(S), \varPhi)$ of S if

1. There exists a minimal simulation relation ϱ of P by $MP_b(S)$, yielding the matching graph $MG(P, MP_b(S))$; and
2. For every SCSG G induced by an SCC of $MG(P, MP_b(S))$ and its induced SCSGs G_P and $G_{MP_b(S)}$, G_P models the formula $\varPhi(G_{MP_b(S)})$, denoted $G_P \models \varPhi(G_{MP_b(S)})$, if $\varPhi(G_{MP_b(S)})$ evaluates to true in the following assignment $\beta(G_P) : \varSigma \rightarrow \{\mathit{true}, \mathit{false}\}$ to the propositions $l \in \varSigma$:
$$\beta(G_P)(l) = \begin{cases} \mathit{true}, & \begin{array}{c} l = (q, x', q') \in \delta_{MP_b(S)} \wedge \exists((q_P, q), (x, x'), (q'_P, q')) \in \delta_{MG} : \\ q_P \in Q_{G_P} \wedge (q_P, x, q'_P) \notin \delta_{G_P} \end{array} \\ \mathit{false}, & \text{otherwise.} \end{cases}$$

As the matching procedure is computationally expensive, it is not intended to decide whether a state machine P is a b-partner of a state machine S. To this end, composing P and S and model checking $S \oplus P$ is more efficient. Definition 31 rather shows how a b-operating guideline represents a b-partner. It serves as a precondition for verifying accordance of two state machines in Sect. 4.

Example 8. Figure 5 shows the matching graph $MG(U, MP_1(B))$, and Figure 3.2 shows the matching graph $MG(V, MP_1(B))$. The matching graph $MG(V, MP_1(B))$ has three induced SCSGs C_1–C_3 (see Fig. 3.2). For C_1, $(v_0, !o, v_1)$ assigns true to $(s_0, !o, s_1)$ of G_0 (defined in Example 7) and evaluates $\varPhi(G_0)$ (see Example 7) to true. A similar argumentation holds for C_2. The projection of C_3 onto V does not have any outgoing transition. As $\varPhi(C_{3\,MP_1(B)})$ (i.e., $\varPhi(G_3)$ in Example 7) is not true, it is evaluated to false and we conclude that V does not match with the 1-operating guideline of B.

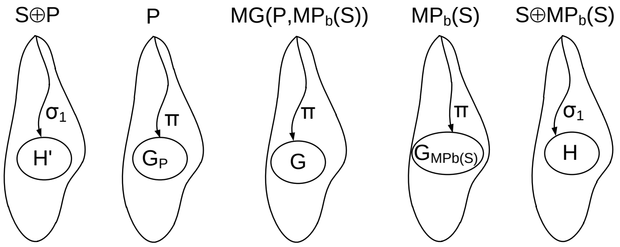


Fig. 6. Ingredients used for the proof of Theorem 32.

3.3 Justification

In this section, we prove that a b-operating guideline of S characterizes all b-partners of S.

Theorem 32. *For any two state machines P and S, P matches with a b-operating guideline $(MP_b(S), \Phi)$ of S iff P is a b-partner of S.*

The proof takes as ingredients the matching graph $MG(P, MP_b(S))$, the state machines P and $MP_b(S)$, and the transition systems $S \oplus P$ and $S \oplus MP_b(S)$. More precisely, the proof uses the relation of an SCC G of $MG(P, MP_b(S))$ and its induced SCSGs G_P and $G_{MP(S)}$ of P and $MP_b(S)$ and their corresponding components in $S \oplus P$ and $S \oplus MP_b(S)$; see Fig. 6 for an illustration.

Before we formalize the necessary properties used in the proof, we give a proof outline with the help of Fig. 6.

For the implication, we choose an arbitrary SCC H' of $S \oplus P$. We have to show that it can be left or contains a final state. To this end, we use the existence of a simulation relation of $S \oplus P$ by $S \oplus MP_b(S)$ (Lemma 33) to determine an SCS H of $S \oplus MP_b(S)$. We project H' onto P and H onto $MP_b(S)$ yielding G_P and $G_{MP_b(S)}$, respectively, from which we construct an SCC G of $MG(P, MP(S))$. From $G_P \models \Phi(G_{MP_b(S)})$ we can show that H' has the desired property.

More difficult is the reverse implication. Here, we start with an SCC G of $MG(P, MP_b(S))$ and its induced SCSGs G_P and $G_{MP_b(S)}$. For a given clause Φ_i of $\Phi(G_{MP_b(S)})$, we construct an SCSG H that defines Φ_i using Lemma 34. From H we construct an SCSG H' of $S \oplus P$, which can be used to derive properties of G_P (see Lemma 35) from which we can conclude that G_P models Φ_i.

Lemma 33. *For any b-partner P of S, $S \oplus MP_b(S)$ simulates $S \oplus P$.*

Proof. To improve readability, we leave out the subscript b throughout this proof.

By assumption, P is a b-partner of S; so the minimal simulation relation $\varrho \subseteq Q_P \times Q_{MP(S)}$ exists. We construct a simulation relation $\varrho' \subseteq Q_{S\oplus P} \times Q_{S\oplus MP(S)}$ from ϱ inductively as follows:

Base: If $(\hat{q}_P, \hat{q}) \in \varrho$, so $((\hat{q}_S, \hat{q}_P, []), (\hat{q}_S, \hat{q}, [])) \in \varrho'$.

Step: Suppose $((q_S, q_P, \mathcal{B}), (q_S, q, \mathcal{B})) \in \varrho'$. If $(q_S, q_P, \mathcal{B}) \xrightarrow{[x,S]}_{\delta_{S\oplus P}} (q_S', q_P, \mathcal{B}')$, then $(q_S, q, \mathcal{B}) \xrightarrow{[x,S]}_{\delta_{S\oplus MP(S)}} (q_S', q, \mathcal{B}')$ and $((q_S', q_P, \mathcal{B}'), (q_S', q, \mathcal{B}')) \in \varrho'$. Likewise, if $(q_S, q_P, \mathcal{B}) \xrightarrow{[x,P]}_{\delta_{S\oplus P}} (q_S, q_P', \mathcal{B}')$, then by the definition of ϱ we conclude that there exists an $x' \in I \cup O \cup T_{MP(S)}$ such that $(q_S, q, \mathcal{B}) \xrightarrow{[x',MP(S)]}_{\delta_{S\oplus MP(S)}} (q_S, q', \mathcal{B}')$ and $(x = x' \vee (x \in T_P \wedge x' \in T_{MP(S)}))$. Thus, we have $((q_S, q_P', \mathcal{B}'), (q_S, q', \mathcal{B}')) \in \varrho'$. □

The next lemma relates $MP_b(S)$ and $S \oplus MP_b(S)$.

Lemma 34. *Let S be a state machine for which $MP_b(S)$ exists and G be an SCSG of $MP_b(S)$ with $\hat{q} \xrightarrow{\pi}_{\delta^*_{MP_b(S)}} q' \in Q_G$.*

For all subgraphs H of $S \oplus MP_b(S)$ that are valid w.r.t. G and for all states $(q_S, q, \mathcal{B}) \in Q_H$, there exist a state $\hat{q}_S \in \hat{Q}_S$ and a trace $\sigma = \sigma_1\sigma_2$ with $(\hat{q}_S, \hat{q}, [\,]) \xrightarrow{\sigma}_{\delta^*_{S\oplus MP_b(S)}} (q_S, q, \mathcal{B})$ and $\sigma_1|_{MP_b(S)} = \pi$ and $q' \xrightarrow{\sigma_2|_{MP_b(S)}}_{\delta^*_G} q$.

Proof. To improve readability, we leave out the subscript b throughout this proof.

Let $(q_S, q, \mathcal{B}) \in Q_H$. We have $\hat{q} \xrightarrow{\pi}_{\delta^*_{MP(S)}} q'$ by assumption, and $q \in Q_G$ follows from H being valid w.r.t. G. Now, with the definition of an SCSG, there exists a trace π' with $q' \xrightarrow{\pi'}_{\delta^*_{MP(S)}} q$. Together with the most preciseness of $MP(S)$ justified by Theorem 32, we conclude, in particular by Proposition 20, that there exists a state $\hat{q}_S \in \hat{Q}_S$ and a trace $\sigma = \sigma_1\sigma_2$ with $(\hat{q}_S, \hat{q}, [\,]) \xrightarrow{\sigma}_{\delta^*_{S\oplus MP(S)}} (q_S, q, \mathcal{B})$ and $\sigma_1|_{MP(S)} = \pi$ and $\sigma_2|_{MP(S)} = \pi'$. So it remains to show that σ_2 is also a trace of H. To see this, consider the state $(q_S, q, \mathcal{B}) \in Q_H$. By construction of $MP(S)$, we have $(q_S, \mathcal{B}) \in k(q)$. If $\pi\pi'$ is the empty sequence, this means that $(q_S, \mathcal{B}) \in cl(\hat{Q}_S \times \{[]\})$ which means that there is the claimed sequence from an initial state to $(q_S, q, \mathcal{B})$. Otherwise, let $\pi\pi' = \pi^* x$ and q^* the state reached by π^* in $MP(S)$, i.e., $q^* \xrightarrow{x}_{\delta_{MP(S)}} q$. By construction of $MP(S)$, $k(q) = cl(\text{set-step}(k(q^*, x))$. By definition of closure and set-step, we conclude that there is some $(q_S^*, \mathcal{B}^*) \in k(q^*)$ such that $(q_S, q, \mathcal{B})$ is reachable in $S \oplus MP(S)$ from $(q_S^*, q^*, \mathcal{B}^*)$ by first executing x (by set-step) and then executing transitions from S (by closure). Iterating the same argument for $(q_S^*, q^*, \mathcal{B}^*)$ until $\pi\pi'$ is reduced to the empty sequence, yields the desired sequence. □

As Figures 4(a) and 4(b) show, the subgraph H is not necessarily connected.

Next, we show that from the subgraph H' we can derive an assignment for G_P. In the proof, we construct from H' a subgraph H of $S \oplus P$ that is valid w.r.t. to G_P.

Lemma 35. *Let $S \oplus P$ be weakly terminating, C be an SCC of $MG(P, MP_b(S))$, and G be the SCSG induced by C. Let G_P and $G_{MP_b(S)}$ be the induced SCGS of G in P and $MP_b(S)$. Let Φ_i be a clause of $\Phi(G_{MP_b(S)})$. Then, $G_P \models \Phi_i$.*

Proof. To improve readability, we leave out the subscript b throughout this proof.

Let H be the subgraph of $S \oplus MP(S)$ that is valid w.r.t. $G_{MP(S)}$ and according to which Φ_i has been added to $\Phi(G_{MP(S)})$. Let π be a path in $MG(P, MP(S))$ that leads from an initial state to some state in G.

We construct from H the following subgraph $H' = (Q, \delta, F)$ of $S \oplus P$:

- $Q = \{(q_S, q_P, \mathcal{B}) \mid \exists q \in Q_{G_{MP(S)}} : (q_S, q, \mathcal{B}) \in Q_H \wedge (q_P, q) \in Q_G\}$,
- $$\begin{aligned} \delta = \quad & \{(q_S, q_P, \mathcal{B}) \xrightarrow{[x,S]}_{\delta} (q'_S, q_P, \mathcal{B}') \mid (q_S, q_P, \mathcal{B}), (q'_S, q_P, \mathcal{B}') \in Q \\ & \quad \wedge \exists q \in Q_{G_{MP(S)}} : (q_P, q) \in Q_G \wedge (q_S, q, \mathcal{B}) \xrightarrow{[x,S]}_{\delta_H} (q'_S, q, \mathcal{B}')\} \\ \cup \; & \{(q_S, q_P, \mathcal{B}) \xrightarrow{[x,P]}_{\delta} (q_S, q'_P, \mathcal{B}') \mid (q_S, q_P, \mathcal{B}), (q'_S, q_P, \mathcal{B}') \in Q \\ & \quad \wedge \exists q, q' \in Q_{G_{MP(S)}} : ((q_P, q), (x, x'), (q'_P, q')) \in \delta_G \\ & \quad \wedge (q_S, q, \mathcal{B}) \xrightarrow{[x', MP(S)]}_{\delta_H} (q_S, q', \mathcal{B}')\}, \end{aligned}$$
- $F = \{(q_S, q_P, [\,]) \in Q \mid \exists q \in Q_{MP(S)} : (q_S, q, [\,]) \in F_H \wedge (q_P, q) \in F_{MG}\}$.

Let $(q_S, q, \mathcal{B}) \in Q_H$. From Lemma 34, we conclude that there exists a trace $\sigma = \sigma_1\sigma_2$ with $(\hat{q}_S, \hat{q}, [\,]) \xrightarrow{\sigma}_{S \oplus MP(S)} (q_S, q, \mathcal{B})$ and $\sigma_1|_{MP(S)} = \pi$ and $q' \xrightarrow{\sigma_2|_{MP(S)}}_{\delta_{G_{MP(S)}}}$. We shall now construct a corresponding trace in $S \oplus P$ that proves reachability of H'. W.l.o.g., assume $\sigma = y_1x_1 \ldots y_nx_ny_{n+1}$ with traces $y_1, \ldots, y_{n+1}$ in S and traces $x_1, \ldots, x_n$ in $MP(S)$. By induction over the length of σ, we have for all $(q'_S, q', \mathcal{B}') \in Q_{S \oplus MP(S)}$ and $(q'_S, q'_P, \mathcal{B}') \in Q_{S \oplus P}$ with $(q'_P, q') \in Q_{MG}$, if $(q'_S, q', \mathcal{B}') \xrightarrow{y_ix_i}_{S \oplus MP(S)} (q''_S, q'', \mathcal{B}'')$ then $(q'_S, q'_P, \mathcal{B}') \xrightarrow{y_ix'_i}_{S \oplus P} (q''_S, q''_P, \mathcal{B}'')$ with $((q'_P, q'), (x'_i, x_i), (q''_P, q'')) \in \delta_{MG}$. For y_i observe that the message bag and the state of S are the same in both states, and for x'_i recall that these steps are determined by δ_{MG}. The relation δ then directly follows from the definition of Q. The set F is defined according to the definition of a final state in $S \oplus MP(S)$. By the definition of Q_{MG}, $(q_P, q) \in Q_{MG}$ and existence of $(q_S, q, [\,]) \in F_H$ imply $q_P \in F_P$ and hence $(q_S, q_P, [\,]) \in F_{H'}$.

The subgraph H' is valid w.r.t G_P: The projection of H' onto the states and transitions of P yields G_P. Furthermore, completeness of the transition relation δ can be concluded from the completeness of the transition relation δ_H and Proposition 18.

We shall use the fact that H' is valid w.r.t. G_P to prove that G_P models Φ_i. As $S \oplus P$ is weakly terminating, H' contains a final state or can be left by an edge of S or P (Proposition 9(3)). If H' contains a final state $(q_S, q_P, [\,])$ and $(q_P, q) \in \varrho$ (ϱ is the minimal simulation relation of P by $MP(S)$), then q is a final state by the construction of ϱ and so is the respective state $(q_S, q, [\,]) \in Q_H$. Therefore, Φ_i is defined as *true*. Now assume that H' does not contain a final state but can be left by an x-labeled edge. Suppose this edge originates from S. Then, it can also be executed in the respective state in Q_H. If its target is not in Q_H, then H defines Φ_i to be true; otherwise, if its target is in Q_H, the corresponding target in H' must be in Q by the construction of H' from H. Suppose this x-labeled edge originates from P. Then it is also a leaving edge in G_P and by the construction of ϱ in $G_{MP(S)}$, and so it is in H. Thus, G_P would assign true to this edge (no matter whether it leaves H or is internal to H) and hence model Φ_i. □

Now, we have all the ingredients to prove Theorem 32.

Proof (of Theorem 32). To improve readability, we leave out the subscript b throughout this proof.

$\Rightarrow$: Let P match with $(MP(S), \Phi)$. Then, $MP(S)$ simulates P. From $TS^0(S)$ simulates $MP(S)$ (Lemma 21) and transitivity of the simulation relation, we conclude that $TS^0(S)$ simulates P. As a consequence, $S \oplus P$ does not violate the bound b. So it suffices to show that $S \oplus P$ weakly terminates, from which we can conclude that P is a b-partner of S.

Let H' be the induced SCSG of some SCC of $S \oplus P$. Let σ_1 be a trace of $S \oplus P$ to a state of H' with $(\hat{q}_S, \hat{q}_P, [\,]) \xrightarrow{\sigma_1}_{\delta^*_{S \oplus P}} (q'_S, q'_P, \mathcal{B}') \in Q_{H'}$. Let $\sigma_1 = y_1 x_1 \dots y_n x_n y_{n+1}$ with traces $y_1, \dots, y_{n+1}$ in S and traces $x_1, \dots, x_n$ in P. By Lemma 33, $S \oplus MP(S)$ simulates $S \oplus P$ using relation $\varrho' \subseteq Q_{S \oplus P} \times Q_{S \oplus MP(S)}$. Thus, there exists an SCSG H in $S \oplus MP(S)$ whose states form an SCS with ϱ' relates the states of H' and H (note that $S \oplus MP(S)$ may have more behavior than $S \oplus P$, so H is not necessarily induced by an SCC). By the construction of ϱ', a trace $\sigma'_1 = y_1 x'_1 \dots y_n x'_n y_{n+1}$, which is derived from σ_1 by replacing every trace x_i of P with the corresponding trace x'_i of $MP(S)$ according to ϱ', can be executed in $S \oplus MP(S)$ leading to a state of H. Let G_P be the projection of H' onto the states and transitions of P and $G_{MP(S)}$ be the projection of H onto the states and transitions of $MP(S)$. By construction, H is valid w.r.t. $G_{MP(S)}$ and H' is valid w.r.t. G_P. Let further $\pi = (x_1, x'_1) \dots (x_n, x'_n)$. By the choice of H', G_P is an SCSG whose states form an SCC and $G_{MP(S)}$ is an SCSG whose states form an SCS. So the matching graph $MG(P, MP(S))$, which exists by assumption, has an SCSG G induced by an SCC such that G is reachable via trace π, and G induces the SCSGs G_P and $G_{MP(S)}$.

By assumption, P matches with $(MP(S), \Phi)$ and, therefore, G_P models $\Phi(G_{MP(S)})$. Let H define the clause Φ_i of $\Phi(G_{MP(S)})$. Then, G_P models Φ_i. We distinguish three cases: (1) The SCSG H contains a final state (and, thus, is not considered for defining $\Phi(G_{MP(S)})$). Then, there exists a final state of H' by the definition of ϱ'. (2) The SCSG H can be left by a transition originating from S (and, thus, is not considered for defining $\Phi(G_{MP(S)})$). Then, the corresponding state in H' enables the same transition, and it leaves H' as otherwise ϱ' would relate a state of H' with a state which is not in H. (3) H can be left by a transition originating from $MP(S)$. By the definition of matching, a corresponding edge leaves G_P and thus H'. In all three cases, Proposition 9 holds and hence $S \oplus P$ weakly terminates.

$\Leftarrow$: Let P be a b-partner of S. This implies the existence of a most-permissive b-partner of S by Lemma 23 from which we conclude the existence of $MP(S)$. By Definition 11, $MP(S)$ simulates P. So let $MG(P, MP(S))$ be the matching graph. It remains to show that Definition 31(2) holds.

To do so, let G be an SCSG induced by an SCC of $MG(P, MP(S))$ and π be a trace of $MG(P, MP(S))$ to a state of G with $(\hat{q}_P, \hat{q}) \xrightarrow{\pi}_{\delta^*_{MG}} (q'_P, q') \in Q_G$. Let G_P and $G_{MP(S)}$ denote the induced SCGSs of G in P and $MP(S)$. Let Φ_i be a clause in $\Phi(G_{MP(S)})$. Then, there exists a subgraph H of $S \oplus MP(S)$ that is valid w.r.t. $G_{MP(S)}$ and defines Φ_i with

- H does not contain a final state; and
- No transition originating from S leaves H.

Lemma 34 shows that we can construct H from G and π. Given H, we conclude with Lemma 35 that G_P models Φ_i. Thus, Definition 31(2) holds and P matches with $(MP(S), \Phi)$. □

4 Deciding Accordance

In this section, we demonstrate the main application for operating guidelines. Given two open systems S and R, we want to decide whether R can safely replace S. Safe replacement of an open system is formalized by the accordance relation. Accordance requires that every partner of S is also a partner of R. An open system may have infinitely many partners, so we must check inclusion of two infinite sets. Because the set of all partners of an open system can be represented in a finite manner using the operating guideline, we may use the operating guidelines of S and R to decide whether R accords with S.

Definition 36 (accordance). Let S and R be state machines with $I_S = I_R$ and $O_S = O_R$. Then, R *b-accords with* S if every b-partner P of S is also a b-partner of R.

It is easy to see that b-accordance is a preorder and by generalizing the composition operator $\oplus$ such that the composition of two state machines is not necessarily a closed transition system, *b*-accordance is even a precongruence for the operator $\oplus$.

To decide b-accordance of S and R, we need to relate their operating guidelines, in particular the assigned formulae. As Φ_S uses transitions of $MP(S)$ as literals, and Φ_R transitions of $MP(R)$, we need to transform these literals into a common domain. The natural candidate is to use transitions of the matching graph as literals.

Definition 37 (formula projection). Let S and R be state machines such that, for their operating guidelines $(MP(S), \Phi_S)$ and $(MP(R), \Phi_R)$, the matching graph $MG(MP(S), MP(R)) = (Q, \hat{Q}, I, O, T, \delta, F)$ exists. Let G be an SCSG of the matching graph and G_S and G_R the respective projections to $MP(S)$ and $MP(R)$. For the formula $\Phi_S(G_S)$, its projection $\Phi_S(G_S)|_{MG}$ replaces every transition literal (q_1, x, q_2) in $\Phi_S(G_S)$ by

$$\bigvee_{q'_1, q'_2, x' : (q_1, q'_1) \in G \wedge ((q_1, q'_1), (x, x'), (q_2, q'_2)) \in \delta} ((q_1, q'_1), (x, x'), (q_2, q'_2)) \quad .$$

Correspondingly, for the formula $\Phi_R(G_R)$, its projection $\Phi_R(G_R)|_{MG}$ replaces every transition literal (q'_1, x', q'_2) in $\Phi_R(G_R)$ by

$$\bigvee_{q_1, q_2, x : (q_1, q'_1) \in G \wedge ((q_1, q'_1), (x, x'), (q_2, q'_2)) \in \delta} ((q_1, q'_1), (x, x'), (q_2, q'_2)) \quad .$$

The next lemma shall be used to prove our main result.

Lemma 38. *Let P be simulated by $MP(S)$.*

1. *P and the matching graph $MG(P, MP(S))$ are bisimilar.*
2. *If P is also simulated by $MP(R)$, then the two matching graphs $MG(P, MP(S))$ and $MG(P, MP(R))$ are bisimilar.*

Proof. (1) The matching graph $MG(P, MP(S))$ exists because of the assumed simulation. The bisimulation follows from the definition of a matching graph.

(2) The two matching graphs $MG(P, MP(S))$ and $MG(P, MP(R))$ exist because of the assumed simulations. By the first item, P and $MG(P, MP(S))$ are bisimilar and P and $MG(P, MP(R))$ are bisimilar. As bisimulation is an equivalence, we conclude that $MG(P, MP(S))$ and $MG(P, MP(R))$ are bisimilar, too. □

The main claim of this section is that accordance of R by S can be decided on their operating guidelines. To this end, we define a refinement relation on operating guidelines that consists of two conditions. First, $MP(R)$ must simulate $MP(S)$. Informally, if R accords with S then, in particular, $MP(S)$ is a partner of R. Second, every partner of S must not livelock with R and thus also satisfies the formulae of the operating guideline of R. To decide this, we investigate all SCSGs G of the matching graph $MG(MP(S), MP(R))$. Recall that matching requires that the projection of an SCC G^* onto P evaluates the formula of the projection of G^* onto MG. This formula must imply the formula $\Phi_R(G_{MP(R)})$, also projected to MG.

Theorem 39 (checking accordance). *For any two state machines S and R with operating guidelines $(MP_b(S), \Phi_S)$ and $(MP_b(R), \Phi_R)$, R b-accords with S iff the following two items hold:*

1. *There exists a minimal simulation relation $\varrho \subseteq Q_{MP_b(S)} \times Q_{MP_b(R)}$ of $MP_b(S)$ by $MP_b(R)$ yielding the matching graph $MG = MG(MP_b(S), MP_b(R))$.*
2. *For every SCSG G of $MG(MP_b(S), MP_b(R))$, the formula $\Phi_S(G_{MP_b(S)})|_{MG} \implies \Phi_R(G_{MP_b(R)})|_{MG}$ is a tautology.*

Proof. To improve readability, we leave out the subscript b throughout this proof.

⇒: Assume R accords with S. We show that items (1) and (2) hold. To this end, we proceed according to the following agenda:

1. Establish a simulation of $MP(S)$ by $MP(R)$ thus proving (1);
2. Assume that (2) is not true, for some SCSG G and some assignment β;
3. Construct a state machine P from $MP(S)$, G, and β;
4. Establish a simulation of P by $MP(S)$;
5. Show that P is a partner of S using Theorem 32 and the fact that β satisfies $\Phi_S(G_{MP(S)})|_{MG}$;
6. Establish a simulation of P by $MP(R)$;
7. Show that P is not a partner of R using Theorem 32 and the fact that β violates $\Phi_R(G_{MP(R)})|_{MG}$.

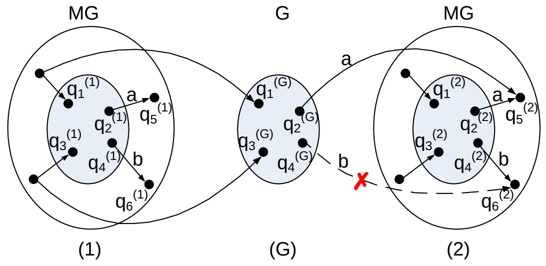


Fig. 7. Illustration for the proof of the implication of Theorem 39. We have $\beta((q_2, a, q_5)) = \mathit{true}$ and $\beta((q_4, b, q_6)) = \mathit{false}$

8. Conclude that R does not accord with S, in contradiction to the assumption, thus proving (2).

Ad 1. The state machine $MP(S)$ is a partner of S and, by assumption, must be a partner of R. So there exists a minimal simulation relation $\varrho \subseteq Q_{MP(S)} \times Q_{MP(R)}$ according to Theorem 32 and $MG = MG(MP(S), MP(R))$, the matching graph, exists.

Ad 2. Assume that there exists an SCSG $G \subseteq Q_{MG}$ where the formula in (2) is not a tautology. This means that there is a particular assignment β where $\Phi_S(G_{MP(S)})|_{MG}$ is true while $\Phi_R(G_{MP(R)})|_{MG}$ is false.

Ad 3. We construct a state machine P that consists of two copies of MG and, additionally, of a copy of G. The first copy of MG is only connected to the copy of G, and the copy of G is only connected to the second copy of MG; see Fig. 7. We refer to a state $q \in Q_{MG}$ as $q^{(1)}$ if it belongs to the first copy of MG and as $q^{(2)}$ if it belongs to the second copy of MG. States of the copy of G are referred to as $q^{(G)}$. Let $Q^{(1)} = \{q^{(1)} \mid q \in Q_{MG}\}$, $Q^{(2)} = \{q^{(2)} \mid q \in Q_{MG}\}$, and $Q^{(G)} = \{(q^{(G)} \mid q \in Q_G\}$; that is, Q_P is supposed to be the disjoint union of $Q^{(1)}$, $Q^{(2)}$, and $Q^{(G)}$.

Initial states of P are exactly the initial states of the first copy of MG. Final states are all those $q^{(1)}$, $q^{(2)}$, and $q^{(G)}$ where q is final in MG.

The three parts of P are connected by additional transitions.

- Add transition $(q^{(1)}, x, q'^{(G)})$ iff (a.) $(q, x, q') \in \delta_{MG}$ and (b.) $q' \in G$.
- Add transition $(q^{(G)}, x, q'^{(2)})$ iff (a.) $(q, x, q') \in \delta_{MG}$, (b.) $q \in G$, and (c.) $\beta((q, x, q')) = \mathit{true}$.

Informally, states from which G as part of the first copy of MG can be entered, get additional transitions to enter the explicit copy of G instead. For each transition that leaves G in MG, the explicit copy of G is left at according position in P, jumping to the second copy of MG, but only with those transitions where β assigns true.

By this construction, every SCSG of P contains only states with the same superscript.

Ad 4. Consider the following relation $\varrho_{P,MP(S)} \subseteq Q_P \times Q_{MP(S)}$: $\varrho_{P,MP(S)} = \{((q_{MP(S)}, q_{MP(R)})^{(x)}), q_{MP(S)}) \mid x \in \{1,2,G\}, (q_{MP(S)}, q_{MP(R)})^{(x)} \in Q^{(x)}$ and $(q_{MP(S)}, q_{MP(R)})^{(x)}$ is reachable in $P\}$.

$\varrho_{P,MP(S)}$ is just a projection of copies of MG to $MP(S)$ and of G to $MP(S)$. In addition, the transitions linking the three parts of P directly correspond to transitions of MG. It is thus easy to see that $\varrho_{P,MP(S)}$ is a simulation relation. Moreover, because every reachable state of P occurs exactly once in $\varrho_{P,MP(S)}$, it is necessarily the minimal simulation relation. Thus the matching graph $MG(P, MP(S))$ exists. Using the mapping ϕ with $\phi(((q_{MP(S)}, q_{MP(R)})^{(x)}, q_{MP(S)})) = (q_{MP(S)}, q_{MP(R)})^{(x)}$, it is easy to verify that the reachable part of P and $MG(P, MP(S))$ are actually isomorphic.

Ad 5. Consider the matching graph $MG(P, MP(S))$ and one of its SCSG H induced by some SCC of this matching graph. As there are no strongly connected components that span over more than one part of P, we can distinguish three cases for H. In the first case, H_P has states in $Q^{(1)}$. We know that MG is bisimilar to $MP(S)$ (Lemma 38(1)) and $MP(S)$ is a partner of S. For this reason, H_P must have sufficiently many leaving transitions to satisfy $\Phi_S(H_{MP(S)})$. The same holds if H_P consists of states in $Q^{(2)}$. In the third case, H_P consists of states in $Q^{(G)}$. Since P and $MG(P, MP(S))$ are isomorphic, the matching graph will cover all states and transitions of G, i.e., $H_P = G$.

By Definition 37, this means that $\Phi_S(G_{MP(S)})$ is true if all those literals (q_S, x, q'_S) are true where there exist q_R, x', and q'_R where $(q_S, q_R) \xrightarrow{(x,x')}_{MG} (q'_S, q'_R)$ and $\beta(((q_S, q_R), (x, x'), (q'_S, q'_R))) = \mathit{true}$. Let l be such a literal (q_S, x, q'_S). As for all edges of MG where β assigns true and which have a source in G, there is a transition in P that leaves (the copy of) G, this means that l is also true in the assignment β^* that is constructed for G in Theorem 32. As $\Phi_S(G_{MP(S)})$ is monotonic (negation free), this suffices to conclude that $\Phi_S(G_{MP(S)})$ is satisfied by β^*. Consequently, P is a partner of S.

Ad 6. Consider the following relation $\varrho_{P,MP(R)} \subseteq Q_P \times Q_{MP(R)}$: $\varrho_{P,MP(R)} = \{((q_{MP(S)}, q_{MP(R)})^{(x)}, q_{MP(R)}) \mid x \in \{1,2,G\}, (q_{MP(S)}, q_{MP(R)})^{(x)} \in Q^{(x)}$ and $(q_{MP(S)}, q_{MP(R)})^{(x)}$ is reachable in $P\}$.

$\varrho_{P,MP(R)}$ is just a projection of copies of MG to $MP(R)$ and of G to $MP(R)$. In addition, the transitions linking the three parts of P directly correspond to transitions of MG. It is thus easy to see that $\varrho_{P,MP(R)}$ is a simulation relation. Moreover, because every reachable state of P occurs exactly once in $\varrho_{P,MP(R)}$, it is necessarily the minimal simulation relation. Thus the matching graph $MG(P, MP(R))$ exists. Using the mapping ϕ with $\phi(((q_{MP(S)}, q_{MP(R)})^{(x)}, q_{MP(R)})) = (q_{MP(S)}, q_{MP(R)})^{(x)}$, it is easy to verify that the reachable part of P and $MG(P, MP(R))$ are actually isomorphic.

Ad 7. We show that P is not a partner of R. To this end, we apply Theorem 32 to the copy of G as part of P and its isomorphic counterpart G^* in $MG(P, MP(R))$. The projection of G^* to P is obviously G while the projection to $MP(R)$ is $G_{MP(R)}$. That is, we evaluate $\Phi_R(G_{MP(R)})$ in the assignment β^* constructed from G. We show that a literal $(q_{MP(R)}, x', q'_{MP(R)})$ can be true only if the disjunction that replaces $(q_{MP(R)}, x', q'_{MP(R)})$ in Definition 37 is evaluated to true by β. Because $\Phi_R(G_{MP(R)})$ is monotonic (negation free) and $\Phi_R(G_{MP(R)})|_{MG}$ is not satisfied by β, we can then conclude that $\Phi_R(G_{MP(R)})$ is not satisfied by β^*. By Theorem 32, this shows that P is not a partner of R.

Hence, consider a literal $(q_{MP(R)}, x', q'_{MP(R)})$ that is true in β^*. By Definition 31, this means that there are $q_P, q'_P \in Q_P$ and x such that $((q_P, q_{MP(R)}), (x, x'), (q'_P, q'_{MP(R)})) \in \delta_{MG(P,MP(R))}$, $q_P \in Q_G$, and $(q_P, x, q'_P) \notin \delta_P$. By the construction of P and the specific simulation relation $\varrho_{P,MP(R)}$, this means that there exist $q_{MP(S)}, q'_{MP(S)} \in Q_{MP(S)}$ and x such that $(((q_{MP(S)}, q_{MP(R)}), q_{MP(R)}), (x, x'), ((q'_{MP(S)}, q'_{MP(R)}), q'_{MP(R)})) \in \delta_{MG(P,MP(R))}$, $(q_{MP(S)}, q_{MP(R)}) \in Q_G$, and $((q_{MP(S)}, q_{MP(R)}), (x, x'), (q'_{MP(S)}, q'_{MP(R)})) \notin \delta_G$. By construction, $((q_{MP(S)}, q_{MP(R)}), (x, x'), (q'_{MP(S)}, q'_{MP(R)})) \notin \delta_G$ implies that $\beta(((q_{MP(S)}, q_{MP(R)}), (x, x'), (q'_{MP(S)}, q'_{MP(R)}))) = \mathit{true}$. Thus, the disjunction that replaces $(q_{MP(R)}, x', q'_{MP(R)})$ in Definition 37 is true in β which remained to show.

Ad 8. We have by (5.) that P is a partner of S but, by (7.) P is not a partner of R. This contradicts the general assumption that R accords with S. Consequently, the assumption made in (2.) must by false and hence the claim (2) of the theorem must hold.

$\Leftarrow$: Assume items (1) and (2) hold. Let P be a partner of S. We have to show that P is also a partner of R (i.e., P matches with the operating guideline of R).

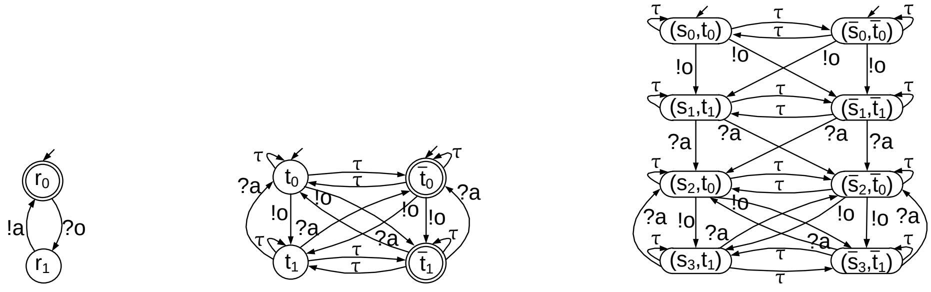
First, we show that the matching graph $MG(P, MP(R))$ exists. By assumption, the matching graphs $MG(P, MP(S))$ and $MG(MP(S), MP(R))$ exist. As a simulation relation is transitive, we conclude that $MP(R)$ simulates P. As $MP(R)$ is by construction deterministic, this simulation relation is even minimal. Thus, $MG(P, MP(R))$ exists, proving the first item of Definition 31.

It remains to prove that the second item of Definition 31 is satisfied. Let G'' be an SCSG induced by an SCC of $MG(P, MP(R))$, and let G''_P and $G''_{MP(R)}$ be the induced SCSGs of G''. We have to show that $G''_P \models \Phi_R(G''_{MP(R)})$.

From the bisimulation between the matching graphs $MG(P, MP(S))$ and $MG(P, MP(R))$ (see Lemma 38(2)), we conclude the existence of an SCSG G' (induced by an SCC) of $MG(P, MP(S))$ which is bisimilar to G''. As G' and G'' are bisimilar, for their projections we have $G'_S = G''_S$ and $G'_{MP(S)}$ and $G''_{MP(R)}$ are bisimilar. Moreover, from P being a partner of S, we conclude by Theorem 32 and Definition 31 that $G'_P \models \Phi_S(G'_{MP(S)})$ in the assignment β_P.

We now show that β_P can be mapped on the domain of edges of R, yielding an assignment in which $G''_P \models \Phi_R(G''_{MP(R)})$. As $G'_{MP(S)}$ and $G''_{MP(R)}$ are bisimilar, there exists an SCSG G of MG with $G_{MP(S)} = G'_{MP(S)}$ and $G_{MP(R)} = G''_{MP(R)}$.

Consider a literal $(q_{MP(S)}, x_S, q'_{MP(S)})$ that is true in β_P. This means that there are $q_P, q'_P \in Q_P$ and x such that $((q_P, q_{MP(S)}), (x, x_S), (q'_P, q'_{MP(S)})) \in \delta_{MG(P,MP(R))}$, $q_P \in Q_G$, and $(q_P, x, q'_P) \notin \delta_P$ by Definition 31. Replacing the formula $\Phi_S(G'_{MP(S)})$ by its projection $\Phi_S(G'_{MP(S)})|_{MG}$ according to Definition 37, replaces every $(q_{MP(S)}, x_S, q'_{MP(S)})$ by a disjunction of literals. Thus,



(a) R (b) $MP_1(R)$) (c) $MG(MP_1(B), MP_1(R))$

Fig. 8. State machine R 1-accords with state machine B.

$\beta_P(q_{MP(S)}, x_S, q'_{MP(S)}) = true$ implies this disjunction is evaluated to true in β_P.

Now, using the second item of Theorem 39, we know that $\Phi_R(G_{MP(R)})|_{MG}$ evaluates to true in β_P. We use again Definition 37 to establish a relation between the formulae $\Phi_R(G_{MP(R)})|_{MG}$ and $\Phi_R(G_{MP(R)}) = \Phi_R(G''_{MP(R)})$. As Definition 37 replaces a disjunction of literals by a single literal and $\Phi_R(G_R)|_{MG}$ evaluates to true in β_P, this literal is only evaluated to true in β_P if the disjunction is. Thus β_P is the assignment in which $G''_P \models \Phi_R(G''_{MP(R)})$. As a consequence, with Definition 31 we have P is also a partner of R. □

Example 9. Figure 8(a) depicts a new buyer R. It provides the possibility to also terminate without interaction. All other functions of B remain unchanged. To prove that R 1-accords with B, we apply Theorem 39. Figure 8 shows $MP_1(R)$ and the matching graph $MG(MP_1(B), MP_1(R))$. Thus the first item of Theorem 39 holds. Two check the second item, we consider the SCSGs of the matching graph. Consider the SCSG $G = (\{(\bar{s}_0, \bar{t}_0)\}, \{\}, \{\})$ and the formulae $\Phi_B(G_{MP(B)})$ (i.e., $\Phi(G_1)$ in Example 7) and $\Phi_B(G_{MP(R)})$. The latter formula is true because all respective subgraphs of $R \oplus MP_1(R)$ contain a final state. Thus, the implication of these formulae is a tautology. The same argumentation holds for any SCSG with the state set $\{(s_0, t_0), (\bar{s}_0, \bar{t}_0)\}$.

5 Related Work

5.1 Operating guidelines

The presented operating guideline for weak termination builds on the notion of an overapproximation for state machines and the most permissive partner for weak termination, as defined by Wolf [53]. Moreover, it generalizes operating guidelines for deadlock freedom as introduced by Lohmann et al. [29]. Whereas in the case of deadlock freedom, it sufficed to annotate every state of the underlying most permissive partner by a Boolean formula, every strongly connected subgraph has to be annotated in the case of weak termination.

The construction of the most permissive partner is related to supervisory control [43]. Given a system specification as a finite automaton, supervisory control asks whether there *exists* an environment such that the composition of the system and the environment satisfies some temporal logic property. Weak termination, also known as nonconflicting or nonblocking, is an important property in the field of discrete event systems [43,17]. Temporal synthesis is an instance of supervisory control where a partner is synthesized directly from a temporal specification; see the overview of Kupferman [24]. It ensures that the synthesized system realizes the property with either the maximal environment (i.e., an environment that offers all inputs in every state) or all environments. There also exist approaches to restrict the environments under consideration by adding assumptions [19] and to bound the size of the synthesized system [46]. However, controller synthesis is just a transient step in our approach to construct an operating guideline that characterizes all partners—a concept that is not yet known in the fields of temporal synthesis and supervisory control. Moreover, to the best of our knowledge, the notion of a most precise partner has not been considered in controller synthesis.

Operating guidelines are related to work of de Alfaro and Henzinger on interface automata [6,7]. Interface automata implicitly characterize the set of required input actions and allowed output actions at every state. Our main contribution is the characterization of all partners in a style that is much more explicit than in interface automata. The explicit notion enables simple procedures for exploring operating guidelines. Moreover, for us, the Boolean annotations play a completely different role than the predicates, for example, in [48]. Instead of data dependencies, our annotations express choices in the control flow.

Operating guidelines complement work on robust model checking by Kupferman and Vardi [25]. An open system S robustly satisfies a temporal logic property ψ if for every environment E such that $S \oplus E$ is deadlock free holds, $S \oplus E$ satisfies ψ. Although we consider only weak termination rather than arbitrary temporal logic properties but in an asynchronous rather than a synchronous setting as in [25], operating guidelines represent all partners, also in case S does not robustly satisfy ψ.

5.2 Accordance

Classification Accordance has been introduced as conflicting preorder in [31] and subcontract in [13] for synchronous communication. As noticed by Malik et al. [31] and Bravetti and Zavattaro [13], should testing [16,37,44]—also known as fair testing—is the coarsest traditional precongruence that implies accordance, but accordance does not imply should testing. Bravetti and Zavattaro [13] give a counterexample, whereas Malik et al. [31] also show that accordance implies a new variant of failures preorder.

Mooij et al. [36] show when accordance and should testing coincide by restricting the tests, on the one hand and by restricting the considered sets of system, on the other hand. They identify three differences between should testing and accordance: First, in the case of accordance, the composed system must terminate together (using a synchronized termination action), whereas in the case of should testing, only the test needs to perform an (unsynchronized) "success" action. In other word, the notion of a partner is symmetric whereas a test is an asymmetric notion. Second, whereas in the case of accordance there is no information about the states of the open system after any occurrence of the termination action, in the case of should testing, there is information about each state of the open system. Third, a state machine can have uncoverable states—that is, states that cannot be visited with any partner. In the case of accordance, there is no information about uncoverable states of the process (because no partner visits them), whereas in the case of should testing, there is information about each state of the process (i.e., those states are visited by tests). This, in particular, causes that accordance does not imply the trace preorder.

The notion of uncoverable states has been introduced as the set of certain conflicts by Malik et al. [31]. A similar phenomenon occurs in the safe-must preorder [12], where a process and its observer must reach a success state before reaching a catastrophic (i.e., diverging) one.

Work on Automata Ware and Malik [50] give a state-based characterization of accordance in the setting of synchronous communication—the characterizations in [31,13,36] are defined in a testing like setting. Based on this characterization, they provide a decision procedure. The idea is to compare for a state of S and a state of R the traces leading to a final state. This decision procedure is similar to the one for should testing in [44], where for a pair of states of S and R the tree failures are compared. In contrast, we decide accordance on the operating guidelines of S and R rather than on S and R. Moreover, we consider a setting of asynchronous communication rather than synchronous communication as Ware and Malik [50]. Recently, Ware and Malik [51] introduce accordance-preserving abstraction techniques to speed up the decision of accordance.

The presented solution to decide accordance generalizes previous work by Stahl et al. [47] on deadlock freedom to weak termination. Likewise, it generalizes work on acyclic open systems by Aalst et al. [5] to cyclic open systems.

We decide accordance in another style than Alfaro and Henzinger for interface automata [7], and in another setting (e.g., asynchronous unqueued communication). Likewise, also the work on protocol automata of Beyer et al [10] considers synchronous communication but arbitrary temporal logic properties.

Benatallah et al. [9] restrict their model to deterministic automata that communicate synchronously. For such models accordance coincides with the simulation preorder.

Pathak et al. [41] focus on a substitutability notion that preserves an arbitrary property, which is expressed by a μ-calculus formula. The approach assumes a synchronous communication model and is based on partial model checking [8]. Oster and Basu [40] extend the work of [41] to asynchronous communication by introducing a buffer process that can buffer one message for each channel.

Work on Petri Nets Vogler's invariant reachability (IR) equivalence [49] is used for the replaceability of open Petri nets. As the main difference to accordance, it is an asymmetric notion that focuses only on the tests (i.e., the partners) and not on the open system being tested. Vogler proves in [49] that IR-equivalence coincides with should testing.

Van der Aalst et al. [5] present a decision procedure for accordance in case the open systems, which are modeled as open Petri nets, are acyclic. In this paper, we do not put any restriction on the structure of an open system.

A compositional refinement check for accordance is presented by Van der Aalst et al. in [3]. The approach is restricted to composition that have a tree structure and the only a sufficient criterion is to decide accordance is given.

Martens [32] considers accordance, but his decision procedure is only sufficient. In contrast, we presented a decision procedure on operating guidelines which is sufficient and necessary.

Van der Aalst and Basten [2] consider synchronously communicating Petri nets and present projection inheritance as a refinement relation. Projection inheritance is based on branching bisimulation [21] and is finer than (i.e., implies) accordance; see [4]. Likewise weak bisimulation used as a refinement relation by Bonchi et at. [11] is finer than accordance.

Work on Process Calculi Bravetti and Zavattaro study the subcontract preorder for different communication paradigms. They consider synchronous handshake in [13] and asynchronous communication via unbounded message queues in [14]. In addition, a stronger notion is introduced ensuring, whenever a message can be sent, the other service is ready to receive this message. Systems that behave this way are strongly compliant [15]. Bravetti and Zavattaro show that except for unbounded message queues, the notion of should testing implies the corresponding subcontract preorder. In the setting of of unbounded message queues decidability has not been proved.

Fournet et al. [20] consider CCS processes of asynchronous message passing software components. They present stuck-free conformance which, in contrast to accordance, only excludes deadlocks. As shown in [20], the CSP stable-failures preorder [45] (which, for finitely branching processes without divergences, is equivalent to must testing; see [38]) does not imply stuck-free conformance, and stuck-free conformance is strictly larger than the refusal preorder of Phillips [42]. Thus, accordance does not imply stuck-free conformance.

The sub-contract preorder from [26,18] is an asymmetric notation that only focuses on the test (i.e., the partner) rather than the composition of open system and test. Moreover, it only requires that the test never gets stuck. Laneve and Padovani prove in [26] that this sub-contract relation coincides with must testing [39]. Must testing is known to be incomparable to should testing.

5.3 Applications of Operating Guidelines

In the case of deadlock freedom, operating guidelines proved their usefulness in a multitude of applications. The algorithm to decide accordance and thus to decide whether one open system can replace another one has been presented in this paper and in the case of deadlock freedom in [47]. Another application is to decide whether a running instance of an open system can be migrated to a state of another open system [27]. To this end, states of the current instance have to be related to states of the new instance. Accordance checks the inclusion of two sets of partners. Kaschner and Wolf [23] show that union, intersection, and complement can be implemented for sets of partners, yielding a way to calculate with sets of open systems.

Matching checks containment of an open system in the set of open systems represented by an operating guideline. It can be used for service discovery [29]. In the case, an open system is misbehaving, the operating guideline can be used to correct this open system [28]. The characterization of all partners of an open system may also yield useful positive test cases while the complement of that characterization might include negative test cases [22]. Furthermore, operating guidelines may serve as a starting point for computing a public view of an open system (i.e., an abstract view) [35].

6 Conclusion

We have studied open systems that communicate via asynchronous message passing, and presented a data structure to characterize the possibly infinite set of partners of an open system S in a finite manner. Thereby a partner of S is an open system such that the composition is a closed system that has the possibility to always reach a final state, a property that is of highest relevance in practice. As an application for our data structure, we showed that we can decide accordance of two open systems on their partner characterization. Although the computation of this data structure is exponential (both in time and space) in the number of states of the open system, a first prototype in our tool Wendy [30] shows that, despite the worst case complexity, the approach is tractable [52].

In future work we aim to implement the procedure for deciding accordance. Furthermore, we want to study which properties other than weak termination yield a partner characterization as the one presented. In addition, we are interested in implementing the operation complement, union and intersection of sets of partners for the given representation as for deadlock freedom [23].

References

1. Aalst, W.M.P.v.d.: The application of Petri nets to workflow management. The Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)
2. Aalst, W.M.P.v.d., Basten, T.: Inheritance of workflows: an approach to tackling problems related to change. Theoretical Computer Science 270(1-2), 125–203 (2002)
3. Aalst, W.M.P.v.d., Hee, K.M.v., Massuthe, P., Sidorova, N., Werf, J.M.E.M.v.d.: Compositional service trees. In: Petri Nets 2009. LNCS, vol. 5606, pp. 283–302. Springer (2009)

4. Aalst, W.M.P.v.d., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: From public views to private views - correctness-by-design for services. In: WS-FM 2007. LNCS, vol. 4937, pp. 139–153. Springer (2008)
5. Aalst, W.M.P.v.d., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: Multiparty contracts: Agreeing and implementing interorganizational processes. Comput. J. 53(1), 90–106 (2010)
6. de Alfaro, L., Henzinger, T.A.: Interface automata. In: ESEC / SIGSOFT FSE. pp. 109–120 (2001)
7. de Alfaro, L., Henzinger, T.A.: Interface theories for component-based design. In: EMSOFT 2001. LNCS, vol. 2211, pp. 148–165. Springer (2001)
8. Andersen, H.R., Lind-Nielsen, J.: Partial model checking of modal equations: A survey. STTT 2(3), 242–259 (1999)
9. Benatallah, B., Casati, F., Toumani, F.: Representing, analysing and managing web service protocols. Data Knowl. Eng. 58(3), 327–357 (2006)
10. Beyer, D., Chakrabarti, A., Henzinger, T.A.: Web service interfaces. In: WWW 2005. pp. 148–159. ACM (2005)
11. Bonchi, F., Brogi, A., Corfini, S., Gadducci, F.: On the use of behavioural equivalences for web services' development. Fundam. Inform. 89(4), 479–510 (2008)
12. Boreale, M., De Nicola, R., Pugliese, R.: Basic observables for processes. Inf. Comput. 149(1), 77–98 (1999)
13. Bravetti, M., Zavattaro, G.: A foundational theory of contracts for multi-party service composition. Fundam. Inform. 89(4), 451–478 (2008)
14. Bravetti, M., Zavattaro, G.: Contract compliance and choreography conformance in the presence of message queues. In: WS-FM 2008. LNCS, vol. 5387, pp. 37–54. Springer (2009)
15. Bravetti, M., Zavattaro, G.: A theory of contracts for strong service compliance. Mathematical Structures in Computer Science 19(3), 601–638 (2009)
16. Brinksma, E., Rensink, A., Vogler, W.: Fair testing. In: CONCUR 1995. LNCS, vol. 962, pp. 313–327. Springer (1995)
17. Cassandras, C.G., Lafortune, S.: Introduction to Discrete Event Systems. Springer, 2 edn. (2007)
18. Castagna, G., Gesbert, N., Padovani, L.: A theory of contracts for web services. ACM Trans. Program. Lang. Syst. 31(5) (2009)
19. Chatterjee, K., Henzinger, T.A., Jobstmann, B.: Environment assumptions for synthesis. In: CONCUR 2008. LNCS, vol. 5201, pp. 147–161. Springer (2008)
20. Fournet, C., Hoare, C.A.R., Rajamani, S.K., Rehof, J.: Stuck-Free Conformance. In: CAV 2004. LNCS, vol. 3114, pp. 242–254. Springer (2004)
21. Glabbeek, R.v., Weijland, W.: Branching Time and Abstraction in Bisimulation Semantics. Journal of the ACM 43(3), 555–600 (1996)
22. Kaschner, K.: Conformance testing for asynchronously communicating services. In: ICSOC 2011. LNCS, vol. 7084, pp. 108–124. Springer (2011)
23. Kaschner, K., Wolf, K.: Set algebra for service behavior: Applications and constructions. In: BPM 2009. LNCS, vol. 5701, pp. 193–210. Springer (2009)
24. Kupferman, O.: Recent challenges and ideas in temporal synthesis. In: SOFSEM 2012. LNCS, vol. 7147, pp. 88–98. Springer (2012)
25. Kupferman, O., Vardi, M.Y.: Interactive Computation - The New Paradigm, chap. Verification of Open Systems, pp. 97–118. Springer (2006)
26. Laneve, C., Padovani, L.: The must preorder revisited. In: CONCUR 2007. LNCS, vol. 4703, pp. 212–225. Springer (2007)
27. Liske, N., Lohmann, N., Stahl, C., Wolf, K.: Another approach to service instance migration. In: ICSOC 2009. LNCS, vol. 5900, pp. 607–621. Springer (2009)

28. Lohmann, N.: Correcting deadlocking service choreographies using a simulation-based graph edit distance. In: BPM 2008. LNCS, vol. 5240, pp. 132–147. Springer (2008)
29. Lohmann, N., Massuthe, P., Wolf, K.: Operating guidelines for finite-state services. In: ICATPN 2007. LNCS, vol. 4546, pp. 321–341. Springer (2007)
30. Lohmann, N., Weinberg, D.: Wendy: A tool to synthesize partners for services. Fundam. Inform. 113(3-4), 295–311 (2011)
31. Malik, R., Streader, D., Reeves, S.: Conflicts and fair testing. Int. J. Found. Comput. Sci. 17(4), 797–814 (2006)
32. Martens, A.: Analyzing web service based business processes. In: FASE 2005. LNCS, vol. 3442, pp. 19–33. Springer (2005)
33. Massuthe, P., Serebrenik, A., Sidorova, N., Wolf, K.: Can I find a partner? Undecidability of partner existence for open nets. Inf. Process. Lett. 108(6), 374–378 (2008)
34. Milner, R.: Communication and Concurrency. Prentice-Hall, Inc. (1989)
35. Mooij, A.J., Parnjai, J., Stahl, C., Voorhoeve, M.: Constructing replaceable services using operating guidelines and maximal controllers. In: WS-FM 2010. LNCS, vol. 6551, pp. 116–130. Springer (2011)
36. Mooij, A.J., Stahl, C., Voorhoeve, M.: Relating fair testing and accordance for service replaceability. J. Log. Algebr. Program. 79(3-5), 233–244 (2010)
37. Natarajan, V., Cleaveland, R.: Divergence and fair testing. In: ICALP 1995. LNCS, vol. 944, pp. 648–659. Springer (1995)
38. Nicola, R.D.: Extensional equivalences for transition systems. Acta Inf. 24(2), 211–237 (1987)
39. Nicola, R.D., Hennessy, M.: Testing equivalences for processes. Theor. Comput. Sci. 34, 83–133 (1984)
40. Oster, Z.J., Basu, S.: Extending substitutability in composite services by allowing asynchronous communication with message buffers. In: ICTAI 2009. pp. 572–575. IEEE Computer Society (2009)
41. Pathak, J., Basu, S., Honavar, V.: On Context-Specific Substitutability of Web Services. In: ICWS 2007. pp. 192–199. IEEE Computer Society (2007)
42. Phillips, I.: Refusal testing. Theor. Comput. Sci. 50, 241–284 (1987)
43. Ramadge, P.J., Wonham, W.M.: Supervisory control of a class of discrete event processes. SIAM J. Control and Optimization 25(1), 206–230 (1987)
44. Rensink, A., Vogler, W.: Fair testing. Inf. Comput. 205(2), 125–198 (2007)
45. Roscoe, A.W.: The Theory and Practice of Concurrency. Prentice Hall Series in Computer Science, Prentice Hall (1998)
46. Schewe, S., Finkbeiner, B.: Bounded synthesis. In: ATVA 2007. LNCS, vol. 4762, pp. 474–488. Springer (2007)
47. Stahl, C., Massuthe, P., Bretschneider, J.: Deciding substitutability of services with operating guidelines. In: ToPNoC II. pp. 172–191. LNCS 5460, Springer (2009)
48. Tripakis, S., Lickly, B., Henzinger, T.A., Lee, E.A.: A theory of synchronous relational interfaces. ACM Trans. Program. Lang. Syst. 33(4), 14 (2011)
49. Vogler, W.: Modular Construction and Partial Order Semantics of Petri Nets, LNCS, vol. 625. Springer (1992)
50. Ware, S., Malik, R.: A state-based characterisation of the conflict preorder. In: FOCLASA 2011. EPTCS, vol. 58, pp. 34–48 (2011)
51. Ware, S., Malik, R.: Conflict-preserving abstraction of discrete event systems using annotated automata. Discrete Event Dynamic Systems 22(4), 451–477 (2012)
52. Weinberg, D.: Deciding Service Substitution – Termination Guaranteed. Phd thesis, Universität Rostock, Germany (2012)

53. Wolf, K.: Does my service have partners? In: ToPNoC II. pp. 152–171. LNCS 5460, Springer (2009)