An Algebraic Theory of Interface Automata

Chris Chilton[a], Bengt Jonsson[b], Marta Kwiatkowska[a]

[a]*Department of Computer Science, University of Oxford, UK*
[b]*Department of Information Technology, Uppsala University, Sweden*

Abstract

We formulate a compositional specification theory for interface automata, where a component model specifies the allowed sequences of input and output interactions with the environment. A trace-based linear-time refinement is provided, which is the weakest preorder preserving substitutivity of components, and is weaker than the classical alternating simulation defined on interface automata. Since our refinement allows a component to be refined by refusing to produce any output, we also define a refinement relation that guarantees safety and progress. The theory includes the operations of parallel composition to support the structural composition of components, logical conjunction and disjunction for independent development, hiding to support abstraction of interfaces, and quotient for incremental synthesis of components. Our component formulation highlights the algebraic properties of the specification theory for both refinement preorders, and is shown to be fully abstract with respect to observation of communication mismatches. Examples of independent and incremental component development are provided.

Keywords: component-based design, interfaces, specification theory, compositionality, refinement, substitutivity, synthesis

1. Introduction

The interface automata of de Alfaro and Henzinger [11] are an influential formalism for modelling the interactions between components and their environment. Components are assumed to communicate by synchronisation of input and output (I/O) actions, with the understanding that outputs are non-blocking. If an output is issued when a component is unwilling to receive it, a communication mismatch is said to occur. This allows one to reason about the allowed behaviours of the environment, which is crucial in assume-guarantee reasoning, for example.

An important paradigm for developing complex reactive systems is component-based design, which can be underpinned by a specification theory. A specification captures the requirements for a component to function in an intended system context, while operators and refinement relations allow for the composing and comparing of specifications in analogy with how components are composed and refined towards an overall system design. Substitutive refinement is essential for dynamic systems of components in this respect, as it allows for the replacement of components without introducing errors to the system.

The original theory of interface automata defines a substitutive refinement in terms of alternating simulation [2], along with a parallel composition operator for observing component interaction. In subsequent papers, variants of the framework have been extended with additional operators, including conjunction (defined by Doyen et al. for synchronous automata [15]) and quotient for supporting incremental development (defined by Bhaduri and Ramesh for deterministic automata [4]).

In this article, we formulate a theory for components that is conceptually similar to interface automata, but is based on a linear-time notion of substitutive refinement involving trace containment. We define a specification theory for component behaviours, which includes the operations of: *parallel composition* for structural composition of components; *conjunction* for supporting independent development, by constructing a component that will work in any environment compatible with at least one of its arguments; *disjunction* for constructing a component that has an environment compatible for both of its operands; *hiding* to support abstraction in hierarchical development; and *quotient* for incrementally synthesising new components to satisfy partial requirements. We prove compositionality for all the operations and show that the specification theory enjoys strong algebraic properties.

Our formalism addresses the following shortcomings of the interface automata theory as formulated by de Alfaro and Henzinger in [11]:

- Alternating simulation is conceptually more complex than refinement based on trace containment, which is standard in widely used theories such as CSP [5] and I/O automata [27, 20]. Further, alternating simulation is overly strong in comparison to our refinement based on traces, which is the weakest preorder preserving compatibility with the environment.

- It is not clear how to extend a refinement relation based on alternating

simulation so that it also preserves liveness properties. This should be contrasted with the conceptually simple handling of liveness properties in formalisms such as I/O automata, which use trace inclusion. In the case of our refinement, we are able to extend it with the notion of quiescence to guarantee observational progress, in addition to substitutivity. We prove that compositionality results for all the operations continue to hold for this enhanced refinement.

The contribution of this article is therefore a compositional, linear-time specification theory for interface automata based on fully abstract substitutive and progress-preserving refinement. Our framework includes all desirable operations on components known from the literature, and satisfies strong algebraic properties, including the characterisation of conjunction and disjunction, respectively, as the meet and join of the refinement preorder. The theory naturally supports a component-based design process that starts with initial design considerations, from which the operations of the theory are applied compositionally in a stepwise fashion, relying on substitutivity to guarantee that no errors will be introduced, even if components are refined at runtime.

A preliminary version of this article appeared as [6], where we introduced the operations of parallel, conjunction and quotient, but did not consider an extension with quiescence. To demonstrate the applicability of the theory to component-based design, the quotient operation was used to synthesise mediator components in [18] and [3]. Furthermore, the flexibility and expressiveness of the theory has been shown through a compositional assume-guarantee reasoning framework [8, 9] and a real-time extension [10, 7].

1.1. Related Work

Interface automata. The models in this article are conceptually similar to interface automata [11], which are essentially finite state automata with I/O distinction on actions. A key difference is that our refinement preorder is a linear-time alternative to the alternating simulation of Alur et al. [2] defined on interface automata. Both refinements are substitutive, but alternating simulation is overly strong due to the conflict between non-determinism in the automaton and the selection of a matching transition to complete the simulation. Effectively, our notion of parallel composition is the same as for interface automata, except that we encode inconsistency due to communication mismatches explicitly in the model. To the best of our knowledge, conjunction and disjunction have not been defined on interface automata,

although Doyen et al. [15] define conjunction (called shared refinement) on a synchronous component model. A definition of quotient has been provided for deterministic interface automata by Bhaduri and Ramesh [4], which mirrors the method developed by Verhoeff [40].

I/O automata. Due to Lynch and Tuttle [27], and Jonsson [20], I/O automata are highly similar to interface automata, except that each state is required to be input-enabled. This input receptiveness means that communication mismatches cannot arise between a component and its environment. Consequently, substitutive refinement can be cast in terms of trace containment [20]. The operation of parallel composition is defined in the same way as for interface automata, except that consideration need not be given to inconsistencies. Conjunction can be defined as a synchronous product, meaning that its set of traces is the intersection of its operands' traces. Disjunction can be defined similarly. Hiding is already defined on outputs by Jonsson [20], and quotient can be defined in a straightforward manner as demonstrated by Drissi and von Bochmann [16].

We mention a process-algebraic characterisation of I/O automata due to de Nicola and Segala [13], which is also applicable to interface automata, since a process exhibits chaotic behaviour on receiving a non-enabled input. Refinement is defined by trace inclusion, but this does not extend to inconsistent trace containment. Consequently, the theory is not able to distinguish a non-enabled input from one that is enabled and can subsequently behave chaotically. Furthermore, high-level operations such as conjunction and quotient are not defined. Note that the CCS of Milner [28] merely has a syntactic distinction of inputs from outputs, so we give it no further attention.

Logic LTSs. Devised by Lüttgen and Vogler [25], these are labelled transition system (LTS) models, without I/O distinction, augmented by an inconsistency predicate on states. A number of compositional operators are considered (parallel composition, conjunction, disjunction, external choice, and hiding [26]), and refinement is given by ready-simulation, a branching time relation that requires the refining component not to introduce any new inconsistency and equality of offered actions at each state in the simulation chain. This formulation of refinement differs from our intuition behind substitutivity, meaning that their operations, such as conjunction, are incomparable to ours. Taking inspiration from [25], in Section 4 we formulate an operational model of components that are I/O automata augmented by an

inconsistency predicate for indicating communication mismatches (and, consequently, non-enabled inputs), making our formalism achieve similar goals as interface automata, but with notation and semantics derived from I/O automata and Logic LTSs.

Circuit trace structures. Dill [14] presents a trace-based theory for modelling circuits, with I/O distinction, which conceptually uses the same basic semantic model as in our framework. A circuit can be characterised by prefix closed sets of traces, which corresponds exactly with our component model in Section 2. Dill's conformance is also similar to our substitutive refinement, except that we generalise this to allow non-identical (static) interfaces, but his liveness extension is based on infinite traces, rather than finite traces and quiescence. Additionally, we formulate a richer collection of compositional operators.

Wolf [41] extends Dill's trace structures to synchronous circuits, where the refinement preorder is based on inclusion between trace sets as in our work, except that processes are restricted to having the same interfaces. As in [14], conjunction, disjunction and quotient are not considered. Process spaces introduced by Negulescu [29, 30] are an abstract theory of process executions which allows one to separate assumptions on the environment from the requirements of processes. Dill's trace structures can be instantiated in this theory, but our results are not derivable. Meanwhile, Passerone [31] defines an algebra that consists of a set of denotations, called agents, for the elements of a model, and of the main operations that the model provides to compose and manipulate agents. The operators considered are inspired by those of [14], but again do not capture the rich collection we consider.

Receptive process theory. Josephs et al. [22] formulate an I/O extension of CSP [17] for modelling asynchronous circuits. The work differs from ours in that processes must communicate through unbounded buffers, which eliminates the possibility of communication errors arising through non-enabledness of inputs. Avoiding this, Josephs [21] formulates a theory of receptive processes, where components must communicate directly with one another. This has connections to our liveness framework, since a receptive process is modelled by means of its failures (communication mismatches and divergences) and quiescent traces (violations of liveness). Consequently, the refinement relation is similar to our progress-sensitive refinement, except that we give an explicit treatment of divergence. Josephs' work does not consider conjunction

and quotient (the latter is defined on the restricted class of delay-insensitive networks [23], where it is referred to as factorisation; however, this does not match our setting).

Modal interfaces. A modal specification characterises sequences of (non I/O) interactions between a component and its environment, along with modalities on the interactions, indicating whether an interaction may or must be possible. Raclet et al. [33, 35, 34] introduce a specification theory for modal specifications that considers a substitutive refinement relation, along with the operations of parallel composition, conjunction and quotient [32]. Their notion of liveness and progress is based on must-modalities, and thus differs from our trace-based formulation. The theory is extended in [35, 34] to modal interfaces (modal specifications with I/O distinction), where a mapping is given from deterministic interface automata without hidden actions to modal interfaces. This is similar to the theory of Larsen et al. [24], except that: a number of technical issues are resolved, relating to compatibility and parallel composition; refinement is based on trace-containment, rather than being game-based; and additional compositional operators are defined.

A weakness of [35, 34] is that the compositionality results for the different operators must be given with respect to either strong or weak refinement relations (the former for parallel and quotient, the latter for conjunction) when the components to be composed have dissimilar alphabets. This has repercussions for parallel composition, which is an asynchronous operator on interface automata, but is treated synchronously on modal interfaces by a lifting on alphabets. This lifting is essentially equivalent to requiring that a refining component is enabled in every state on each input that is not in the interface of the original component. Consequently, there are also differences between the quotient operators of the two frameworks, since they should be the adjoint of their respective parallel operations.

Ioco-testing theory. Our work is related to the ioco theory for model based testing due to Tretmans [38], which only considers the operators of parallel composition, hiding and choice. Aarts and Vaandrager [1] show the similarities between interface automata and the ioco theory. A key result of that paper relates quiescence-extended alternating simulation refinement on interface automata with the ioco relation, under determinism of models. In comparison with our framework, this implies that the ioco relation coincides with our progress-sensitive refinement for components free of divergence.

1.2. Outline

Section 2 begins by introducing a trace-based theory of interface automata, and defines substitutive refinement, along with the collection of compositional operators. In Section 3, we extend the trace-based theory by formulating a refinement preorder guaranteeing substitutivity together with preservation of progress, for which we generalise the compositional operators. An operational theory of components is presented in Section 4, for both the substitutive and progress-sensitive frameworks. A detailed comparison of our work with interface automata is given in Section 5, while Section 6 concludes.

2. A Trace-Based Theory of Substitutable Components

In this section, we introduce a trace-based representation for components modelled as interface automata. The formulation captures the essential information relating to whether a component can work in an arbitrary environment without introducing communication mismatches, which is vital for checking substitutability of components. Based on this representation, we introduce a weakest refinement relation preserving safe substitutivity of components and provide definitions of compositional operators for our theory.

Definition 1 (Component). *A* component $\mathcal{P}$ *is a tuple* $\langle \mathcal{A}_\mathcal{P}^I, \mathcal{A}_\mathcal{P}^O, T_\mathcal{P}, F_\mathcal{P} \rangle$ *in which* $\mathcal{A}_\mathcal{P}^I$ *and* $\mathcal{A}_\mathcal{P}^O$ *are disjoint sets referred to as the inputs and outputs respectively (the union of which is denoted by* $\mathcal{A}_\mathcal{P}$*),* $T_\mathcal{P} \subseteq \mathcal{A}_\mathcal{P}^*$ *is a set of observable traces, and* $F_\mathcal{P} \subseteq \mathcal{A}_\mathcal{P}^*$ *is a set of inconsistent traces. The trace sets must satisfy the constraints:*

1. $F_\mathcal{P} \subseteq T_\mathcal{P}$
2. $T_\mathcal{P}$ *is prefix closed*
3. *If* $t \in T_\mathcal{P}$ *and* $t' \in (\mathcal{A}_\mathcal{P}^I)^*$*, then* $tt' \in T_\mathcal{P}$
4. *If* $t \in F_\mathcal{P}$ *and* $t' \in \mathcal{A}_\mathcal{P}^*$*, then* $tt' \in F_\mathcal{P}$*.*

If $\epsilon \notin T_\mathcal{P}$*, we say that* $\mathcal{P}$ *is unrealisable, and is realisable contrariwise.*

The sets $\mathcal{A}_\mathcal{P}^I$ and $\mathcal{A}_\mathcal{P}^O$ make up the *interface* of $\mathcal{P}$, i.e., the interaction primitives that the component is willing to observe, while the trace sets encode the possible interaction sequences over the component's interface. $T_\mathcal{P}$ consists of all *observable* sequences of interactions that can arise between

the component and the environment. As inputs are controlled by the environment, any trace in $T_{\mathcal{P}}$ is extendable by a sequence of inputs, since the component cannot prevent these inputs from being issued (this is referred to as *input-receptivity*). Traces contained in $F_{\mathcal{P}}$ are deemed to be *inconsistent*, which can encode, e.g., run-time errors and communication mismatches. As interface automata are not required to be input receptive, we use $F_{\mathcal{P}}$ to record the traces in $T_{\mathcal{P}}$ that involve non-enabled inputs. Under this treatment of inputs, we say that our theory is *not* input enabled, even though $T_{\mathcal{P}}$ is closed under input extensions. Once an inconsistency has arisen, the resulting behaviour is unspecified, so we assume that subsequent observations of the component are chaotic.

Throughout this article, we use the following running example to demonstrate the suitability of our framework for component-based design. The interested reader is referred to [3] for an application of the theory to mediator synthesis and to [7] for further examples, including examples of timed components.

Example 1. *A multi-function device capable of printing and scanning is modelled as a component* Device *in Figure 1. The device can be placed in* print_mode *or* scan_mode*, can receive* job_details*, and can* print *and* scan*. From the perspective of the device, actions* print *and* scan *should be treated as outputs (indicated by* !*), while all other actions are inputs (indicated by* ?*).*

Concerning the diagrammatic representation, the interface of a component is given by the actions labelling transitions in the figure (note that, in general, the interface may contain actions that do not occur in a component's behaviour). For compactness, we avoid giving an explicit representation for input transitions immediately leading to an inconsistent state, since they can be inferred from the input-receptivity of observable traces. Furthermore, at this stage of the article, whether a node is a circle or a square is irrelevant; the distinction will become apparent in Section 3, Example 7.

From hereon let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components with signatures $\langle \mathcal{A}^I_{\mathcal{P}}, \mathcal{A}^O_{\mathcal{P}}, T_{\mathcal{P}}, F_{\mathcal{P}} \rangle$, $\langle \mathcal{A}^I_{\mathcal{Q}}, \mathcal{A}^O_{\mathcal{Q}}, T_{\mathcal{Q}}, F_{\mathcal{Q}} \rangle$ and $\langle \mathcal{A}^I_{\mathcal{R}}, \mathcal{A}^O_{\mathcal{R}}, T_{\mathcal{R}}, F_{\mathcal{R}} \rangle$ respectively.

Notation. Let $\mathcal{A}$ and $\mathcal{B}$ be sets of actions. For a trace t, write $t \restriction \mathcal{A}$ for the projection of t onto $\mathcal{A}$. Now for $T \subseteq \mathcal{A}^*$, write $T \restriction \mathcal{B}$ for $\{t \restriction \mathcal{B} : t \in T\}$, $T \Uparrow \mathcal{B}$ for $\{t \in \mathcal{B}^* : t \restriction \mathcal{A} \in T\}$ and $T \uparrow \mathcal{B}$ for $T \cdot (\mathcal{B} \setminus \mathcal{A}) \cdot (\mathcal{A} \cup \mathcal{B})^*$.

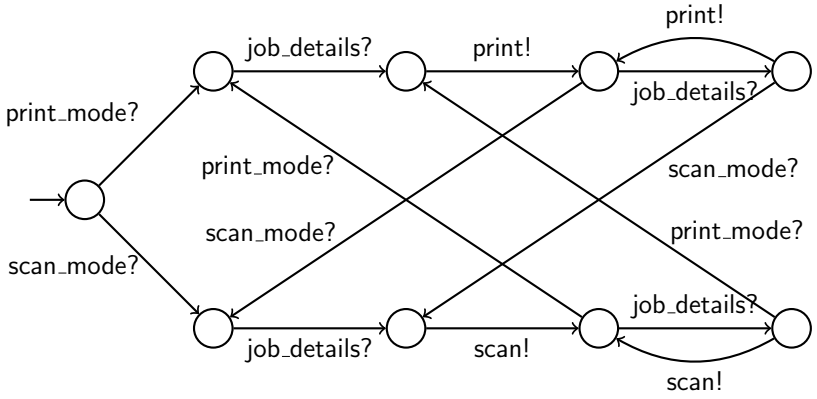

Figure 1: Multi-function Device.

2.1. Refinement

The refinement relation on components should support safe substitutivity, meaning that, for $\mathcal{Q}$ to be used in place of $\mathcal{P}$, we require that $\mathcal{Q}$ exists safely in *every* environment that is safe for $\mathcal{P}$. Whether an environment is safe or not for a component depends on the interaction sequences between the two. The affirmative holds if the environment can prevent the component from performing an inconsistent trace. As outputs are controlled by the component, it follows that a safe environment must refuse to issue an input on any trace from which there is a sequence of output actions that allow the trace to become inconsistent.

Given a component $\mathcal{P}$, we can formulate the most general safe component $\mathcal{E}(\mathcal{P})$, containing all of $\mathcal{P}$'s observable and inconsistent traces, but satisfying the additional property: if $t \in T_{\mathcal{P}}$ and there exists $t' \in (\mathcal{A}^O_{\mathcal{P}})^*$ such that $tt' \in F_{\mathcal{P}}$, then $t \in F_{\mathcal{E}(\mathcal{P})}$. This has the effect of making the component immediately inconsistent whenever it has the potential to become inconsistent under its own control. If the environment respects this safe component, by not issuing any input that results in an inconsistent trace, then the component can never encounter an inconsistent trace. Note that if $\epsilon \in F_{\mathcal{E}(\mathcal{P})}$ then there is no environment that can prevent $\mathcal{P}$ from performing an inconsistent trace. However, for uniformity we still refer to $\mathcal{E}(\mathcal{P})$ as the safe component of $\mathcal{P}$.

Definition 2. *The* safe component *for* $\mathcal{P}$ *is defined as* $\mathcal{E}(\mathcal{P}) = \langle \mathcal{A}^I_{\mathcal{P}}, \mathcal{A}^O_{\mathcal{P}}, T_{\mathcal{P}} \cup F_{\mathcal{E}(\mathcal{P})}, F_{\mathcal{E}(\mathcal{P})} \rangle$*, where* $F_{\mathcal{E}(\mathcal{P})} = \{t \in T_{\mathcal{P}} : \exists t' \in (\mathcal{A}^O_{\mathcal{P}})^* \cdot tt' \in F_{\mathcal{P}}\} \cdot \mathcal{A}^*_{\mathcal{P}}$.

Based on safe components, we can now give the formal definition of substitutive refinement.

Definition 3 (Refinement). *$\mathcal{Q}$ is said to be a* refinement *of $\mathcal{P}$, written $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$, iff:*

I1. $\mathcal{A}^I_{\mathcal{P}} \subseteq \mathcal{A}^I_{\mathcal{Q}}$

I2. $\mathcal{A}^O_{\mathcal{Q}} \subseteq \mathcal{A}^O_{\mathcal{P}}$

I3. $\mathcal{A}^I_{\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$

I4. $T_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}^*_{\mathcal{P}} \subseteq T_{\mathcal{E}(\mathcal{P})}$

I5. $F_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}^*_{\mathcal{P}} \subseteq F_{\mathcal{E}(\mathcal{P})}$.

For $\mathcal{Q}$ to be a refinement of $\mathcal{P}$, the interface of $\mathcal{Q}$ must be substitutable for the interface of $\mathcal{P}$, meaning that $\mathcal{Q}$ must be willing to accept all of $\mathcal{P}$'s inputs, while it must produce only a subset of $\mathcal{P}$'s outputs, as witnessed by I1 and I2. Condition I3 ensures that $\mathcal{P}$ and $\mathcal{Q}$ are *compatible*, that is, they are not allowed to mix action types. In [6] we did not impose this constraint, as it is not necessary to guarantee substitutivity. However, in this article we choose to include the constraint for three reasons: (i) it is not necessarily meaningful to convert outputs into inputs during refinement; (ii) compositionality of hiding does not hold without this constraint; and (iii) mixing of action types is problematic for assume-guarantee reasoning, which deals with the behaviour of the environment.

Condition I4 ensures that the observable behaviour of $\mathcal{Q}$ is contained within the behaviour of $\mathcal{P}$, except for when an input in $\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{P}}$ is encountered. An environment for $\mathcal{P}$ cannot issue such an input, so we are not concerned with the ensuing behaviour of $\mathcal{Q}$, which should be unconstrained. Finally, condition I5 ensures that $\mathcal{Q}$ cannot introduce any new errors that are not in $\mathcal{P}$'s behaviour. Note that checking $F_{\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{P}} \subseteq F_{\mathcal{P}}$ would be too strong to use for the last clause, as we are only interested in trace containment up to the point where an environment can issue a bad input, from which the component can become inconsistent autonomously.

Definition 4. *$\mathcal{P}$ and $\mathcal{Q}$ are said to be* equivalent*, written $\mathcal{P} \equiv_{imp} \mathcal{Q}$, iff $\mathcal{P} \sqsubseteq_{imp} \mathcal{Q}$ and $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$.*

Lemma 1. *Refinement is reflexive, and is transitive subject to preservation of action types:* $\mathcal{R} \sqsubseteq_{imp} \mathcal{Q}$, $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ *and* $\mathcal{A}^I_{\mathcal{R}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$ *implies* $\mathcal{R} \sqsubseteq_{imp} \mathcal{P}$.

Proof. Reflexivity is trivial, while transitivity follows by the transitivity of subset inclusion. □

We are now in a position to define the compositional operators of our theory. In general, the compositional operators are only partially defined, specifically on components that are said to be *composable*. This is a syntactic check on the interfaces of the components to be composed, which ensures that their composition is meaningful. For each operator, we state the required composability constraints.

2.2. Parallel Composition

The parallel composition of two components yields a component representing the combined effect of its operands running asynchronously. The composition is obtained by synchronising on common actions and interleaving on independent actions. This makes sense even in the presence of non-blocking outputs, because communication mismatches arising through non-enabledness of inputs automatically appear as inconsistent traces in the composition, on account of our component formulation. To support broadcasting, we make the assumption that inputs and outputs synchronise to produce outputs. As the outputs of a component are controlled locally, we also assume that the output actions of the components to be composed are disjoint, in which case we say that the components are *composable*. In practice, components that are not composable can be made so by employing renaming.

Definition 5. *Let* $\mathcal{P}$ *and* $\mathcal{Q}$ *be composable for parallel, i.e.,* $\mathcal{A}^O_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{Q}} = \emptyset$. *Then* $\mathcal{P} \,||\, \mathcal{Q}$ *is the component* $\langle \mathcal{A}^I_{\mathcal{P}||\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}||\mathcal{Q}}, T_{\mathcal{P}||\mathcal{Q}}, F_{\mathcal{P}||\mathcal{Q}} \rangle$, *where:*

- $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} = (\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}) \setminus (\mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}})$
- $\mathcal{A}^O_{\mathcal{P}||\mathcal{Q}} = \mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}}$
- $T_{\mathcal{P}||\mathcal{Q}} = [(T_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \cap (T_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}})] \cup F_{\mathcal{P}||\mathcal{Q}}$
- $F_{\mathcal{P}||\mathcal{Q}} = [(T_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \cap (F_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}})] \cdot \mathcal{A}^*_{\mathcal{P}||\mathcal{Q}} \cup$
  $[(F_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \cap (T_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}})] \cdot \mathcal{A}^*_{\mathcal{P}||\mathcal{Q}}$.

In words, the observable traces of the composition are simply those traces that are inconsistent, plus any trace whose projection onto $\mathcal{A}_\mathcal{P}$ is a trace of $\mathcal{P}$ and whose projection onto $\mathcal{A}_\mathcal{Q}$ is a trace of $\mathcal{Q}$. A trace is inconsistent if it has a prefix whose projection onto the alphabet of one of the components is inconsistent and the projection onto the alphabet of the other component is an observable trace of that component.

The definition of parallel composition for interface automata as defined by de Alfaro and Henzinger [11] also includes backward propagation of inconsistencies. This is not necessary in our framework, since an equivalent technique is implicitly performed as part of refinement, as we do not insist on a canonical representation for components. A canonical representation for a component could easily be obtained by applying the $\mathcal{E}$ operator, should this be considered desirable. In such a setting, the class of component representations would be the safe components (cf. Definition 2).

Lemma 2. *Parallel composition is associative and commutative.*

Proof. Commutativity is trivial. For associativity, we show that $F_{\mathcal{P}||(\mathcal{Q}||\mathcal{R})} = F_{(\mathcal{P}||\mathcal{Q})||\mathcal{R}}$, given that the T-set equivalence is similar. As a shorthand, we use $\mathcal{A}$ to denote $\mathcal{A}_{\mathcal{P}||(\mathcal{Q}||\mathcal{R})}$, which is equal to $\mathcal{A}_{(\mathcal{P}||\mathcal{Q})||\mathcal{R}}$.

$$
\begin{aligned}
F_{\mathcal{P}||(\mathcal{Q}||\mathcal{R})} &= [T_\mathcal{P} \Uparrow \mathcal{A} \cap F_{\mathcal{Q}||\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \text{ (i)} \\
&\quad \cup [F_\mathcal{P} \Uparrow \mathcal{A} \cap T_{\mathcal{Q}||\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&\quad = [T_\mathcal{P} \Uparrow \mathcal{A} \cap ((F_\mathcal{Q} \Uparrow \mathcal{A}_{\mathcal{Q}||\mathcal{R}} \cap T_\mathcal{R} \Uparrow \mathcal{A}_{\mathcal{Q}||\mathcal{R}}) \cdot \mathcal{A}^*_{\mathcal{Q}||\mathcal{R}}) \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \text{ (ii)} \\
&\quad \cup [T_\mathcal{P} \Uparrow \mathcal{A} \cap ((T_\mathcal{Q} \Uparrow \mathcal{A}_{\mathcal{Q}||\mathcal{R}} \cap F_\mathcal{R} \Uparrow \mathcal{A}_{\mathcal{Q}||\mathcal{R}}) \cdot \mathcal{A}^*_{\mathcal{Q}||\mathcal{R}}) \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \text{ (iii)} \\
&\quad \cup [F_\mathcal{P} \Uparrow \mathcal{A} \cap (T_\mathcal{Q} \Uparrow \mathcal{A}_{\mathcal{Q}||\mathcal{R}} \cap T_\mathcal{R} \Uparrow \mathcal{A}_{\mathcal{Q}||\mathcal{R}}) \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&\quad \cup [F_\mathcal{P} \Uparrow \mathcal{A} \cap F_{\mathcal{Q}||\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \text{ contained within (i), so within (ii) and (iii)} \\
&\quad = [T_\mathcal{P} \Uparrow \mathcal{A} \cap (F_\mathcal{Q} \Uparrow \mathcal{A} \cap T_\mathcal{R} \Uparrow \mathcal{A})] \cdot \mathcal{A}^* \\
&\quad \cup [T_\mathcal{P} \Uparrow \mathcal{A} \cap (T_\mathcal{Q} \Uparrow \mathcal{A} \cap F_\mathcal{R} \Uparrow \mathcal{A})] \cdot \mathcal{A}^* \\
&\quad \cup [F_\mathcal{P} \Uparrow \mathcal{A} \cap (T_\mathcal{Q} \Uparrow \mathcal{A} \cap T_\mathcal{R} \Uparrow \mathcal{A})] \cdot \mathcal{A}^* \\
&\quad = [(T_\mathcal{P} \Uparrow \mathcal{A} \cap F_\mathcal{Q} \Uparrow \mathcal{A}) \cap T_\mathcal{R} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&\quad \cup [(T_\mathcal{P} \Uparrow \mathcal{A} \cap T_\mathcal{Q} \Uparrow \mathcal{A}) \cap F_\mathcal{R} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&\quad \cup [(F_\mathcal{P} \Uparrow \mathcal{A} \cap T_\mathcal{Q} \Uparrow \mathcal{A}) \cap T_\mathcal{R} \Uparrow \mathcal{A}] \cdot \mathcal{A}^*
\end{aligned}
$$

$$
\begin{aligned}
&= [(T_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}} \cap F_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \Uparrow \mathcal{A} \cap T_{\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&\cup [(T_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}} \cap T_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \Uparrow \mathcal{A} \cap F_{\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&\cup [(F_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}} \cap T_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \Uparrow \mathcal{A} \cap T_{\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&= [F_{\mathcal{P}||\mathcal{Q}} \Uparrow \mathcal{A} \cap T_{\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&\cup [T_{\mathcal{P}||\mathcal{Q}} \Uparrow \mathcal{A} \cap F_{\mathcal{R}} \Uparrow \mathcal{A}] \cdot \mathcal{A}^* \\
&= F_{(\mathcal{P}||\mathcal{Q})||\mathcal{R}}
\end{aligned}
$$

□

The following result shows that parallel composition is monotonic on refinement, subject to restrictions on the interfaces to be composed and composability. A corollary of this result is that mutual refinement is a congruence for parallel, subject (only) to composability.

Theorem 1. *Let $\mathcal{P}$, $\mathcal{Q}$, $\mathcal{P}'$ and $\mathcal{Q}'$ be components such that $\mathcal{P}$ and $\mathcal{Q}$ are composable, $\mathcal{A}_{\mathcal{P}'} \cap \mathcal{A}_{\mathcal{Q}'} \cap \mathcal{A}_{\mathcal{P}||\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{P}} \cap \mathcal{A}_{\mathcal{Q}}$ and $\mathcal{A}^O_{\mathcal{P}||\mathcal{Q}} \cap \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'} = \emptyset$. If $\mathcal{P}' \sqsubseteq_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq_{imp} \mathcal{Q}$, then $\mathcal{P}' \,||\, \mathcal{Q}' \sqsubseteq_{imp} \mathcal{P} \,||\, \mathcal{Q}$.*

Proof. It is easy to show that the conditions on alphabets are satisfied. To show $t \in F_{\mathcal{E}(\mathcal{P}'||\mathcal{Q}')} \cap \mathcal{A}^*_{\mathcal{P}||\mathcal{Q}}$ implies $t \in F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$ (and respectively for the T-sets), assume that the result holds for all strict prefixes of t. So there exists $t'' \in (\mathcal{A}^O_{\mathcal{P}'||\mathcal{Q}'})^*$ such that $tt'' \in F_{\mathcal{P}'||\mathcal{Q}'}$. By the conditions on alphabets, it also follows that $tt'' \in \mathcal{A}^*_{\mathcal{P}||\mathcal{Q}}$. It can now be shown that $tt'' \upharpoonright \mathcal{A}_{\mathcal{P}} = tt'' \upharpoonright \mathcal{A}_{\mathcal{P}'}$ (and similarly for $\mathcal{A}_{\mathcal{Q}}$ and $\mathcal{A}_{\mathcal{Q}'}$), for suppose that there exists $a \in \mathcal{A}_{\mathcal{P}} \setminus \mathcal{A}_{\mathcal{P}'}$ on the trace tt''. Then $a \in \mathcal{A}^O_{\mathcal{P}}$, which implies $a \notin \mathcal{A}_{\mathcal{P}'||\mathcal{Q}'}$, as $a \notin \mathcal{A}_{\mathcal{Q}'}$ by compatibility and composability. Instead, if $a \in \mathcal{A}_{\mathcal{P}'} \setminus \mathcal{A}_{\mathcal{P}}$, then $a \in \mathcal{A}^I_{\mathcal{P}'}$. As $a \in \mathcal{A}_{\mathcal{P}'||\mathcal{Q}'} \cap \mathcal{A}_{\mathcal{P}||\mathcal{Q}}$, it must hold that $a \in \mathcal{A}^I_{\mathcal{Q}}$, but, by the conditions of the theorem, it would follow that $a \in \mathcal{A}^I_{\mathcal{P}}$, which is contradictory.

So, without loss of generality, suppose $tt'' \upharpoonright \mathcal{A}_{\mathcal{P}'} \in F_{\mathcal{P}'}$ and $tt'' \upharpoonright \mathcal{A}_{\mathcal{Q}'} \in T_{\mathcal{Q}'}$. By refinement at the component level, it follows that $tt'' \upharpoonright \mathcal{A}_{\mathcal{P}} \in F_{\mathcal{E}(\mathcal{P})}$ and $tt'' \upharpoonright \mathcal{A}_{\mathcal{Q}} \in T_{\mathcal{E}(\mathcal{Q})}$. From this, it is easy to show that $tt'' \in F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$, and so $t \in F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$ as required. The T-set containment is a simplification, since it is not necessary to consider the t'' extension. □

Parallel composition is claimed to be monotonic for modal interfaces without any conditions on the interfaces (except for composability), according to Raclet et al. in [34]. This is due to the authors using strong refinement, which

is more restrictive than $\sqsubseteq_{imp}$, since it requires that all actions in $\mathcal{A}^I_{\mathcal{P}'} \setminus \mathcal{A}^I_{\mathcal{P}}$ and $\mathcal{A}^I_{\mathcal{Q}'} \setminus \mathcal{A}^I_{\mathcal{Q}}$ never produce unsafe behaviour (see Section 2.7 for a more detailed consideration).

Example 2. *The most liberal* User *that can interact with the* Device *(shown in Figure 1) is a component obtained from* Device *by interchanging inputs and outputs (given that we do not explicitly represent traces making the component receptive). The definition of parallel composition guarantees that the composition of the* Device *along with the resultant* User *is free of inconsistencies (i.e., communication mismatches), and is a transition system equal to that of the* Device *and the* User*, but with all actions converted to outputs.*

Note that, if a user wished to perform the trace print_mode! scan_mode!*, then this would also be a trace in the parallel composition, since* print_mode? scan_mode? *is a trace of* Device*, albeit an inconsistent one, which is why it is not explicitly represented in Figure 1. Consequently, the trace would also be inconsistent in the parallel composition.*

2.3. Conjunction

The conjunction operator on components can be thought of as supporting independent development, in the sense that it yields the coarsest component that will work in any environment safe for at least one of its operands. Consequently, the conjunction of components is the coarsest component that is a refinement of its operands (i.e. is the meet operator), which is why it is frequently referred to as the *shared refinement* operator [15, 35].

In a number of frameworks, including that of Logic LTSs due to Lüttgen and Vogler [25], conjunction represents synchronous parallel composition, formed as the intersection of the good behaviours of the components to be composed. In contrast, our conjunction is a *substitutive refinement* of each component. Therefore, an input must be accepted in the conjunction if at least one of the components accepts it, while an input should be accepted in the synchronous parallel only if all of the appropriately alphabetised components accept it.

Conjunction is only defined on composable components, where $\mathcal{P}$ and $\mathcal{Q}$ are composable for conjunction if the sets $\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}$ and $\mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}}$ are disjoint.

Definition 6. *Let $\mathcal{P}$ and $\mathcal{Q}$ be components composable for conjunction, i.e., such that the sets $\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}$ and $\mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}}$ are disjoint. Then $\mathcal{P} \wedge \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}}, T_{\mathcal{P}\wedge\mathcal{Q}}, F_{\mathcal{P}\wedge\mathcal{Q}} \rangle$, where:*

- $\mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}} = \mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}$
- $\mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}} = \mathcal{A}^O_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{Q}}$
- $T_{\mathcal{P}\wedge\mathcal{Q}} = (T_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})) \cap (T_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}}))$
- $F_{\mathcal{P}\wedge\mathcal{Q}} = (F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})) \cap (F_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}}))$.

The T and F sets are defined such that any trace in the conjunction is a trace of both $\mathcal{P}$ and $\mathcal{Q}$, unless if there is an input along the trace that does not belong in the alphabet of one of the components (say $\mathcal{Q}$). On encountering such an input, the remainder of the trace would be in $T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}}$, which has the effect of leaving the behaviour of $\mathcal{P}$ unconstrained.

Lemma 3. *Conjunction is associative, commutative and idempotent.*

Proof. Obvious, given the algebraic properties of the set operations. □

The following theorem demonstrates that conjunction really does correspond to the meet operator, and that it is monotonic under refinement, subject to composability.

Theorem 2. *Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$, be components composable for conjunction. Then:*

- $\mathcal{P} \wedge \mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ *and* $\mathcal{P} \wedge \mathcal{Q} \sqsubseteq_{imp} \mathcal{Q}$
- $\mathcal{R} \sqsubseteq_{imp} \mathcal{P}$ *and* $\mathcal{R} \sqsubseteq_{imp} \mathcal{Q}$ *implies* $\mathcal{R} \sqsubseteq_{imp} \mathcal{P} \wedge \mathcal{Q}$
- $\mathcal{P}' \sqsubseteq_{imp} \mathcal{P}$ *and* $\mathcal{Q}' \sqsubseteq_{imp} \mathcal{Q}$ *implies* $\mathcal{P}' \wedge \mathcal{Q}' \sqsubseteq_{imp} \mathcal{P} \wedge \mathcal{Q}$.

Proof. For the first claim, we consider just inconsistent trace containment (the proof for observable traces being similar) on $\mathcal{P} \wedge \mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$. Let $t \in F_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})} \cap \mathcal{A}^*_{\mathcal{P}}$. Then there exists t' a prefix of t and $t'' \in (\mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}})^*$, such that $t't'' \in F_{\mathcal{P}\wedge\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{P}}$. By the definition of conjunction, we have $t't'' \in F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})$ and so $t't'' \in F_{\mathcal{E}(\mathcal{P})}$, given $t't'' \in \mathcal{A}^*_{\mathcal{P}}$.

For the second claim, we again show the containment on inconsistent traces, as the proof for the observable traces is near identical. Let $t \in F_{\mathcal{E}(\mathcal{R})} \cap \mathcal{A}^*_{\mathcal{P}\wedge\mathcal{Q}}$. Then without loss of generality, either $t \in \mathcal{A}^*_{\mathcal{P}}$, which from $\mathcal{R} \sqsubseteq_{imp} \mathcal{P}$ implies $t \in F_{\mathcal{E}(\mathcal{P})}$, or there is a prefix $t'a$ of t such that $t' \in \mathcal{A}^*_{\mathcal{P}}$ and $a \in$

$\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}_{\mathcal{P}}$. By the induction hypothesis on the strict prefix t', it follows that $t' \in T_{\mathcal{E}(\mathcal{P})}$ and so $t'a \in T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}}$, hence $t \in T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}}$. A similar argument can be applied for $\mathcal{R} \sqsubseteq_{imp} \mathcal{Q}$, and so $t \in F_{\mathcal{P} \wedge \mathcal{Q}}$ as required.

For the third claim, we know by the first part that $\mathcal{P}' \wedge \mathcal{Q}' \sqsubseteq_{imp} \mathcal{P}'$ and $\mathcal{P}' \wedge \mathcal{Q}' \sqsubseteq_{imp} \mathcal{Q}'$, from which $\mathcal{P}' \wedge \mathcal{Q}' \sqsubseteq_{imp} \mathcal{P}$ and $\mathcal{P}' \wedge \mathcal{Q}' \sqsubseteq_{imp} \mathcal{Q}$ can be deduced by transitivity. The result now follows by the second claim. Note that the compatibility conditions for transitivity may not hold, but this does not matter, since the problematic cases are when $\mathcal{A}^I_{\mathcal{Q}'} \cap (\mathcal{A}^O_{\mathcal{P}} \setminus \mathcal{A}^O_{\mathcal{Q}})$ or $\mathcal{A}^I_{\mathcal{P}'} \cap (\mathcal{A}^O_{\mathcal{Q}} \setminus \mathcal{A}^O_{\mathcal{P}})$ are non-empty. To circumvent the problem, for each of $\mathcal{P}$ and $\mathcal{Q}$ it is possible to construct components $\mathcal{P}''$ and $\mathcal{Q}''$ that have output sets $\mathcal{A}^O_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{Q}}$, obtained by deleting all traces containing outputs not in this set. Then it certainly holds that $\mathcal{P}' \wedge \mathcal{Q}' \sqsubseteq_{imp} \mathcal{P}''$ and $\mathcal{P}' \wedge \mathcal{Q}' \sqsubseteq_{imp} \mathcal{Q}''$, from which it is straightforward to show, by the definition of conjunction, that $\mathcal{P} \wedge \mathcal{Q} = \mathcal{P}'' \wedge \mathcal{Q}''$. □

Example 3. *To demonstrate conjunction, we consider a device that is capable of printing and faxing documents. The behaviour of this device is shown in Figure 2. Note how this device is capable of* print*ing multiple documents after having received* job_details *(indicated by the self-loop labelled with* print*).*

The conjunction of the original multi-function device (capable of printing and scanning, shown in Figure 1) along with this new printing/faxing device is shown in Figure 3. The resulting device is responsive to the inputs that can be issued for each of the separate devices, but is only willing to perform functions that can be executed by both. Therefore, the resulting device is unable to scan *or* fax *documents, even though it can be placed in these modes. Moreover, the device is only able to* print *a single document after having received* job_details*. Such behaviour may seem unnecessarily restrictive and undesirable; however, the resulting device is the most general that can be used safely in place of the original printing/scanning device and the printing/faxing device. Consequently, the resulting device can only introduce communication mismatches that both of the original devices can introduce.*

One reason why the conjunction in Figure 3 is so restrictive is that it cannot perform any output action that is not in the interface of both conjuncts. If we improve on this situation by extending the set of actions of the device in Figure 1 with fax_mode *and* fax*, and extending the set of actions of the device in Figure 2 with* scan_mode *and* scan*, so that the components to be conjoined have identical interfaces, then the conjunction is a component as*

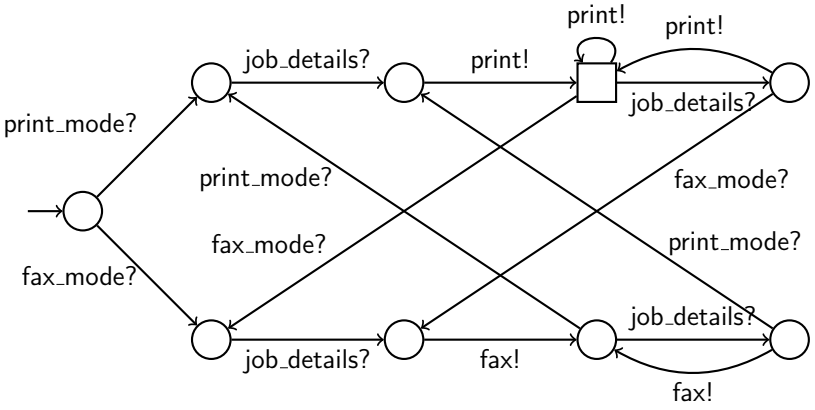

Figure 2: A printing and faxing device.

shown in Figure 4. This device is capable of scan*ning and* fax*ing documents, but cannot be placed in* scan_mode *after it has been placed in* fax_mode *and vice versa, although it can still be switched into* print_mode *and back.*

We remark that if, instead, we used conjunction defined as the intersection of behaviours (i.e. synchronous parallel, as in [25]), this would yield a device that cannot be used safely in place of either. The problem is that the behaviour would be unspecified when the device is placed in either scan_mode *or* fax_mode*, which means it will not work in any environment compatible with the printing/scanning device, nor the printing/faxing device.*

2.4. Disjunction

Disjunction is the dual of conjunction, so corresponds to the join operator on the refinement preorder. Therefore, the disjunction of a collection of components is the finest component that they each refine, meaning that the disjunction will work in environments safe for both of its operands. Composability of components under disjunction is the same as for conjunction.

Definition 7. *Let $\mathcal{P}$ and $\mathcal{Q}$ be components composable for disjunction, i.e., such that the sets $\mathcal{A}^I_\mathcal{P} \cup \mathcal{A}^I_\mathcal{Q}$ and $\mathcal{A}^O_\mathcal{P} \cup \mathcal{A}^O_\mathcal{Q}$ are disjoint. Then $\mathcal{P} \vee \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}}, T_{\mathcal{P}\vee\mathcal{Q}}, F_{\mathcal{P}\vee\mathcal{Q}} \rangle$, where:*

- $\mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}} = \mathcal{A}^I_\mathcal{P} \cap \mathcal{A}^I_\mathcal{Q}$
- $\mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}} = \mathcal{A}^O_\mathcal{P} \cup \mathcal{A}^O_\mathcal{Q}$

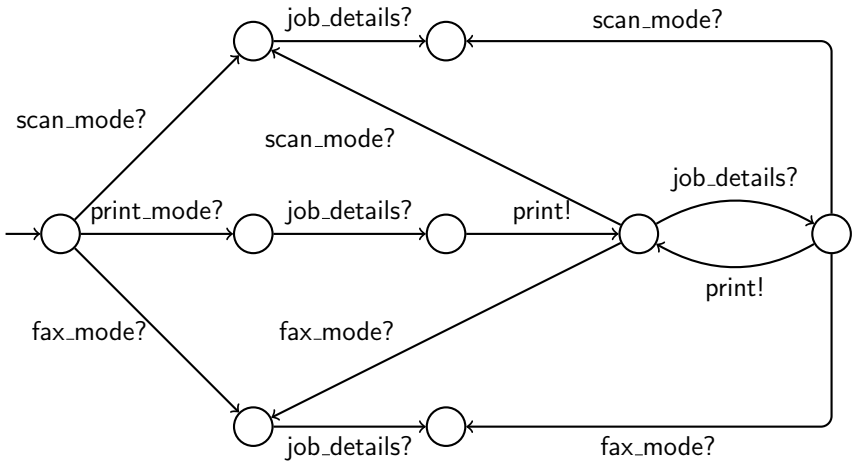

Figure 3: The conjunction of the printing/scanning and printing/faxing devices.

- $T_{\mathcal{P}\vee\mathcal{Q}} = [(T_{\mathcal{P}} \cup T_{\mathcal{Q}}) \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}] \cup F_{\mathcal{P}\vee\mathcal{Q}}$
- $F_{\mathcal{P}\vee\mathcal{Q}} = [(F_{\mathcal{P}} \cup F_{\mathcal{Q}}) \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}] \cdot \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$.

Essentially, as the disjunction should be refined by its arguments, the behaviours of $\mathcal{P}$ and $\mathcal{Q}$ should be contained within the behaviour of $\mathcal{P} \vee \mathcal{Q}$. Similarly, if a trace is inconsistent in one of $\mathcal{P}$ or $\mathcal{Q}$, then it must also be inconsistent within the disjunction.

Lemma 4. *Disjunction is associative, commutative and idempotent.*

Proof. Commutativity and idempotence are trivial. For associativity, we show that $F_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})} = F_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}}$, since the T-set equivalence follows by the same reasoning.

$$\begin{aligned}
F_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})} &= [(F_{\mathcal{P}} \cup F_{\mathcal{Q}\vee\mathcal{R}}) \cap \mathcal{A}^*_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})}] \cdot \mathcal{A}^*_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})} \\
&= [(F_{\mathcal{P}} \cup [(F_{\mathcal{Q}} \cup F_{\mathcal{R}}) \cap \mathcal{A}^*_{\mathcal{Q}\vee\mathcal{R}}] \cdot \mathcal{A}^*_{\mathcal{Q}\vee\mathcal{R}}) \cap \mathcal{A}^*_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})}] \cdot \mathcal{A}^*_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})} \\
&= [(F_{\mathcal{P}} \cup (F_{\mathcal{Q}} \cup F_{\mathcal{R}})) \cap \mathcal{A}^*_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})}] \cdot \mathcal{A}^*_{\mathcal{P}\vee(\mathcal{Q}\vee\mathcal{R})} \\
&= [((F_{\mathcal{P}} \cup F_{\mathcal{Q}}) \cup F_{\mathcal{R}}) \cap \mathcal{A}^*_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}}] \cdot \mathcal{A}^*_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}} \\
&= [([(F_{\mathcal{P}} \cup F_{\mathcal{Q}}) \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}] \cdot \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}} \cup F_{\mathcal{R}}) \cap \mathcal{A}^*_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}}] \cdot \mathcal{A}^*_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}} \\
&= [(F_{\mathcal{P}\vee\mathcal{Q}} \cup F_{\mathcal{R}}) \cap \mathcal{A}^*_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}}] \cdot \mathcal{A}^*_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}} \\
&= F_{(\mathcal{P}\vee\mathcal{Q})\vee\mathcal{R}}
\end{aligned}$$

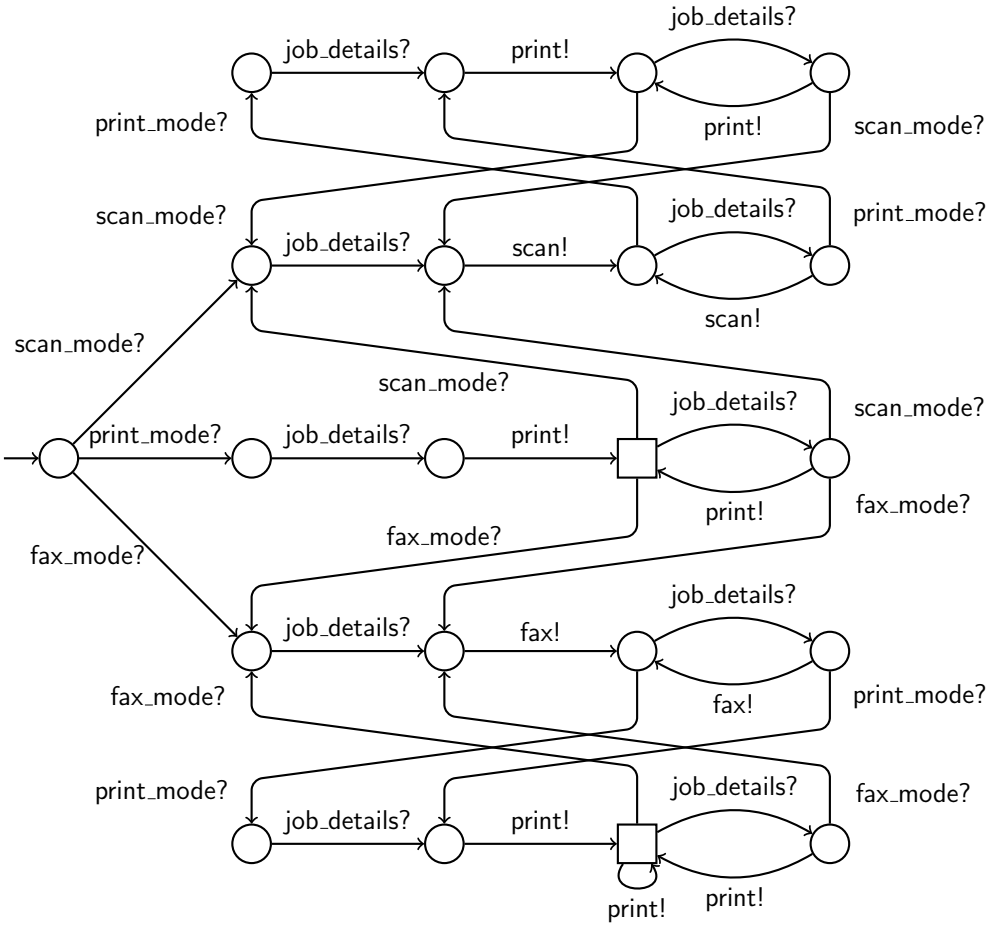

Figure 4: The conjunction of the printing/scanning and printing/faxing devices when the components have identical interfaces incorporating all actions.

□

As for conjunction, disjunction has an analogous set of algebraic properties, obtained by reversing the direction of refinement.

Theorem 3. *Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$, be components composable for disjunction. Then:*

- $\mathcal{P} \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$ *and* $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$
- $\mathcal{P} \sqsubseteq_{imp} \mathcal{R}$ *and* $\mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$ *implies* $\mathcal{P} \vee \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$
- $\mathcal{P}' \sqsubseteq_{imp} \mathcal{P}$ *and* $\mathcal{Q}' \sqsubseteq_{imp} \mathcal{Q}$ *implies* $\mathcal{P}' \vee \mathcal{Q}' \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$.

Proof. For the first claim, suppose $t \in F_{\mathcal{E}(\mathcal{P})} \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$. Then there exists a prefix t' of t and a trace $t'' \in (\mathcal{A}^O_{\mathcal{P}})^*$ such that $t't'' \in F_{\mathcal{P}}$. Necessarily, $t't'' \in \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$, and so $t't'' \in F_{\mathcal{P}\vee\mathcal{Q}}$. This implies $t \in F_{\mathcal{E}(\mathcal{P}\vee\mathcal{Q})}$ as required. The T-set containment is similar, and so $\mathcal{P} \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$. Equivalently, it is straightforward to show $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$.

For the second claim, suppose $t \in F_{\mathcal{E}(\mathcal{P}\vee\mathcal{Q})} \cap \mathcal{A}^*_{\mathcal{R}}$. Then there exists t', a prefix of t and $t'' \in (\mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}})^*$, such that $t't'' \in F_{\mathcal{P}\vee\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{R}}$ and, without loss of generality, $t't'' \in (F_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}) \cdot \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$. Therefore there is a prefix t_p of $t't''$ such that $t_p \in F_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{R}}$. From $\mathcal{P} \sqsubseteq_{imp} \mathcal{R}$, it follows that $t_p \in F_{\mathcal{E}(\mathcal{R})}$ and so $t \in F_{\mathcal{E}(\mathcal{R})}$ as required. Showing the T-set containment is similar.

For the third claim, we know by the first part that $\mathcal{P} \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$ and $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$, from which $\mathcal{P}' \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$ and $\mathcal{Q}' \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$ can be deduced by transitivity (assuming the compatibility constraints are satisfied). The result now follows by the second claim. When the compatibility constraints are not satisfied, it must be because $\mathcal{A}^I_{\mathcal{P}'} \cap \mathcal{A}^O_{\mathcal{Q}}$ or $\mathcal{A}^I_{\mathcal{Q}'} \cap \mathcal{A}^O_{\mathcal{P}}$ is non-empty. It is possible to construct components $\mathcal{P}''$ and $\mathcal{Q}''$ with input actions $\mathcal{A}^I_{\mathcal{P}'} \cap \mathcal{A}^I_{\mathcal{Q}'}$, obtained from $\mathcal{P}'$ and $\mathcal{Q}'$ by deleting all traces containing an input not in $\mathcal{A}^I_{\mathcal{P}'} \cap \mathcal{A}^I_{\mathcal{Q}'}$. Then certainly $\mathcal{P}'' \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$ and $\mathcal{Q}'' \sqsubseteq_{imp} \mathcal{P} \vee \mathcal{Q}$, from which the result can be deduced by observing that $\mathcal{P}' \vee \mathcal{Q}' = \mathcal{P}'' \vee \mathcal{Q}''$. □

Example 4. *A user wishing to use a multi-function device is non-deterministically allocated the printing/scanning device (Figure 1) or the printing/faxing device (Figure 2). The most general behaviour allowed by the user (such that communication mismatches are not introduced) is obtained by inverting the inputs and outputs on the disjunction of the two devices. The disjunction is shown in Figure 5.*

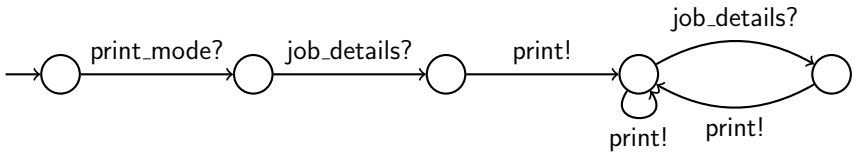

Figure 5: The disjunction of the printing/scanning and printing/faxing devices.

2.5. Hiding

We introduce hiding to support abstraction for hierarchical development. Hiding is a unary operator on components that has the effect of contracting the interface by removing an action. Taking intuition from a simple analogy in which inputs correspond to buttons and outputs correspond to lights, the resulting behaviour of a component under hiding of action b is as follows:

- If b is an input, then the b-button will never be pressed. This means that no behaviour is observable beyond a b on a trace, so all traces should be pruned on encountering a b.
- If b is an output, then hiding suppresses the visibility of the b-light. The component should thus silently skip over b, which corresponds to projecting out b from all traces.

From this, we give the formal definition, which is dependent on the type of action to be hidden.

Definition 8. *Let $\mathcal{P}$ be a component and let b be an action. The hiding of b in $\mathcal{P}$ is a component $\mathcal{P}/b = \langle \mathcal{A}^I_{\mathcal{P}/b}, \mathcal{A}^O_{\mathcal{P}/b}, T_{\mathcal{P}/b}, F_{\mathcal{P}/b} \rangle$, where:*

- $\mathcal{A}^I_{\mathcal{P}/b} = \mathcal{A}^I_{\mathcal{P}} \setminus \{b\}$
- $\mathcal{A}^O_{\mathcal{P}/b} = \mathcal{A}^O_{\mathcal{P}} \setminus \{b\}$
- $T_{\mathcal{P}/b} = \begin{cases} T_{\mathcal{P}} \upharpoonright \mathcal{A}_{\mathcal{P}/b} & \text{if } b \in \mathcal{A}^O_{\mathcal{P}} \\ T_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}/b} & \text{otherwise} \end{cases}$
- $F_{\mathcal{P}/b} = \begin{cases} F_{\mathcal{P}} \upharpoonright \mathcal{A}_{\mathcal{P}/b} & \text{if } b \in \mathcal{A}^O_{\mathcal{P}} \\ F_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}/b} & \text{otherwise.} \end{cases}$

The soundness of this definition requires careful consideration when b is an output. For a trace $tb \in T_{\mathcal{P}}$ and input $a \in \mathcal{A}^I_{\mathcal{P}}$, observe that ta is a safe trace of $\mathcal{P}/b$ (i.e., $ta \in T_{\mathcal{P}/b} \setminus F_{\mathcal{P}/b}$) iff both ta and tba are safe traces of $\mathcal{P}$. Taking intuition from b being a hidden light, this behaviour is correct since it cannot be known precisely when the light will illuminate, so it is only safe for the environment to issue the input a after t if the component is willing to accept a both before and after the light has been silently illuminated.

Theorem 4. *Let $\mathcal{P}$ and $\mathcal{Q}$ be components and let b be an action. If $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$, then $\mathcal{Q}/b \sqsubseteq_{imp} \mathcal{P}/b$.*

Proof. Begin by noting that $\mathcal{E}(\mathcal{Q})/b = \mathcal{E}(\mathcal{Q}/b)$ (and similarly for $\mathcal{P}$). In the case that $b \in \mathcal{A}^I_{\mathcal{Q}}$, let $t \in F_{\mathcal{E}(\mathcal{Q}/b)} \cap \mathcal{A}^*_{\mathcal{P}/b}$. Then $t \in F_{\mathcal{E}(\mathcal{Q})/b}$ and so $t \in F_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}^*_{\mathcal{Q}/b} \cap \mathcal{A}^*_{\mathcal{P}/b}$. By $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ we have $t \in F_{\mathcal{E}(\mathcal{P})} \cap \mathcal{A}^*_{\mathcal{P}/b}$. This means that $t \in F_{\mathcal{E}(\mathcal{P}/b)}$ as required. The observable trace containment can be shown similarly. Note that this case also applies when $b \notin \mathcal{A}_{\mathcal{P}} \cup \mathcal{A}_{\mathcal{Q}}$.

For the case when $b \in \mathcal{A}^O_{\mathcal{P}}$, assume that $t \in F_{\mathcal{E}(\mathcal{Q}/b)} \cap \mathcal{A}^*_{\mathcal{P}/b}$, from which we know $t \in F_{\mathcal{E}(\mathcal{Q})/b}$. Suppose there is a $t' \in F_{\mathcal{E}(\mathcal{Q})}$ such that $t' \upharpoonright \mathcal{A}_{\mathcal{Q}/b} = t$. Then $t' \in \mathcal{A}^*_{\mathcal{P}}$ and so from $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ it follows that $t' \in F_{\mathcal{E}(\mathcal{P})}$. As $t' \in \mathcal{A}^*_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{Q}}$, it follows that $t' \upharpoonright \mathcal{A}_{\mathcal{P}/b} = t$. Hence $t \in F_{\mathcal{E}(\mathcal{P})/b}$, which implies $t \in F_{\mathcal{E}(\mathcal{P}/b)}$ as required. The T-set containment is similar. □

Example 5. *Disaster strikes and the* Device *becomes broken such that it will no longer* scan *documents (depicted as* BrokenDevice *in Figure 6). As a result, the* BrokenDevice *should not be placed in* scan_mode*. The updated behaviour of the device is given by* BrokenDevice / scan_mode*, as shown in Figure 7. The resulting component model contracts the interface of the* BrokenDevice *by being indifferent to* scan_mode *requests.*

2.6. Quotient

The final operation that we consider is that of quotient, which provides functionality to synthesise components from a global specification and partial implementation. Given a component representing a system $\mathcal{R}$, together with an implementation of one component $\mathcal{P}$ in the system $\mathcal{R}$, the quotient yields the coarsest component for the remaining part of $\mathcal{R}$ to be implemented. Thus, the parallel composition of the quotient with $\mathcal{P}$ should be a refinement of $\mathcal{R}$. Therefore, quotient can be thought of as the adjoint of parallel composition.

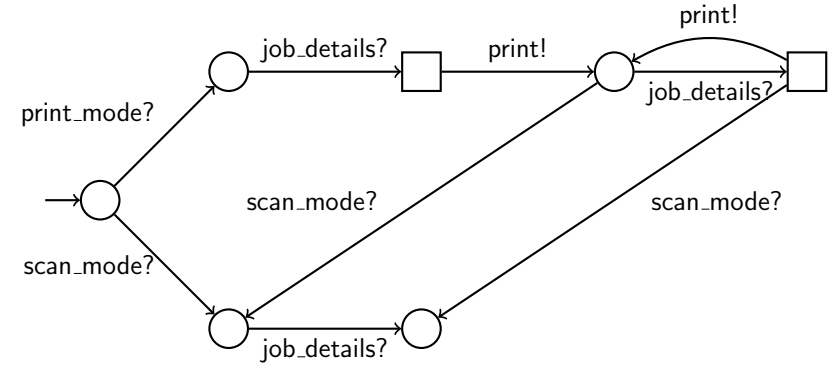

Figure 6: BrokenDevice without the ability to scan.

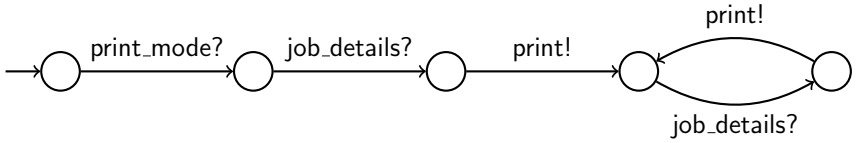

Figure 7: BrokenDevice after hiding the scan_mode functionality.

A necessary condition for the existence of the quotient is that $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{R}}$, otherwise refinement will fail on the alphabet containment checks.

Definition 9. *Let $\mathcal{P}$ and $\mathcal{R}$ be components such that $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{R}}$. The quotient of $\mathcal{P}$ from $\mathcal{R}$ is the component $\mathcal{R}/\mathcal{P}$ with signature $\langle \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}, \mathcal{A}^O_{\mathcal{R}/\mathcal{P}}, T_{\mathcal{R}/\mathcal{P}}, F_{\mathcal{R}/\mathcal{P}} \rangle$, where:*

- $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^I_{\mathcal{R}} \setminus \mathcal{A}^I_{\mathcal{P}}$
- $\mathcal{A}^O_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^O_{\mathcal{R}} \setminus \mathcal{A}^O_{\mathcal{P}}$
- $T_{\mathcal{R}/\mathcal{P}}$ *is the largest prefix-closed and input-receptive subset of* $\{t \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \upharpoonright \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}} \implies t' \in T_{\mathcal{E}(\mathcal{R})}\} \cap$ $\{t \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \upharpoonright \mathcal{A}_{\mathcal{P}} \in F_{\mathcal{P}} \implies t' \in F_{\mathcal{E}(\mathcal{R})}\}$
- $F_{\mathcal{R}/\mathcal{P}} = \{t \in T_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \upharpoonright \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}} \implies t' \in F_{\mathcal{E}(\mathcal{R})}\}$.

Explaining the intuition behind the definition, observe that whenever $\mathcal{R}$ is inconsistent, the parallel composition of $\mathcal{P}$ and the quotient can be inconsistent, so the quotient itself can be inconsistent. Similarly, if a trace is not in $\mathcal{P}$, then it will not be encountered in the composition $\mathcal{P} \mid\mid \mathcal{R}/\mathcal{P}$, hence it should be inconsistent in the quotient (so that we obtain the least refined solution). These two conditions correlate with $t \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P} \implies t \in F_{\mathcal{E}(\mathcal{R})}$ in the definition of $F_{\mathcal{R}/\mathcal{P}}$.

If $\mathcal{P}$ is inconsistent on a trace t when $\mathcal{R}$ is not inconsistent, then the parallel composition of $\mathcal{P}$ and the quotient would be inconsistent if t is in the quotient. Similarly, if t is a trace of $\mathcal{P}$, but not of $\mathcal{R}$, then the parallel composition would have a behaviour that is not in $\mathcal{R}$, if t were included in the quotient. Both of these situations are problematic, since the composition of $\mathcal{P}$ and the quotient would not be a refinement of $\mathcal{R}$. Consequently, the quotient must suppress the last output on its behaviour of this trace, so that the composition can never encounter the inconsistency (or additional behaviour) that $\mathcal{P}$ will introduce. In our definition, this correlates with the requirement that $T_{\mathcal{R}/\mathcal{P}}$ is the largest input-receptive set satisfying the conditions that $t \upharpoonright \mathcal{A}_\mathcal{P} \in F_\mathcal{P} \implies t \in F_{\mathcal{E}(\mathcal{R})}$ and $t \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P} \implies t \in T_{\mathcal{E}(\mathcal{R})}$.

Although $\mathcal{R}/\mathcal{P}$ is always defined when $\mathcal{A}^O_\mathcal{P} \subseteq \mathcal{A}^O_\mathcal{R}$, it may not be a realisable component, even if both $\mathcal{R}$ and $\mathcal{P}$ are realisable. Unfortunately, there is no syntactic check on the interfaces of $\mathcal{R}$ and $\mathcal{P}$ that can determine whether $\mathcal{R}/\mathcal{P}$ is realisable or not. This can only be inferred by examining the behaviours of $\mathcal{R}$ and $\mathcal{P}$.

Theorem 5. *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components. Then $\mathcal{P} \mid\mid \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$ iff:*

- *$\mathcal{R}/\mathcal{P}$ is defined (i.e., $\mathcal{A}^O_\mathcal{P} \subseteq \mathcal{A}^O_\mathcal{R}$)*
- *$\mathcal{P} \mid\mid (\mathcal{R}/\mathcal{P}) \sqsubseteq_{imp} \mathcal{R}$*
- *$\mathcal{A}^I_\mathcal{Q} = \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$ implies $\mathcal{Q} \sqsubseteq_{imp} \mathcal{R}/\mathcal{P}$.*

Proof. For the first claim, if $\mathcal{P} \mid\mid \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$, then $\mathcal{A}^O_{\mathcal{P}\mid\mid\mathcal{Q}} \subseteq \mathcal{A}^O_\mathcal{R}$. As $\mathcal{A}^O_{\mathcal{P}\mid\mid\mathcal{Q}} = \mathcal{A}^O_\mathcal{P} \cup \mathcal{A}^O_\mathcal{Q}$, it follows that $\mathcal{A}^O_\mathcal{P} \subseteq \mathcal{A}^O_\mathcal{R}$ i.e., the quotient is defined. Instead, if $\mathcal{R}/\mathcal{P}$ is defined, then $\mathcal{A}^O_\mathcal{P} \subseteq \mathcal{A}^O_\mathcal{R}$. Taking $\mathcal{Q} = \langle \mathcal{A}^I_\mathcal{R}, \mathcal{A}^O_\mathcal{R} \setminus \mathcal{A}^O_\mathcal{P}, \emptyset, \emptyset \rangle$ gives $\mathcal{P} \mid\mid \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$.

For the second claim, let $t \in F_{\mathcal{E}(\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P}))} \cap \mathcal{A}^*_\mathcal{R}$. Then there exists a prefix t' of t and $t'' \in (\mathcal{A}^O_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})})^*$ such that $t't'' \in F_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})} \cap \mathcal{A}^*_\mathcal{R}$. Without loss of generality, suppose there is no prefix of $t't''$ in $F_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})}$. Then either $t't'' \upharpoonright$

$\mathcal{A}_\mathcal{P} \in F_\mathcal{P}$ and $t't'' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} \in T_{\mathcal{R}/\mathcal{P}}$, or $t't'' \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P}$ and $t't'' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} \in F_{\mathcal{R}/\mathcal{P}}$. If the former holds, then $t't'' \in F_{\mathcal{E}(\mathcal{R})}$ by the definition of $T_{\mathcal{R}/\mathcal{P}}$, which implies $t \in F_{\mathcal{E}(\mathcal{R})}$. In the case of the latter, if follows by the definition of $F_{\mathcal{R}/\mathcal{P}}$ that $t't'' \in F_{\mathcal{E}(\mathcal{R})}$, from which it can be deduced that $t \in F_{\mathcal{E}(\mathcal{R})}$. Now suppose that $t \in (T_{\mathcal{P}||(\mathcal{R}/\mathcal{P})} \setminus F_{\mathcal{E}(\mathcal{P}||(\mathcal{R}/\mathcal{P}))}) \cap \mathcal{A}^*_\mathcal{R}$. Then it follows that $t \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P}$ and $t \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} \in T_{\mathcal{R}/\mathcal{P}}$. By the definition of $T_{\mathcal{R}/\mathcal{P}}$, it follows that $t \in T_{\mathcal{E}(\mathcal{R})}$ as required.

For the third claim, first suppose $t \in F_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$. Then there exists a prefix t' of t and $t'' \in (\mathcal{A}^O_\mathcal{Q})^*$ such that $t't'' \in F_\mathcal{Q}$. Note that $t't'' \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$. Let $t''' \in \mathcal{A}^*_\mathcal{R}$ be an arbitrary trace such that $t''' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t't''$. If $t''' \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P}$, then $t''' \in F_{\mathcal{E}(\mathcal{R})}$, since $\mathcal{P} \,||\, \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$. Therefore, by the arbitrariness of t''', it follows that $t't'' \in F_{\mathcal{R}/\mathcal{P}}$ unless $t't'' \notin T_{\mathcal{R}/\mathcal{P}}$ (which can only be if prefix-closure or input-receptiveness does not hold, but this would imply $t't'' \notin F_\mathcal{Q}$). Hence $t \in F_{\mathcal{E}(\mathcal{R}/\mathcal{P})}$ as required. Now suppose that $t \in (T_\mathcal{Q} \setminus F_{\mathcal{E}(\mathcal{Q})}) \cap \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$. Again, let $t''' \in \mathcal{A}^*_\mathcal{R}$ be an arbitrary trace such that $t''' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t$. If $t''' \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P}$, then $t''' \in T_{\mathcal{E}(\mathcal{R})}$, and if $t''' \upharpoonright \mathcal{A}_\mathcal{P} \in F_\mathcal{P}$, then $t''' \in F_{\mathcal{E}(\mathcal{R})}$, since $\mathcal{P} \,||\, \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$. By the arbitrariness of t''', it follows that $t \in T_{\mathcal{R}/\mathcal{P}}$. □

This definition of quotient generalises that supplied in [6] and[4], both of which require that the interface of $\mathcal{R}/\mathcal{P}$ synchronises with all actions of $\mathcal{P}$. Although in this article we take $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^I_\mathcal{R} \setminus \mathcal{A}^I_\mathcal{P}$, our definition works for any set such that $\mathcal{A}^I_\mathcal{R} \setminus \mathcal{A}^I_\mathcal{P} \subseteq \mathcal{A}^I_{\mathcal{R}/\mathcal{P}} \subseteq \mathcal{A}_\mathcal{R}$, with the results of Theorem 5 continuing to hold. In other words, the quotient operation can be parameterised on the set $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$ of input actions of $\mathcal{R}/\mathcal{P}$. For any such choice of $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$, the construction of $T_{\mathcal{R}/\mathcal{P}}$ and $F_{\mathcal{R}/\mathcal{P}}$ for this extended set of inputs remains unchanged from Definition 9 (having redefined $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$). Consequently, we can take $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^I_\mathcal{R} \cup \mathcal{A}^O_\mathcal{P}$, which allows the interface of the quotient to observe all actions of $\mathcal{P}$ and hence capture more specific behaviours. In general, it is not possible to start with the original quotient $\mathcal{R}/\mathcal{P}$ (having inputs $\mathcal{A}^I_\mathcal{R} \setminus \mathcal{A}^I_\mathcal{P}$) and refine it to a component $\mathcal{Q}$ over the extended set of inputs such that $\mathcal{P} \,||\, \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$ can be inferred, since parallel composition has interface restrictions for monotonicity to hold (cf. Theorem 1).

The next theorem shows that quotient is well-behaved with respect to refinement.

Theorem 6. *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components such that $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$.*

- *If $\mathcal{Q}/\mathcal{R}$ is defined and $\mathcal{A}^I_{\mathcal{R}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$, then $\mathcal{Q}/\mathcal{R} \sqsubseteq_{imp} \mathcal{P}/\mathcal{R}$.*
- *If $\mathcal{R}/\mathcal{P}$ is defined and $(\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{P}}) \cap \mathcal{A}_{\mathcal{R}} = \emptyset$, then $\mathcal{R}/\mathcal{Q} \sqsupseteq_{imp} \mathcal{R}/\mathcal{P}$.*

Proof. For the first property, note that definedness of $\mathcal{Q}/\mathcal{R}$ implies definedness of $\mathcal{P}/\mathcal{R}$. Consequently, $\mathcal{R} \parallel (\mathcal{Q}/\mathcal{R}) \sqsubseteq_{imp} \mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$. The constraint $\mathcal{A}^I_{\mathcal{R}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$ ensures that transitivity holds, from which we derive $\mathcal{R} \parallel (\mathcal{Q}/\mathcal{R}) \sqsubseteq_{imp} \mathcal{P}$. Hence $\mathcal{Q}/\mathcal{R} \sqsubseteq_{imp} \mathcal{P}/\mathcal{R}$ by Theorem 5.

For the second property, definedness of $\mathcal{R}/\mathcal{P}$ implies definedness of $\mathcal{R}/\mathcal{Q}$. From $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$, we obtain $\mathcal{Q} \parallel (\mathcal{R}/\mathcal{P}) \sqsubseteq_{imp} \mathcal{P} \parallel (\mathcal{R}/\mathcal{P})$ by Theorem 1 (the conditions of which are satisfied by $(\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{P}}) \cap \mathcal{A}_{\mathcal{R}} = \emptyset$). By Theorem 5 we know $\mathcal{P} \parallel (\mathcal{R}/\mathcal{P}) \sqsubseteq_{imp} \mathcal{R}$, and so we obtain $\mathcal{Q} \parallel (\mathcal{R}/\mathcal{P}) \sqsubseteq_{imp} \mathcal{R}$ by transitivity (Lemma 1), given that $(\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{P}}) \cap \mathcal{A}_{\mathcal{R}} = \emptyset$ ensures that action types are not mixed. Finally, by Theorem 5, it follows that $\mathcal{R}/\mathcal{Q}$ is the minimal solution to $\mathcal{Q} \parallel X \sqsubseteq_{imp} \mathcal{R}$, and so $\mathcal{R}/\mathcal{P} \sqsubseteq_{imp} \mathcal{R}/\mathcal{Q}$, given that $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^I_{\mathcal{R}/\mathcal{Q}}$. □

Example 6. *To demonstrate quotient, we assume that the action* job_details *can encode two types of behaviour, depending on the mode of the device. When* Device *is in* print_mode*, the* job_details *should encode information pertaining to printing, such as the document to be printed. Conversely, when* Device *is in* scan_mode*, the* job_details *should contain information indicative of scanning functionality, such as the resolution at which scanning must be performed. This essentially means that, after the* job_details *have been sent to* Device*, the device mode may not be changed until the current job has been* print*ed or* scan*ned. This constraint is represented by the component* Constraint *in Figure 8. The* Constraint *component is an observer that generates errors when bad sequences of actions are seen, which is why all actions are treated as inputs. The behaviour of the constrained device is given by* Device || Constraint*.*

The most general behaviour of a user that interacts with the constrained device is given by the quotient User2 = ErrorFree/(Device || Constraint) *(as depicted in Figure 9).* ErrorFree *is the component having a single state with a self-loop for each action (treated as an output). As* ErrorFree *does not possess any inconsistent states, the quotient operation guarantees that* User2 || Device || Constraint *is free of inconsistencies, hence* User2 || Device *conforms to the behaviour of* Constraint*.*

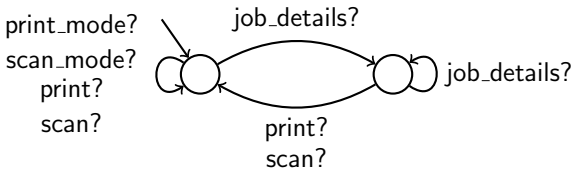

Figure 8: Constraint on job_details.

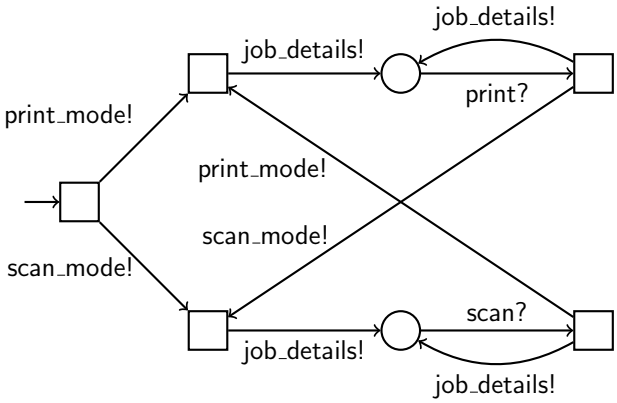

Figure 9: Component representing User2.

The computation of quotient can be automated. Applications of quotient to mediator synthesis were demonstrated by Inverardi and Tivoli [18], and Bennaceur et al. [3], with the latter presenting a prototype implementation.

2.7. Full Abstraction

In this section, we demonstrate that our refinement relation precisely characterises safe substitutivity of components, by means of a testing framework that places components in parallel with an arbitrary environment and checks for inconsistency. Based on this testing scenario, we show that $\equiv_{imp}$ is fully abstract for the full collection of operators in the specification theory.

Definition 10. *Let $\mathcal{P}$ and $\mathcal{Q}$ be components. Then $\mathcal{Q}$ is inconsistency substitutable for $\mathcal{P}$, denoted by $\mathcal{Q} \sqsubseteq^{F}_{imp} \mathcal{P}$, iff $\epsilon \in F_{\mathcal{E}(\mathcal{Q})}$ implies $\epsilon \in F_{\mathcal{E}(\mathcal{P})}$.*

From this definition, we can show that $\sqsubseteq_{imp}$ is the weakest preorder representing safe-substitutivity.

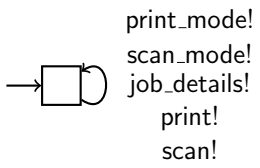

Figure 10: An ErrorFree component

Theorem 7. *Let $\mathcal{P}$ and $\mathcal{Q}$ be components such that $\mathcal{A}^I_\mathcal{P} \subseteq \mathcal{A}^I_\mathcal{Q}$, $\mathcal{A}^O_\mathcal{Q} \subseteq \mathcal{A}^O_\mathcal{P}$ and $\mathcal{A}^I_\mathcal{Q} \cap \mathcal{A}^O_\mathcal{P} = \emptyset$. Then:*

$$\mathcal{Q} \sqsubseteq_{imp} \mathcal{P} \text{ iff } \forall \mathcal{R} \cdot \mathcal{A}^O_\mathcal{R} = \mathcal{A}^I_\mathcal{P} \text{ and } \mathcal{A}^I_\mathcal{R} = \mathcal{A}^O_\mathcal{Q} \implies \mathcal{Q} \,||\, \mathcal{R} \sqsubseteq^F_{imp} \mathcal{P} \,||\, \mathcal{R}.$$

Proof. First suppose $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$. Then, from the constraint on the interface for $\mathcal{R}$, we have that $\mathcal{Q} \,||\, \mathcal{R} \sqsubseteq_{imp} \mathcal{P} \,||\, \mathcal{R}$ by Theorem 1, since the constraints for that Theorem are satisfied. Hence $\mathcal{Q} \,||\, \mathcal{R} \sqsubseteq^F_{imp} \mathcal{P} \,||\, \mathcal{R}$ as required.

For the other direction, suppose that $\mathcal{Q} \not\sqsubseteq_{imp} \mathcal{P}$. Then there exists a smallest t such that $t \in F_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}^*_\mathcal{P}$ and $t \notin F_{\mathcal{E}(\mathcal{P})}$, or $t \in T_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}^*_\mathcal{P}$ and $t \notin T_{\mathcal{E}(\mathcal{P})}$.

In the case of the former, it follows (by the minimality of t) that there exists $t' \in (\mathcal{A}^O_\mathcal{Q})^*$ such that $tt' \in F_\mathcal{Q}$, and, moreover, $tt' \in \mathcal{A}^*_\mathcal{P}$. Consequently, $tt' \in \mathcal{A}^*_\mathcal{R}$, so we construct an $\mathcal{R}$ such that $F_\mathcal{R} = \emptyset$ and $T_\mathcal{R}$ is the smallest set containing tt' that makes $\mathcal{R}$ a component. Now $tt' \in T_\mathcal{R}$ implies $tt' \in F_{\mathcal{Q}||\mathcal{R}}$, and so $\epsilon \in F_{\mathcal{E}(\mathcal{Q}||\mathcal{R})}$ given $tt' \in (\mathcal{A}^O_{\mathcal{Q}||\mathcal{R}})^*$. However, as $t \notin F_{\mathcal{E}(\mathcal{P})}$, it follows that $t \notin F_{\mathcal{E}(\mathcal{P}||\mathcal{R})}$, hence $\epsilon \notin F_{\mathcal{E}(\mathcal{P}||\mathcal{R})}$, which means $\mathcal{Q} \,||\, \mathcal{R} \not\sqsubseteq^F_{imp} \mathcal{P} \,||\, \mathcal{R}$ as required.

In the case of the latter, it is sufficient to consider $t \in T_\mathcal{Q} \cap \mathcal{A}^*_\mathcal{P}$. Note that $t \in \mathcal{A}^*_\mathcal{R}$ by definition, so we construct an $\mathcal{R}$ such that $F_\mathcal{R} = \{t'' \in \mathcal{A}^*_\mathcal{R} : t \text{ is a prefix of } t''\}$ and $T_\mathcal{R}$ is the least set making $\mathcal{R}$ a component. Therefore $t \in F_{\mathcal{Q}||\mathcal{R}}$, which yields $\epsilon \in F_{\mathcal{E}(\mathcal{Q}||\mathcal{R})}$ given $t \in (\mathcal{A}^O_{\mathcal{Q}||\mathcal{R}})^*$. However, as $t \notin T_{\mathcal{E}(\mathcal{P})}$, it follows that $t \notin T_{\mathcal{E}(\mathcal{P}||\mathcal{R})}$, hence $\epsilon \notin T_{\mathcal{E}(\mathcal{P}||\mathcal{R})}$. From this we obtain $\epsilon \notin F_{\mathcal{E}(\mathcal{P}||\mathcal{R})}$, so $\mathcal{Q} \,||\, \mathcal{R} \not\sqsubseteq^F_{imp} \mathcal{P} \,||\, \mathcal{R}$ as required. □

The conditions on the interfaces of $\mathcal{P}$ and $\mathcal{Q}$ are required for Theorem 7 to hold, since $\mathcal{Q} \,||\, \mathcal{R} \sqsubseteq_{imp} \mathcal{P} \,||\, \mathcal{R}$ does not imply that $\mathcal{A}^I_\mathcal{P} \subseteq \mathcal{A}^I_\mathcal{Q}$, $\mathcal{A}^O_\mathcal{Q} \subseteq \mathcal{A}^O_\mathcal{P}$ and $\mathcal{A}^I_\mathcal{Q} \cap \mathcal{A}^O_\mathcal{P} = \emptyset$.

We now elaborate on our claim that $\sqsubseteq_{imp}$ is the weakest preorder preserving substitutivity of components. Indeed, Theorem 7 supports this claim, but this is due to our formulation of an environment, in that we assume $\mathcal{A}^O_\mathcal{R} \cap$

$(\mathcal{A}^I_\mathcal{Q} \setminus \mathcal{A}^I_\mathcal{P}) = \emptyset$. If this constraint does not hold, then our refinement is not the weakest preorder preserving substitutivity. To see why, consider the components $\mathcal{P} = \langle \mathcal{A}^I, \mathcal{A}^O, \mathcal{A}^*, \emptyset \rangle$ and $\mathcal{Q} = \langle \mathcal{A}^I \dot\cup \{x\}, \mathcal{A}^O, (\mathcal{A} \dot\cup \{x\})^*, \mathcal{A}^* \uparrow \{x\} \rangle$ for which $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$. Then the component $\mathcal{R} = \langle \mathcal{A}^O, \mathcal{A}^I \dot\cup \{x\}, (\mathcal{A} \dot\cup \{x\})^*, \emptyset \rangle$ is an environment safe for $\mathcal{P}$ but not for $\mathcal{Q}$ (note that in Theorem 7 this $\mathcal{R}$ is not treated as an environment for $\mathcal{P}$).

Supposing that we wish $\mathcal{R}$ to be classified as a valid environment for $\mathcal{P}$, it becomes necessary to reformulate the definition of refinement in order to maintain the weakest substitutive preorder property. The necessary alteration essentially requires that $\mathcal{A}^I_\mathcal{Q} = \mathcal{A}^I_\mathcal{P}$. We choose not to adopt this convention, since our substitutive preorder would no longer be weaker than the alternating refinement defined by de Alfaro and Henzinger for interface automata. There is, of course, no reason why a user of our theory could not adopt this stronger requirement.

From our characterisation of $\sqsubseteq_{imp}$ as the weakest substitutive preorder, we obtain a full abstraction result for $\equiv_{imp}$ on the specification theory, with respect to checking of inconsistency equivalence $\equiv^F_{imp}$ (i.e., $\sqsubseteq^F_{imp} \cap \sqsupseteq^F_{imp}$). Our definition of full abstraction is taken from van Glabbeek [39] (Definition 16), which means that $\equiv_{imp}$ is the coarsest congruence for the operators of our specification theory with respect to simple inconsistency equivalence.

Corollary 1. *Substitutive equivalence $\equiv_{imp}$ is fully abstract for parallel composition, conjunction, disjunction, hiding and quotient with respect to observational equivalence of inconsistency.*

Proof. Note that, under $\equiv_{imp}$, none of the alphabet constraints (other than those for composability) are required for the compositionality results to hold in Theorems 1, 2, 3, 4 and 6. Consequently, $\equiv_{imp}$ is a congruence for all of the compositional operators. Taking this along with Theorem 7 shows that $\equiv_{imp}$ is the coarsest such equivalence with respect to observational equivalence of inconsistency. □

We do not obtain full abstraction for $\sqsubseteq_{imp}$, since the compositional operators do not form a pre-congruence under $\sqsubseteq_{imp}$, due to the compatibility constraints. The constraints are, however, automatically satisfied for $\equiv_{imp}$.

3. Extending the Component Theory: Preservation of Progress

A perceived shortcoming of interface automata (and hence our theory in Section 2) is that the principle of substitutivity requires a refining component to be no more expressive on the output it can produce, in comparison to the behaviour of the original. In fact, the most refined component will have an interface that is unwilling to produce any external stimuli whatsoever. Refinement resulting in absence of external behaviour is frequently seen in the literature, one such example being the trace semantics of CSP [17], in which every process can be refined by the deadlocked process STOP. Such refinements preserve safety, but they do not require any meaningful computation to be performed. To resolve this issue, the refinement relation should be adapted by instilling a notion of liveness/progress.

In this section, we adapt the substitutive refinement relation of Section 2.1 by forcing a refining component to make progress whenever the original can. Our choice of progress is based on the notion of *quiescence*; a trace is said to be quiescent just if it cannot be extended by an output. Quiescence differs from deadlock in that a deadlocked component is unwilling to accept any input (or produce any output), whereas a quiescent component may be able to accept input. The updated refinement relation requires substitutability, as in Section 2.1, but also that any non-quiescent trace of the original component is non-quiescent in the refining component. Our choice of quiescence, in place of fairness sets [37, 36], is motivated by the desire to utilise only finite-length traces, as in Section 2. In addition to quiescence, a component should not be allowed to make progress by performing an unbounded amount of internal computation. As a result, our refinement relation must also take into account the divergence of a component. Note that, in contrast to CSP [17], we do *not* require divergent traces to be extension closed.

The remainder of this section presents an updated component formulation, together with the formal definition of the substitutive and progress-sensitive refinement relation. Revised definitions for the compositional operators are presented, and the algebraic results are re-established.

Definition 11. *A* progress-sensitive component $\mathcal{P}$ *(henceforth referred to as a component) is a tuple* $\langle \mathcal{A}^I_{\mathcal{P}}, \mathcal{A}^O_{\mathcal{P}}, T_{\mathcal{P}}, F_{\mathcal{P}}, D_{\mathcal{P}}, K_{\mathcal{P}} \rangle$ *in which* $\langle \mathcal{A}^I_{\mathcal{P}}, \mathcal{A}^O_{\mathcal{P}}, T_{\mathcal{P}}, F_{\mathcal{P}} \rangle$ *is a component as in Definition 3, and:*

- $D_{\mathcal{P}}$ *is a set of extended divergent traces such that* $F_{\mathcal{P}} \subseteq D_{\mathcal{P}} \subseteq T_{\mathcal{P}}$
- $K_{\mathcal{P}}$ *is a set of extended quiescent traces such that* $\{t \in T_{\mathcal{P}} : \nexists o \in \mathcal{A}_{\mathcal{P}}^{O} \cdot to \in T_{\mathcal{P}}\} \cup D_{\mathcal{P}} \subseteq K_{\mathcal{P}} \subseteq T_{\mathcal{P}}$.

The set $D_{\mathcal{P}}$ consists of all divergent and inconsistent traces of $\mathcal{P}$, while $K_{\mathcal{P}}$ also contains the quiescent traces of $\mathcal{P}$. Note that, due to the possibility of internal computation (which introduces non-deterministic behaviour), the quiescent traces of a component are not completely determined by $T_{\mathcal{P}}$ and $F_{\mathcal{P}}$. In our framework, a separate treatment of divergence is given in order to guarantee that a refining component makes observable progress. This is in contrast to the receptive process theory of Josephs [21] and the work of Jonsson [19], for example.

We now redefine $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ to be components with signatures $\langle \mathcal{A}_{\mathcal{P}}^{I}, \mathcal{A}_{\mathcal{P}}^{O}, T_{\mathcal{P}}, F_{\mathcal{P}}, D_{\mathcal{P}}, K_{\mathcal{P}} \rangle$, $\langle \mathcal{A}_{\mathcal{Q}}^{I}, \mathcal{A}_{\mathcal{Q}}^{O}, T_{\mathcal{Q}}, F_{\mathcal{Q}}, D_{\mathcal{Q}}, K_{\mathcal{Q}} \rangle$ and $\langle \mathcal{A}_{\mathcal{R}}^{I}, \mathcal{A}_{\mathcal{R}}^{O}, T_{\mathcal{R}}, F_{\mathcal{R}}, D_{\mathcal{R}}, K_{\mathcal{R}} \rangle$ respectively.

3.1. Refinement

As in Section 2.1, refinement of component $\mathcal{Q}$ by component $\mathcal{P}$ needs to talk about the most general safe representations $\mathcal{E}(\mathcal{P})$ and $\mathcal{E}(\mathcal{Q})$. This carries across to the new setting effortlessly, by taking $D_{\mathcal{E}(\mathcal{P})} = D_{\mathcal{P}} \cup F_{\mathcal{E}(\mathcal{P})}$ and $K_{\mathcal{E}(\mathcal{P})} = K_{\mathcal{P}} \cup F_{\mathcal{E}(\mathcal{P})}$. Based on this, we give the formal definition of refinement.

Definition 12. $\mathcal{Q}$ *is said to be a* progress-sensitive refinement *of* $\mathcal{P}$*, written* $\mathcal{Q} \sqsubseteq_{imp}^{l} \mathcal{P}$*, iff* $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$*,* $D_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}_{\mathcal{P}}^{*} \subseteq D_{\mathcal{E}(\mathcal{P})}$ *and* $K_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}_{\mathcal{P}}^{*} \subseteq K_{\mathcal{E}(\mathcal{P})}$.

By $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ we mean refinement as in Definition 3 after having projected out $D_{\mathcal{P}}$, $K_{\mathcal{P}}$, $D_{\mathcal{Q}}$ and $K_{\mathcal{Q}}$ from $\mathcal{P}$ and $\mathcal{Q}$; this condition guarantees that $\mathcal{Q}$ is substitutable for $\mathcal{P}$. The additional constraints $D_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}_{\mathcal{P}}^{*} \subseteq D_{\mathcal{E}(\mathcal{P})}$ and $K_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}_{\mathcal{P}}^{*} \subseteq K_{\mathcal{E}(\mathcal{P})}$ ensure that $\mathcal{Q}$ is only allowed to diverge when $\mathcal{P}$ can diverge, and can only be quiescent when $\mathcal{P}$ is quiescent. It is these final clauses that force a refining component to make observable progress whenever the original can.

Equivalence of components, indicated using $\equiv_{imp}^{l}$, can easily be defined by means of mutual refinement, i.e., is equal to $\sqsubseteq_{imp}^{l} \cap (\sqsubseteq_{imp}^{l})^{-1}$.

Lemma 5. *Progress-sensitive refinement is reflexive, and transitive subject to preservation of action types.*

Proof. Follows by the exact same reasoning as in Lemma 1. □

3.2. Parallel Composition

As parallel composition is not related to refinement, the definition remains largely unchanged, excepting the sets of extended divergent and quiescent traces. To compute these sets, it is straightforward to observe that a trace is divergent in the parallel composition if its projection onto the alphabet of at least one of the components is a divergent trace, and is quiescent if its projections onto the alphabets of both components are quiescent.

Definition 13. *Let $\mathcal{P}$ and $\mathcal{Q}$ be composable for parallel. Then $\mathcal{P} \,||_l\, \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}||\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}||\mathcal{Q}}, T_{\mathcal{P}||\mathcal{Q}}, F_{\mathcal{P}||\mathcal{Q}}, D_{\mathcal{P}||\mathcal{Q}}, K_{\mathcal{P}||\mathcal{Q}} \rangle$, where:*

- $D_{\mathcal{P}||\mathcal{Q}} = [(D_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \cap (T_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}})] \cup$
  $[(T_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \cap (D_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}})] \cup F_{\mathcal{P}||\mathcal{Q}}$
- $K_{\mathcal{P}||\mathcal{Q}} = [(K_{\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \cap (K_{\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{P}||\mathcal{Q}})] \cup D_{\mathcal{P}||\mathcal{Q}}$.

Given the effect of divergence and quiescence on parallel composition, it is not surprising that the monotonicity result is unchanged.

Theorem 8. *Let $\mathcal{P}$, $\mathcal{P}'$, $\mathcal{Q}$ and $\mathcal{Q}'$ be components such that $\mathcal{P}$ and $\mathcal{Q}$ are composable, $\mathcal{A}_{\mathcal{P}'} \cap \mathcal{A}_{\mathcal{Q}'} \cap \mathcal{A}_{\mathcal{P}||\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{P}} \cap \mathcal{A}_{\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'} \cap \mathcal{A}^O_{\mathcal{P}||\mathcal{Q}} = \emptyset$. If $\mathcal{P}' \sqsubseteq^l_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^l_{imp} \mathcal{Q}$, then $\mathcal{P}' \,||_l\, \mathcal{Q}' \sqsubseteq^l_{imp} \mathcal{P} \,||_l\, \mathcal{Q}$.*

Proof. By Theorem 1, we know that the T and F-set containments hold. In the difficult case, suppose $t \in D_{\mathcal{P}'||\mathcal{Q}'} \setminus F_{\mathcal{E}(\mathcal{P}'||\mathcal{Q}')}$. Then, without loss of generality, we know $t \restriction \mathcal{A}_{\mathcal{P}'} \in D_{\mathcal{P}'}$ and $t \restriction \mathcal{A}_{\mathcal{Q}'} \in T_{\mathcal{Q}'}$. By the alphabet constraints (as elaborated in the proof of Theorem 1) it follows that $t \restriction \mathcal{A}_{\mathcal{P}'} = t \restriction \mathcal{A}_{\mathcal{P}}$ and $t \restriction \mathcal{A}_{\mathcal{Q}'} = t \restriction \mathcal{A}_{\mathcal{Q}}$. Hence, from $\mathcal{P}' \sqsubseteq^l_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^l_{imp} \mathcal{Q}$, it follows that $t \restriction \mathcal{A}_{\mathcal{P}} \in D_{\mathcal{E}(\mathcal{P})}$ and $t \restriction \mathcal{A}_{\mathcal{Q}} \in T_{\mathcal{E}(\mathcal{Q})}$, yielding $t \in D_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$ as required. The quiescent trace containment is similar. □

3.3. Conjunction

As conjunction corresponds to the meet operator on the refinement preorder, its definition in the progress-sensitive setting is substantially altered. In particular, we require that a trace in the conjunction can only be quiescent if it is permitted to be quiescent in both of the components to be conjoined. For substitutability, it is necessary to synchronise on outputs, which means that the conjunction can introduce new undesirable quiescence. Hence, it is necessary to perform a backward pruning, which removes an output at an earlier stage to avoid violating the constraints on quiescence later on. Of course, removing outputs at an earlier stage can introduce more quiescence, so pruning must be applied iteratively.

Definition 14. *Let $\mathcal{P}$ and $\mathcal{Q}$ be composable for conjunction. Then $\mathcal{P} \wedge_l \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}}, T_{\mathcal{P}\wedge\mathcal{Q}} \setminus Err, F_{\mathcal{P}\wedge\mathcal{Q}} \setminus Err, D_{\mathcal{P}\wedge\mathcal{Q}} \setminus Err, K_{\mathcal{P}\wedge\mathcal{Q}} \setminus Err \rangle$, where:*

- $D_{\mathcal{P}\wedge\mathcal{Q}} = (D_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})) \cap (D_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}}))$
- $K_{\mathcal{P}\wedge\mathcal{Q}} = (K_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})) \cap (K_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}}))$
- *Err is the smallest set containing* $\{t \in T_{\mathcal{P}\wedge\mathcal{Q}} : \exists t' \in (\mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})^* \cdot tt' \notin K_{\mathcal{P}\wedge\mathcal{Q}}$ *and* $\forall o \in \mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}} \cdot tt'o \notin T_{\mathcal{P}\wedge\mathcal{Q}} \setminus Err\} \cdot \mathcal{A}^*_{\mathcal{P}\wedge\mathcal{Q}}$.

Err captures the quiescent traces in $\mathcal{P} \wedge \mathcal{Q}$ that are not quiescent in both $\mathcal{P}$ and $\mathcal{Q}$. These traces correspond to a clash of requirements between safety and progress, so are subsequently removed from the behaviour of $\mathcal{P} \wedge_l \mathcal{Q}$. In removing these traces, we can introduce further quiescence, which is why Err is defined as a least fixed point. Note that, unlike in the original definition, the conjunction of two realisable components may not be realisable.

Theorem 9. *Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$ be components composable for conjunction. Then:*

- $\mathcal{P} \wedge_l \mathcal{Q} \sqsubseteq^l_{imp} \mathcal{P}$ *and* $\mathcal{P} \wedge_l \mathcal{Q} \sqsubseteq^l_{imp} \mathcal{Q}$
- $\mathcal{R} \sqsubseteq^l_{imp} \mathcal{P}$ *and* $\mathcal{R} \sqsubseteq^l_{imp} \mathcal{Q}$ *implies* $\mathcal{R} \sqsubseteq^l_{imp} \mathcal{P} \wedge_l \mathcal{Q}$
- $\mathcal{P}' \sqsubseteq^l_{imp} \mathcal{P}$ *and* $\mathcal{Q}' \sqsubseteq^l_{imp} \mathcal{Q}$ *implies* $\mathcal{P}' \wedge_l \mathcal{Q}' \sqsubseteq^l_{imp} \mathcal{P} \wedge_l \mathcal{Q}$.

Proof. For the first claim, we just need to show divergent and quiescent trace containment, which is a straightforward modification to Theorem 2. The proof for observable and inconsistent trace containment remains unchanged.

For the second claim, under the assumption that $T_{\mathcal{E}(\mathcal{R})} \cap Err = \emptyset$, the observable and inconsistent trace containments remain as in Theorem 2, and the divergent and inconsistent trace containments are a straightforward extension. We therefore need to show that $T_{\mathcal{E}(\mathcal{R})} \cap Err = \emptyset$, by proving that $T_{\mathcal{E}(\mathcal{R})} \cap X_i = \emptyset$ for each $i \in \mathbb{N}$, where X_i is the i-th approximation of Err defined as a fixed point. Clearly the result holds for $i = 0$ (since $X_0 = \emptyset$), so we show that it holds for $i = k + 1$ given that it holds for $i = k$. Suppose $t \in T_{\mathcal{E}(\mathcal{R})} \cap X_{k+1}$. Then by Theorem 2 we know $t \in T_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})} \cap X_{k+1}$ (since $Err \subseteq \mathcal{A}^*_{\mathcal{P}\wedge\mathcal{Q}}$), which means that there exists $t' \in (\mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})^*$ such that $tt' \notin K_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})}$ and $\forall o \in \mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}} \cdot tt'o \notin T_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})} \setminus X_k$. From $tt' \in T_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})} \cap \overline{K_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})}}$, we know that $tt' \in T_{\mathcal{E}(\mathcal{R})} \cap \overline{K_{\mathcal{E}(\mathcal{R})}}$. Thus, there exists $o' \in \mathcal{A}^O_{\mathcal{R}}$ such that $tt'o' \in T_{\mathcal{E}(\mathcal{R})}$, which means that $tt'o' \in T_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})}$. Hence $tt'o' \in X_k$, but this implies $tt'o' \notin T_{\mathcal{E}(\mathcal{R})}$, which is contradictory.

For the third claim, under the assumption that $(T_{\mathcal{E}(\mathcal{P}'\wedge\mathcal{Q}')} \setminus Err_{\mathcal{P}'\wedge\mathcal{Q}'}) \cap Err_{\mathcal{P}\wedge\mathcal{Q}} = \emptyset$, the observable and inconsistent trace containments follow as before in Theorem 2, and the divergent and quiescent containments can be shown similarly. To show that $(T_{\mathcal{E}(\mathcal{P}'\wedge\mathcal{Q}')} \setminus Err_{\mathcal{P}'\wedge\mathcal{Q}'}) \cap Err_{\mathcal{P}\wedge\mathcal{Q}} = \emptyset$, $Err_{\mathcal{P}\wedge\mathcal{Q}}$ can be approximated as for the previous claim. The proof is then a straightforward modification, having noted that $t \in T_{\mathcal{E}(\mathcal{P}'\wedge\mathcal{Q}')} \setminus Err_{\mathcal{P}'\wedge\mathcal{Q}'}$ and $t' \in (\mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})^*$ implies $tt' \in T_{\mathcal{E}(\mathcal{P}'\wedge\mathcal{Q}')} \setminus Err_{\mathcal{P}'\wedge\mathcal{Q}'}$. □

Example 7. *In the progress-sensitive setting, we assume that a square node in a figure indicates non-quiescent behaviour, meaning that some output must occur. Based on this, we now consider the conjunction of* Device *in Figure 1 with the printing and faxing device of Figure 2, under the assumption that the components have identical interfaces incorporating all actions. The T, F and K sets (prior to the removal of the Err traces) can be obtained from the pictorial representation of the original Figure 4 (note that the D set is empty, since there are no τ transitions).*

The upper non-quiescent state in Figure 4 is problematic, because the behaviour is quiescent in reality, since no output can be offered. Therefore, this state must be removed (including the last output from which there is a sequence of inputs leading to this state, according to the definition of Err), which leads to the deletion of the immediately preceding print *transition. Note*

that the circular state to the left of the upper quiescent state does not need to be removed, since it is allowed to be quiescent. The lower non-quiescent state is not problematic, because the component can always perform the print *self-loop. Consequently, the actual conjunction, after having removed the Err traces, is shown in Figure 11.*

3.4. Disjunction

Recall that the definition of conjunction is complicated by the fact that, after a common trace, one of the components may be quiescent while the other is not. It is this behaviour that forces us to prune the traces contained in Err, which are subject to the conflicts of requirements between progress and safety. Being the dual of conjunction, the disjunctive operator does not share a similar fate, since the disjunction can always avoid conflicts by being less strict on the requirements of safety and progress.

Definition 15. *Let $\mathcal{P}$ and $\mathcal{Q}$ be composable for disjunction. Then $\mathcal{P} \vee_l \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}}, T_{\mathcal{P}\vee\mathcal{Q}}, F_{\mathcal{P}\vee\mathcal{Q}}, D_{\mathcal{P}\vee\mathcal{Q}}, K_{\mathcal{P}\vee\mathcal{Q}} \rangle$, where:*

- $D_{\mathcal{P}\vee\mathcal{Q}} = [(D_{\mathcal{P}} \cup D_{\mathcal{Q}}) \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}] \cup F_{\mathcal{P}\vee\mathcal{Q}}$
- $K_{\mathcal{P}\vee\mathcal{Q}} = [(K_{\mathcal{P}} \cup K_{\mathcal{Q}}) \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}] \cup F_{\mathcal{P}\vee\mathcal{Q}}$.

Under progress-sensitive refinement, the algebraic properties of disjunction continue to hold.

Theorem 10. *Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$ be components composable for disjunction. Then:*

- $\mathcal{P} \sqsubseteq^l_{imp} \mathcal{P} \vee_l \mathcal{Q}$ *and* $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{P} \vee_l \mathcal{Q}$
- $\mathcal{P} \sqsubseteq^l_{imp} \mathcal{R}$ *and* $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{R}$ *implies* $\mathcal{P} \vee_l \mathcal{Q} \sqsubseteq^l_{imp} \mathcal{R}$
- $\mathcal{P}' \sqsubseteq^l_{imp} \mathcal{P}$ *and* $\mathcal{Q}' \sqsubseteq^l_{imp} \mathcal{Q}$ *implies* $\mathcal{P}' \vee_l \mathcal{Q}' \sqsubseteq^l_{imp} \mathcal{P} \vee_l \mathcal{Q}$.

Proof. A straightforward extension of Theorem 3. The divergent and quiescent trace containment proofs are identical to showing containment of the observable traces. □

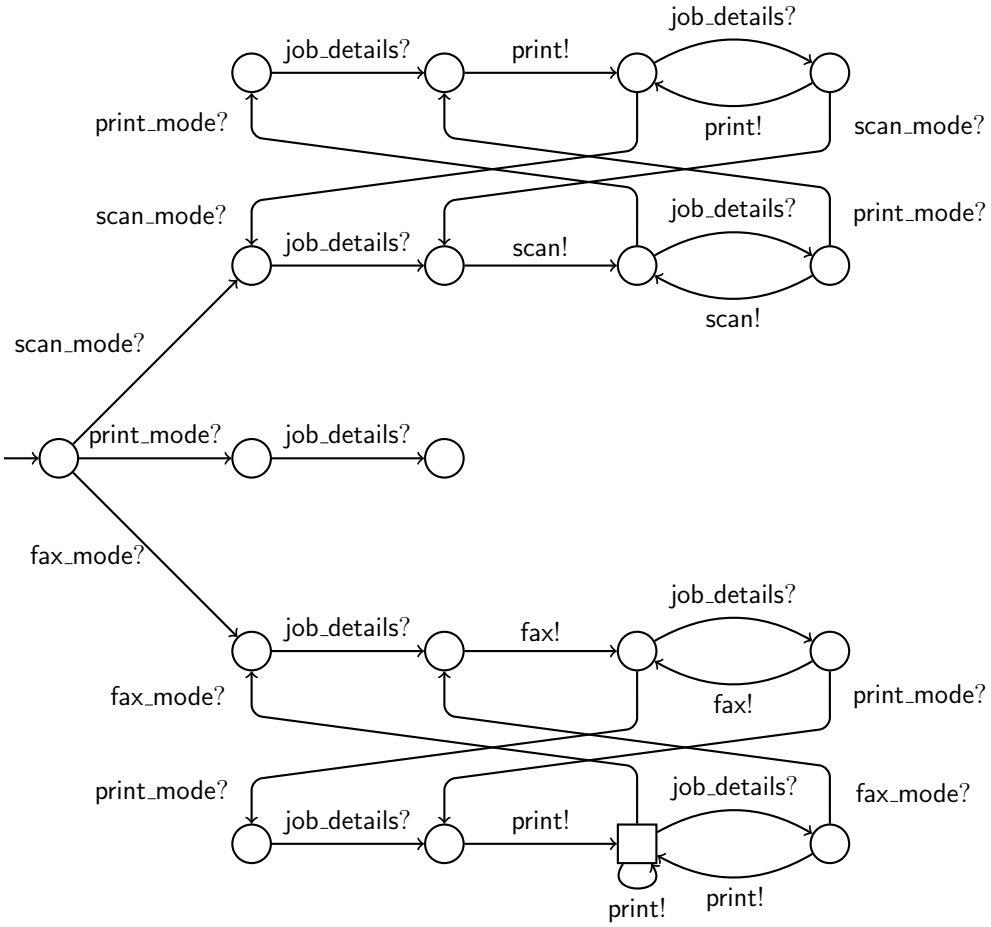

Figure 11: Progress-sensitive conjunction of the printing/scanning and printing/faxing devices when the components have identical interfaces incorporating all actions

3.5. Hiding

The removal of inputs from a component's interface can have no effect on the quiescence or divergence of traces. This is not true for outputs in our setting, although there are a number of ways to handle quiescence. Therefore, the reasoning needs careful attention, once we have considered the definition.

Definition 16. *Let $\mathcal{P}$ be a component and let b be an action. The hiding of b in $\mathcal{P}$ is a component $\mathcal{P} /_l b = \langle \mathcal{A}^I_{\mathcal{P}/b}, \mathcal{A}^O_{\mathcal{P}/b}, T_{\mathcal{P}/b}, F_{\mathcal{P}/b}, D_{\mathcal{P}/b}, K_{\mathcal{P}/b} \rangle$, where:*

- $D_{\mathcal{P}/b} = \begin{cases} D_{\mathcal{P}} \upharpoonright \mathcal{A}_{\mathcal{P}/b} \cup \mathsf{div} & \text{if } b \in \mathcal{A}^O_{\mathcal{P}} \\ D_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}/b} & \text{otherwise} \end{cases}$
- $K_{\mathcal{P}/b} = \begin{cases} K_{\mathcal{P}} \upharpoonright \mathcal{A}_{\mathcal{P}/b} \cup \mathsf{div} & \text{if } b \in \mathcal{A}^O_{\mathcal{P}} \\ K_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}/b} & \text{otherwise} \end{cases}$
- $\mathsf{div} = \{t \upharpoonright \mathcal{A}_{\mathcal{P}/b} : t \in T_{\mathcal{P}} \text{ and } \forall i \in \mathbb{N} \cdot tb^i \in T_{\mathcal{P}}\}$.

According to our definition, in the case that b is an output, divergence can be introduced after a trace t under two circumstances. The first is when there is a sequence of b actions leading to a divergent trace, while the second corresponds to the introduction of divergence outright, whereby t can be extended by an arbitrary number of b actions. This makes sense, and is common to a number of formulations of hiding (e.g., CSP [17]).

In the case of quiescence, a trace t is quiescent if t can diverge, or if there is a sequence of b actions leading to a quiescent state. This means that, if a component can only produce the single output b and cannot diverge after the trace t, then it is not necessarily the case that the component becomes quiescent on t after hiding b. This formulation of quiescence is justified since, immediately after the trace t, the component can perform internal computation, which can affect the subsequently offered outputs. This can be seen clearly in the operational setting (see Section 4.5), and corresponds to the notion that quiescence should only be considered in stable states. Moreover, this interpretation ensures that hiding is compositional under refinement.

Theorem 11. *Let $\mathcal{P}$ and $\mathcal{Q}$ be components and let b be an action. If $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$, then $\mathcal{Q} /_l b \sqsubseteq_{imp} \mathcal{P} /_l b$.*

Proof. The divergent and quiescent trace containments follow by the same reasoning as in Theorem 4 when $b \in \mathcal{A}^I_{\mathcal{Q}}$ or $b \notin \mathcal{A}_{\mathcal{P}} \cup \mathcal{A}_{\mathcal{Q}}$, and the observable and inconsistent containments are entirely unchanged. When $b \in \mathcal{A}^O_{\mathcal{P}}$, suppose that $t \in D_{\mathcal{Q}/b} \cap \mathcal{A}^*_{\mathcal{P}/b}$. Then there exists $t' \in T_{\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{P}}$ such that $t' \upharpoonright \mathcal{A}_{\mathcal{Q}/b} = t$ and $t' \in D_{\mathcal{Q}}$ or $\forall i \in \mathbb{N} \cdot t'b^i \in T_{\mathcal{Q}}$. In the case of the former, it follows that $t' \in D_{\mathcal{E}(\mathcal{P})}$, and so $t \in D_{\mathcal{E}(\mathcal{P}/b)}$ as required, given $t' \upharpoonright \mathcal{A}_{\mathcal{Q}/b} = t' \upharpoonright \mathcal{A}_{\mathcal{P}/b}$. For the latter case, $t' \in T_{\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{P}}$ implies $t' \in T_{\mathcal{E}(\mathcal{P})}$, and moreover, $t'b^i \in T_{\mathcal{E}(\mathcal{P})}$ for each $i \in \mathbb{N}$. Hence $t \in D_{\mathcal{E}(\mathcal{P}/b)}$. Quiescent trace containment is similar. □

3.6. Quotient

The definition of quotient remains largely unchanged from the substitutive case, except for the need to remove two types of trace:

1. Quiescent (resp. divergent) traces in the parallel composition of $\mathcal{P}$ and $\mathcal{R}/\mathcal{P}$ that are non-quiescent (resp. non-divergent) in $\mathcal{R}$. As we are unable to alter the traces of $\mathcal{P}$, it is necessary to prune all behaviour from (and including) the last available output in $\mathcal{A}^O_{\mathcal{R}/\mathcal{P}}$ on the projection of these traces onto $\mathcal{A}_{\mathcal{R}/\mathcal{P}}$, in order to avoid reaching such conflicts.

2. Traces of $\mathcal{R}/\mathcal{P}$ that introduce new quiescence conflicts, after having repeatedly removed traces satisfying this or the previous condition.

Definition 17. *Let $\mathcal{P}$ and $\mathcal{R}$ be components such that $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{R}}$. The quotient of $\mathcal{P}$ from $\mathcal{R}$ is the component $\mathcal{R} \,/_l\, \mathcal{P}$ with signature $\langle \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}, \mathcal{A}^O_{\mathcal{R}/\mathcal{P}}, T_{\mathcal{R}/_l\mathcal{P}}, F_{\mathcal{R}/_l\mathcal{P}}, D_{\mathcal{R}/_l\mathcal{P}}, K_{\mathcal{R}/_l\mathcal{P}} \rangle$, where:*

- $X_{\mathcal{R}/_l\mathcal{P}} = X^l_{\mathcal{R}/\mathcal{P}} \setminus Err$ *for* $X \in \{T, F, D, K\}$
- $T^l_{\mathcal{R}/\mathcal{P}}$ *is the largest prefix-closed and input-receptive subset of* $T_{\mathcal{R}/\mathcal{P}} \cap \{t \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \upharpoonright \mathcal{A}_{\mathcal{P}} \in D_{\mathcal{P}} \implies t' \in D_{\mathcal{E}(\mathcal{R})}\}$
- $F^l_{\mathcal{R}/\mathcal{P}} = T^l_{\mathcal{R}/\mathcal{P}} \cap F_{\mathcal{R}/\mathcal{P}}$
- $D^l_{\mathcal{R}/\mathcal{P}} = \{t \in T^l_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \upharpoonright \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}} \implies t' \in D_{\mathcal{E}(\mathcal{R})}\}$

- $K^l_{\mathcal{R}/\mathcal{P}} = \{t \in T^l_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \upharpoonright \mathcal{A}_{\mathcal{P}} \in K_{\mathcal{P}} \implies t' \in K_{\mathcal{E}(\mathcal{R})}\}$
- *Err is the smallest set containing* $\{t \in T^l_{\mathcal{R}/\mathcal{P}} : \exists t' \in (\mathcal{A}^I_{\mathcal{R}/\mathcal{P}})^* \cdot tt' \notin K^l_{\mathcal{R}/\mathcal{P}} \text{ and } \forall o \in \mathcal{A}^O_{\mathcal{R}/\mathcal{P}} \cdot tt'o \notin T^l_{\mathcal{R}/\mathcal{P}} \setminus Err\} \cdot \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$.

The definition of $T^l_{\mathcal{R}/\mathcal{P}}$ ensures that, by the intersection with $T_{\mathcal{R}/\mathcal{P}}$, any trace of $\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P})$ must also be in $\mathcal{R}$, and that any inconsistent trace of $\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P})$ is also inconsistent in $\mathcal{R}$. The additional constraint intersected with $T_{\mathcal{R}/\mathcal{P}}$ ensures that $\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P})$ only diverges when $\mathcal{R}$ can diverge. For $\mathcal{R} \,/_l\, \mathcal{P}$ to be the least refined solution to $\mathcal{P} \,||_l\, X \sqsubseteq^l_{imp} \mathcal{R}$, any trace in $T^l_{\mathcal{R}/\mathcal{P}}$ is:

- inconsistent, when it is not a trace of $\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P})$ or is inconsistent in $\mathcal{R}$; is
- divergent, when it is not a trace of $\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P})$ or is divergent in $\mathcal{R}$; and is
- quiescent, when it is not a trace of $\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P})$, is not quiescent in $\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P})$ (due to $\mathcal{P}$ not being quiescent) or is quiescent in $\mathcal{R}$.

The resulting trace sets $X^l_{\mathcal{R}/\mathcal{P}}$ for $X \in \{T, F, D, K\}$ do not form a component, since it does not follow that $\{t \in T^l_{\mathcal{R}/\mathcal{P}} : \nexists o \in \mathcal{A}^O_{\mathcal{R}/\mathcal{P}} \cdot to \in T^l_{\mathcal{R}/\mathcal{P}}\}$ is a subset of $K^l_{\mathcal{R}/\mathcal{P}}$. *Err* is defined to capture such conflicts, which are subsequently removed from the quotient. As the removal of traces can introduce quiescence, the set is defined as a fixed point. Due to the possibility of *Err* capturing all traces in $T^l_{\mathcal{R}/\mathcal{P}}$, it follows that (as for conjunction) the quotient of two realisable components may not be realisable, and this can only be determined by examining the behaviours of $\mathcal{P}$ and $\mathcal{R}$. However, the quotient is always defined when $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{R}}$.

Theorem 12. *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components. Then $\mathcal{P} \,||_l\, \mathcal{Q} \sqsubseteq^l_{imp} \mathcal{R}$ iff:*

- *$\mathcal{R} \,/_l\, \mathcal{P}$ is defined (i.e., $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{R}}$)*
- *$\mathcal{P} \,||_l\, (\mathcal{R} \,/_l\, \mathcal{P}) \sqsubseteq^l_{imp} \mathcal{R}$*
- *$\mathcal{A}^I_{\mathcal{Q}} = \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$ implies $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{R} \,/_l\, \mathcal{P}$.*

Proof. The reasoning for the first claim is identical to that in Theorem 5. For the second claim, inconsistent and observable trace containment follows by Theorem 5, having noticed that $F_{\mathcal{R}/_l\mathcal{P}} \subseteq F_{\mathcal{R}/\mathcal{P}}$ and $T_{\mathcal{R}/_l\mathcal{P}} \subseteq T_{\mathcal{R}/\mathcal{P}}$. For divergent traces, suppose $t \in (D_{\mathcal{P}||_l(\mathcal{R}/_l\mathcal{P})} \setminus F_{\mathcal{E}(\mathcal{P}||_l(\mathcal{R}/_l\mathcal{P}))}) \cap \mathcal{A}^*_{\mathcal{R}}$. Then either $t \upharpoonright \mathcal{A}_{\mathcal{P}} \in D_{\mathcal{P}}$ and $t \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} \in T_{\mathcal{R}/_l\mathcal{P}}$, or $t \upharpoonright \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}}$ and $t \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} \in D_{\mathcal{R}/_l\mathcal{P}}$. For the former, it follows by the definition of $T^l_{\mathcal{R}/\mathcal{P}}$ that $t \in D_{\mathcal{E}(\mathcal{R})}$, while, for the latter case, it follows by the definition of $D^l_{\mathcal{R}/\mathcal{P}}$ that $t \in D_{\mathcal{E}(\mathcal{R})}$, as required. For the quiescent containment, suppose $t \in (K_{\mathcal{P}||_l(\mathcal{R}/_l\mathcal{P})} \setminus D_{\mathcal{E}(\mathcal{P}||_l(\mathcal{R}/_l\mathcal{P}))}) \cap \mathcal{A}^*_{\mathcal{R}}$. Therefore, $t \upharpoonright \mathcal{A}_{\mathcal{P}} \in K_{\mathcal{P}}$ and $t \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} \in K_{\mathcal{R}/_l\mathcal{P}}$. By the definition of $K^l_{\mathcal{R}/\mathcal{P}}$ it follows that $t \in K_{\mathcal{E}(\mathcal{R})}$ as required.

For the third claim, we first show that $X_{\mathcal{E}(\mathcal{Q})} \subseteq X^l_{\mathcal{E}(\mathcal{R}/\mathcal{P})}$ for each $X \in \{T, F, D, K\}$ (noting that $X^l_{\mathcal{E}(\mathcal{R}/\mathcal{P})} = X^l_{\mathcal{R}/\mathcal{P}}$), and thereafter show that $T_{\mathcal{E}(\mathcal{Q})} \cap Err = \emptyset$, from which it can be inferred that $X_{\mathcal{E}(\mathcal{Q})} \subseteq X_{\mathcal{R}/_l\mathcal{P}}$. First note that, if $t \in T_{\mathcal{E}(\mathcal{Q})}$, then certainly $t \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$, since $\mathcal{A}^I_{\mathcal{Q}} = \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$ and $\mathcal{R} /_l \mathcal{P}$ has the largest possible set of outputs. Now begin by supposing that $t \in F_{\mathcal{E}(\mathcal{Q})}$. Then there exists a prefix t' of t and $t'' \in (\mathcal{A}^O_{\mathcal{Q}})^*$ such that $t't'' \in F_{\mathcal{Q}}$. Note that $t't'' \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$. By Theorem 5, it follows that $t't'' \in F_{\mathcal{R}/\mathcal{P}} \cap T_{\mathcal{R}/\mathcal{P}}$. Therefore, we must show that $t't'' \in T^l_{\mathcal{R}/\mathcal{P}}$. So let $t''' \in \mathcal{A}^*_{\mathcal{R}}$ be an arbitrary trace such that $t''' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t't''$. If $t''' \upharpoonright \mathcal{A}_{\mathcal{P}} \in D_{\mathcal{P}}$, then $t''' \in D_{\mathcal{E}(\mathcal{R})}$ as required, since $t''' \in D_{\mathcal{P}||_l\mathcal{Q}}$ and $\mathcal{P} \,||_l\, \mathcal{Q} \sqsubseteq^l_{imp} \mathcal{R}$. Thus $t't'' \in T^l_{\mathcal{R}/\mathcal{P}}$, unless if the trace is removed due to non prefix-closure/input-receptiveness. But if either of these do not hold, then it can be shown that $t't'' \notin T_{\mathcal{E}(\mathcal{Q})}$. Consequently, $t \in F^l_{\mathcal{E}(\mathcal{R}/\mathcal{P})}$ as required. Now suppose that $t \in (D_{\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{Q})})$. Then, for any $t''' \in \mathcal{A}^*_{\mathcal{R}}$ such that $t''' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t$, if $t''' \upharpoonright \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}}$, then $t''' \in D_{\mathcal{P}||_l\mathcal{Q}}$, which by $\mathcal{P} \,||_l\, \mathcal{Q} \sqsubseteq^l_{imp} \mathcal{R}$ yields $t''' \in D_{\mathcal{E}(\mathcal{R})}$. Hence $t \in D^l_{\mathcal{R}/\mathcal{P}}$ as required. Similarly, if $t \in K_{\mathcal{Q}} \setminus D_{\mathcal{E}(\mathcal{Q})}$, then for any $t''' \in \mathcal{A}^*_{\mathcal{R}}$ such that $t''' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t$, it holds that, if $t''' \upharpoonright \mathcal{A}_{\mathcal{P}} \in K_{\mathcal{P}}$, then $t''' \in K_{\mathcal{E}(\mathcal{R})}$, since $\mathcal{P} \,||_l\, \mathcal{Q} \sqsubseteq^l_{imp} \mathcal{R}$ and it must be the case that $t''' \in K_{\mathcal{P}||_l\mathcal{Q}}$. Thus $t \in K^l_{\mathcal{R}/\mathcal{P}}$ as required.

For the final part of the third claim, in order to demonstrate that $T_{\mathcal{E}(\mathcal{Q})} \cap Err = \emptyset$, we show $T_{\mathcal{E}(\mathcal{Q})} \cap X_i = \emptyset$ for each $i \in \mathbb{N}$, where X_i is the i-th iteration of finding the fixed point defining Err. When $i = 0$, $X_i = \emptyset$, so the result trivially holds. Now suppose $i = k + 1$, and assume that the result holds for $i = k$. If $t \in T_{\mathcal{E}(\mathcal{Q})} \cap X_{k+1}$, then we know $t \in T^l_{\mathcal{R}/\mathcal{P}} \cap X_{k+1}$ by the previous part. Consequently, there exists $t' \in (\mathcal{A}^I_{\mathcal{R}/\mathcal{P}})^*$ such that $tt' \notin K^l_{\mathcal{R}/\mathcal{P}}$ and $\nexists o \in \mathcal{A}^O_{\mathcal{R}/\mathcal{P}} \cdot tt'o \in T^l_{\mathcal{R}/\mathcal{P}} \setminus X_k$. Note that $tt' \in T_{\mathcal{E}(\mathcal{Q})}$, and by the previous part $tt' \notin K_{\mathcal{Q}}$ as $tt' \notin K^l_{\mathcal{R}/\mathcal{P}}$. Hence, there exists $o' \in \mathcal{A}^O_{\mathcal{Q}}$

such that $tt'o' \in T_{\mathcal{E}(\mathcal{Q})}$, which implies $tt'o' \in T^l_{\mathcal{R}/\mathcal{P}}$. It therefore follows that $tt'o' \in X_k$ so that $tt'o' \in T^l_{\mathcal{R}/\mathcal{P}} \setminus X_k$ holds. But by the induction hypothesis, this allows us to conclude that $tt'o' \notin T_{\mathcal{E}(\mathcal{Q})}$, which is contradictory. Thus $T_{\mathcal{E}(\mathcal{Q})} \cap X_{k+1} = \emptyset$ and so $T_{\mathcal{E}(\mathcal{Q})} \cap Err = \emptyset$. □

Theorem 13. *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components such that $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{P}$.*

- *If $\mathcal{Q} /_l \mathcal{R}$ is defined and $\mathcal{A}^I_{\mathcal{R}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$, then $\mathcal{Q} /_l \mathcal{R} \sqsubseteq^l_{imp} \mathcal{P} /_l \mathcal{R}$.*
- *If $\mathcal{R} /_l \mathcal{P}$ is defined and $(\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{P}}) \cap \mathcal{A}_{\mathcal{R}} = \emptyset$, then $\mathcal{R} /_l \mathcal{Q} \sqsupseteq^l_{imp} \mathcal{R} /_l \mathcal{P}$.*

Proof. The proof is the same as in Theorem 6 when using Theorems 8 and 12 in place of Theorems 1 and 5, and Lemma 5 in place of Lemma 1. □

Example 8. *To demonstrate quotient in the quiescent framework, suppose that a user wishes to interact with* BrokenDevice *(Figure 6), but without ever reaching a quiescent state, i.e., a point from which the system as a whole is blocked waiting for input. Note that* User2 *(as shown in Figure 9) is not a suitable candidate, since, after placing* BrokenDevice *in* scan_mode *and sending* job_details*, the system becomes blocked due to* BrokenDevice *never offering to* scan*. (Note that the allocation of quiescent and non-quiescent behaviours in* BrokenDevice *and* User2 *has been added arbitrarily, since the quiescent conditions on* BrokenDevice *do not follow from those on* Device*, and similarly for* User2*.)*

We generate a satisfying user as User3 = ErrorFree $/_l$ BrokenDevice*, the result of which is shown in Figure 12.* ErrorFree *is the component having chaotic behaviour over all actions, which we treat as outputs (Figure 10). As* ErrorFree *does not have inconsistencies, and moreover is non-quiescent (since the single node is a square), it follows that* User3 $||_l$ BrokenDevice *is both inconsistency free and does not become quiescent.*

The quotient is computed in two phases: first the computation of the T, F, D and K sets is performed, after which traces in Err are removed. The first phase generates a component equal to BrokenDevice*, but with inputs and outputs interchanged, along with circular and square nodes. In the second phase, we see that the trace* ⟨scan_mode, job_details⟩ *is quiescent, but is required to make progress. We therefore remove the trace* ⟨scan_mode, job_details⟩ *from*

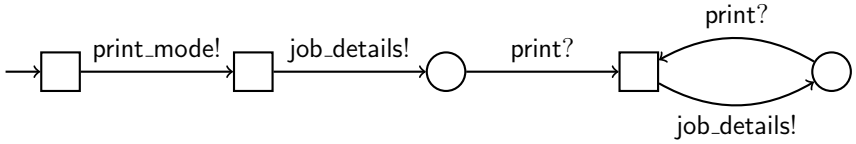

Figure 12: Component representing User3

the T, F, D and K sets. But now the trace $\langle$scan_mode$\rangle$ *becomes quiescent, so we must also remove this trace. The empty trace* ϵ *is not quiescent, since* print_mode *can be performed. Consequently, the behaviour of* User3 *must never place the* BrokenDevice *into* scan_mode*, since any trace exhibiting* scan_mode *is contained within Err. This is shown in Figure 12.*

As a variant of the example, if BrokenDevice *had all circular nodes, then the T, F and D sets of the quotient would remain as in* BrokenDevice*, having interchanged inputs and outputs, and the K set would be equal to F, meaning all nodes are squares. But then the trace* $\langle$print_mode, job_details$\rangle$ *would be quiescent, as is the trace* $\langle$scan_mode, job_details$\rangle$*, so these traces would be included within the Err set and subsequently removed. Consequently, all states would have to be pruned, meaning that no safe user, ensuring progress, can exist.*

4. Operational Theory of Components

In this section, we outline an operational representation for components, and demonstrate the relationship between these operational models and the trace-based models of Sections 2 and 3. From this, we supply operational definitions for the compositional operators of our theory.

Definition 18. *An* operational component P *is a tuple* $\langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, S_{\mathsf{P}}, \longrightarrow_{\mathsf{P}}, s^0_{\mathsf{P}}, F_{\mathsf{P}} \rangle$*, where:*

- $\mathcal{A}^I_{\mathsf{P}}$ *is a finite set of input actions*
- $\mathcal{A}^O_{\mathsf{P}}$ *is a finite set of output actions, disjoint from* $\mathcal{A}^I_{\mathsf{P}}$
- S_{P} *is a finite set of states*
- $\longrightarrow_{\mathsf{P}} \subseteq S_{\mathsf{P}} \times (\mathcal{A}_{\mathsf{P}} \cup \{\tau\}) \times S_{\mathsf{P}}$ *is the transition relation*

- $s^0_{\mathsf{P}} \in S_{\mathsf{P}}$ *is the designated initial state*
- $F_{\mathsf{P}} \subseteq S_{\mathsf{P}}$ *is the set of inconsistent states.*

The transition relation satisfies the properties that: (i) $\bot_{\mathsf{P}} \xrightarrow{a}_{\mathsf{P}} \bot_{\mathsf{P}}$ *for each* $a \in \mathcal{A}_{\mathsf{P}} \cup \{\tau\}$ *and* $\bot_{\mathsf{P}} \in F_{\mathsf{P}}$*; and (ii) for each* $s \in S_{\mathsf{P}}$ *and* $a \in \mathcal{A}^I_{\mathsf{P}}$ *there exists* $s' \in S_{\mathsf{P}}$ *such that* $s \xrightarrow{a}_{\mathsf{P}} s'$*. These conditions ensure that all states are input-receptive, and that the inconsistent states are chaotic.*

If a component does not have an initial state we say that it is unrealisable*, and is realisable otherwise.*

It is important that the set of states S_{P} is finite, so that divergence of a state can be determined in finite time. This allows us to decide which inputs are safe, and which outputs may eventually be issued, for a particular state.

Notation. A relation $\overset{\epsilon}{\Longrightarrow}_{\mathsf{P}} \subseteq S_{\mathsf{P}} \times S_{\mathsf{P}}$ is defined by $p \overset{\epsilon}{\Longrightarrow}_{\mathsf{P}} p'$ iff $p(\xrightarrow{\tau}_{\mathsf{P}})^* p'$. Generalising $\overset{\epsilon}{\Longrightarrow}_{\mathsf{P}}$ for visible actions $a \in \mathcal{A}$, we obtain $p \overset{a}{\Longrightarrow}|_{\mathsf{P}} p'$ iff there exists p_a such that $p \overset{\epsilon}{\Longrightarrow}_{\mathsf{P}} p_a \xrightarrow{a}_{\mathsf{P}} p'$, and $p \overset{a}{\Longrightarrow}_{\mathsf{P}} p'$ iff there exists p_a such that $p \overset{a}{\Longrightarrow}|_{\mathsf{P}} p_a \overset{\epsilon}{\Longrightarrow}_{\mathsf{P}} p'$. The extension to words $w = a_1 \ldots a_n$ is defined in the natural way by $p \overset{w}{\Longrightarrow}_{\mathsf{P}} p'$ iff $p \overset{a_1}{\Longrightarrow}_{\mathsf{P}} \ldots \overset{a_n}{\Longrightarrow}_{\mathsf{P}} p'$.

Henceforth, let P, Q and R be operational components with signatures $\langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, S_{\mathsf{P}}, \longrightarrow_{\mathsf{P}}, s^0_{\mathsf{P}}, F_{\mathsf{P}} \rangle$, $\langle \mathcal{A}^I_{\mathsf{Q}}, \mathcal{A}^O_{\mathsf{Q}}, S_{\mathsf{Q}}, \longrightarrow_{\mathsf{Q}}, s^0_{\mathsf{Q}}, F_{\mathsf{Q}} \rangle$ and $\langle \mathcal{A}^I_{\mathsf{R}}, \mathcal{A}^O_{\mathsf{R}}, S_{\mathsf{R}}, \longrightarrow_{\mathsf{R}}, s^0_{\mathsf{R}}, F_{\mathsf{R}} \rangle$ respectively.

4.1. Refinement

We now give semantic mappings from operational models to trace-based models that preserve both substitutive and progress-sensitive behaviour.

Definition 19. *Let* P *be an operational component. Then* $[\![\mathsf{P}]\!]$ *is the trace-based component* $\langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, T_{[\![\mathsf{P}]\!]}, F_{[\![\mathsf{P}]\!]} \rangle$*, where* $T_{[\![\mathsf{P}]\!]} = \{t : \exists s \in S_{\mathsf{P}} \cdot s^0_{\mathsf{P}} \overset{t}{\Longrightarrow}_{\mathsf{P}} s\}$ *and* $F_{[\![\mathsf{P}]\!]} = \{t : \exists s \in F_{\mathsf{P}} \cdot s^0_{\mathsf{P}} \overset{t}{\Longrightarrow}_{\mathsf{P}} s\}$*.*

The trace-based representation of an operational model simply records the component's interface, and its sets of observable and inconsistent traces.

Definition 20. *Let* P *be an operational component. Then* $[\![\mathsf{P}]\!]^l$ *is the progress-sensitive trace-based component* $\langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, T_{[\![\mathsf{P}]\!]}, F_{[\![\mathsf{P}]\!]}, D_{[\![\mathsf{P}]\!]^l}, K_{[\![\mathsf{P}]\!]^l} \rangle$*, where:*

- $D_{[\![\mathsf{P}]\!]^l} = \{t : \exists s \cdot s^0_{\mathsf{P}} \overset{t}{\Longrightarrow}_{\mathsf{P}} s$ *and* s *can diverge*$\}$

- $K_{[\![\mathsf{P}]\!]^l} = \{t : \exists s \cdot s^0_{\mathsf{P}} \stackrel{t}{\Longrightarrow}_{\mathsf{P}} s \text{ and } \nexists o \in \mathcal{A}^O_{\mathsf{P}} \cup \{\tau\} \cdot s \stackrel{o}{\longrightarrow}_{\mathsf{P}}\} \cup D_{[\![\mathsf{P}]\!]^l}$.

The progress-sensitive trace-based representation of an operational model includes the constituents of a standard trace-based component, together with a set of extended divergent traces and a set of extended quiescent traces. The inclusion of inconsistent traces within the divergent and quiescent trace sets is a condition of being a progress-sensitive component (cf. Definition 11). Note that $D_{[\![\mathsf{P}]\!]^l}$ includes all inconsistent traces, since every inconsistent state is divergent. Moreover, only stable states (without outgoing τ transitions) are able to be quiescent (although the extended quiescent trace set includes divergences). This has similarities with the stable-failures and failures-divergences models of CSP [17].

Based on these mappings to trace-based models, the justification of which is presented in Section 4.7, we can formulate definitions of refinement on operational models.

Definition 21. *Let* P *and* Q *be operational components. Then* Q *is a substitutable refinement of* P*, written* $\mathsf{Q} \sqsubseteq_{op} \mathsf{P}$*, iff* $[\![\mathsf{Q}]\!] \sqsubseteq_{imp} [\![\mathsf{P}]\!]$*. Similarly,* Q *is a substitutable and progress-sensitive refinement of* P*, written* $\mathsf{Q} \sqsubseteq^l_{op} \mathsf{P}$*, iff* $[\![\mathsf{Q}]\!]^l \sqsubseteq^l_{imp} [\![\mathsf{P}]\!]^l$.

As in the trace-based setting, we can define the safe representation of an operational component that becomes inconsistent as soon as the original component has the potential to become inconsistent under its own control.

Definition 22. *The safe representation of the operational component* P *is itself an operational component* $\mathcal{E}(\mathsf{P}) = \langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, S_{\mathsf{P}}, \longrightarrow_{\mathcal{E}(\mathsf{P})}, F_{\mathcal{E}(\mathsf{P})} \rangle$*, where:*

- $F_{\mathcal{E}(\mathsf{P})}$ *is the smallest set containing* F_{P} *and satisfying the property: if* $s \stackrel{a}{\longrightarrow}_{\mathsf{P}} s'$ *with* $a \in \mathcal{A}^O_{\mathsf{P}} \cup \{\tau\}$ *and* $s' \in F_{\mathcal{E}(\mathsf{P})}$*, then* $s \in F_{\mathcal{E}(\mathsf{P})}$.
- $\longrightarrow_{\mathcal{E}(\mathsf{P})}$ *is the smallest set containing* $\longrightarrow_{\mathsf{P}}$ *that is also chaotic on all states contained in* $F_{\mathcal{E}(\mathsf{P})}$.

We now present operational definitions for all the operators considered in the trace-based section with respect to both the substitutive and progress-sensitive refinement preorders. For each operator, we make explicit the relationship with the trace-based definition. This allows the compositionality results from the trace-based sections to carry across to this operational setting.

4.2. Parallel Composition

We give a single operational definition of parallel composition applicable to both the substitutive and progress-sensitive refinements.

Definition 23. *Let* P *and* Q *be components composable for parallel. Then the parallel composition of* P *and* Q *is the component* $\mathsf{P} \mid\mid \mathsf{Q} = \mathsf{P} \mid\mid_l \mathsf{Q} = \langle \mathcal{A}^I, \mathcal{A}^O, S, \longrightarrow, s_0, F \rangle$*, where:*

- $\mathcal{A}^I = (\mathcal{A}^I_{\mathsf{P}} \cup \mathcal{A}^I_{\mathsf{Q}}) \setminus (\mathcal{A}^O_{\mathsf{P}} \cup \mathcal{A}^O_{\mathsf{Q}})$
- $\mathcal{A}^O = \mathcal{A}^O_{\mathsf{P}} \cup \mathcal{A}^O_{\mathsf{Q}}$
- $S = S_{\mathsf{P}} \times S_{\mathsf{Q}}$
- $\longrightarrow$ *is the smallest relation respecting the chaotic nature of inconsistent states and satisfying the following rules:*

  P1. If $p \xrightarrow{a}_{\mathsf{P}} p'$ *with* $a \in \mathcal{A}_{\mathsf{P}} \setminus \mathcal{A}_{\mathsf{Q}} \cup \{\tau\}$*, then* $(p, q) \xrightarrow{a} (p', q)$

  P2. If $q \xrightarrow{a}_{\mathsf{Q}} q'$ *with* $a \in \mathcal{A}_{\mathsf{Q}} \setminus \mathcal{A}_{\mathsf{P}} \cup \{\tau\}$*, then* $(p, q) \xrightarrow{a} (p, q')$

  P3. If $p \xrightarrow{a}_{\mathsf{P}} p'$ *and* $q \xrightarrow{a}_{\mathsf{Q}} q'$ *with* $a \in \mathcal{A}_{\mathsf{P}} \cap \mathcal{A}_{\mathsf{Q}}$*, then* $(p, q) \xrightarrow{a} (p', q')$*.*

- $s_0 = (s^0_{\mathsf{P}}, s^0_{\mathsf{Q}})$
- $F = (S_{\mathsf{P}} \times F_{\mathsf{Q}}) \cup (F_{\mathsf{P}} \times S_{\mathsf{Q}})$*.*

Conditions P1 to P3 ensure that the parallel composition of components interleaves on independent actions and synchronises on common actions. For P3, given the parallel composability constraint, synchronisation can take place between an output and an input, or two inputs.

The following theorem shows the relationship between parallel composition on operational and trace-based components. Consequently, the monotonicity results from the trace-based sections are applicable here.

Theorem 14. *Let* P *and* Q *be components composable for parallel composition. Then* $[\![\mathsf{P} \mid\mid \mathsf{Q}]\!] = [\![\mathsf{P}]\!] \mid\mid [\![\mathsf{Q}]\!]$ *and* $[\![\mathsf{P} \mid\mid_l \mathsf{Q}]\!]^l = [\![\mathsf{P}]\!]^l \mid\mid_l [\![\mathsf{Q}]\!]^l$*.*

Proof. Trivial, as the trace-based definition of parallel composition interleaves on independent actions and synchronises on common actions. This is precisely captured by the operational definition. □

4.3. Conjunction

We now formulate an operational definition of conjunction. As this operator corresponds to the meet of the refinement preorder, its definition depends on the refinement type we are considering. For substitutive refinement, we have a straightforward definition that considers the enabled actions in any pair of states. When considering the progress-sensitive refinement, we first apply the substitutive definition, but then have to prune bad states that violate progress. These bad states are defined inductively.

Definition 24. *Let* P *and* Q *be components composable for conjunction such that, without loss of generality,* $\bot_{\mathsf{P}} \in F_{\mathsf{P}}$ *and* $\bot_{\mathsf{Q}} \in F_{\mathsf{Q}}$*. Then the substitutive conjunction of* P *and* Q *is a component* $\mathsf{P} \wedge \mathsf{Q} = \langle \mathcal{A}^I_{\mathsf{P}} \cup \mathcal{A}^I_{\mathsf{Q}}, \mathcal{A}^O_{\mathsf{P}} \cap \mathcal{A}^O_{\mathsf{Q}}, S, \longrightarrow, s_0, F \rangle$*, where:*

- $S = S_{\mathsf{P}} \times S_{\mathsf{Q}}$
- $\longrightarrow$ *is the smallest relation satisfying the following rules:*

  C1. If $a \in \mathcal{A}_{\mathsf{P}} \cap \mathcal{A}_{\mathsf{Q}}$*,* $p \stackrel{a}{\Longrightarrow}|_{\mathcal{E}(\mathsf{P})} p'$ *and* $q \stackrel{a}{\Longrightarrow}|_{\mathcal{E}(\mathsf{Q})} q'$*, then* $(p, q) \stackrel{a}{\longrightarrow} (p', q')$

  C2. If $a \in \mathcal{A}^I_{\mathsf{P}} \setminus \mathcal{A}^I_{\mathsf{Q}}$ *and* $p \stackrel{a}{\Longrightarrow}|_{\mathcal{E}(\mathsf{P})} p'$*, then* $(p, q) \stackrel{a}{\longrightarrow} (p', \bot_{\mathsf{Q}})$

  C3. If $a \in \mathcal{A}^I_{\mathsf{Q}} \setminus \mathcal{A}^I_{\mathsf{P}}$ *and* $q \stackrel{a}{\Longrightarrow}|_{\mathcal{E}(\mathsf{Q})} q'$*, then* $(p, q) \stackrel{a}{\longrightarrow} (\bot_{\mathsf{P}}, q')$

  C4. If p *does not diverge and* $p \stackrel{\tau}{\longrightarrow}_{\mathcal{E}(\mathsf{P})} p'$*, then* $(p, q) \stackrel{\tau}{\longrightarrow} (p', q)$

  C5. If q *does not diverge and* $q \stackrel{\tau}{\longrightarrow}_{\mathcal{E}(\mathsf{Q})} q'$*, then* $(p, q) \stackrel{\tau}{\longrightarrow} (p, q')$

  C6. If p *diverges and* q *diverges, then* $(p, q) \stackrel{\tau}{\longrightarrow} (p, q)$*.*

- $s_0 = (s^0_{\mathsf{P}}, s^0_{\mathsf{Q}})$
- $F = F_{\mathcal{E}(\mathsf{P})} \times F_{\mathcal{E}(\mathsf{Q})}$*.*

In contrast to the definition in [6], here we give a more elaborate handling of τ transitions in order to use the same base definition for conjunction under substitutivity and progress. The original definition permitted τ transitions to proceed independently, which allows the conjunction to diverge if at least one of the components can diverge. However, this is not acceptable under our progress-sensitive refinement preorder. Instead, we must only allow the conjunction to diverge on occasions when both components are willing to

diverge. This is achieved by condition C6 and the fact that the remaining conditions work on the τ-closure of the components.

We now inductively define the pruned conjunction of two components, which is used for defining conjunction under the progress-sensitive preorder.

Definition 25. *Let* P *and* Q *be components composable for conjunction. The progress-sensitive conjunction of* P *and* Q*, denoted* $\mathsf{P} \wedge_l \mathsf{Q}$*, is obtained from* $\mathsf{P} \wedge \mathsf{Q}$ *by pruning all states in* X*, the smallest set defined inductively by:*

- *If* p *is stable,* $p \xrightarrow{o}_{\mathcal{E}(\mathsf{P})}$ *for some* $o \in \mathcal{A}^O_{\mathsf{P}}$*, and* $\nexists a \in \mathcal{A}^O_{\mathsf{P}\wedge\mathsf{Q}} \cdot (p,q) \xrightarrow{a} (p',q')$ *with* $(p',q') \notin X$*, then* $(p,q) \in X$
- *If* q *is stable,* $q \xrightarrow{o}_{\mathcal{E}(\mathsf{Q})}$ *for some* $o \in \mathcal{A}^O_{\mathsf{Q}}$*, and* $\nexists a \in \mathcal{A}^O_{\mathsf{P}\wedge\mathsf{Q}} \cdot (p,q) \xrightarrow{a} (p',q')$ *with* $(p',q') \notin X$*, then* $(p,q) \in X$
- *If* $(p,q) \xrightarrow{a} (p',q')$ *for* $a \in \mathcal{A}^I_{\mathsf{P}\wedge\mathsf{Q}}$ **implies** $(p',q') \in X$*, then* $(p,q) \in X$*.*

Note that $\mathsf{P} \wedge_l \mathsf{Q}$ may prune the initial state in $\mathsf{P} \wedge \mathsf{Q}$, in which case we say that $\mathsf{P} \wedge_l \mathsf{Q}$ is unrealisable. As for parallel, there is a correspondence between conjunction at the operational and trace-based levels.

Theorem 15. *Let* P *and* Q *be operational components composable for conjunction. Then* $[\![\mathsf{P} \wedge \mathsf{Q}]\!] = [\![\mathsf{P}]\!] \wedge [\![\mathsf{Q}]\!]$ *and* $[\![\mathsf{P} \wedge_l \mathsf{Q}]\!]^l = [\![\mathsf{P}]\!]^l \wedge_l [\![\mathsf{Q}]\!]^l$*.*

Proof. Showing $[\![\mathsf{P} \wedge \mathsf{Q}]\!] = [\![\mathsf{P}]\!] \wedge [\![\mathsf{Q}]\!]$ is trivial, since if $t \in T_{[\![\mathsf{P}\wedge\mathsf{Q}]\!]}$, then $(s^0_{\mathsf{P}}, s^0_{\mathsf{Q}}) \stackrel{t}{\Longrightarrow}_{\mathsf{P}\wedge\mathsf{Q}} (p,q)$. If $s^0_{\mathsf{P}} \stackrel{t}{\Longrightarrow}_{\mathsf{P}} p$, then $t \in T_{[\![\mathsf{P}]\!]}$, while if $s^0_{\mathsf{P}} \stackrel{t}{\not\Longrightarrow}_{\mathsf{P}} p$, then $t \in T_{[\![\mathsf{P}]\!]} \uparrow \mathcal{A}^I_{\mathsf{Q}}$. Similarly for Q. Either way, $t \in T_{[\![\mathsf{P}]\!]\wedge[\![\mathsf{Q}]\!]}$. The other direction is similar, as is the inconsistent trace containment.

To show that $[\![\mathsf{P} \wedge_l \mathsf{Q}]\!]^l = [\![\mathsf{P}]\!]^l \wedge_l [\![\mathsf{Q}]\!]^l$, it is sufficient to prove that $t \in T_{[\![\mathsf{P}\wedge\mathsf{Q}]\!]}$ implies: $t \in Err$ iff $\forall p,q \cdot (s^0_{\mathsf{P}}, s^0_{\mathsf{Q}}) \stackrel{t}{\Longrightarrow}_{\mathsf{P}\wedge\mathsf{Q}} (p,q)$ implies $(p,q) \in X$. This can be demonstrated in a straightforward manner using an inductive argument by approximating Err and X, which are both obtained as fixed points. □

4.4. Disjunction

As the trace-based definition of disjunction does not need to prune error traces, the operational definition of disjunction is applicable to both the substitutive and progress-sensitive refinements.

Definition 26. *Let* P *and* Q *be components composable for disjunction. Then the disjunction of* P *and* Q *is the component* $\mathsf{P} \vee \mathsf{Q} = \mathsf{P} \vee_l \mathsf{Q} = \langle \mathcal{A}^I_{\mathsf{P}} \cap \mathcal{A}^I_{\mathsf{Q}}, \mathcal{A}^O_{\mathsf{P}} \cup \mathcal{A}^O_{\mathsf{Q}}, S, \longrightarrow, s_0, F \rangle$*, where:*

- $S = \{s_0\} \cup S_{\mathsf{P}} \cup S_{\mathsf{Q}}$*, for* $s_0 \notin S_{\mathsf{P}}, S_{\mathsf{Q}}$
- $\longrightarrow$ *is the smallest relation respecting the chaotic nature of inconsistent states that contains* $\longrightarrow_{\mathsf{P}}$ *and* $\longrightarrow_{\mathsf{Q}}$ *restricted to* $\mathcal{A}_{\mathsf{P} \vee \mathsf{Q}}$*, and the transitions* $s_0 \xrightarrow{\tau} s^0_{\mathsf{P}}$ *and* $s_0 \xrightarrow{\tau} s^0_{\mathsf{Q}}$
- $F = F_{\mathsf{P}} \cup F_{\mathsf{Q}}$.

A correspondence can be shown between the two forms of operational disjunction and the trace-based versions.

Theorem 16. *Let* P *and* Q *be components composable for disjunction. Then* $[\![\mathsf{P} \vee \mathsf{Q}]\!] = [\![\mathsf{P}]\!] \vee [\![\mathsf{Q}]\!]$ *and* $[\![\mathsf{P} \vee_l \mathsf{Q}]\!]^l = [\![\mathsf{P}]\!]^l \vee_l [\![\mathsf{Q}]\!]^l$.

Proof. Obvious given the definition of disjunction in both the substitutive and progress-sensitive trace-based frameworks. □

4.5. Hiding

Since hiding is not concerned with the refinement preorder, it has a common definition for both the substitutive and progress frameworks.

Definition 27. *Let* P *be a component and let* b *be an action. The hiding of* b *from* P *is the component* $\mathsf{P}/b = \mathsf{P} /_l b = \langle \mathcal{A}^I_{\mathsf{P}} \setminus \{b\}, \mathcal{A}^O_{\mathsf{P}} \setminus \{b\}, S_{\mathsf{P}}, \longrightarrow, s^0_{\mathsf{P}}, F_{\mathsf{P}} \rangle$*, where:*

H1. If $p \xrightarrow{a}_{\mathsf{P}} p'$ *and* $a \neq b$*, then* $p \xrightarrow{a} p'$

H2. If $p \xrightarrow{b}_{\mathsf{P}} p'$ *and* $b \in \mathcal{A}^O_{\mathsf{P}}$*, then* $p \xrightarrow{\tau} p'$.

As for all of the previously considered operators, there is a natural correspondence between hiding on operational and trace-based models.

Theorem 17. *Let* P *be a component, and let* b *be an arbitrary action. Then* $[\![\mathsf{P}/b]\!] = [\![\mathsf{P}]\!]/b$ *and* $[\![\mathsf{P} /_l b]\!]^l = [\![\mathsf{P}]\!]^l /_l b$.

Proof. Trivial given the trace-based definition of hiding. □

4.6. Quotient

The operational definition of quotient needs to consider all resolutions of non-determinism in the components to be composed. For simplicity, we therefore restrict to deterministic components without τ-transitions. We begin by giving an operational definition of quotient for which we must prune a number of states that violate inconsistency containment on the substitutive refinement preorder. We then extend the pruning so that it removes violations of the quiescence containment on the progress-sensitive refinement relation. To improve the clarity of our definition, we further assume that the quotient can observe all of R's actions.

Definition 28. *Let* P *and* R *be deterministic components such that* $\mathcal{A}^O_\mathsf{P} \subseteq \mathcal{A}^O_\mathsf{R}$ *and, without loss of generality,* $\bot_\mathsf{R} \in F_\mathsf{R}$*. Then the quotient is the component* $\mathsf{R}/\mathsf{P} = \langle \mathcal{A}^I_{\mathsf{R}/\mathsf{P}}, \mathcal{A}^O_{\mathsf{R}/\mathsf{P}}, S_{\mathsf{R}/\mathsf{P}}, \longrightarrow, s_0, F_{\mathsf{R}/\mathsf{P}} \rangle$*, where:*

- $\mathcal{A}^I_{\mathsf{R}/\mathsf{P}} = \mathcal{A}^I_\mathsf{R} \cup \mathcal{A}^O_\mathsf{P}$
- $\mathcal{A}^O_{\mathsf{R}/\mathsf{P}} = \mathcal{A}^O_\mathsf{R} \setminus \mathcal{A}^O_\mathsf{P}$
- $S_{\mathsf{R}/\mathsf{P}} = (S_\mathsf{R} \times S_\mathsf{P}) \setminus G$
- $\longrightarrow$ *is the smallest relation respecting the chaotic nature of inconsistent states and satisfying the following rules:*

  Q1. *If* $a \in \mathcal{A}_{\mathsf{R}/\mathsf{P}} \setminus \mathcal{A}_\mathsf{P}$ *and* $r \xrightarrow{a}_\mathsf{R} r'$*, then* $(r,p) \xrightarrow{a} (r',p)$

  Q2. *If* $a \in \mathcal{A}_{\mathsf{R}/\mathsf{P}} \cap \mathcal{A}_\mathsf{P}$*,* $r \xrightarrow{a}_\mathsf{R} r'$ *and* $p \xrightarrow{a}_\mathsf{P} p'$*, then* $(r,p) \xrightarrow{a} (r',p')$

  Q3. *If* $a \in \mathcal{A}_{\mathsf{R}/\mathsf{P}} \cap \mathcal{A}_\mathsf{P}$ *and* $p \not\xrightarrow{a}_\mathsf{P}$*, then* $(r,p) \xrightarrow{a} (\bot_\mathsf{R}, p)$
- $s_0 = (s^0_\mathsf{R}, s^0_\mathsf{P})$
- $F_{\mathsf{R}/\mathsf{P}} = (F_{\mathcal{E}(\mathsf{R})} \times S_\mathsf{P}) \setminus G$
- $G \subseteq S_\mathsf{R} \times S_\mathsf{P}$ *is the smallest set satisfying:*

  G1. *If* $r \notin F_{\mathcal{E}(\mathsf{R})}$ *and* $p \in F_\mathsf{P}$*, then* $(r,p) \in G$

  G2. *If* $a \in \mathcal{A}^O_\mathsf{R} \cap \mathcal{A}^O_\mathsf{P}$*,* $r \not\xrightarrow{a}_\mathsf{R}$ *and* $p \xrightarrow{a}_\mathsf{P}$*, then* $(r,p) \in G$

G3. If $(r,p) \xrightarrow{a} (r',p')$, $a \in \mathcal{A}^I_{\mathsf{R}/\mathsf{P}}$ *and* $(r',p') \in G$, *then* $(r,p) \in G$.

Conditions Q1 and Q2 essentially correspond to the parallel composition of P and R, whereby the two components synchronise on common actions, and interleave on the independent actions of R. Independent actions of P must be inputs, so they are irrelevant to the quotient, since an environment safe for R will never issue them. Condition Q3 states that the quotient can become inconsistent on an input that is never issued by P (meaning the action is an output of P). Conditions G1 and G2 capture situations where substitutivity would be violated, while G3 propagates the violation backwards to a point where the quotient can avoid it, by not producing an output from which the environment can, under its own control, reach the violation.

As quotient is the adjoint of parallel composition under the refinement relation, we must give an alternative characterisation for the progress-sensitive framework. We do this by removing states that introduce quiescence errors in the definition above.

Definition 29. *Let* P *and* R *be deterministic components such that* $\mathcal{A}^O_{\mathsf{P}} \subseteq \mathcal{A}^O_{\mathsf{R}}$. *Then the progress-sensitive quotient is the component* $\mathsf{R} \mathbin{/_l} \mathsf{P}$ *obtained from* R/P *by removing states contained within the smallest* X*-set defined by:*

- *If* $\exists o \in \mathcal{A}^O_{\mathsf{R}} \cdot r \xrightarrow{o}_{\mathsf{R}}$, $\nexists a \in \mathcal{A}^O_{\mathsf{P}} \cdot p \xrightarrow{a}_{\mathsf{P}}$ *and* $\nexists b \in \mathcal{A}^O_{\mathsf{R}/\mathsf{P}} \cdot (r,p) \xrightarrow{b}_{\mathsf{R}/\mathsf{P}} (r',p')$ *with* $(r',p') \notin X$, *then* $(r,p) \in X$
- *If* $(r,p) \xrightarrow{a} (r',p')$, $a \in \mathcal{A}^I_{\mathsf{R}/\mathsf{P}}$ *and* $(r',p') \in X$, *then* $(r,p) \in X$.

As usual, the operational definitions are closely related to the trace-based definitions.

Theorem 18. *Let* P *and* R *be deterministic components such that* $\mathcal{A}^O_{\mathsf{P}} \subseteq \mathcal{A}^O_{\mathsf{R}}$. *Then* $[\![\mathsf{R}/\mathsf{P}]\!] = [\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]$ *and* $[\![\mathsf{R} \mathbin{/_l} \mathsf{P}]\!]^l = [\![\mathsf{R}]\!]^l \mathbin{/_l} [\![\mathsf{P}]\!]^l$.

Proof. First show for the safety setting that $t \in T_{[\![\mathsf{R}/\mathsf{P}]\!]} \iff t \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ and $t \in F_{[\![\mathsf{R}/\mathsf{P}]\!]} \iff t \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

Begin by supposing $t \in T_{[\![\mathsf{R}/\mathsf{P}]\!]}$, and assume the result holds for all strict prefixes of t. Then there exists $(r,p) \notin G$ such that $(s^0_{\mathsf{R}}, s^0_{\mathsf{P}}) \xRightarrow{t} (r,p)$. By the definition of $\longrightarrow$ it follows that either $s^0_{\mathsf{R}} \xRightarrow{t}_{\mathsf{R}} r$ and $s^0_{\mathsf{P}} \xRightarrow{t\upharpoonright\mathcal{A}_{\mathsf{P}}}_{\mathsf{P}} p$, or there exists a prefix $t'a$ of t such that $s^0_{\mathsf{R}} \xRightarrow{t'}_{\mathsf{R}} r'$, $s^0_{\mathsf{P}} \xRightarrow{t'\upharpoonright\mathcal{A}_{\mathsf{P}}}_{\mathsf{P}} p'$ and $p' \not\xrightarrow{a}_{\mathsf{P}}$ with

$a \in \mathcal{A}_{\mathsf{P}}^{O}$. For the former, it holds that $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{[\![\mathsf{P}]\!]}$ and $t \in T_{[\![\mathsf{R}]\!]}$. Hence $t \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$, noting that prefix closure holds by the induction hypothesis, and $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in F_{[\![\mathsf{P}]\!]}$ implies $t \in F_{\mathcal{E}([\![\mathsf{R}]\!])}$, otherwise at least one of G1-3 would have been satisfied. For the latter, it follows that $t' \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$, and so $t'a \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ given $t'a \upharpoonright \mathcal{A}_{\mathsf{P}} \notin T_{[\![\mathsf{P}]\!]}$. Hence $t \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

Now suppose that $t \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Then (i) $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{[\![\mathsf{P}]\!]}$ implies $t \in T_{\mathcal{E}([\![\mathsf{R}]\!])}$ and (ii) $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in F_{[\![\mathsf{P}]\!]}$ implies $t \in F_{\mathcal{E}(\mathsf{R})}$. By (i) it follows that G2 cannot hold on any prefix, and by (i) it follows that G1 cannot hold on any prefix. Further, G3 cannot hold since $T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ is required to be input-receptive. Therefore, either $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{[\![\mathsf{P}]\!]}$ or there exists a prefix $t'a$ of t such that $t' \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{[\![\mathsf{P}]\!]}$ and $t'a \upharpoonright \mathcal{A}_{\mathsf{P}} \notin T_{[\![\mathsf{P}]\!]}$ with $a \in \mathcal{A}_{\mathsf{P}}^{O}$. For the former, it follows by Q1-3 that there exists (r, p) such that $(s_{\mathsf{R}}^0, s_{\mathsf{P}}^0) \stackrel{t}{\Longrightarrow} (r, p)$, giving $t \in T_{[\![\mathsf{R}/\mathsf{P}]\!]}$. For the latter, there exists (r, p) such that $(s_{\mathsf{R}}^0, s_{\mathsf{P}}^0) \stackrel{t'}{\Longrightarrow} (r, p)$, from which, by Q3, $(r, p) \stackrel{a}{\longrightarrow} (\bot_{\mathsf{R}}, p)$, which is inconsistent. Hence $t'a \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$, which implies $t \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$, thus giving $t \in T_{[\![\mathsf{R}/\mathsf{P}]\!]}$.

Now consider $t \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$, and assume there is no strict prefix of t contained in this set. By the minimality of t, it follows $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{[\![\mathsf{P}]\!]}$ and $t \in F_{\mathcal{E}([\![\mathsf{R}]\!])}$. Consequently, there exists p such that $s_{\mathsf{P}}^0 \stackrel{t\upharpoonright\mathcal{A}_{\mathsf{P}}}{\Longrightarrow}_{\mathsf{P}} p$ and there exists r such that $s_{\mathsf{R}}^0 \stackrel{t}{\Longrightarrow}_{\mathsf{R}} r$. Hence $(s_{\mathsf{R}}^0, s_{\mathsf{P}}^0) \stackrel{t}{\Longrightarrow} (r, p)$ with $(r, p) \in F_{\mathsf{R}/\mathsf{P}}$, providing G1-3 do not hold. But none of these conditions can hold, otherwise $t \notin T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Therefore, $(r, p) \notin G$, implying $t \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$.

Finally, suppose that $t \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$ and there is no strict prefix of t also in this set. Then there exists $(r, p) \in F_{\mathsf{R}/\mathsf{P}}$ such that $(s_{\mathsf{R}}^0, s_{\mathsf{P}}^0) \stackrel{t}{\Longrightarrow} (r, p)$ with $(r, p) \notin G$. So $s_{\mathsf{R}}^0 \stackrel{t}{\Longrightarrow}_{\mathsf{R}} r$ and either $s_{\mathsf{P}}^0 \stackrel{t}{\Longrightarrow}_{\mathsf{P}} p$ or $t \equiv t'a$ with $a \in \mathcal{A}_{\mathsf{P}}^{O} \cap \mathcal{A}_{\mathsf{R}}^{O}$ such that $s_{\mathsf{P}}^0 \stackrel{t'}{\Longrightarrow}_{\mathsf{P}} p'$ for some p' and $p' \stackrel{a}{\not\longrightarrow}_{\mathsf{P}}$. For the former, it follows that $t \in F_{\mathcal{E}([\![\mathsf{R}]\!])}$ and $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{[\![\mathsf{P}]\!]}$, thus yielding $t \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ given $t \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. For the latter, $t \upharpoonright \mathcal{A}_{\mathsf{P}} \notin T_{[\![\mathsf{P}]\!]}$ implies $t \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

For the liveness equivalence, it is sufficient to show that $t \in Err_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ iff $t \in X_{\mathsf{R}/\mathsf{P}}$. This can be demonstrated in a straightforward manner using an inductive argument on the approximations of $Err_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ and $X_{\mathsf{R}/\mathsf{P}}$. Note that the definition of $Err_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ can be greatly simplified, as we assume $\mathcal{A}_{\mathsf{R}} = \mathcal{A}_{\mathsf{R}/\mathsf{P}}$ along with determinism, the latter of which implies divergence freedom. $\square$

4.7. Full Abstraction

The close correspondence between the operational and trace-based models allows us to present a full abstraction result for the operational framework. This relies on showing that operational refinement $\sqsubseteq_{op}$ given in terms of trace containment can be equated with contextual checking of inconsistency in the operational models.

Definition 30. *Let* P *and* Q *be operational components. Then* Q *is said to be inconsistency substitutable for* P*, denoted by* $\mathsf{Q} \sqsubseteq^{F}_{op} \mathsf{P}$*, iff an inconsistent state reachable from* s^0_{Q} *by hidden and output actions implies there is an inconsistent state reachable from* s^0_{P} *by hidden and output actions.*

From this, $\mathsf{Q} \sqsubseteq_{op} \mathsf{P}$ can be characterised by $\sqsubseteq^{F}_{op}$ when considering the environments that Q and P can interact with. This shows that $\sqsubseteq_{op}$ is the weakest preorder preserving substitutivity.

Theorem 19. *Let* P *and* Q *be operational components such that* $\mathcal{A}^I_{\mathsf{P}} \subseteq \mathcal{A}^I_{\mathsf{Q}}$*,* $\mathcal{A}^O_{\mathsf{Q}} \subseteq \mathcal{A}^O_{\mathsf{P}}$ *and* $\mathcal{A}^I_{\mathsf{Q}} \cap \mathcal{A}^O_{\mathsf{P}} = \emptyset$*. Then:*

$$\mathsf{Q} \sqsubseteq_{op} \mathsf{P} \textit{ iff } \forall \mathsf{R} \cdot \mathcal{A}^O_{\mathsf{R}} = \mathcal{A}^I_{\mathsf{P}} \textit{ and } \mathcal{A}^I_{\mathsf{R}} = \mathcal{A}^O_{\mathsf{Q}} \implies \mathsf{Q} \,||\, \mathsf{R} \sqsubseteq^{F}_{op} \mathsf{P} \,||\, \mathsf{R}.$$

Proof. A straightforward modification to Theorem 7. □

Based on this result, it is straightforward to show full abstraction.

Corollary 2. *Operational equivalence* $\equiv_{op}$ *is fully abstract for parallel composition, conjunction, disjunction, hiding and quotient with respect to observational equivalence of inconsistency.*

Proof. Same reasoning as in Corollary 1 (with updated references). □

5. On the Relationship with Interface Automata

In this section, we relate our operational theory of components to the interface automata defined in [11]. We show that the theory of interface automata can be embedded within our framework, and demonstrate that the alternating refinement relation is stronger than our substitutive preorder.

5.1. Interface Automata

We recall a general definition of interface automata [11], which, unlike the restrictions imposed in [12], permits hidden transitions and does not insist on determinism of inputs. Thus, an interface automaton can be thought of as a finite-state machine with transitions labelled by input, output or τ actions, and does not require input enabledness in each state.

Definition 31. *An* interface automaton P *is a tuple* $\langle S_{\mathsf{P}}, \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, \longrightarrow_{\mathsf{P}}, s^0_{\mathsf{P}} \rangle$*, where:*

- S_{P} *is a finite set of states*
- $\mathcal{A}^I_{\mathsf{P}}$ *is a finite set of input actions*
- $\mathcal{A}^O_{\mathsf{P}}$ *is a finite set of output actions, disjoint from* $\mathcal{A}^I_{\mathsf{P}}$
- $\longrightarrow_{\mathsf{P}} \subseteq S_{\mathsf{P}} \times (\mathcal{A}_{\mathsf{P}} \cup \{\tau\}) \times S_{\mathsf{P}}$ *is the transition relation*
- $s^0_{\mathsf{P}} \in S_{\mathsf{P}}$ *is the designated initial state.*

Substitutive refinement of interface automata is given by means of alternating simulation [2], with a covariant inclusion on inputs and contravariant inclusion on outputs. Again, we reproduce the general definition from [11], which is free of unnecessary restrictions. First, we introduce two shorthands for simplifying the definition:

- $Act^I_{\mathsf{P}}(p) \triangleq \{a \in \mathcal{A}^I_{\mathsf{P}} : p \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{P}} p' \text{ implies } p' \stackrel{a}{\longrightarrow}_{\mathsf{P}}\}$
- $Act^O_{\mathsf{P}}(p) \triangleq \{a \in \mathcal{A}^O_{\mathsf{P}} : p \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{P}} \stackrel{a}{\longrightarrow}_{\mathsf{P}}\}$.

The set $Act^I_{\mathsf{P}}(p)$ denotes the input actions that may safely be issued when P is in state p. Any action in $Act^I_{\mathsf{P}}(p)$ must therefore be enabled in any state reachable from p by hidden transitions. On the other hand, $Act^O_{\mathsf{P}}(p)$ represents the output actions of P that the environment must be willing to accept. Thus, this set is the collection of outputs enabled in any state reachable from p by hidden transitions. We now give the formal definition of alternating refinement.

Definition 32. *Interface automaton* Q *is said to be an* alternating refinement *of* P*, written* $\mathsf{Q} \sqsubseteq_{IA} \mathsf{P}$*, just if* $\mathcal{A}^I_{\mathsf{P}} \subseteq \mathcal{A}^I_{\mathsf{Q}}$*,* $\mathcal{A}^O_{\mathsf{Q}} \subseteq \mathcal{A}^O_{\mathsf{P}}$*, and* $s^0_{\mathsf{Q}}\ R\ s^0_{\mathsf{P}}$*, where* $R \subseteq S_{\mathsf{Q}} \times S_{\mathsf{P}}$ *is an alternating simulation satisfying the property: if* $q\ R\ p$*, then:*

AS1. $Act^I_{\mathsf{P}}(p) \subseteq Act^I_{\mathsf{Q}}(q)$

AS2. $Act^O_{\mathsf{Q}}(q) \subseteq Act^O_{\mathsf{P}}(p)$

AS3. For each $a \in Act^I_{\mathsf{P}}(p) \cup Act^O_{\mathsf{Q}}(q)$ *and for each* $q \stackrel{a}{\Longrightarrow}|_{\mathsf{Q}} q'$*, there exists* $p \stackrel{a}{\Longrightarrow}|_{\mathsf{P}} p'$ *such that* $q'\ R\ p'$*.*

Conditions AS1 and AS2 require that q can safely accept any input that p is willing to accept, while q will only produce a subset of outputs that p can produce. Condition AS3 propagates this constraint on to the common successor states.

5.1.1. Relation with Operational Components

We now indicate how to map interface automata to the operational components as defined in Section 4. The mapping must add additional transitions for the non-enabled inputs to a special inconsistent state $\perp$.

Definition 33. *Let* P *be an interface automaton. Then the corresponding operational component is* $[\![\mathsf{P}]\!]^{IA} = \langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, S_{\mathsf{P}} \cup \{\perp\}, \longrightarrow, s^0_{\mathsf{P}}, \{\perp\}\rangle$*, where:*

$$\begin{aligned}\longrightarrow = \longrightarrow_{\mathsf{P}} &\cup \{(s, a, \perp) : s \in S_{\mathsf{P}},\ a \in \mathcal{A}^I_{\mathsf{P}} \text{ and } \nexists s' \cdot s \stackrel{a}{\longrightarrow}_{\mathsf{P}} s'\} \\ &\cup \{(\perp, a, \perp) : a \in \{\tau\} \cup \mathcal{A}_{\mathsf{P}}\}.\end{aligned}$$

Given this definition, it should be straightforward to see that interface automata are a subclass of our operational components, in particular, the components that can only become inconsistent by seeing a bad input, and that are not permitted to be inconsistent up front.

The following theorem shows the relationship between alternating refinement and the substitutive preorder of our modelling framework.

Theorem 20. *Let* P *and* Q *be interface automata. Then* $\mathsf{Q} \sqsubseteq_{IA} \mathsf{P}$ *implies* $[\![\mathsf{Q}]\!]^{IA} \sqsubseteq_{op} [\![\mathsf{P}]\!]^{IA}$*.*

Proof. Begin by supposing $\mathsf{Q} \sqsubseteq_{IA} \mathsf{P}$ and let t be the smallest trace such that $t \in F_{\mathcal{E}([\![\mathsf{Q}]\!]^{IA})} \cap \mathcal{A}^*_{\mathsf{P}}$ and $t \notin F_{\mathcal{E}([\![\mathsf{P}]\!]^{IA})}$. By definition of interface automata, it follows that $t \in F_{[\![\mathsf{Q}]\!]^{IA}}$ and $t \notin F_{[\![\mathsf{P}]\!]^{IA}}$ as the automata can only be inconsistent on seeing a bad input. Moreover, as the automata cannot be inconsistent up front, it follows that $t \equiv t'a$ with $a \in \mathcal{A}^I_{\mathsf{P}}$. By minimality of t, we know $t' \in T_{[\![\mathsf{P}]\!]^{IA}} \setminus F_{[\![\mathsf{P}]\!]^{IA}}$ and also that $t' \in T_{[\![\mathsf{Q}]\!]^{IA}} \setminus F_{[\![\mathsf{Q}]\!]^{IA}}$. Consequently, for each state q' such that $s^0_{\mathsf{Q}} \stackrel{t'}{\Longrightarrow}_{\mathsf{Q}} q'$, it follows that there exists p' such

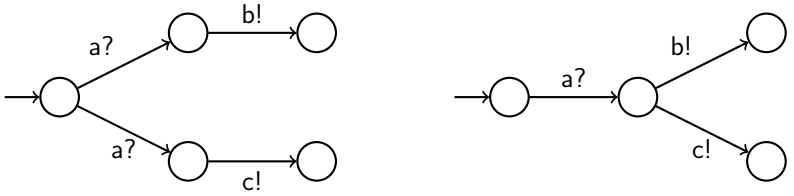

Figure 13: Interface automata distinguishing alternating simulation and $\sqsubseteq_{imp}$.

that $s^0_{\mathsf{P}} \stackrel{t'}{\Longrightarrow}_{\mathsf{P}} p'$, where at each intermediate state AS1 and AS2 hold, and $q' \; R \; p'$ for an alternating simulation R. For at least one of these q', it follows that $q' \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{Q}} \stackrel{a}{\not\longrightarrow}_{\mathsf{Q}}$ (given $t'a \in F_{[\![\mathsf{Q}]\!]^{IA}}$). However, as $t'a \in T_{[\![\mathsf{P}]\!]^{IA}} \setminus F_{[\![\mathsf{P}]\!]^{IA}}$ it follows that $a \in Act^I_{\mathsf{P}}(p')$. Hence AS1 is violated, meaning $q' \not R \; p'$, which is contradictory. Therefore, $F_{[\![\mathsf{Q}]\!]^{IA}} \cap \mathcal{A}^*_{\mathsf{P}} \subseteq F_{[\![\mathsf{P}]\!]^{IA}}$ as required.

Now suppose that $\mathsf{Q} \sqsubseteq_{IA} \mathsf{P}$ and let t be the smallest trace such that $t \in (T_{[\![\mathsf{Q}]\!]^{IA}} \setminus F_{[\![\mathsf{Q}]\!]^{IA}}) \cap \mathcal{A}^*_{\mathsf{P}}$ and $t \notin T_{[\![\mathsf{P}]\!]^{IA}}$. It therefore follows that $t = t'a$ with $a \in \mathcal{A}^O_{\mathsf{Q}}$, and $t \in (\mathcal{A}_{\mathsf{P}} \cap \mathcal{A}_{\mathsf{Q}})^*$. Consequently, for each state q' such that $s^0_{\mathsf{Q}} \stackrel{t'}{\Longrightarrow}_{\mathsf{Q}} q'$, it follows that there exists p' such that $s^0_{\mathsf{P}} \stackrel{t'}{\Longrightarrow}_{\mathsf{P}} p'$, where at each intermediate state AS1 and AS2 hold, and $q' \; R \; p'$ for an alternating simulation R. For at least one of these q', it follows that $q' \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{Q}} \stackrel{a}{\longrightarrow}_{\mathsf{Q}}$, hence $a \in Act^O_{\mathsf{Q}}(q')$. However, as $t'a \notin T_{[\![\mathsf{P}]\!]^{IA}}$ it follows that $a \notin Act^O_{\mathsf{P}}(p')$ for any p' reachable under t'. Hence AS2 is violated, meaning $q' \not R \; p'$, which again is contradictory. As a result, $T_{[\![\mathsf{Q}]\!]^{IA}} \cap \mathcal{A}^*_{\mathsf{P}} \subseteq T_{[\![\mathsf{P}]\!]^{IA}}$. □

Being a branching-time relation, alternating refinement is too strong for substitutivity. This is demonstrated by the interface automata in Figure 13. The automaton on the left is an alternating refinement of the one on the right, but not vice-versa, whereas the component representations of the automata are substitutively equivalent in our framework under $\equiv_{op}$. Consequently, it is not the case in Theorem 20 that $[\![\mathsf{Q}]\!]^{IA} \sqsubseteq_{op} [\![\mathsf{P}]\!]^{IA}$ implies $\mathsf{Q} \sqsubseteq_{IA} \mathsf{P}$.

The existence of a matching transition in condition AS3 is the cause of this asymmetry in the expressive power of alternating refinement and our substitutive preorder. If we restrict to deterministic interface automata, the choice of successor is determined, and so the two refinements coincide.

Theorem 21. *Let* P *and* Q *be deterministic interface automata. Then* $\mathsf{Q} \sqsubseteq_{IA} \mathsf{P}$ *iff* $[\![\mathsf{Q}]\!]^{IA} \sqsubseteq_{op} [\![\mathsf{P}]\!]^{IA}$.

Proof. Based on Theorem 20, alternating simulation implies our trace-based refinement. So suppose $\mathsf{Q} \not\sqsubseteq_{IA} \mathsf{P}$. Then there exists a smallest trace t such that $s^0_{\mathsf{Q}} \stackrel{t}{\Longrightarrow}_{\mathsf{Q}} q'$, but no state p' such that $s^0_{\mathsf{P}} \stackrel{t}{\Longrightarrow}_{\mathsf{P}} p'$ and $q' \; R \; p'$. Note that by determinism q' is uniquely defined, as is p' if it exists. If p' exists, then $q' \not R \; p'$ meaning either AS1 or AS2 is violated. If AS1 is violated, then $q' \stackrel{a}{\not\longrightarrow}_{\mathsf{Q}}$ while $p' \stackrel{a}{\longrightarrow}_{\mathsf{P}}$ for some $a \in \mathcal{A}^I_{\mathsf{P}}$. Hence $ta \in F_{[\![\mathsf{Q}]\!]^{IA}}$ while $ta \notin F_{[\![\mathsf{P}]\!]^{IA}}$, which implies $[\![\mathsf{Q}]\!]^{IA} \not\sqsubseteq_{op} [\![\mathsf{P}]\!]^{IA}$. Instead, if AS2 is violated, then $q' \stackrel{a}{\longrightarrow}_{\mathsf{Q}}$ while $p' \stackrel{a}{\not\longrightarrow}_{\mathsf{P}}$ for some $a \in \mathcal{A}^O_{\mathsf{Q}}$. Hence $ta \in T_{[\![\mathsf{Q}]\!]^{IA}}$ while $ta \notin T_{[\![\mathsf{P}]\!]^{IA}}$, which also implies $[\![\mathsf{Q}]\!]^{IA} \not\sqsubseteq_{op} [\![\mathsf{P}]\!]^{IA}$. The final possibility is that p' does not exist, in which case $t \equiv t'a$, and $s^0_{\mathsf{P}} \stackrel{t'}{\Longrightarrow}_{\mathsf{P}}$ while $s^0_{\mathsf{P}} \stackrel{t}{\not\Longrightarrow}_{\mathsf{P}}$. As $\mathsf{Q} \not\sqsubseteq_{IA} \mathsf{P}$, it follows that $a \in \mathcal{A}^O_{\mathsf{Q}}$, but there is no matching transition in P. Consequently, $t \in T_{[\![\mathsf{Q}]\!]^{IA}}$, but $t \notin T_{[\![\mathsf{P}]\!]^{IA}}$, which yields $[\![\mathsf{Q}]\!]^{IA} \not\sqsubseteq_{op} [\![\mathsf{P}]\!]^{IA}$ as required. □

It is worth pointing out that the definition of alternating refinement in [12], which applies only to input-deterministic interface automata, is also too strong for substitutivity, since the original definition of alternating refinement relates more input-deterministic models than the later definition.

5.1.2. Compositional Operators

In this section, we briefly remark on the relation between the composition operators for interface automata and our operational framework.

Parallel composition of interface automata P and Q can be defined as $[\![\mathsf{P}]\!]^{IA} \mid\mid [\![\mathsf{Q}]\!]^{IA}$, after propagating inconsistencies backwards over output and τ transitions, and removing the resultant inconsistent states. The obtained model is an interface automaton only if the initial state remains. This also provides a characterisation of *compatibility* for interface automata: P and Q are compatible only if, after performing the parallel composition as just defined, the initial state remains.

Conjunction is more problematic to define, because of the discrepancies between alternating simulation and our substitutive refinement. If we consider only deterministic interface automata, for which the refinements coincide, conjunction of interface automata P and Q can be defined as $[\![\mathsf{P}]\!]^{IA} \wedge [\![\mathsf{Q}]\!]^{IA}$, after having pruned all inconsistent states. Disjunction can be defined similarly.

Hiding is also straightforward, in that removal of b from interface automaton P is given by $[\![\mathsf{P}]\!]^{IA}/b$, once all inconsistent states have been removed.

As quotient for interface automata is only defined on deterministic models

[4], alternating refinement and our substitutive refinement coincide. Therefore, the quotient of interface automaton P from R is given by the removal of inconsistent states from $[\![\mathsf{R}]\!]^{IA}/[\![\mathsf{P}]\!]^{IA}$, but is only defined when $[\![\mathsf{R}]\!]^{IA}/[\![\mathsf{P}]\!]^{IA}$ is realisable, the latter meaning that an initial state exists.

6. Conclusion and Future Work

We have developed a compositional specification theory for components that may be modelled operationally, closely mirroring actual implementations, or in an abstract manner by means of trace structures. Both frameworks admit linear-time refinement relations, defined in terms of traces, which correspond to substitutivity and progress-sensitive substitutivity respectively. We define the operations of parallel composition, conjunction, disjunction, hiding and quotient, and prove that the induced equivalence is a congruence for these operations, allowing us to provide full abstraction results. The simplicity of our formalism facilitates compositional reasoning about the temporal ordering of interactions needed for assume-guarantee inference (contracts), both for safety [8] and (progress-sensitive) liveness properties [9], as well as timed contracts [7].

Acknowledgments. The authors are supported by EU FP7 project CONNECT (231167) and ERC Advanced Grant VERIWARE. We would also like to thank the anonymous reviewers for their insightful comments.

References

[1] Fides Aarts and Frits Vaandrager. Learning I/O Automata. In Paul Gastin and François Laroussinie, editors, *Concurrency Theory, Proc. 21st International Conference (CONCUR'10)*, volume 6269 of *Lecture Notes in Computer Science*, pages 71–85. Springer, 2010.

[2] Rajeev Alur, Thomas A. Henzinger, Orna Kupferman, and Moshe Y. Vardi. Alternating Refinement Relations. In Davide Sangiorgi and Robert de Simone, editors, *Concurrency Theory, Proc. 9th International Conference (CONCUR'98)*, volume 1466 of *Lecture Notes in Computer Science*, pages 163–178. Springer, 1998.

[3] Amel Bennaceur, Chris Chilton, Malte Isberner, and Bengt Jonsson. Automated Mediator Synthesis: Combining Behavioural and Ontological Reasoning. In Robert M. Hierons, Mercedes G. Merayo, and Mario

Bravetti, editors, *Software Engineering and Formal Methods, Proc. 11th International Conference (SEFM'13)*, volume 8137 of *Lecture Notes in Computer Science*, pages 274–288. Springer, 2013.

[4] Purandar Bhaduri and S. Ramesh. Interface synthesis and protocol conversion. *Formal Aspects of Computing*, 20(2):205–224, March 2008.

[5] Stephen D. Brookes, Charles A. R. Hoare, and A. William Roscoe. A Theory of Communicating Sequential Processes. *Journal of the ACM*, 31(3):560–599, June 1984.

[6] Taolue Chen, Chris Chilton, Bengt Jonsson, and Marta Kwiatkowska. A Compositional Specification Theory for Component Behaviours. In Helmut Seidl, editor, *Programming Languages and Systems, Proc. 21st European Symposium on Programming (ESOP'12)*, volume 7211 of *Lecture Notes in Computer Science*, pages 148–168. Springer, 2012.

[7] Chris Chilton. *An Algebraic Theory of Componentised Interaction*. PhD thesis, Department of Computer Science, University of Oxford, 2013.

[8] Chris Chilton, Bengt Jonsson, and Marta Kwiatkowska. Assume-Guarantee Reasoning for Safe Component Behaviours. In Corina Păsăreanu and Gwen Salaün, editors, *Formal Aspects of Component Software, Proc. 9th International Symposium (FACS'12)*, volume 7684 of *Lecture Notes in Computer Science*, pages 92–109. Springer, 2013.

[9] Chris Chilton, Bengt Jonsson, and Marta Kwiatkowska. Compositional assume-guarantee reasoning for input/output component theories. *Science of Computer Programming*, 91 Part A:115–137, October 2014.

[10] Chris Chilton, Marta Kwiatkowska, and Xu Wang. Revisiting Timed Specification Theories: A Linear-Time Perspective. In Marcin Jurdzinski and Dejan Nickovic, editors, *Formal Modeling and Analysis of Timed Systems, Proc. 10th International Conference (FORMATS'12)*, volume 7595 of *Lecture Notes in Computer Science*, pages 75–90. Springer, 2012.

[11] Luca de Alfaro and Thomas A. Henzinger. Interface automata. *ACM SIGSOFT Software Engineering Notes*, 26(5):109–120, September 2001.

[12] Luca de Alfaro and Thomas A. Henzinger. Interface-Based Design. In Manfred Broy, Johannes Grünbauer, David Harel, and Charles A. R.

Hoare, editors, *Engineering Theories of Software Intensive Systems, Proc. NATO Advanced Study Institute*, volume 195 of *NATO Science Series II: Mathematics, Physics and Chemistry*, pages 83–104. Springer, 2005.

[13] Rocco de Nicola and Roberto Segala. A process algebraic view of input/output automata. *Theoretical Computer Science*, 138(2):391–423, February 1995.

[14] David L. Dill. *Trace theory for automatic hierarchical verification of speed-independent circuits*. PhD thesis, Carnegie Mellon University, 1988.

[15] Laurent Doyen, Thomas A. Henzinger, Barbara Jobstmann, and Tatjana Petrov. Interface theories with component reuse. In Luca de Alfaro and Jens Palsberg, editors, *Embedded Software, Proc. 8th ACM International Conference (EMSOFT'08)*, pages 79–88. ACM, 2008.

[16] Jarad Drissi and Gregor von Bochmann. Submodule construction for systems of I/O automata. Technical Report 1133, DIRO, University of Montreal, 1999.

[17] Charles A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[18] Paola Inverardi and Massimo Tivoli. Automatic Synthesis of Modular Connectors via Composition of Protocol Mediation Patterns. In David Notkin, Betty H. C. Cheng, and Klaus Pohl, editors, *Software Engineering, Proc. 35th International Conference (ICSE'13)*, pages 3–12. IEEE Computer Society, 2013.

[19] Bengt Jonsson. A Hierarchy of Compositional Models of I/O Automata. Technical Report SICS:R91:04, Swedish Institute of Computer Science, 1991.

[20] Bengt Jonsson. Compositional specification and verification of distributed systems. *ACM Transactions on Programming Languages and Systems*, 16(2):259–303, March 1994.

[21] Mark B. Josephs. Receptive process theory. *Acta Informatica*, 29(1):17–31, February 1992.

[22] Mark B. Josephs, Charles A. R. Hoare, and He Jifeng. A Theory of Asynchronous Processes. Technical Report PRG-TR-6-89, Oxford University Computing Laboratory, 1989.

[23] Mark B. Josephs and Hemangee K. Kapoor. Controllable delay-insensitive processes. *Fundamenta Informaticae*, 78(1):101–130, January 2007.

[24] Kim G. Larsen, Ulrik Nyman, and Andrzej Wasowski. Modal I/O Automata for Interface and Product Line Theories. In Rocco De Nicola, editor, *Programming Languages and Systems, Proc. 16th European Symposium on Programming (ESOP'07)*, volume 4421 of *Lecture Notes in Computer Science*, pages 64–79. Springer, 2007.

[25] Gerald Lüttgen and Walter Vogler. Conjunction on processes: Full abstraction via ready-tree semantics. *Theoretical Computer Science*, 373(1-2):19–40, March 2007.

[26] Gerald Lüttgen and Walter Vogler. Ready simulation for concurrency: It's logical! *Information and Computation*, 208(7):845–867, July 2010.

[27] Nancy Lynch and Mark Tuttle. An introduction to input/output automata. *CWI Quarterly*, 2(3):219–246, September 1989.

[28] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980.

[29] Radu Negulescu. *Process Spaces and the Formal Verification of Asynchronous Circuits*. PhD thesis, University of Waterloo, 1998.

[30] Radu Negulescu. Process Spaces. In Catuscia Palamidessi, editor, *Concurrency Theory, Proc. 11th International Conference (CONCUR'00)*, volume 1877 of *Lecture Notes in Computer Science*, pages 199–213. Springer, 2000.

[31] Roberto Passerone. *Semantic Foundations for Heterogeneous Systems*. PhD thesis, University of California, Berkeley, 2004.

[32] Jean-Baptiste Raclet. Residual for component specifications. *Electronic Notes in Theoretical Computer Science*, 215:93–110, June 2008.

[33] Jean-Baptiste Raclet, Eric Badouel, Albert Benveniste, Benoît Caillaud, Axel Legay, and Roberto Passerone. Modal Interfaces: Unifying Interface Automata and Modal Specifications. In Samarjit Chakraborty and Nicolas Halbwachs, editors, *Embedded Software, Proc. 7th ACM International Conference (EMSOFT'09)*, pages 87–96. ACM, 2009.

[34] Jean-Baptiste Raclet, Eric Badouel, Albert Benveniste, Benoît Caillaud, Axel Legay, and Roberto Passerone. A modal interface theory for component-based design. *Fundamenta Informaticae*, 108(1-2):119–149, January 2011.

[35] Jean-Baptiste Raclet, Eric Badouel, Albert Benveniste, Benoît Caillaud, and Roberto Passerone. Why are modalities good for Interface Theories? In Stephen Edwards and Walter Vogler, editors, *Application of Concurrency to System Design, Proc. 9th International Conference (ACSD'09)*, pages 119–127. IEEE Computer Society, 2009.

[36] Judi Romijn and Frits Vaandrager. A note on fairness in I/O automata. *Information Processing Letters*, 59(5):245–250, September 1996.

[37] Roberto Segala. Quiescence, fairness, testing, and the notion of implementation. *Information and Computation*, 138(2):194–210, November 1997.

[38] Jan Tretmans. Model-Based Testing and Some Steps towards Test-Based Modelling. In Marco Bernardo and Valérie Issarny, editors, *Formal Methods for Eternal Networked Software Systems, 11th International School on Formal Methods for the Design of Computer, Communication and Software Systems (SFM'11)*, volume 6659 of *Lecture Notes in Computer Science*, pages 297–326. Springer, 2011.

[39] Rob J. van Glabbeek. Full Abstraction in Structural Operational Semantics (Extended Abstract). In Maurice Nivat, Charles Rattray, Teodor Rus, and Giuseppe Scollo, editors, *Algebraic Methodology and Software Technology, Proc. 3rd International Conference (AMAST'93)*, Workshops in Computing, pages 75–82. Springer, 1994.

[40] Tom Verhoeff. *A Theory of Delay-Insensitive Systems*. PhD thesis, Department of Mathematics and Computing Science, Eindhoven University of Technology, 1994.

[41] Elisabeth S. Wolf. *Hierarchical Models of Synchronous Circuits for Formal Verification and Substitution*. PhD thesis, Stanford University, 1995.