Nondeterministic Modal Interfaces*

Ferenc Bujtor[1], Sascha Fendrich[2], Gerald Lüttgen[2], and Walter Vogler[1]

[1] Institut für Informatik, University of Augsburg, Germany
{walter.vogler,ferenc.bujtor}@informatik.uni-augsburg.de
[2] Software Technologies Research Group, University of Bamberg, Germany
{gerald.luettgen,sascha.fendrich}@swt-bamberg.de

Abstract. Interface theories are employed in the component-based design of concurrent systems. They often emerge as combinations of Interface Automata (IA) and Modal Transition Systems (MTS), e.g., Nyman et al.'s IOMTS, Bauer et al.'s MIO, Raclet et al.'s MI or our MIA. In this paper, we generalise MI to *nondeterministic* interfaces, for which we resolve the longstanding conflict between unspecified inputs being allowed in IA but forbidden in MTS. With this solution we achieve, in contrast to related work, an *associative* parallel composition, a *compositional* preorder, a conjunction on interfaces with *dissimilar alphabets* supporting perspective-based specifications, and a quotienting operator for decomposing *nondeterministic* specifications in a single theory.

1 Introduction

Interface theories [2,7,8,15,16,18] support the component-based design of concurrent systems and offer a semantic framework for, e.g., software contracts [1] and web services [4]. Several such theories are based on de Alfaro and Henzinger's *Interface Automata* (IA) [10], whose distinguishing feature is a parallel composition on labelled transition systems with inputs and outputs, where receiving an unexpected input is regarded as an error, i.e., a communication mismatch. All states are pruned from which entering an error state cannot be prevented by the environment, rather than leaving the parallel composition fully undefined as in [2].

Various researchers have combined IA with Larsen's *Modal Transition Systems* (MTS) [14], which features may- and must-transitions to express allowed and required behaviour, resp. In a refinement of an interface, all required behaviour must be preserved and no disallowed behaviour may be added. Whereas in IA outputs are optional, they may now be enforced in theories combining IA and MTS, such as Nyman et al.'s IOMTS [15], Bauer et al.'s MIO [2], Raclet et al.'s *Modal Interfaces* (MI) [18] and our *Modal Interface Automata* (MIA) [16,17]. In this paper we extend MI to nondeterministic systems, yielding the most general approach to date and permitting new applications, e.g., for dealing with races in networks. We built upon our prior work in [17], from which we adopt disjunctive must-transitions that are needed for operationally defining conjunction, which is another key operator in interface theories and supports perspective-based specification.

* Research support was provided by the DFG (German Research Foundation) under grants LU 1748/3-1 and VO 615/12-1.

Combining IA and MTS is, however, problematic since unspecified inputs are forbidden in MTS, but allowed in IA with arbitrary behaviour afterwards. In IOMTS [15], the MTS-view was adopted and, as a consequence, compositionality of refinement wrt. the parallel operator $\|$ was lost. In [17] we followed the IA-view but found that resolving the conflict is essential for a more flexible conjunction. In our new MIA, we can optionally express the IA-view for state p and input i by an i-may-transition from p to a *special, universal state e* that can be refined in any way; we will need this option when defining $\|$. There is a similar idea in MI [18], but an ordinary state is used there with the consequence that $\|$ is not associative. In contrast to the somewhat related demonic completion as used, e.g., in [11], we do not enforce input-enabledness. With the new feature, our interface theory allows for a proper distinction between may- and must-transitions for inputs, unlike [16,17]. This enables us to define conjunction also on interfaces with dissimilar alphabets via alphabet extension.

As in MI, our MIA is equipped with a multicast parallel composition, where one output can synchronise with several inputs. We also develop a quotienting operator as a kind of inverse of parallel composition $\|$. For a specification P and a given component D, quotienting constructs the most general component Q such that $Q \| D$ refines P. Quotienting is a very practical operator because it can be used for decomposing concurrent specifications stepwise, specifying contracts [3], and reusing components. In contrast to [18], our quotienting permits *nondeterministic* specifications and complements $\|$ rather than a simpler parallel product without pruning.

In summary, our new interface theory MIA generalises and improves upon existing theories combining IA and MTS: parallel composition is commutative and associative (cf. Section 3), quotienting also works for nondeterministic specifications (cf. Section 4), conjunction properly reflects perspective-based specification (cf. Sections 5 and 6), and refinement (cf. Section 2) is compositional and permits alphabet extension (cf. Section 6). A technical report of this paper [5] contains all proofs, more explanations and examples; it also introduces a disjunction and an action scoping operator.

2 Modal Interface Automata: The Setting

In this section we define MIA and its supported operations. Essentially, MIAs are state machines with disjoint input and output alphabets and two transition relations, *may* and *must*, as in MTS [14]. May-transitions describe permitted behaviour, while must-transitions describe required behaviour. Unlike previous versions of MIA [16,17] and other similar theories, we introduce the *universal state e* as an extra constituent.

Definition 1 (Modal Interface Automata). *A* Modal Interface Automaton *(MIA) is a tuple* $(P, I, O, \longrightarrow, \dashrightarrow, p_0, e)$, *where*

- *P is the set of states containing the* initial state p_0 *and the* universal state e,
- *I and O are disjoint sets, the alphabets of* input *and* output actions, *not containing the special internal action τ, and $A =_{df} I \cup O$ is called the* alphabet,
- $\longrightarrow \subseteq P \times (A \cup \{\tau\}) \times (\mathscr{P}_{fin}(P) \setminus \emptyset)$ *is the* disjunctive must-transition *relation, with $\mathscr{P}_{fin}(P)$ being the set of finite subsets of P,*
- $\dashrightarrow \subseteq P \times (A \cup \{\tau\}) \times P$ *is the* may-transition *relation.*

We require (a) for all $\alpha \in A \cup \{\tau\}$ that $p \xrightarrow{\alpha} P'$ implies $\forall p' \in P'.\, p \overset{\alpha}{\dashrightarrow} p'$ (syntactic consistency) *and that (b) e appears in transitions only as the target of input may-transitions.*

Cond. (a) states that whatever is required should be allowed; this syntactic consistency is a natural and standard condition (cf. [14]). Cond. (b) matches the idea for e explained in the introduction. We use this state in the context of parallel composition to represent communication errors. Note that our disjunctive must-transitions have a single label, in contrast to Disjunctive MTS [13].

In the sequel, we identify a MIA $(P, I, O, \longrightarrow, \dashrightarrow, p_0, e)$ with its state set P and, if needed, use index P when referring to one of its components, e.g., we write I_P for I. Similarly, we write, e.g., I_1 instead of I_{P_1} for MIA P_1. In addition, we let i, o, a, ω and α stand for representatives of the alphabets I, O, A, $O \cup \{\tau\}$ and $A \cup \{\tau\}$, resp.; we write $A = I/O$ when highlighting inputs I and outputs O in an alphabet A, and we define $\hat{a} =_{\text{df}} a$ and $\hat{\tau} =_{\text{df}} \varepsilon$ (the empty word). Furthermore, outputs and internal actions are called *local* actions since they are controlled locally by P. For convenience, we let $p \xrightarrow{a} p'$, $p \overset{a}{\not\longrightarrow}$ and $p \overset{a}{\not\dashrightarrow}$ denote $p \xrightarrow{a} \{p'\}$, $\nexists p'.\, p \xrightarrow{a} p'$ and $\nexists p'.\, p \overset{a}{\dashrightarrow} p'$, resp. In figures, we often refer to an action a as $a?$ if $a \in I$, and as $a!$ if $a \in O$. Must-transitions (may-transitions) are drawn using solid, possibly splitting arrows (dashed arrows); any depicted must-transition also implicitly represents the underlying may-transition(s).

We now define *weak* must- and may-transition relations that abstract from transitions labelled by τ. The following definition is equivalent to the one in [17].

Definition 2 (Weak Transition Relations). *We define* weak *must-transition and* weak *may-transition relations, $\Longrightarrow$ and $=\!\!\Rightarrow$ resp., as the smallest relations satisfying the conditions $P' \overset{\varepsilon}{\Longrightarrow} P'$ for finite $P' \subseteq P$, $p \overset{\varepsilon}{=\!\!\Rightarrow} p$ as well as:*

(a) $P' \overset{\hat{\alpha}}{\Longrightarrow} P''$, $p'' \in P''$ and $p'' \xrightarrow{\tau} P'''$ implies $P' \overset{\hat{\alpha}}{\Longrightarrow} (P'' \setminus \{p''\}) \cup P'''$,

(b) $P' \overset{\varepsilon}{\Longrightarrow} P'' = \{p_1, \ldots, p_n\}$ and $\forall j.\, p_j \xrightarrow{a} P_j$ implies $P' \overset{a}{\Longrightarrow} \bigcup_{j=1}^{n} P_j$,

(c) $p \overset{\varepsilon}{=\!\!\Rightarrow} p'' \overset{\tau}{\dashrightarrow} p'$ implies $p \overset{\varepsilon}{=\!\!\Rightarrow} p'$,

(d) $p \overset{\varepsilon}{=\!\!\Rightarrow} p'' \overset{\alpha}{\dashrightarrow} p''' \overset{\varepsilon}{=\!\!\Rightarrow} p'$ implies $p \overset{\alpha}{=\!\!\Rightarrow} p'$.

For $\{p'\} \overset{\hat{\alpha}}{\Longrightarrow} P''$ we often write $p' \overset{\hat{\alpha}}{\Longrightarrow} P''$. Mostly for inputs a, we also use relation compositions $\xrightarrow{a}\overset{\varepsilon}{\Longrightarrow}$ and $\overset{a}{\dashrightarrow}\overset{\varepsilon}{=\!\!\Rightarrow}$ resp., i.e., where leading τs are disallowed. Observe that $p \xrightarrow{a}\overset{\varepsilon}{\Longrightarrow} P'$ implies $p \overset{a}{\Longrightarrow} P'$, and $p \overset{a}{\dashrightarrow}\overset{\varepsilon}{=\!\!\Rightarrow} p'$ implies $p \overset{a}{=\!\!\Rightarrow} p'$.

Now we define our refinement relation. It is a weak alternating simulation conceptually similar to the observational modal refinement found, e.g., in [12]. A notable aspect, originating from IA [10], is that inputs must be matched immediately, i.e., only trailing τs are allowed. Intuitively, this is due to parallel composition requiring that a signal sent from one system must be received immediately; otherwise, it is considered an error (a communication mismatch). Since one wishes not to introduce new errors during refinement, a refined system must immediately provide all specified inputs.

We treat the universal state e as completely underspecified, i.e., we decree that any state refines it. This is only possible since e is not an ordinary state. We define our refinement preorder for MIAs with common input and output alphabets; we relax this in Section 6.

Definition 3 (MIA Refinement). *Let P, Q be MIAs with common input/output alphabets. A relation $\mathscr{R} \subseteq P \times Q$ is a* MIA-refinement relation *if for all $(p,q) \in \mathscr{R}$ with $q \neq e_Q$:*

(i) $p \neq e_P$,
(ii) $q \xrightarrow{i} Q'$ *implies* $\exists P'.\, p \xrightarrow{i}\overset{\varepsilon}{\Longrightarrow} P'$ *and* $\forall p' \in P' \, \exists q' \in Q'.\, (p',q') \in \mathscr{R}$,
(iii) $q \xrightarrow{\omega} Q'$ *implies* $\exists P'.\, p \overset{\hat{\omega}}{\Longrightarrow} P'$ *and* $\forall p' \in P' \, \exists q' \in Q'.\, (p',q') \in \mathscr{R}$,
(iv) $p \overset{i}{\dashrightarrow} p'$ *implies* $\exists q'.\, q \overset{i}{\dashrightarrow}\overset{\varepsilon}{\Longrightarrow} q'$ *and* $(p',q') \in \mathscr{R}$,
(v) $p \overset{\omega}{\dashrightarrow} p'$ *implies* $\exists q'.\, q \overset{\hat{\omega}}{\Longrightarrow} q'$ *and* $(p',q') \in \mathscr{R}$.

We write $p \sqsubseteq q$ and say that p MIA-refines *q if there exists a MIA-refinement relation $\mathscr{R}$ such that $(p,q) \in \mathscr{R}$, and we let $p \sqsupseteq\sqsubseteq q$ stand for $p \sqsubseteq q$ and $q \sqsubseteq p$. Furthermore, we extend these notations to MIAs, write $P \sqsubseteq Q$ if $p_0 \sqsubseteq q_0$, and use $\sqsupseteq\sqsubseteq$ analogously.*

MIA refinement $\sqsubseteq$ is a preorder and the largest MIA-refinement relation. The preorder property is quite subtle to prove due to the weak transition relations.

3 Parallel Composition

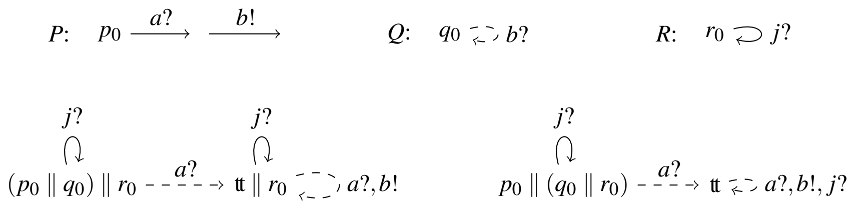
IA [9,10] is equipped with an interleaving parallel operator, where an action occurring as an input in one interface is synchronised with the same action occurring as an output in some other interface; the synchronised action is hidden, i.e., labelled by τ. Since our work builds upon MI [18] we instead consider here a parallel composition, where the synchronisation of an interface's output action involves all concurrently running interfaces that have the action as input. We define a parallel operator $\|$ on MIA in two stages. First, a standard product $\otimes$ between two MIAs is introduced.

Definition 4 (Parallel Product). *MIAs P_1, P_2 are* composable *if $O_1 \cap O_2 = \emptyset$. For such MIAs we define the* product *$P_1 \otimes P_2 = ((P_1 \times P_2) \dot{\cup} \{e_{12}\}, I, O, \longrightarrow, \dashrightarrow, (p_{01}, p_{02}), e_{12})$, where $I =_{df} (I_1 \cup I_2) \setminus (O_1 \cup O_2)$ and $O =_{df} O_1 \cup O_2$ and where $\longrightarrow$ and $\dashrightarrow$ are the smallest relations satisfying the following conditions:*

(PMust1)	$(p_1,p_2) \xrightarrow{\alpha} P_1' \times \{p_2\}$	*if*	$p_1 \xrightarrow{\alpha} P_1'$ *and* $\alpha \notin A_2$
(PMust2)	$(p_1,p_2) \xrightarrow{\alpha} \{p_1\} \times P_2'$	*if*	$p_2 \xrightarrow{\alpha} P_2'$ *and* $\alpha \notin A_1$
(PMust3)	$(p_1,p_2) \xrightarrow{a} P_1' \times P_2'$	*if*	$p_1 \xrightarrow{a} P_1'$ *and* $p_2 \xrightarrow{a} P_2'$ *for some a*
(PMay1)	$(p_1,p_2) \overset{\alpha}{\dashrightarrow} (p_1',p_2)$	*if*	$p_1 \overset{\alpha}{\dashrightarrow} p_1'$ *and* $\alpha \notin A_2$
(PMay2)	$(p_1,p_2) \overset{\alpha}{\dashrightarrow} (p_1,p_2')$	*if*	$p_2 \overset{\alpha}{\dashrightarrow} p_2'$ *and* $\alpha \notin A_1$
(PMay3)	$(p_1,p_2) \overset{a}{\dashrightarrow} (p_1',p_2')$	*if*	$p_1 \overset{a}{\dashrightarrow} p_1'$ *and* $p_2 \overset{a}{\dashrightarrow} p_2'$ *for some a.*

From the parallel product, parallel composition is obtained by pruning, i.e., one removes errors and states leading up to errors via local actions, so called *illegal* states. This cuts all input transitions leading to an illegal state.

In [6] we have shown that de Alfaro and Henzinger have defined pruning in an inappropriate way in [9]. We remedied this by cutting not only an i-transition from some state p to an illegal state, but also all other i-transitions from p. Now, in [6,9], p can be refined by a state with an i-transition and arbitrary behaviour afterward; we express this by introducing an i-may-transition to the universal state.



P: $p_0 \xrightarrow{a?} \xrightarrow{b!}$
Q: q_0 $b?$
R: r_0 $j?$
$j?$
$(p_0 \parallel q_0) \parallel r_0 \overset{a?}{\dashrightarrow} \mathtt{tt} \parallel r_0$ $a?, b!$
$p_0 \parallel (q_0 \parallel r_0) \overset{a?}{\dashrightarrow} \mathtt{tt}$ $a?, b!, j?$

Fig. 1. Differences of our state e to $\mathtt{tt}$ in [18], where $A_P = \{a\}/\{b\}$, $A_Q = \{b\}/\emptyset$ and $A_R = \{j\}/\emptyset$

Definition 5 (Parallel Composition). *Given a parallel product $P_1 \otimes P_2$, a state (p_1, p_2) is a* new error *if there is some $a \in A_1 \cap A_2$ such that (a) $a \in O_1$, $p_1 \overset{a}{\dashrightarrow}$ and $p_2 \overset{a}{\not\longrightarrow}$, or (b) $a \in O_2$, $p_2 \overset{a}{\dashrightarrow}$ and $p_1 \overset{a}{\not\longrightarrow}$. It is an* inherited error *if one of its components is a universal state, i.e., if it is of the form (e_1, p_2) or (p_1, e_2).*

We define the set $E \subseteq P_1 \times P_2$ of illegal *states as the least set such that $(p_1, p_2) \in E$ if (i) (p_1, p_2) is a new or inherited error or (ii) $(p_1, p_2) \overset{\omega}{\dashrightarrow} (p_1', p_2')$ and $(p_1', p_2') \in E$.*

Should the initial state be an illegal state, i.e., $(p_{01}, p_{02}) \in E$, then e_{12} becomes the initial – and thus the only reachable – state of the parallel composition $P_1 \parallel P_2$.

Otherwise, $P_1 \parallel P_2$ is obtained from $P_1 \otimes P_2$ by pruning *illegal states as follows. If there is a state $(p_1, p_2) \notin E$ with $(p_1, p_2) \overset{i}{\dashrightarrow} (p_1', p_2') \in E$ for some $i \in I$, then all must- and may-transitions labelled i and starting at (p_1, p_2) are removed, and a single transition $(p_1, p_2) \overset{i}{\dashrightarrow} e_{12}$ is added. Furthermore, all states in E, all unreachable states (except for e_{12}), and all their incoming and outgoing transitions are removed. If $(p_1, p_2) \in P_1 \parallel P_2$, we write $p_1 \parallel p_2$ and call p_1 and p_2* compatible.

In [18], Raclet et al. use a similar approach to pruning: they introduce a state we denote as $\mathtt{tt}$, which has only input may-transitions as incoming transitions. Furthermore, it has a may-loop for every action of the parallel composition so that it can be refined by any state, much like our universal state (cf. Def. 3(i)). To see the difference, condsider the MIAs P, Q, R in Figure 1, where we construct $(P \parallel Q) \parallel R$ according to [18]. Since $\mathtt{tt}$ is an ordinary state, it is combined with r_0 inheriting the j-must-loop. In our approach, the combination with r_0 is an inherited error, and e does not have any must-transitions.

More importantly, there is the severe problem that parallel composition in [18] is not associative. Consider again the systems P, Q and R in Fig. 1; their parallel compositions shown are not equivalent according to $\sqsupseteq\sqsubseteq$ (and the equivalence in [18]). Note that our example does not rely on the multicast aspect of our parallel composition; it works just as well for IA parallel composition.

Theorem 6 (Associativity of Parallel Composition). *Parallel composition is associative in the sense that, for MIAs P, Q and R, if $(P \parallel Q) \parallel R$ is defined, then $P \parallel (Q \parallel R)$ is defined as well and they are isomorphic, and vice versa.*

Theorem 7 (Compositionality of Parallel Composition). *Let P_1, P_2 and Q_1 be MIAs and $P_1 \sqsubseteq Q_1$. Assume that Q_1 and P_2 are composable, then (a) P_1 and P_2 are composable, and (b) $P_1 \parallel P_2 \sqsubseteq Q_1 \parallel P_2$, and $P_1 \parallel P_2$ is compatible if $Q_1 \parallel P_2$ is.*

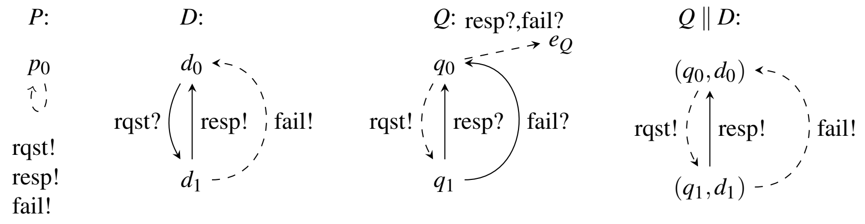


Fig. 2. $Q = P /\!/ D$ with $q_0 = p_0 /\!/ d_0$ and $q_1 = p_0 /\!/ d_1$, where the alphabets are $A_P = \emptyset/\{\text{rqst, resp, fail}\}$, $A_D = \{\text{rqst}\}/\{\text{resp, fail}\}$, $A_Q = \{\text{resp, fail}\}/\{\text{rqst}\}$ and $A_{Q\parallel D} = \emptyset/\{\text{rqst, resp, fail}\}$

4 Quotienting

The quotient operation is a kind of inverse or adjoined operation to parallel composition. It equips the theory with a means for component reuse and incremental, component-based specification. To describe the participants in a quotient operation we use the letters P for the specification, D for the divisor (the already implemented component) and Q for the quotient or its refinements. Given MIAs P and D, the quotient is the coarsest MIA Q such that $Q \parallel D \sqsubseteq P$ holds; we call this inequality the *defining inequality of the quotient*. We write $P /\!/ D$ for the quotient if it exists.

We demonstrate quotienting with the simple client-server application of Figure 2. The server takes the role of the already given component D. It can receive a request and answers with a response. Additionally, the server may implement a failure as answer. When composed in parallel, client Q and server D are supposed to form a *closed* system, i.e., all shared actions are outputs. Thus, the parallel composition of client and server must refine the overall specification P. A specification for the client is then obtained as the quotient $Q = P /\!/ D$. Figure 2 gives a preview of this Q according to our construction below. Client Q may implement the sending of a request, and if so, it must be receptive for a response and a failure. If one of the latter two transitions were of may-modality, this would cause a communication mismatch in the parallel composition with D. The may-transitions resp? and fail! from q_0 to e_Q only exist to make Q as coarse as possible; they disappear in the parallel composition with D. Now, it is easy to check that the defining inequality $Q \parallel D \sqsubseteq P$ is satisfied. The example also shows that, in general, we do not have equality of $(P /\!/ D) \parallel D$ and P.

We define the quotient for a restricted set of MIAs, namely where the specification P has no τs and where the divisor D is may-deterministic without τs. We call D *may-deterministic* if $d \overset{\alpha}{\dashrightarrow} d'$ and $d \overset{\alpha}{\dashrightarrow} d''$ implies $d' = d''$. Due to syntactic consistency, a may-deterministic MIA has no disjunctive must-transitions, i.e., the target sets of must-transitions are singletons. In addition, we exclude the pathological case where P has some state p and input i with $p \overset{i}{\dashrightarrow} e_P$ and $\exists p' \neq e_P.\, p \overset{i}{\dashrightarrow} p'$. Recall that transitions $p \overset{i}{\dashrightarrow} e_P$ are meant to express the following situation: (a) input i is not specified at p, but at the same time (b) p shall be refinable as in IA [10] by a state with an i-transition and arbitrary subsequent behaviour. Despite these restrictions, our quotient significantly generalises that of MI [18], which considered deterministic specifications and divisors only. In the following, we call MIAs P, D satisfying our restrictions a *quotient pair*.

4.1 Definition and Main Result

Like most other operators we define the quotient in two stages, where $\mathrm{may}_P(p,\alpha)$ stands for $\{p' \in P \mid p \overset{\alpha}{\dashrightarrow}_P p'\}$.

Definition 8 (Pseudo-quotient). *Let* $(P, I_P, O_P, \longrightarrow_P, \dashrightarrow_P, p_0, e_P)$, $(D, I_D, O_D, \longrightarrow_D, \dashrightarrow_D, d_0, e_D)$ *be a quotient pair with* $A_D \subseteq A_P$ *and* $O_D \subseteq O_P$, *and* $I =_{df} I_P \cup O_D$ *and* $O =_{df} O_P \setminus O_D$. *The* pseudo-quotient of P over D *is defined as the MIA* $(\{(e_P, e_D)\}, I, O, \emptyset, \emptyset, (e_P, e_D), (e_P, e_D))$ *if* $p_0 = e_P$. *Otherwise,* $P \oslash D =_{df} (P \times D, I, O, \longrightarrow, \dashrightarrow, (p_0, d_0), (e_P, e_D))$, *where the transition relations are defined by:*

$$\begin{array}{llll}
\text{(QMust1)} & (p,d) \xrightarrow{a} P' \times \{d\} & \textit{if} & p \xrightarrow{a}_P P' \textit{ and } a \notin A_D \\
\text{(QMust2)} & (p,d) \xrightarrow{a} P' \times \{d'\} & \textit{if} & p \xrightarrow{a}_P P' \textit{ and } d \xrightarrow{a}_D d' \\
\text{(QMust3)} & (p,d) \xrightarrow{a} P' \times \{d'\} & \textit{if} & P' =_{df} \mathrm{may}_P(p,a) \neq \emptyset,\ e_P \notin P', \\
 & & & d \overset{a}{\dashrightarrow}_D d' \textit{ and } a \in O_D \\
\text{(QMay1)} & (p,d) \overset{a}{\dashrightarrow} (p',d) & \textit{if} & p \overset{a}{\dashrightarrow}_P p' \neq e_P \textit{ and } a \notin A_D \\
\text{(QMay2)} & (p,d) \overset{a}{\dashrightarrow} (p',d') & \textit{if} & p \overset{a}{\dashrightarrow}_P p' \neq e_P \textit{ and } d \xrightarrow{a}_D d' \\
\text{(QMay3)} & (p,d) \overset{a}{\dashrightarrow} (p',d') & \textit{if} & p \overset{a}{\dashrightarrow}_P p',\ e_P \notin \mathrm{may}_P(p,a), \\
 & & & d \overset{a}{\dashrightarrow}_D d' \textit{ and } a \notin O_P \cap I_D \\
\text{(QMay4)} & (p,d) \overset{a}{\dashrightarrow} (e_P,e_D) & \textit{if} & e_P \in \mathrm{may}_P(p,a) \quad \textit{(note: } a \in I_P \subseteq I) \\
\text{(QMay5)} & (p,d) \overset{a}{\dashrightarrow} (e_P,e_D) & \textit{if} & p \neq e_P,\ d \overset{a}{\not\dashrightarrow}_D \textit{ and } a \in A_D \setminus (O_P \cap I_D)
\end{array}$$

Regarding the definition of the alphabets we follow [8] and [18]; there is, however, a choice regarding the input alphabet, which we discuss in Sec. 6. The intuition behind a state (p,d) in $P \oslash D$ is that (p,d) composed in parallel with d refines state p, and that (p,d) should be coarsest wrt. MIA refinement satisfying this condition. With this in mind, we now justify some of the above rules intuitively.

Rule (QMust1) is necessary due to the following consideration. If P has an a-must-transition where a is unknown to D, this can only originate from an a-must-transition in the quotient Q that we wish to construct; in order to be most permissive, each $p' \in P'$ must have a match in $Q \parallel D$. The corresponding consideration is true for Rule (QMay1), which also establishes syntactic consistency for Rule (QMust1).

Rule (QMust3) ensures that (p,d) and d are compatible in case of an output of d. An application of this rule can be seen in Fig. 2 for action fail? at $q_1 = p_0 // d_1$. Syntactic consistency results from Rules (QMay2) and (QMay3); note that $a \in O_D$ implies $a \notin I_D$.

Rule (QMay5) makes $P \oslash D$ as coarse as possible. The input a-may-transitions introduced here just disappear in $(P \oslash D) \parallel D$, since a is blocked by D. This can be seen in Fig. 2 for actions resp? and fail? at $q_0 = p_0 // d_0$ and in $Q \parallel D$ at (q_0, d_0).

$P \oslash D$ is indeed a MIA. We have already argued for syntactic consistency. All rules ensure $p \neq e_P$; hence, $e_{P \oslash D}$ has no outgoing transitions. Incoming transitions of $e_{P \oslash D}$ can only arise from Rules (QMay4) or (QMay5), which are only applicable for $a \in I$.

Up to now, we have only defined the pseudo-quotient. Considering a candidate pair (p,d), for some combinations of modalities and assignments of actions to input or output, it is impossible that p is refined by a state resulting from a parallel composition with d. We call such states *impossible states* and remove them from the pseudo-quotient states. For example, for $p \xrightarrow{a}$ and $d \overset{a}{\dashrightarrow}$ such that $d \not\xrightarrow{a}$, no parallel composition with d refines p. While may-transitions can be refined by removing them and disjunctive transitions can be refined to subsets of their targets to prevent the reachability of impossible states, all states having a must-transition to only impossible states must also be removed.

Definition 9 (Quotient). *Let $P \oslash D$ be the pseudo-quotient of P over D. The set $G \subseteq P \times D$ of* impossible states *is defined as the least set refining the following rules:*

(G1)	$p \xrightarrow{a}_P$ *and* $d \not\xrightarrow{a}_D$ *and* $a \in A_D$	*implies*	$(p,d) \in G$
(G2)	$p \neq e_P$ *and* $p \overset{a}{\not\dashrightarrow}_P$ *and* $d \overset{a}{\dashrightarrow}_D$ *and* $a \in O_D$	*implies*	$(p,d) \in G$
(G3)	$p \neq e_P$ *and* $d = e_D$	*implies*	$(p,d) \in G$
(G4)	$(p,d) \xrightarrow{a}_{P \oslash D} R'$ *and* $R' \subseteq G$	*implies*	$(p,d) \in G$

The quotient *$P//D$ is obtained from $P \oslash D$ by deleting all states $(p,q) \in G$. This also removes any may- or must-transition exiting and any may-transition entering a deleted state. Deleted states are also removed from targets of disjunctive must-transitions. If $(p,d) \in P//D$, we write $p//d$. If $(p_0,d_0) \notin P//D$, the quotient P over D is not defined.*

Rule (G1) is obvious since (p,d) cannot ensure that $p \xrightarrow{a}_P$ is matched if d has no a-must-transition, as an a-may-transition or even a forbidden action at d can in no case compose to a refinement of a must-transition at p. Rule (G2) captures the situation where d has an output a that is forbidden at p. Offering an a-must-input in the quotient would lead to a transition in the parallel composition with d, while not offering it would lead to an error; both would not refine p. Rule (G3) captures the division by e_D: state e_D in parallel with any state is universal and does not refine $p \neq e_P$. Finally, Rule (G4) propagates back all impossibilities that cannot be avoided by refining.

Note that $P//D$ is a MIA. Quotienting yields the coarsest MIA satisfying the defining inequality; proving this statement involves showing that the definedness of $\|$ and $//$ is mutually preserved across refinement. Operator $//$ is also monotonous at the left.

Theorem 10 ($//$ is a Quotient Operator wrt. $\|$). *Let P, D be a quotient pair and Q be a MIA such that $A_D \subseteq A_P$, $O_D \subseteq O_P$, $O_Q = O_P \setminus O_D$ and $I_Q = I_P \cup O_D$. Then, $Q \sqsubseteq P//D$ iff $Q \| D \sqsubseteq P$.*

Theorem 11 (Monotonicity of $//$ wrt. $\sqsubseteq$). *Let P_1, P_2, D be MIAs with $P_1 \sqsubseteq P_2$. If $P_1//D$ is defined and P_2, D are a quotient pair, then $P_2//D$ is defined and $P_1//D \sqsubseteq P_2//D$.*

4.2 Discussion

For $Q \| D \sqsubseteq P$ to hold, $Q \| D$ and P must have the same input alphabet and the same output alphabet. Thus, we must have $O_Q = O_P \setminus O_D$ and $I_Q \supseteq I_P \setminus I_D$. Concerning the input actions in D, quotient Q can listen to them but does not have to. Hence,

$I_Q \subseteq I_P \setminus I_D \cup A_D = I_P \cup O_D$. The more inputs Q has, the easier it is to supply the behaviour ensuring $Q \parallel D \sqsubseteq P$. Thus, we have chosen the input alphabet $I_P \cup O_D$ for our quotient $P//D$, just as is done in [8] and [18]. When comparing some Q to $P//D$ in Thm. 10, Q necessarily has the same input and output alphabets as $P//D$, by Def. 3.

Quotient operators for interface theories have already been discussed by Raclet et al. [18] and Chilton et al. [7]. Our quotient $Q = P//D$ is most similar to [18], where D is assumed to be may-deterministic, P and D have no internal transitions, and $I_Q = I_P \cup O_D$. However, also P must be may-deterministic there, whereas we additionally allow nondeterminism and disjunctive must-transitions in P.

In addition, we have corrected some technical shortcomings of MI [18]. Its quotient operation ignores compatibility so that quotienting is an adjoint to the parallel product but *not* to parallel composition. This has been recognised in a technical report [3], which unfortunately employs a changed setting without a universal state.

5 Conjunction

Besides parallel composition and quotienting, conjunction is one of the most important operators of interface theories. It allows one to specify different perspectives of a system separately, from which an overall specification can be determined. More formally, the conjunction should be the coarsest specification that refines the given perspective specifications, i.e., it should characterise the greatest lower bound of the refinement preorder. In the sequel, we define conjunction on MIAs with common alphabets, as we did for MIA refinement. Similar to parallel composition, we first present a conjunctive product and, in a second step, remove state pairs with contradictory specifications.

Definition 12 (Conjunctive Product). *Consider two MIAs $(P, I, O, \longrightarrow_P, \dashrightarrow_P, p_0, e_P)$ and $(Q, I, O, \longrightarrow_Q, \dashrightarrow_Q, q_0, e_Q)$ with common alphabets. The* conjunctive product *is defined as $P\&Q =_{df} (P \times Q, I, O, \longrightarrow, \dashrightarrow, (p_0, q_0), (e_P, e_Q))$, satisfying the following rules plus the symmetric rules of (OMust1), (IMust1), (EMust1), (May1), (EMay1):*

$$\begin{array}{lll}
\text{(OMust1)} & (p,q) \xrightarrow{\omega} \{(p',q') \mid p' \in P',\, q \overset{\hat{\omega}}{\Longrightarrow}_Q q'\} & \textit{if } p \xrightarrow{\omega}_P P' \textit{ and } q \overset{\hat{\omega}}{\Longrightarrow}_Q \\
\text{(IMust1)} & (p,q) \xrightarrow{i} \{(p',q') \mid p' \in P',\, q \overset{i}{\dashrightarrow}\overset{\varepsilon}{\Longrightarrow}_Q q'\} & \textit{if } p \xrightarrow{i}_P P' \textit{ and } q \overset{i}{\dashrightarrow}\overset{\varepsilon}{\Longrightarrow}_Q \\
\text{(EMust1)} & (p,e_Q) \xrightarrow{\alpha} P' \times \{e_Q\} \ \textit{if } p \xrightarrow{\alpha}_P P' & \\
\text{(May1)} & (p,q) \overset{\tau}{\dashrightarrow} (p',q) & \textit{if } p \overset{\tau}{\Longrightarrow}_P p' \\
\text{(OMay)} & (p,q) \overset{\omega}{\dashrightarrow} (p',q') & \textit{if } p \overset{\omega}{\Longrightarrow}_P p' \textit{ and } q \overset{\omega}{\Longrightarrow}_Q q' \\
\text{(IMay)} & (p,q) \overset{i}{\dashrightarrow} (p',q') & \textit{if } p \overset{i}{\dashrightarrow}\overset{\varepsilon}{\Longrightarrow}_P p' \textit{ and } q \overset{i}{\dashrightarrow}\overset{\varepsilon}{\Longrightarrow}_Q q' \\
\text{(EMay1)} & (p,e_Q) \overset{\alpha}{\dashrightarrow} (p',e_Q) & \textit{if } p \overset{\alpha}{\dashrightarrow}_P p'
\end{array}$$

Note that this definition is similar to the one in [17], except for the treatment of inputs and the universal state. The conjunctive product is inherently different from the parallel product. Single transitions are defined through weak transitions, e.g., as in Rules (OMust1), (IMust1) and (May1), and τ-transitions synchronise by Rule (OMay). Furthermore, as given by Rules (EMust1) and (EMay1), a universal state is a neutral element for the conjunctive product, whereas it is absorbing for the parallel product.

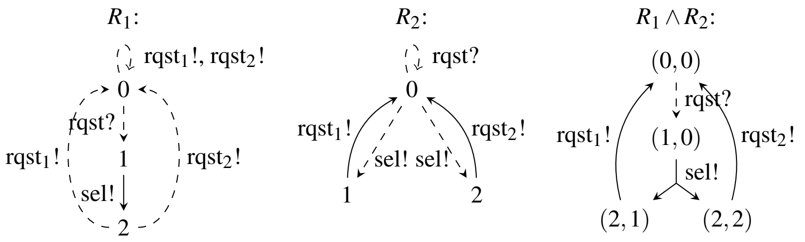


Fig. 3. Conjunction on MIAs may lead to disjunctive transitions

Definition 13 (Conjunction). *Given a conjunctive product $P\&Q$, the set $F \subseteq P \times Q$ of* inconsistent states *is defined as the least set satisfying for all $p \neq e_P$ and $q \neq e_Q$:*

(F1) $(p \xrightarrow{o}_P \text{ and } q \overset{o}{\not\Longrightarrow}_Q) \text{ or } (p \overset{o}{\not\Longrightarrow}_P \text{ and } q \xrightarrow{o}_Q)$ *implies* $(p,q) \in F$

(F2) $(p \xrightarrow{i}_P \text{ and } q \overset{i}{\not\dashrightarrow}_Q) \text{ or } (p \overset{i}{\not\dashrightarrow}_P \text{ and } q \xrightarrow{i}_Q)$ *implies* $(p,q) \in F$

(F3) $(p,q) \xrightarrow{\alpha} R' \text{ and } R' \subseteq F$ *implies* $(p,q) \in F$

The conjunction $P \wedge Q$ is obtained by deleting all states $(p,q) \in F$ from $P\&Q$. This also removes any may- or must-transition exiting and any may-transition entering a deleted state; in addition, deleted states are removed from targets of disjunctive must-transitions. We write $p \wedge q$ for (p,q) of $P \wedge Q$; all such states are defined and consistent by construction. If $(p_0,q_0) \in F$, then the conjunction of P and Q does not exist.

An example of conjunction is given in Fig. 3. MIAs R_1 and R_2 can be understood as requirements for a server front-end that routes between a client and at least one of two back-ends. MIA R_1 specifies that, after getting a client's request (rqst?), a back-end selection (sel!) must be performed, after which the request can be forwarded to one of the two back-ends (rqst$_1$!, rqst$_2$!). MIA R_2 specifies that, with the selection, it is decided to which one of the back-ends the request will be forwarded (rqst$_1$!, rqst$_2$!).

In $R_1 \wedge R_2$, the selection process (sel!) is given by a *disjunctive* must-transition. Such a requirement cannot be specified in a deterministic theory, such as MI [18] which our theory extends. Although one might approximate the disjunctive sel! by individual selection actions sel$_1$! and sel$_2$! for each back-end, the conjunction would either have both actions as may-transitions and thus allow one to omit both, or would have both actions as must-transitions, disallowing a server application with only one back-end.

Theorem 14 ($\wedge$ is And). *Let P and Q be MIAs with common alphabets. Then, (i) $(\exists R.\ R \sqsubseteq P$ and $R \sqsubseteq Q)$ iff $P \wedge Q$ defined. Further, in case $P \wedge Q$ is defined and for any R: (ii) $R \sqsubseteq P$ and $R \sqsubseteq Q$ iff $R \sqsubseteq P \wedge Q$.*

Clearly, conjunction is commutative. Further, as a consequence of the above theorem, (i) it is also associative and (ii) MIA refinement is compositional wrt. conjunction.

6 Alphabet Extension

So far, MIA refinement is only defined on MIAs with the same alphabets. This is insufficient for supporting perspective-based specification, where an overall specification is conjunctively composed of smaller specifications, each addressing one 'perspective' (e.g., a single system requirement) and referring only to actions that are relevant to that perspective. Hence, it is useful to extend conjunction and thus MIA refinement to dissimilar alphabets in such a way that we can add new inputs and outputs in a refinement step. For this purpose we introduce alphabet extension as an operation on MIAs, similar to [17] and also to *weak extension* in [18]. More precisely, we add may-loops for all new actions to each state, except the universal state.

Definition 15 (Alphabet Extension & Refinement). *Given a MIA* $(P, I, O, \longrightarrow, \dashrightarrow, p_0, e)$ *and disjoint action sets* I' *and* O' *satisfying* $I' \cap A = \emptyset = O' \cap A$*, where* $A =_{df} I \cup O$*, the* alphabet extension of P by I' and O' *is given by* $[P]_{I',O'} =_{df} (P, I \cup I', O \cup O', \longrightarrow, \dashrightarrow', p_0, e)$ *for* $\dashrightarrow' =_{df} \dashrightarrow \cup \{(p, a, p) \mid p \in P \setminus \{e\}, a \in I' \cup O'\}$*. We often write* $[p]_{I',O'}$ *for* p *as state of* $[P]_{I',O'}$*, or conveniently* $[p]$ *in case* I'*,* O' *are understood from the context.*

For MIAs P *and* Q *with* $p \in P$*,* $q \in Q$*,* $I_P \supseteq I_Q$ *and* $O_P \supseteq O_Q$*, we define* $p \sqsubseteq' q$ *if* $p \sqsubseteq [q]_{I_P \setminus I_Q, O_P \setminus O_Q}$*. Since* $\sqsubseteq'$ *extends* $\sqsubseteq$ *to MIAs with different alphabets, we write* $\sqsubseteq$ *for* $\sqsubseteq'$ *and abbreviate* $[q]_{I_P \setminus I_Q, O_P \setminus O_Q}$ *by* $[q]_P$*; the same notations are used for* P *and* Q*.*

Compositionality of parallel composition as in Thm. 7 is preserved by the extended refinement relation as long as alphabet extension does not yield new communications.

Theorem 16 (Compositionality of Parallel Composition). *Let* P_1*,* P_2*,* Q *be MIAs such that* Q *and* P_2 *are composable and* $P_1 \sqsubseteq Q$*. Assume further that, for* $I' =_{df} I_1 \setminus I_Q$ *and* $O' =_{df} O_1 \setminus O_Q$*, we have* $(I' \cup O') \cap A_2 = \emptyset$*. Then: (a)* P_1 *and* P_2 *are composable, and (b) if* Q *and* P_2 *are compatible, then so are* P_1 *and* P_2 *and* $P_1 \parallel P_2 \sqsubseteq Q \parallel P_2$*.*

Our conjunction operator may be lifted to conjuncts with dissimilar alphabets by defining $P \wedge' Q =_{\mathrm{df}} [P]_Q \wedge [Q]_P$; the lifted operator $\wedge'$ satisfies the analogue of Thm. 14.

7 Conclusions and Future Work

We presented an extension of Raclet et al.'s modal interface theory [18] to *nondeterministic* systems. To do so we resolved, for the first time properly, the conflict between unspecified inputs being allowed in interface theories derived from de Alfaro and Henzinger's Interface Automata [10] but forbidden in Modal Transition Systems [14]. To this end, we introduced a special universal state, which enabled us to achieve compositionality (in contrast to [15]) as well as associativity (in contrast to [18]) for parallel composition; this also allowed for a more practical support of perspective-based specification when compared to [16,17]. In addition, we defined a quotienting operator that permits the decomposition of *nondeterministic* specifications and takes *pruning* in parallel composition into account (in contrast to [18]).

Regarding future work, we wish to explore the choice of alphabets for quotienting and relax the determinism requirement on divisors. We also intend to implement our theory in MICA (see `http://www.irisa.fr/s4/tools/mica/`) or the MIO Workbench [2].

Acknowledgments. We thank the reviewers for their comments and suggestions.

References

1. Bauer, S.S., David, A., Hennicker, R., Guldstrand Larsen, K., Legay, A., Nyman, U., Wąsowski, A.: Moving from specifications to contracts in component-based design. In: de Lara, J., Zisman, A. (eds.) FASE 2012. LNCS, vol. 7212, pp. 43–58. Springer, Heidelberg (2012)
2. Bauer, S.S., Mayer, P., Schroeder, A., Hennicker, R.: On weak modal compatibility, refinement, and the MIO Workbench. In: Esparza, J., Majumdar, R. (eds.) TACAS 2010. LNCS, vol. 6015, pp. 175–189. Springer, Heidelberg (2010)
3. Benveniste, A., Caillaud, B., Nickovic, D., Passerone, R., Raclet, J.B., Reinkemeier, P., Sangiovanni-Vincentelli, A., Damm, W., Henzinger, T.A., Larsen, K.G.: Contracts for system design. Tech. Rep. 8147, INRIA (November 2012)
4. Beyer, D., Chakrabarti, A., Henzinger, T.A., Seshia, S.A.: An application of web-service interfaces. In: ICWS, pp. 831–838. IEEE (2007)
5. Bujtor, F., Fendrich, S., Lüttgen, G., Vogler, W.: Nondeterministic modal interfaces. Tech. Rep. 2014-06, Institut für Informatik, Universität Augsburg (2014)
6. Bujtor, F., Vogler, W.: Error-pruning in interface automata. In: Geffert, V., Preneel, B., Rovan, B., Štuller, J., Tjoa, A.M. (eds.) SOFSEM 2014. LNCS, vol. 8327, pp. 162–173. Springer, Heidelberg (2014)
7. Chen, T., Chilton, C., Jonsson, B., Kwiatkowska, M.Z.: A compositional specification theory for component behaviours. In: Seidl, H. (ed.) ESOP 2012. LNCS, vol. 7211, pp. 148–168. Springer, Heidelberg (2012)
8. Chilton, C.: An Algebraic Theory of Componentised Interaction. Ph.D. thesis, Oxford (2013)
9. de Alfaro, L., Henzinger, T.A.: Interface automata. In: FSE, pp. 109–120. ACM (2001)
10. de Alfaro, L., Henzinger, T.A.: Interface-based design. In: Engineering Theories of Software-Intensive Systems. NATO Science Series, vol. 195. Springer (2005)
11. De Nicola, R., Segala, R.: A process algebraic view of input/output automata. Theor. Comput. Sci. 138(2), 391–423 (1995)
12. Hüttel, H., Larsen, K.G.: The use of static constructs in a modal process logic. In: Meyer, A.R., Taitslin, M.A. (eds.) Logic at Botik 1989. LNCS, vol. 363, pp. 163–180. Springer, Heidelberg (1989)
13. Larsen, K., Xinxin, L.: Equation solving using modal transition systems. In: LICS, pp. 108–117. IEEE (1990)
14. Larsen, K.G.: Modal specifications. In: Sifakis, J. (ed.) CAV 1989. LNCS, vol. 407, pp. 232–246. Springer, Heidelberg (1990)
15. Larsen, K.G., Nyman, U., Wąsowski, A.: Modal I/O automata for interface and product line theories. In: De Nicola, R. (ed.) ESOP 2007. LNCS, vol. 4421, pp. 64–79. Springer, Heidelberg (2007)
16. Lüttgen, G., Vogler, W.: Modal interface automata. LMCS 9(3) (2013)
17. Lüttgen, G., Vogler, W.: Richer interface automata with optimistic and pessimistic compatibility. ECEASST 66 (2013), an extended version has been submitted to Acta Informatica
18. Raclet, J.B., Badouel, E., Benveniste, A., Caillaud, B., Legay, A., Passerone, R.: A modal interface theory for component-based design. Fund. Inform. 108(1-2), 119–149 (2011)